UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

LUCIANO SOLER

Compactação de Vídeo Escalável

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Dante Augusto Couto Barone
Orientador

Prof. Dr. José Valdeni de Lima
Co-orientador

Porto Alegre, setembro de 2006.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Soler, Luciano

Compactação de Vídeo Escalável / Luciano Soler – Porto Alegre: Programa de Pós-Graduação em Computação, 2006.

175 f.:il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2006. Orientador: Dante Augusto Couto Barone; Co-orientador: José Valdeni de Lima.

1.Codificação Escalável. 2.Televisão Digital 3.Codificação Avançada de Vídeo. I. Barone, Dante Augusto Couto. II. Lima, José Valdeni de. III. Compactação de Vídeo Escalável.

AGRADECIMENTOS

Antes de tudo, agradeço a Deus por ter me iluminado, dado saúde e paciência para superar todas as adversidades.

Agradeço aos meus pais João Antonio e Leonice Abigail, por serem a base da minha educação e por fazerem com que eu acredite no meu potencial. Avós, irmã, tios e demais familiares que sempre estiveram na torcida por mim.

A minha namorada, Elaine, por ter estado sempre ao meu lado, com seu apoio, afeto e principalmente paciência durante esta difícil jornada.

Aos meus orientadores Prof. Dante e Prof. Valdeni, pelos valiosos conselhos dados no decorrer desse trabalho, pela paciência e pelo incentivo dado em todas as horas difíceis ou não.

Aos companheiros da pós-graduação e da república Paraná, agradeço pela parceria na busca pelo desenvolvimento científico, tecnológico e cultural do nosso país, bem como os divertidos “deliciosos cafezinhos” na “hora do soninho” à tarde.

E a todas as outras pessoas aqui não citadas, mas que me apoiaram nos momentos mais difíceis, comemoraram cada conquista e foram peças fundamentais para que este trabalho fosse concluído.

Enfim, a essas pessoas, deixo meu muito obrigado, na esperança de que este trabalho esteja correspondendo a todas as expectativas.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

AAC	Advanced Audio Coding
ABP	Adaptative BitPlane coding
ASO	Arbitrary Slice Order
ASP	Advanced Simple Profile
ATM	Asynchronous Transfer Mode
ATSC	Advanced Television Systems Committee
AVC	Advanced Vídeo Coding
AWBP	Adaptative Weighted BitPlane coding
BAB	Binary Alpha Block
BIFS	Binary Format for Scenes
CABAC	Context Adaptative Binary Arithmetic Coding
CAE	Context based Arithmetic Encoding
CAVLC	Context Adaptative Variable Length Coding
CBP	Cyclic BitPlane coding
CELP	Coded Excited Linear Prediction
CHC	Conversational High Compression
CIF	Common Intermediate Format
CS	Coefficient Scanning
CWBP	Cyclic Weighted BitPlane coding
DAB	Digital Audio Broadcasting
DCT	Discrete Cosine Transform
DPCM	Diferential PCM
DVB	Digital Vídeo Broadcasting
DVD	Digital Versatile Disc
DWT	Discrete Wavelet Transform
EBCOT	Embedded Block Coding with Optimized Truncation
EI	Enhancement Intra
EOB	End Of Block
EOP	Endo Of Plane

EP	Enhancement P
EZW	Embedded Zero-tree Wavelet
FDIS	Final Draft International Standard
FEC	Forward Error Control
FGS	Fine Granularity Scalability
FIR	Fine Impulse Response
FLC	Fixed Length Coding
FMO	Flexible Macroblock Ordering
GOB	Group of Blocks
GOP	Group of Pictures
GPRS	General Packet Radio System
GSM	Global System Móbile
HD	High Definition
HEC	Header Extension Code
HLP	High Latency Profile
HQPI	High Quality Predicted Image
HQR	High Quality Reference
HTTP	Hyper Text Transfer Protocol
HVXC	Harmonic Vector eXitation Coding
ICT	Irreversible Component Transform
IDCT	Inverse DCT
IDR	Instantaneous Decoding Refresh
IEC	International Engineering Consortium
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPMP	Intellectual Property Management and Protection
ISMA	Internet Streaming Media Alliance
ISO	International Standards Organization
ITU	International Telecommunications Union
JM	Joint Model
JPEG	Joint Photographic Experts Group
JTC	Joint Technical Committee
JVT	Joint Vídeo Team
LAN	Local Área Network
LQR	Low Quality Reference

MBAmp	MacroBlock Allocation Map
Mbone	Multicast Backbone
MCFGS	Motion Compensation FGS
MCTF	Motion Compensation Temporal Filtering
MIME	Multi-Purpose Internet Mail Extensions
MP	Matching Pursuits
MP3	MPEG-1 Audio Layer 3
MPEG	Motion Picture Experts Group
MSB	Most Significant Bit
MTU	Maximum Transfer Unit
MZTE	Multiscale Zero-Tree Entropy
NAL	Network Adaptation Layer
OBMC	Overlapped Block Motion Compensation
OCI	Object Content Information
PBP	Priority Break Point
PCM	Pulse Code Modulation‘
PDA	Personal Digital Assistant
PEZW	Predictive Embedded Zero-tree Wavelet
PSNR	Peak Signal Noise Ratio
QCIF	Quarter CIF
QoS	Quality of Service
RCT	Reversible Component Transform
RD	Rate/Distortion
RLE	Run Length Encoding
ROI	Region of Interest
RTCP	Real-Time Control Protocol
RTP	Real-Time Transport Protocol
RTSP	Real-Time Streaming Protocol
SA-DWT	Shape Adaptative Discrete Wavelet Transform
SD	Standard Definition
SI	Switching Intra
SISC	Scan Interleaving base Shape Coding
SNR	Signal Noise Ration
SP	Switching Predicted
SPIHT	Set Partitioning In Hierarchical Trees

SSL	Switched Single Layer
STB	Set Top Box
TCP	Transpor Control Protocol
UDP	User Datagram Protocol
UMTS	Universal Móbile Telecommunicaion System
UVLC	Universal Variable Length Coding
VCEG	Vídeo Coding Expert Group
VCL	Vídeo Coding Layer
VLD	Variable Length Decoder
VLC	Variable Length Code
VO	Vídeo Object
VOD	Vídeo On Demand
VOL	Vídeo Object Layer
VTC	Visual Texture Coding
WQR	Worst Quality Reference
WTCQ	Wavelet Trellils Coded Quantization
xDSL	Digital Subscriber Line
ZTE	Zero-Tree Entropy
ZTC	Zero-Tree Coding

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A codificação de vídeo é um problema cuja solução deve ser projetada de acordo com as necessidades da aplicação desejada. Neste trabalho, um método de compressão de vídeo com escalabilidade é apresentado, apresentando melhorias dos formatos de compressão atuais.

A escalabilidade corresponde a capacidade de extrair do bitstream completo, conjuntos eficientes de bits que são decodificados oferecendo imagens ou vídeos decodificados com uma variação (escala) segundo uma dada característica da imagem ou vídeo. O número de conjuntos que podem ser extraídos do bitstream completo definem a granularidade da escalabilidade fornecida, que pode ser muito fina ou com passos grossos. Muitas das técnicas de codificação escalável utilizam uma camada base que deve ser sempre decodificada e uma ou mais camadas superiores que permitem uma melhoria em termos de qualidade (SNR), resolução espacial e/ou resolução temporal.

O esquema de codificação escalável final presente na norma MPEG-4 é uma das técnicas mais promissoras, pois pode adaptar-se às características dos canais (Internet) ou terminais que apresentam um comportamento variável ou desconhecido, como velocidade maxima de acesso, variações de largura de banda, erros de canal, etc. Apesar da norma MPEG-4 FGS se afirmar como uma alternativa viável para aplicações de distribuição de vídeo, possui uma quebra significativa de desempenho em comparação com a codificação não escalável de vídeo (perfil ASP da norma MPEG-4 Visual).

Este trabalho tem por objetivo estudar novas ferramentas de codificação de vídeo introduzidas na recente norma H.264/AVC e MPEG-4 Visual, desenvolvendo um modelo que integre a escalabilidade granular presente no MPEG-4 aos avanços na área de codificação presentes no H.264/AVC. Esta estrutura de escalabilidade permite reduzir o custo em termos de eficiência da codificação escalável.

Os resultados apresentados dentro de cada capítulo mostram a eficácia do método proposto bem como idéias para melhorias em trabalhos futuros.

Palavras-Chave: Compressão Escalável, Televisão Digital.

Scalable Compression

ABSTRACT

Video encoding is a problem whose solution should be designed according to the need of intended application. This work presents a method of video compression with scalability that improves the current compression formats.

Scalability represents the extracting capacity of full bitstream, efficient set of bits that are decoded to supply images or decoded videos with a variation according to a given image or video feature. A number of sets that can be extracted from full bitstream defines the supplied scalability granularity, which can be very thin or with thick steps. Most scalable video coding techniques use a base layer which must always be decoded and one or more higher layers which allow improvements in terms of quality (also known as SNR), frame/sampling rate or spatial resolution (for images and video).

The MPEG-4 Fine Granularity Scalable (FGS) video coding scheme is one of the most promising techniques, because it can adapt itself to the features of channels (Internet) or terminals that present an unpredictable or unknown behavior, as maximum speed of access, variations of the bandwidth, channel errors, etc. Although the MPEG-4 FGS standard is a feasible solution for video streaming applications, it shows a significant loss of performance in comparison with non-scalable video coding, in particular the rather efficient Advanced Simple Profile defined in MPEG-4 Visual Standard.

This work aims at studying new tools of video encoding introduced by the recent H.264/AVC norm and Visual MPEG-4, developing a model that integrates the granular scalability present in MPEG-4 to the coding improvements present in H.264/AVC. This new scalability structure allows cost reduction in terms of efficiency of the scalable coding.

The results presented in each chapter show the effectiveness of the proposed method as well as ideas for improvements in future work.

Keywords: Scalable Compression, Digital Television.

1 INTRODUÇÃO

Com a evolução da eletrônica e a explosão da informática, vemos uma sociedade cada vez mais globalizada, clamando por mais conforto e qualidade de vida. Da mesma forma, a evolução das telecomunicações vem unindo povos de todo o mundo, sem distinção de raça, credo ou cor.

Nesse contexto, a necessidade de se comunicar tem buscado esforços para que novas formas de interação sejam disponibilizadas para a sociedade, não somente com a função de integração social, mas também contribuindo para a evolução de outras ciências, como a medicina.

Desde os primeiros *bips* do telégrafo por volta de 1830, passando para sons do rádio e logo em seguida para as imagens em uma televisão, é notável a necessidade da disponibilidade de novas mídias, garantindo a interação cada vez mais real entre as pessoas.

Em meados da década de 90, a banda americana chamada *Severe Time Damage* tornou-se a primeira banda a realizar uma apresentação ao vivo com transmissão de áudio e vídeo (ao vivo) pela internet. Naquela época, a Internet ainda não tinha conhecido as inovações em termos de qualidade e quantidade de os serviços que iriam transformar, anos mais tarde, na maior e mais popular rede de comunicação do mundo.

Atualmente o panorama é bastante diferente, muitas atenções têm sido depositadas na transmissão de imagens mais realísticas e convincentes, além da necessidade de se transmitir informações visuais até mesmo em meios onde o som é o propósito principal, como nos telefones celulares. Logo, um crescente aumento da quantidade de informações necessárias para a transmissão ou armazenamento das imagens tem sido observado, acarretando a necessidade da elaboração de mecanismos mais evoluídos e complexos para processá-las.

O vídeo constitui uma mídia com uma quantidade muito grande de informação, e por isso formas de compactação dessa informação são buscadas incessantemente por pesquisadores. Agregando-se a essa necessidade, são buscados métodos que obtenham bons resultados em relação à fidelidade visual, robustez a erros de transmissão e também baixa complexidade de processamento, já que poderiam ser utilizados em dispositivos portáteis com poder de processamento e consumo de energia reduzidos.

Esta dissertação visa a exploração de métodos para compressão de vídeo, seja explorando tanto métodos já consagrados, bem como novos aprimoramentos de algoritmos e técnicas na área. Diferente de outros desenvolvidos, este trabalho tem foco na codificação avançada de vídeo do H.264 utilizando escalabilidade.

1.1 Video escalável

A codificação de vídeo surgiu para maximar a qualidade do vídeo decodificad a um determinado *bit rate*. Em um sistem clássico de comunicação, codificador comprime o sinal de vídeo com um determinado *bit rate* que é menor ou igual a capacidade do canal e o decodificador reconstrói o sinal utilizando todos os bits recebidos. Este modelo pressupõe que o codificador conhece a capacidade do canal disponível e o decodificador recebe e decodifica todos os bits enviados. No entanto, na prática isto não acontece, pois para muitas aplicações, estes pressoupostos não se verificam, como por exemplo na distribuição de vídeo pré-codificado em que o *bit rate* disponível do canal não é conhecido durante a codificação. Uma das soluções utilizadas na distribuição de vídeo pré-codificado é o modelo de fluxos binários independentes. Neste modelo, a mesma informação é codificada de maneira independente, visando diferentes codificações, em termos de *bit rate*, resolução espacial e temporal. O servidor de vídeo fica então encarregado de distribuir os *bitstreams* codificador em um ou mais canais simultanemente, para um conjunto maior ou menor de receptores. A vantagem deste modelo consiste na sua simplicidade, pois apenas é necessário codificar o conteúdo múltiplas vezes com componentes conhecidos e/ou algoritmos de codificação já existentes. No entanto, este modelo possui uma baixa eficiência uma vez que não explora a dependência entre os diversos *bitstreams* independentes (em grande parte semelhantes uma vez que o conteúdo é o mesmo), repetindo a informação no servidor e, quando necessário, também durante a transmissão. Outra desvantagem deste modelo, consiste no número limitado de representações do vídeo em questão que é necessário conservar, uma vez que sempre que se deseje obter uma representação correspondente a condições que não tenham sido previamente consideradas e não seja disponível no servidor o respectivo conteúdo (com um determinado *bit rate*) é necessário recodificar novamente o conteúdo.

Uma das formas de contornar estes problemas associados a este modelo é através da utilização de uma nova funcionalidade: a escalabilidade na codificação ou seja usando uma representação escalável do vídeo.

A escalabilidade corresponde a capacidade de extrair o *bitstream* total correspondente a um dado conteúdo codificado em certas condições, por exemplo com uma certa qualidade e resolução, denominado neste caso por *bitstream* escalável, subconjuntos eficientes de bits que podem ser utilmente decodificáveis ou seja que oferecem imagens ou vídeos decodificado com uma variação (escala) segundo uma dada característica da imagem ou vídeo (ex: resolução espacial ou qualidade).

Ainda que, um conjunto de *bitstreams* independentes, possa (quase) respeitar a definição de escalabilidade apresentada, considerando como *bitstream* total a soma dos *bitstreams* independentes, na prática a exigência de eficiência faz com que a escalabilidade seja tipicamente alcançada através de um *bitstream* base e camadas incrementais que exploram a redundância em relação a camada base, e eventualmente em relação a outros *bitstreams* incrementais, que permitem oferecer sucessivas imagens ou vídeo decodificados segundo as dimensões de escalabilidade adotadas (resolução temporal ou espacial).

1.2 Contexto e motivação para a codificação escalável

O sistema usado para distribuição de vídeo ou imagens influencia a forma como os esquemas de codificação são projetados. Existem dois modelos principais de distribuição de vídeo: o modelo de transferência completa (*downloading*) e o modelo de transmissão em tempo real (*streaming*). No modelo de transferência completa, o usuário seleciona o vídeo que pretende, espera que este seja transferido para seu terminal na totalidade e, finalmente, utiliza o decodificador adequado para visualizar o conteúdo transferido. Neste modelo, o vídeo consiste em um arquivo binário que pode ser transportado como qualquer arquivo de dado (ex: através de email, ftp, http, etc). No entanto, para arquivos grandes (quando o vídeo possui duração elevada ou elevado *bit rate*), esta solução não é satisfatória, pois o tempo de transferência é elevado e muitas vezes inaceitáveis para usuários com pouco espaço de armazenamento, conexões lentas e paciência limitada. Por outro lado, o modelo de transmissão em tempo real perminte que o usuário visualize o vídeo a medida que este é transferido para o terminal do cliente. Normalmente, a visualização do vídeo começa com um pequeno atraso no início da comunicação e não é necessário espaço de armazenamento para guardar toda a informação recebida. Este modelo e um dos mais populares modelos de distribuição de vídeo na Internet, pois a visualização inicia-se pouco tempo depois do usuário selecionar o conteúdo que pretende visualizar não existindo grande tempo de espera. Além disso, a distribuição de conteúdo ao vivo apenas é possível com o modelo de transmissão em tempo real, pois o vídeo é capturado, codificado, transportado e visualizado enquanto o evento acontece, com um atraso global pequeno.

O contexto desta dissertação explora o modelo de transmissão em tempo real (quer seja para transmissões ao vivo ou não), onde a escalabilidade na codificação de vídeo tem um papel importante, pois a estrutura de hierarquia do *bitstream* permite adaptação dinâmica às características das redes (largura de banda) e dos terminais (capacidade de processamento).

Nos capítulos a seguir são apresentados o estado da arte na área de codificação de vídeo e escalabilidade, bem como as técnicas mais comuns empregadas na codificação para a compressão dos dados de vídeo de maneira temporal e espacial com objetivo de atualizar o leitor nestas técnicas, seguindo a mesma linha, é apresentado ao leitor uma análise das melhores soluções de codificação escalável e uma solução implementada no desenvolver deste trabalho.

2 CODIFICAÇÃO ESCALÁVEL

A compressão é um dos componentes mais importantes de muitos serviços e aplicações multimídia, tais como a distribuição de vídeo na Internet e em redes móveis, acesso a bases de dados multimídia, televisão digital, vídeo, telefonia e videoconferência. As características das redes dos terminais que estas interligam colocam muitos problemas na concepção de um sistema de distribuição de vídeo devido a grande heterogeneidade existente. O vídeo comprimido pode ser transportado em redes com diferentes características em termos de *bit rates,* padrões de erro, atraso e variação temporal destes parâmetros. A codificação de vídeo utiliza técnicas com predição temporal, faz uso de informação passada e/ou futura, e códigos de comprimento variável, o que torna o vídeo comprimido mais vulnerável a erros de transmissão e mais difícil de se adaptar às diferentes características da rede de transporte. Desta forma, a combinação destes dois tipos de técnicas levanta questões importantes:

- Como se pode diminuir o impacto dos erros (de bits ou de perdas de pacotes) que ocorrem durante o transporte do vídeo comprimido?
- De que forma pode-se obter uma representação genérica do vídeo comprimido adequada às características cada vez mais variadas das redes e dos terminais?

Atualmente, muitos estudos têm se desenvolvido na área de codificação escalável de imagem e vídeo. Os principais objetivos no estudo e desenvolvimento de técnicas de codificação escalável são de alcançar uma compressão eficiente (de preferência próxima à da codificação não escalável), elevada flexibilidade em termos de adequação aos meios de transmissão e baixa complexidade. Devido à natureza conflituosa dos requisitos de eficiência, flexibilidade e complexidade, cada técnica de codificação escalável procura um equilíbrio entre estes três fatores. Isto significa que as empresas que pretendem oferecer um serviço de distribuição de vídeo têm de escolher uma solução de codificação de acordo com as características do serviço a oferecer. Por exemplo, no caso da distribuição de vídeo para terminais móveis é necessário ter em conta a complexidade, enquanto que a distribuição de vídeo na Internet este fator não é tão determinante.

2.1 Aplicações

As ferramentas de codificação de fonte são normalmente desenvolvidas com o objetivo de oferecer novas funcionalidades como, por exemplo, capacidades interativas ou mais compressão. Neste contexto, esta seção apresenta um conjunto de aplicações que podem se beneficiar da utilização de ferramentas de codificação escalável do vídeo.

Como é natural, não se pretende aqui descrever todas as aplicações possíveis, mas apenas dar uma idéia das principais áreas em que as técnicas de codificação escalável de vídeo podem ser mais úteis. Para cada aplicação, são mostrados os principais problemas na transmissão de vídeo, a forma como podem ser resolvidos e as funcionalidades requeridas.

2.1.1 Vídeo na Internet

Nos últimos anos, tem se assistido a um enorme crescimento na utilização da Internet. Entretanto, a capacidade da Internet também cresce rapidamente devido a utilização de novas tecnologias e de grandes investimentos em infra-estrutura por parte das empresas de telecomunicações. Outro fenômeno que contribui para a difusão da Internet é o aparecimento da "banda larga", dos mais variados tipos. Deste modo, o espectro de velocidades de acesso a Internet é bastante amplo, pois varia entre modens convencionais (linha discada), modens via cabo, linhas ADSL, etc. Isto significa que os servidores de vídeo existentes na Internet devem servir clientes com *bit rates* muito diferentes, terminais bastante variados e exigências em termos de qualidade de vídeo também muito diferentes. Além desta ampla gama de ligações à Internet, a largura de banda varia quando muitos clientes acessam um material de vídeo num mesmo servidor (pode existir congestionamento de tráfego). Mesmo com estas dificuldades, um vasto conjunto de aplicações multimídia sobre a Internet está surgindo como transmissões de rádio, televisão, noticiários, filmes, ensino a distância, etc. A **figura 2.1** exemplifica este processo para os noticiários via Internet: o primeiro passo consiste na produção do conteúdo multimídia e corresponde a criação da página web, incluindo a codificação e indexação do conteúdo audiovisual; este conteúdo pode ser distribuído diretamente (*live streaming*) ou ser adquirido anteriormente, estando disponível a partir de uma base de dados. Em seguida, o servidor de vídeo e o servidor web distribuem a informação multimídia para cada cliente, resultando numa página web no terminal do cliente, com vários elementos: texto, imagens fixas, vídeo e áudio.

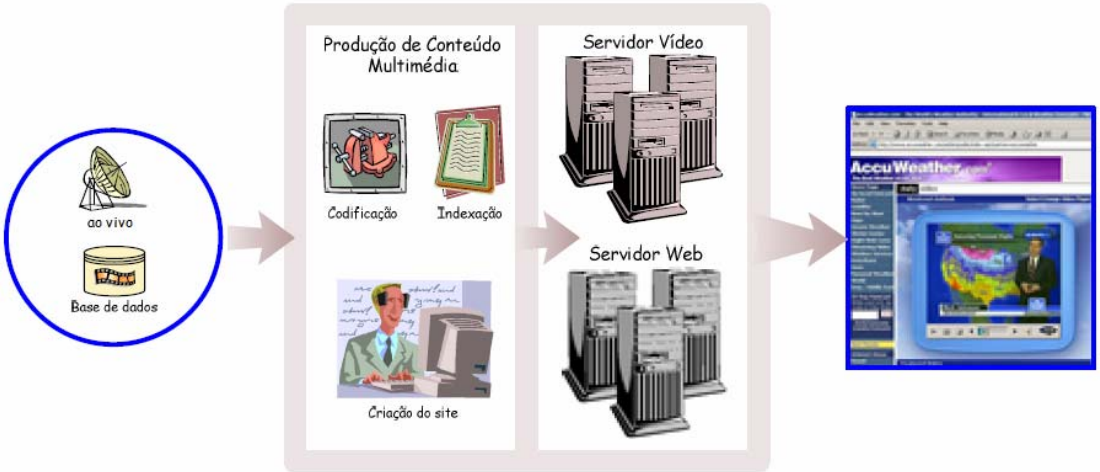


Figura 2.1: Difusão de noticiários na Internet.

Para lidar com esta heterogeneidade de ligações e variações de largura de banda, não é pratico produzir vários *bitstreams* com diferentes características correspondentes ao mesmo conteúdo, uma vez que o servidor teria de possuir vários codificadores de áudio e vídeo operando paralelamente em tempo real ou armazenar diferentes versões codificadas do mesmo conteúdo. Assim sendo, existe a necessidade que o codificador

seja capaz de produzir um *bitstream* escalável de forma que o servidor de vídeo e/ou certos elementos da rede (*routers, gateways*) possam processar facilmente o vídeo codificado, truncando o *bitstream*, para adaptá-lo à capacidade disponível numa determinada ligação sem que haja necessidade de codificar repetidamente o mesmo conteúdo.

2.1.2 Vídeo sobre redes móveis

Nos últimos anos, as comunicações móveis e multimídia tiveram um desenvolvimento muito rápido e um sucesso comercial acentuado. Naturalmente, o sucesso de ambas as áreas vem de uma visão abrangente das telecomunicações: permitir a comunicação de qualquer tipo de dados, em qualquer lugar. A convergência destas duas áreas é agora possível com a introdução das redes móveis de terceira geração que permitem um conjunto amplo de serviços móveis multimídia. De todos os tipos de informação transmitida, o vídeo é o mais exigente em termos de *bit rate*, o que obriga a utilização de algoritmos de compressão que minimizem a largura de banda necessária para o transporte de vídeo. No entanto, sem determinadas precauções, os sinais de vídeo comprimido são extremamente vulneráveis a erros de transmissão. Além disso, as redes móveis não garantem qualidade de serviço (QoS), uma vez que as variações nas características do canal podem introduzir taxas de erros elevadas durante certos períodos de tempo.

Sendo assim, surge a necessidade de se trabalhar com um codificador de vídeo que opere a um *bit rate* baixo (ex: menor que 64kbit/s), e maximize a qualidade de vídeo decodificado na presença de erros e seja capaz de explorar convenientemente a capacidade disponível do canal quando ele varia ao longo do tempo. As principais ferramentas que podem ser usadas por parte do codificador são: a codificação resiliente a erros e a codificação escalável. As técnicas de codificação escalável são utilizadas em ambientes móveis com três objetivos:

- Lidar com a variabilidade da largura de banda.
- Dividir o *bitstream* de vídeo em várias camadas, permitindo uma proteção de erros adequada a importância de cada camada.
- Lidar com a heterogeneidade dos terminais, em termos de *bit rate* e resolução espacial.

Um serviço que anda crescendo muito nos celulares de terceira geração (**figura 2.2**) é a transmissão de vídeo em tempo-real. Este serviço permite que o usuário visualize conteúdo de vídeo sem ser necessário transferir todo o arquivo de vídeo antes de iniciar a visualização.

Figura 2.2: Telefones celulares de terceira geração.

No contexto dos ambientes móveis é importante que a decodificação escalável seja o menos exigente possível em relação à capacidade de processamento e consumo de energia.

2.1.3 Televisão digital

A difusão digital de sinais televisivos é um dos desenvolvimentos mais aguardados na área das telecomunicações há alguns anos. A televisão digital implica uma forma normalizada de codificar os sinais de áudio, vídeo e dados e o seu transporte em vários meios de transmissão (ex: terrestre, cabo, satélite). O lançamento de serviços de televisão digital por todo o mundo já se tornou uma realidade. Muitos projetos de televisão digital – DVB na Europa, DTV nos EUA e ISDB no Japão – utilizam as normas desenvolvidas pelos grupos MPEG para codificação. Apesar de nenhum destes sistemas utilizar técnicas de codificação escalável, eles permitem uma funcionalidade acrescida quando se pretende transmitir simultaneamente um evento em diferentes formatos. Por exemplo, a escalabilidade permite a transmissão de um sinal de vídeo com uma resolução espacial equivalente a televisão analógica, SDTV (*Standard Definition TV*), e alta definição HDTV (*High Definition TV*), de uma forma mais eficiente que a transmissão simultânea de dois sinais de vídeo independentes, um para cada resolução. Deste modo, para um operador de televisão, a codificação escalável de vídeo possui várias vantagens: um maior número de canais simultaneamente difundidos devido ao melhor uso da banda disponível e uma complexidade menor para acomodar serviços deste tipo.

Considerando um ambiente onde existe um número de eventos esportivos, por exemplo, vários jogos de futebol ocorrendo simultaneamente; as estações de TV podem explorar esta tecnologia difundindo todos os jogos, permitindo que os telespectadores possam escolher os jogos que pretendem assistir simultaneamente mesmo que com menor qualidade. Deste modo, podem possuir a opção de ver três ou quatro jogos simultaneamente e/ou comentários dos jogos em diferentes resoluções com diferentes níveis de qualidade (**figura 2.3**).

Figura 2.3: Televisão digital com seleção de objetos de interesse.

Outra área de aplicação televisiva é o ensino a distância, onde os estudantes podem utilizar serviços deste tipo, selecionando, por exemplo, um objeto sobre o qual desejam mais informação e assim visualizar o objeto selecionado com uma melhor resolução e/ou qualidade. A utilização da escalabilidade permite também à operadora de TV alcançar uma audiência mais vasta, através da difusão de eventos selecionados de outros tipos de redes (redes móveis de terceira geração) que possuem requisitos de complexidade, resolução e qualidade diferentes.

2.1.4 Base de dados multimídia

O número de bases de dados que armazenam diversos tipos de informação (ex: áudio, vídeo, texto, imagens, modelos tridimensionais) está aumentando consideravelmente e alguns dos problemas advindos são decorrentes de escalabilidade e de desempenho. Uma base de dados multimídia pode envolver coleções extensas de dados audiovisuais e permite muitas vezes aos seus usuários selecionar o material de interesse em diferentes níveis de resolução e/ou qualidade, o que permite reduzir consideravelmente a quantidade de dados para transferência entre o cliente e o servidor. Por outro lado, estes sistemas também querem oferecer aos usuários a possibilidade de processar ou visualizar o material selecionado com alta resolução e/ou qualidade. Para se poder acessar a base de dados através de um conjunto amplo de terminais e por um maior número de redes é necessário que a base de dados guarde o conteúdo em múltiplas resoluções e níveis de qualidade disponibilizando diferentes versões de acordo com as características dos terminais, redes de acesso e preferências do usuário. A codificação escalável permite que o conteúdo audiovisual seja representado de forma eficiente e ainda possibilita o acesso eficiente para uma vasta audiência de usuários, com características diferentes, a partir de vários tipos de terminais e redes.

Outra funcionalidade muitas vezes desejável é a capacidade de selecionar objetos ou regiões de interesse em uma imagem ou vídeo (visualização de certas regiões e imagens de mapas adquiridas por satélite), o que exige que o servidor envie mais informação apenas da relação de objetos selecionados e/ou de interesse em vez de enviar melhorias de toda a imagem ou vídeo (reduzindo os requisitos de largura de banda). A utilização de técnicas de codificação escalável também permite diminuir os requisitos em termos de capacidade de armazenamento da base de dados, facilitando a manutenção do material multimídia, pois apenas uma versão escalável da imagem ou vídeo é necessária em vez de várias versões com diferentes qualidades e resoluções. Por exemplo, no comércio eletrônico, os usuários podem rapidamente visualizar vários produtos presentes na base de dados em qualidades reduzidas. Quando o cliente encontra alguns produtos interessantes e os seleciona, uma imagem ou vídeo com melhor qualidade e resolução podem então serem visualizados.

2.2 Classificação das técnicas de escalabilidade

Antes de apresentar várias técnicas de escalabilidade é importante classificá-las em categorias. Para fazer uma classificação adequada nos esquemas de codificação escalável de vídeo é essencial distinguir duas dimensões principais de escalabilidade, que são:

- Características da imagem ou do vídeo: Esta dimensão de escalabilidade está relacionada com a capacidade de obter *bitstreams* oferecendo uma variação (escala) segundo uma dada dimensão ou característica da imagem ou vídeo. As características utilizadas têm que permitir obter diferentes representações da mesma seqüência de vídeo ou da mesma imagem; as características mais utilizadas são a resolução temporal, a resolução espacial e a qualidade ou relação sinal ruído (SNR).
- Nível de granularidade: o nível de granularidade da escalabilidade no *bitstream* indica o número de diferentes representações que podem ser obtidas a partir do mesmo *bitstream* codificado, segundo uma ou mais características. O nível de granularidade determina o número de camadas

que se pode obter do *bitstream* escalável de uma forma independente das restantes. Quanto mais fina for a granularidade, mais camadas ou representações podem ser obtidas e mais eficiente pode ser a adaptação as características das redes e terminais.

Deste modo, uma determinada técnica de codificação escalável pode ser sempre analisada segundo duas partes: a primeira é a característica da imagem ou do vídeo, e está relacionada com características do terminal ou ainda as preferências do usuário; a segunda é a granularidade e está relacionada com as características da rede de transmissão (variação da largura de banda) e com o número de representações desejadas.

2.2.1 Características da imagem ou do vídeo

Para caracterizar uma determinada técnica de codificação escalável, é necessário classificá-la de acordo com a dimensão das características da imagem ou vídeo usado para oferecer uma escala de representações. Nesta seção é apresentada a classificação das técnicas de codificação escalável em relação às características da imagem ou vídeo utilizadas para extrair as múltiplas representações do *bitstream*; identificam-se cinco características básicas: resolução espacial, resolução temporal, qualidade ou SNR, conteúdo e uma característica que resulta da estrutura da sintaxe do *bitstream*: a separação de dados. Quando se combinam diversos tipos de características, pode-se alcançar um número superior de representações úteis, o que resulta num novo tipo de escalabilidade: a escalabilidade combinada ou híbrida. As características usadas dão forte influência no nível de granularidade que é possível se obter a partir do *bitstream*, se a característica utilizada for apenas a resolução temporal tornando-se difícil obter uma escalabilidade de granularidade muito elevada.

2.2.1.1 Escalabilidade Espacial

A escalabilidade espacial permite a extração de *bitstreams* com diferentes resoluções espaciais a partir do *bitstream* completo. Esta técnica permite oferecer conteúdos a terminais com diferentes características em termos de resolução espacial a partir do mesmo *bitstream*. A **figura 2.5** ilustra este tipo de escalabilidade com 3 camadas escaláveis.

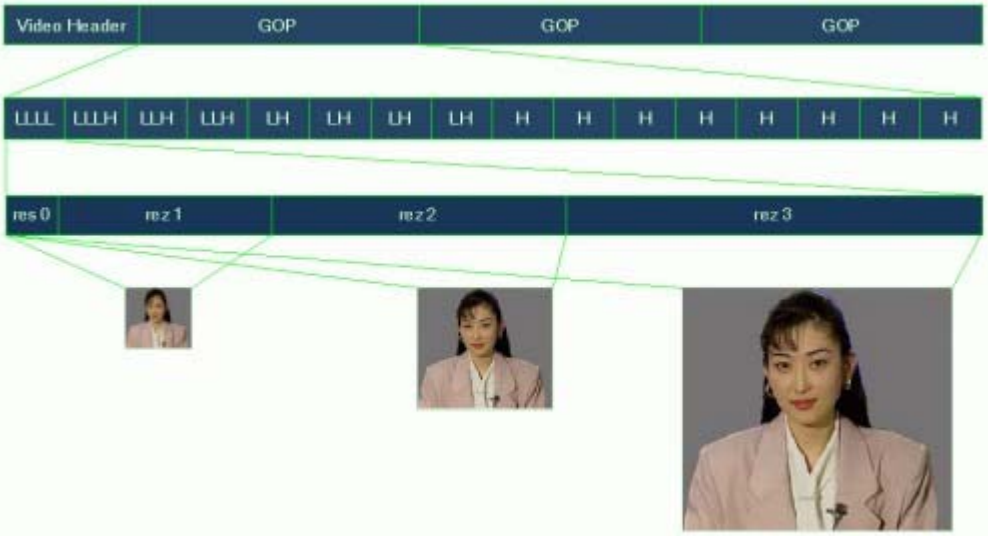


Figura 2.5: 3 camadas de escalabilidade espacial

Na **figura 2.5**, o *bitstream* encontra-se dividido em 3 camadas de escalabilidade espacial. Com a decodificação da primeira camada, o usuário obtém uma versão da imagem ou vídeo original com a menor resolução espacial possível. A decodificação da segunda camada permite obter mais informação que, adicionada à primeira camada, resulta numa imagem reconstruída com uma maior resolução espacial. À medida que se decodificam mais camadas, a imagem reconstruída possui uma resolução espacial cada vez maior, até o nível de resolução espacial máxima. Quando se decodificam mais camadas adicionais, o usuário obtém aumentos sucessivos da resolução espacial da imagem até a resolução original ou resolução máxima estabelecida na fase de codificação. A escalabilidade espacial pode ser obtida através de técnicas de codificação piramidal ou de multi-resolução (BURT, 1983).

2.2.1.2 Escalabilidade temporal

A escalabilidade temporal permite a extração de *bitstreams* correspondentes a diferentes resoluções temporais a partir do *bitstream* completo. A decodificação da primeira camada resulta numa versão do vídeo com baixa resolução temporal e a decodificação progressiva das restantes camadas permite um aumento gradual da resolução temporal, como exemplificado na **figura 2.6**.

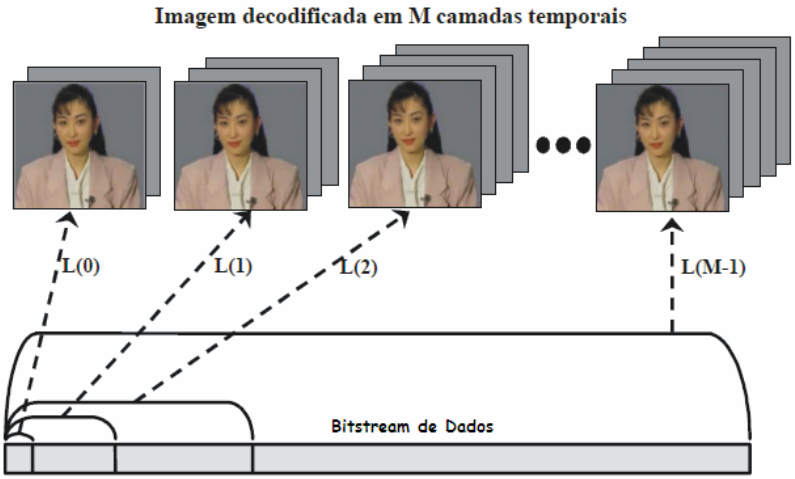


Figura 2.6: M camadas de escalabilidade temporal.

Uma técnica de codificação escalável deve permitir uma qualidade visual excelente na resolução temporal máxima, mas também manter uma qualidade visual aceitável nas resoluções temporais mais baixas. A possibilidade de extrair de um *bitstream* de vídeo diversas resoluções temporais é determinada pelo uso das técnicas de codificação usadas para explorar a correlação entre os quadros (compensação de movimento) e a codificação dos vetores de movimento (CONKLIN, 2000).

2.2.1.3 Escalabilidade SNR

A escalabilidade SNR (*Signal to Noise Ratio*) permite a extração de *bitstreams* correspondentes a diferentes níveis de qualidade a partir do *bitstream* completo. Este tipo de escalabilidade também chamada escalabilidade de qualidade, uma vez que o erro de decodificação está relacionado com a qualidade perceptual da imagem. Neste caso, há um aumento da qualidade da imagem ou vídeo sem variação das resoluções espacial e temporal. Este tipo de escalabilidade codifica sucessivamente o erro de codificação

entre a imagem original e a sua reconstrução numa dada camada. A **figura 2.7a** apresenta um *bitstream* com N camadas de escalabilidade SNR.

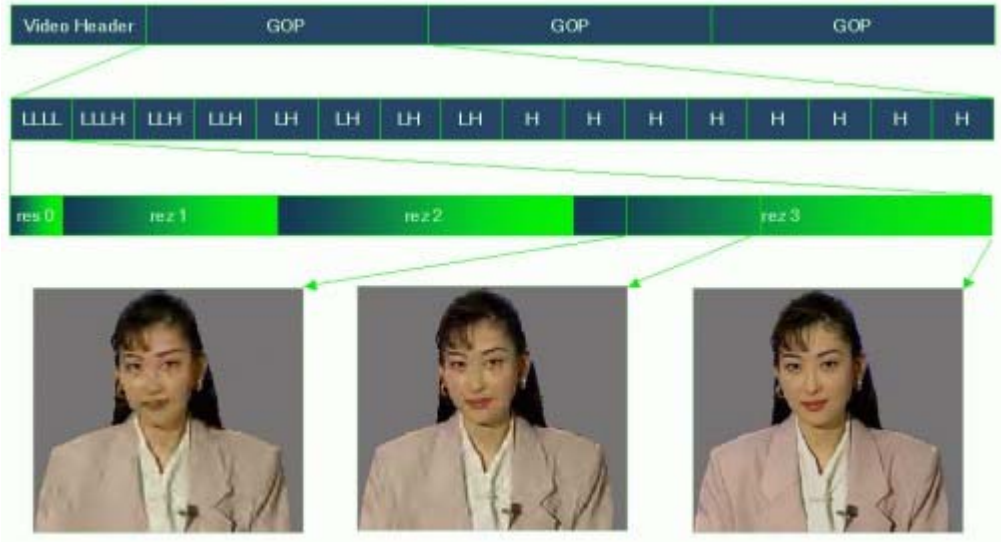


Figura 2.7: 3 camadas de escalabilidade de qualidade ao nível de imagem.

A decodificação da primeira camada fornece uma versão de baixa qualidade do vídeo em questão. A sucessiva decodificação das restantes camadas resulta num aumento da qualidade até ao nível de qualidade máxima. Um caso especial da escalabilidade de SNR é a sua aplicação a regiões ou objetos que fazem parte de uma seqüência de vídeo ou imagem; sendo este tipo de escalabilidade SNR suportado pela norma JPEG2000 através da definição de regiões de interesse e pela norma MPEG-4 Visual (perfil FGS).

2.2.1.4 Escalabilidade de conteúdo

A escalabilidade de conteúdo ou de objeto permite a extração de *bitstreams* correspondentes a cenas com diferentes números de objetos a partir do *bitstream* escalável completo correspondente a cena com todos os objetos. Como é natural, decodificam-se os vários objetos na cena segundo a ordem decrescente da sua importância de modo que, para certa quantidade de recursos, sejam sempre decodificados primeiro os objetos mais relevantes. A **figura 2.8** exemplifica este tipo de escalabilidade para uma cena de vídeo composta por cinco objetos: 2 apresentadores, o fundo, uma logomarca e uma tela com uma bailarina dançando; os vários objetos podem sobrepor-se ou não na cena final conforme a forma como foi gerada. A primeira camada contém um objeto com mais importância (neste caso o apresentador) e a medida que o número de camadas aumenta, são adicionados a cena outros objetos de importância decrescente.

a) b) c) d)

Figura 2.8: Escalabilidade de conteúdo – a) 1 apresentador; b) 2 apresentadores; c) apresentadores, logo e fundo; d) apresentadores, logo e fundo com uma bailarina.

Até o momento, a única norma de codificação que suporta este tipo de escalabilidade é a norma MPEG-4 Visual, através da definição do conceito de objeto de vídeo VO (*Vídeo Object*). Em uma cena de vídeo com vários objetos, um VO pode ser decodificado de uma forma independente dos restantes, permitindo assim que o cliente selecione quais os objetos que pretende decodificar, de acordo com seus recursos. A escalabilidade de conteúdo pode ser combinada com outros tipos de escalabilidade, permitindo que os próprios objetos sejam individualmente escaláveis e assim obter um maior número de representações úteis, a partir do *bitstream* de um único objeto. Por exemplo, a norma MPEG-4 Visual permite que cada objeto seja escalável segundo a dimensão espacial (ou temporal), através da combinação de técnicas de escalabilidade para a textura e forma dos objetos de vídeo.

2.2.1.5 Separação de dados

A escalabilidade por separação de dados permite a extração de *bitstreams* correspondentes a diferentes conjuntos sintáticos de dados a partir do *bitstream* escalável completo. Neste tipo de escalabilidade, o *bitstream* de dados é dividido em duas ou mais camadas, também referidas como partições. A separação de dados, apesar de não ser muitas vezes considerada uma forma de codificação escalável de vídeo, permite que os dados de vídeo codificados sejam divididos em duas ou mais classes de forma útil em termos de decodificação: por exemplo, dados essenciais e dados adicionais. Na separação de dados, a distinção entre estes dois tipos de dados é flexível dependendo do *bit rate* desejado para cada classe. Por exemplo, o *bitstream* pode ser dividido em duas camadas: a primeira contendo informação sintaticamente mais importante, tais como vetores de movimento e coeficientes de baixa freqüência da DCT, e a segunda contendo os coeficientes DCT de alta freqüência. A separação de dados pode ser implementada com uma complexidade inferior quando comparada com outros esquemas de codificação por exigir uma única passagem de codificação.

2.2.1.6 Combinação de tipos de escalabilidade

Os codificadores escaláveis de vídeo podem combinar os cinco tipos de escalabilidade apresentados para formar qualquer tipo de escalabilidade composta. Este tipo de escalabilidade é referido na literatura como escalabilidade combinada ou híbrida e pode combinar dois ou mais tipos de escalabilidade previamente apresentados. Os tipos de escalabilidade híbrida mais utilizados são a escalabilidade espacial/SNR para imagens fixas e a escalabilidade espacial/temporal ou SNR/temporal para seqüências de vídeo.

A principal vantagem da combinação de técnicas de codificação escalável básicas consiste na criação de um *bitstream* com uma granularidade mais elevada. Também permite uma maior flexibilidade por parte do decodificador, pois este pode utilizar mais do que uma característica para representar uma imagem ou vídeo. A utilização de mais de uma dimensão de escalabilidade para representar uma imagem ou vídeo é obtida à custa de um acréscimo de complexidade no codificador e decodificador. A escolha da combinação de escalabilidade depende da aplicação em questão e dos seus requisitos. No transporte de vídeo na Internet, a escalabilidade SNR/temporal é a mais utilizada, uma vez que a escalabilidade espacial possui uma complexidade maior (na norma

MPEG-2 Vídeo) e possui um impacto subjetivo maior quando a resolução espacial varia (de CIF para QCIF) durante a visualização.

2.2.2 Nível de granularidade

O nível de granularidade de um *bitstream* escalável é uma importante medida na classificação das técnicas de escalabilidade, pois indica o número de representações úteis que podem ser extraídas do mesmo *bitstream*. Nesta subseção, é apresentada a classificação das técnicas de codificação escalável segundo a dimensão da granularidade, tendo sido identificadas três categorias: escalabilidade de baixa granularidade, com um número limitado de níveis de granularidade, escalabilidade de elevada granularidade ou contínua no *bit rate*, com um número elevado de níveis de granularidade e a escalabilidade híbrida que combina a escalabilidade de baixa granularidade com a escalabilidade contínua.

2.2.2.1 Escalabilidade de baixa granularidade

Nesta categoria, o *bitstream* pode ser decodificado segundo um conjunto não muito elevado de *bit rates* estabelecidos durante a codificação. O esquema de codificação comprime a seqüência de vídeo em várias camadas sendo uma dessas camadas a camada base, que pode ser decodificada independentemente e fornece a qualidade visual mínima. As outras camadas são camadas de melhoria e, apesar de poderem ser decodificadas de forma independente, só podem ser úteis para melhorar a qualidade ou resolução da imagem quando todas as camadas hierarquicamente inferiores forem também decodificadas. O *bitstream* combinado de todas as camadas fornece a qualidade mais alta, a decodificação da camada base ou qualquer subconjunto de camadas tem sempre qualidade visual inferior do *bitstream* total. Esta técnica permite que a camada base seja codificada com técnicas de codificação de canal mais robustas ou numa rede que suporte serviços diferenciados, a camada base pode ser transportada com uma prioridade mais elevada. Os modos de escalabilidade oferecidos pelas normas MPEG-2 Vídeo e MPEG-4 Visual se encaixam nesta categoria.

2.2.2.2 Escalabilidade de elevada granularidade

Ao contrário da categoria anterior, o *bitstream* pode ser decodificado a qualquer *bit rate* ou seja com elevada granularidade. Esta técnica é muito flexível e permite ao servidor de vídeo adaptar o *bit rate* do vídeo em distribuição às disponibilidades da rede com uma granularidade muito fina, ou seja, com grande eficiência (todos os bits recebidos se tornam úteis). Para que a escalabilidade seja contínua no *bit rate* é necessário que todos os dados comprimidos sejam embutidos num único *bitstream* e possam ser decodificados em diferentes *bit rates* com uma variação muito pequena entre eles. O decodificador recebe os dados comprimidos desde o princípio do *bitstream* até o ponto onde o *bit rate* escolhido seja alcançado. As imagens codificadas com este tipo de escalabilidade podem ser decodificadas progressivamente; o decodificador apenas precisa receber um conjunto muito pequeno de dados para começar a visualizar a imagem. Na compressão de uma única imagem, os bits que possuem a informação mais importante são incluídos no início do *bitstream*, de forma que a qualidade visual seja maximizada para todos os *bit rates*. As normas JPEG, JPEG2000 e o modo VTC (*Visual Texture Coding*) da norma MPEG-4 Visual se encaixam nesta categoria.

2.2.2.3 Escalabilidade híbrida

Este tipo de escalabilidade combina a escalabilidade de baixa granularidade com a escalabilidade contínua. Este esquema de codificação comprime uma seqüência de vídeo em duas camadas: uma camada base e uma camada superior com escalabilidade contínua. No entanto, enquanto na escalabilidade de baixa granularidade a camada superior tem de ser totalmente recebida e decodificada (em conjunto com a camada base) para haver melhoria de qualidade, na escalabilidade híbrida a camada superior pode ser truncada em qualquer ponto. A melhoria de qualidade é proporcional ao número de bits que o decodificador utiliza da camada superior. O codificador apenas necessita conhecer o intervalo de *bit rates* [R_{min}, R_{max}] que irá ser utilizado para consumir o conteúdo. Este tipo de escalabilidade é híbrida uma vez que possui uma camada base independente codificada com R_{min} (sem continuidade de escalabilidade) e uma camada superior com escalabilidade contínua, alcançando um *bit rate* total R_{max}. A escalabilidade fina do MPEG-4 Visual (FGS) (ISO/IEC 14496-2, 2002) encaixa nesta categoria.

2.3 Técnicas de codificação escalável

Para compensar a falta de previsibilidade das características dos terminais e das redes de transmissão, muitas soluções foram propostas. Os esforços para melhorar a qualidade do vídeo decodificado nestas condições podem ser classificados em duas categorias principais:

- Protocolos de transporte: Desenvolvimento de novos protocolos e arquiteturas adequadas ao transporte de vídeo.
- Codificação de fonte: Desenvolvimento de mecanismos de codificação de vídeo que permitam que o conteúdo seja facilmente adaptável às características da rede e dos terminais.

Em relação à primeira categoria, o trabalho desenvolvido concentra-se na criação de um ambiente mais adequando possível ao transporte de vídeo. Exemplos de tecnologias nesta categoria são: serviços diferenciados e integrados, gestão de tráfego, controle de congestionamento, códigos corretores de erros e *interleaving*. As técnicas desenvolvidas na área de transporte de vídeo nas redes de comunicação são importantes para o desempenho de um sistema completo de distribuição de vídeo. Outro componente igualmente importante constitui-se nos algoritmos de codificação do próprio sinal de vídeo ou de imagem. Nesta subseção, são apresentadas técnicas de codificação fonte mais relevantes no contexto desta dissertação, ou seja, as técnicas de codificação escalável de vídeo. Estas técnicas despertam bastante o interesse por parte da comunidade científica e por parte da indústria com vários sistemas de distribuição de vídeo já construídos a partir delas. Além disso, novos estudos, soluções e melhorias às técnicas já existentes são propostas todos os anos, com o principal objetivo de melhorar a eficiência de codificação e flexibilidade de adaptação.

Deste modo, selecionam-se quatro tecnologias de escalabilidade consideradas fundamentais, apresentadas a seguir de forma independente apesar de possuírem algumas semelhanças conceituais entre si. A codificação piramidal será a primeira técnica a ser apresentada e consiste na representação de imagens com vários níveis de resolução espacial / temporal, através de filtros de sub-amostragem e sobre-amostragem; esta técnica é usada em algumas das normas de codificação de imagem e

vídeo (JPEG, MPEG-2 Vídeo). A segunda técnica, codificação em sub-bandas ou codificação *wavelet*, é uma das técnicas mais promissoras na área da codificação escalável de imagens e vídeo. Este tipo de transformadas gera uma representação em camadas da imagem ou vídeo com uma elevada granularidade. A terceira técnica apresentada é baseada na transformada discreta do coseno (DCT – *Discrete Cosine Transform*). Devido à popularidade desta transformada na área de codificação de imagem e vídeo, vários esquemas escaláveis foram propostos e serão aqui descritos. A quarta técnica pode ser usada apenas para vídeo e baseia-se na aplicação de *matching pursuits* às imagens residuais do vídeo. Este esquema permite uma escalabilidade fina e baseia-se na codificação das características mais importantes da imagem (com maior energia) antes das de menor importância.

2.3.1 Codificação piramidal

Através da utilização de técnicas de sub-amostragem e sobre-amostragem é possível construir uma representação piramidal da imagem baseada na resolução espacial. Neste contexto, o termo "piramidal" refere-se a uma estrutura de dados que permite obter um acréscimo de informação à medida que se obtém as diferentes camadas que formam esta estrutura de dados. A imagem no topo da pirâmide tem a resolução espacial mínima. Os níveis restantes da pirâmide permitem reconstruir imagens piramidais com resolução superior; para os níveis mais próximos da base da pirâmide (informação máxima) o acréscimo de informação é inferior. Um dos primeiros trabalhos na área da codificação escalável de imagens fixas utiliza uma estrutura piramidal e foi desenvolvido por Burt (1983); a arquitetura básica adotada é ilustrada na **figura 2.9**. A imagem de entrada é processada por um filtro passa-baixo H(z) e sub-amostrada (2↓) de forma a obter uma representação mais "grosseira" da imagem original, no caso com uma resolução espacial inferior. A imagem sub-amostrada é quantificada (Q_b) e, como contém menos informação que a imagem original, pode ser codificada com um menor número de bits, dando origem a camada base. A imagem da camada base é utilizada como predição para gerar o *bitstream* da camada superior, pois é sobre-amostrada (2↑), filtrada (filtro passa-alto G(z)) e subtraída da imagem original, obtendo-se um erro de predição. Uma vez que as imagens naturais tem uma tendência de concentrar sua energia nas freqüências espaciais mais baixas, este erro de predição (que representa as freqüências mais altas) tem uma energia inferior, o que permite uma maior eficiência de compressão. O resultado é uma hierarquia de dois níveis constituída por uma representação básica da imagem e um erro de predição quantificado (Qs) que permite melhorar a qualidade.

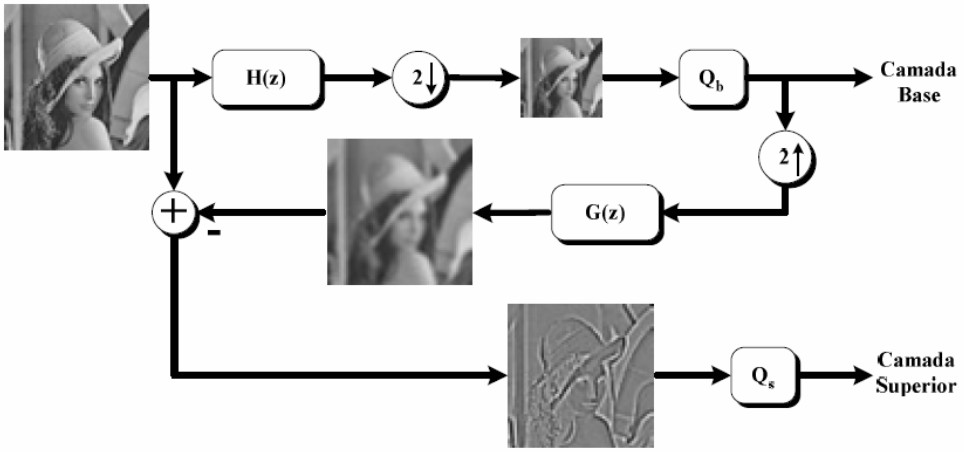


Figura 2.9: Estrutura piramidal de Burt e Adelson com 2 níveis.

Burt e Adelson ampliaram esta arquitetura para um número arbitrário de níveis, através da aplicação recursiva desta decomposição. A **figura 2.10** apresenta o esquema genérico de codificação desenvolvido para o caso específico de quatro *bitstreams*. O operador R representa as operações de filtragem passa-baixo e sub-amostragem e o operador E o filtro passa-alto (complementar do filtro passa-baixo) e a sobre-amostragem.

Neste sistema de codificação piramidal dois tipos distintos de pirâmides são gerados: a pirâmide gaussiana e a pirâmide laplaciana, tal como ilustrado na **figura 2.10**. A pirâmide gaussiana é gerada através da filtragem recursiva e sub-amostragem da imagem original e é referida com este nome porque o filtro utilizado em Burt (1983) tem uma forma aproximadamente gaussiana. A pirâmide laplaciana corresponde a um conjunto de imagens diferença (exceto a imagem do topo), ou seja, os erros de predição que permitem a reconstrução perfeita da imagem original. A pirâmide laplaciana é referida com este nome porque a operação de diferença entre duas imagens gaussianas é muito semelhante aos operadores de Laplace utilizados no processamento de imagens (JAIN, 1989).

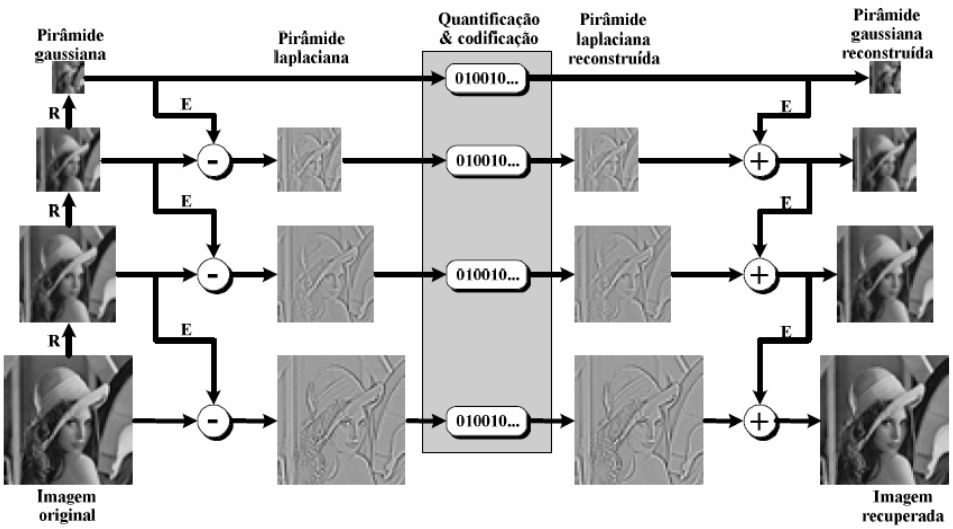


Figura 2.10: Codificação e decodificação escalável com a técnica piramidal.

Na codificação com perdas, o conjunto das imagens da pirâmide laplaciana e a imagem do topo são quantificadas e codificadas. O processo de decodificação tem de começar pela imagem de topo e deve seguir uma abordagem do topo para a base. Em cada nível da pirâmide, a imagem anteriormente decodificada serve como preditor e é somada a correspondente imagem laplaciana depois de filtrada e sobre-amostrada.

A pirâmide gaussiana pode ser entendida como um conjunto de representações em freqüência da imagem original. O primeiro nível da pirâmide laplaciana (imagem com a resolução máxima) é uma representação passa-alta da imagem original, os níveis intermediários correspondem a representações passa-banda e a imagem no topo da pirâmide é uma versão passa-baixa. Este processo de dividir uma imagem em bandas de freqüência é também referido como decomposição em sub-bandas, o que significa que a codificação piramidal pode ser encarada como uma variante da codificação em sub-bandas apresentada a seguir. Na codificação de cada imagem da pirâmide laplaciana (que representa uma banda de freqüências), a sensibilidade do sistema visual humano pode ser explorada para reduzir o *bit rate* total. É bastante conhecido que a sensibilidade do observador é maior para as freqüências espaciais mais baixas; logo mais bits devem

ser usados para codificar este tipo de informação. Uma escolha adequada dos quantificadores permite reduzir o *bit rate* total sem que o observador perceba a degradação da imagem.

Uma das principais desvantagens da estrutura piramidal é o aumento do número de amostras utilizadas para predição; as decomposições piramidais possuem muita informação redundante. Mais concretamente, uma imagem de 512x512 *pixels* é decomposta em uma imagem de 256x256 *pixels* mais uma imagem de 512x512 *pixels* de refinamento, o que obriga a codificar mais 25% dos dados em comparação com a codificação não escalável. No limite, o número total de amostras a ser codificado é no máximo:

$$1^2 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2^2}\right)^2 + \ldots + \left(\frac{1}{2^\infty}\right)^2 = \frac{4}{3}$$

Existe um aumento de 33% em termos de amostras o que é significativo. Em esquemas de codificação de imagem, um aumento de 33% no número de amostras a codificar resulta em uma degradação apreciável na qualidade da imagem. Esta desvantagem pode ser superada se for utilizado um esquema de codificação entrópica eficiente; no entanto, uma solução aceitável deste tipo ainda não foi encontrada.

Este esquema de codificação serviu de ponto de partida para muitos algoritmos de codificação escalável de vídeo e imagem. Chaddha (1995) desenvolveu um esquema baseado numa pirâmide laplaciana com três níveis, quantificação vetorial (estruturada em árvore) de cada nível de codificação entrópica. Medidas de distorção baseadas na DCT e no erro quadrático médio foram incorporadas na medição da distorção da quantificação vetorial. O algoritmo de codificação foi integrado em um sistema completo de distribuição de vídeo oferecendo serviços como o acesso à base de dados e ensino a distância.

Em Illngner (1997) é apresentado um sistema de codificação escalável espacial de vídeo através da extensão da pirâmide laplaciana para uma seqüência de vídeo. A estimação e compensação de movimento são realizadas entre quadros, no mesmo nível da pirâmide, obtendo-se assim um erro de predição que é codificado numa estrutura piramidal. Em Girod (1995) é apresentado um esquema de codificação escalável baseado na decomposição piramidal do vídeo. A principal inovação deste esquema é a introdução de uma pirâmide espaço-temporal combinada com um esquema de quantificação vetorial. Um esquema de compensação de movimento permite explorar a redundância no domínio temporal. Em Girod (1995) e Horn (1996) são estudadas técnicas de compensação de movimento em diferentes níveis da pirâmide (escalas) e técnicas de estimação de movimento hierárquicas de forma a obter uma melhor eficiência de codificação em todos os *bit rates*. A **figura 2.11** mostra a pirâmide utilizada: as camadas 1 e 2 correspondem aos formatos SIF e QSIF, respectivamente, a camada 0 fornece a resolução espacial máxima de acordo com a norma ITU-R 601 (versão americana) e a camada 3 oferece a menor resolução espacial possível, para permitir o transporte do vídeo em redes de baixo *bit rate*. Para permitir uma transmissão robusta em canais com elevadas taxas de erro (ex: difusão de TV digital terrestre), o codificador de fonte é combinado com um esquema de proteção de erros que utiliza códigos com taxas de codificação adequadas a importância de cada camada (HORN, 1996).

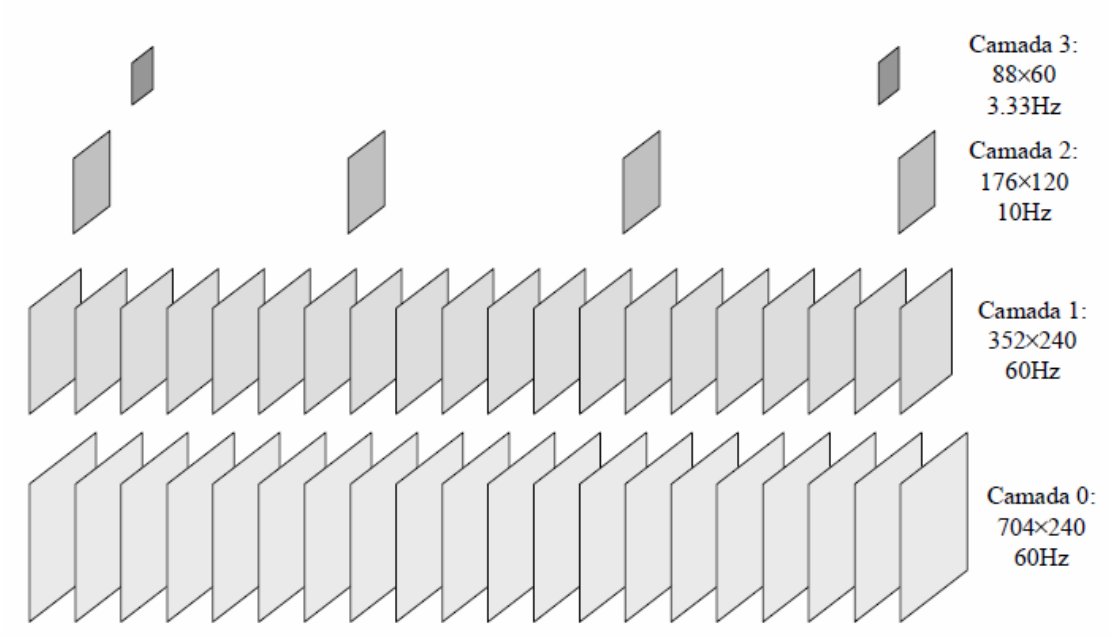


Figura 2.11: Exemplo de pirâmide espaço-temporal

Este tipo de codificador foi também integrado em um servidor de vídeo (HORN, 1997) que permite a transmissão de vídeo na Internet através do protocolo RTP para controle da sessão e transmissão de dados. Em conclusão, a codificação piramidal serviu de ponto de partida para o desenvolvimento de muitos algoritmos utilizados hoje em dia (ex: modo hierárquico incluído na norma JPEG e para a escalabilidade espacial e espaço-temporal incluídas nas normas MPEG-2 Vídeo, H.263+ e MPEG-4 Visual.

2.3.2 Codificação em sub-bandas

A codificação em sub-bandas ou codificação *wavelet* foi introduzida por Crochiere (1976) para codificar áudio, e têm sido amplamente utilizada na codificação de imagens, vídeo e áudio. Esta técnica simples, mas bastante poderosa, é baseada na decomposição do sinal utilizando pares de filtros que permitem uma reconstrução perfeita (ou quase perfeita) do sinal. A operação mais importante num sistema de codificação em sub-bandas é ilustrado na **figura 2.12**: o sinal origem é dividido em dois, cada um com metade das amostras, através de filtragem e sub-amostragem.

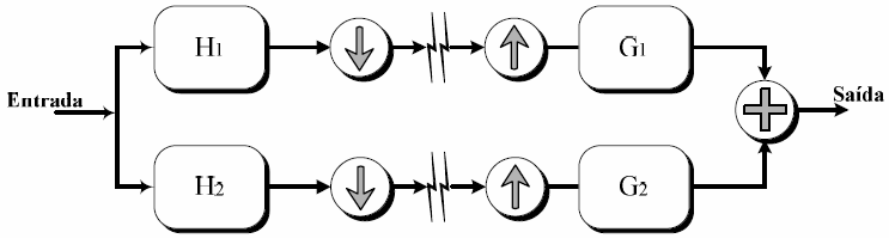


Figura 2.12: Decomposição do sinal em 2 sub-bandas.

Tal como na codificação piramidal, o sinal é decomposto em duas camadas, uma que permite uma representação mais grosseira do sinal e outra que combinada com a primeira permite a recuperação completa do sinal original. Neste caso, a decomposição é realizada através de dois filtros – um filtro passa-baixo $H_1(z)$ e um filtro passa-alto $H_2(z)$ – usualmente referidos como filtros de análise. Os sinais resultantes são sub-

amostrados, quantificados e transmitidos independentemente. Para reconstruir o sinal original, os sinais resultantes são sobre-amostrados, filtrados por um par de filtros de síntese $G_1(z)$ e $G_2(z)$ e, finalmente, adicionados.

Este é o princípio básico de dividir espectralmente o sinal em duas ou mais bandas de freqüência e é adequado para codificação de imagem. Primeiro, as imagens tem tendência a concentrar a sua energia nas freqüências mais baixas. Segundo, o sistema visual humano possui uma sensibilidade menor às freqüências mais altas, o que permite ajustar a distorção da imagem de acordo com critérios perceptuais. Por último, as imagens são processadas de forma diferente de outros codificadores que estruturam a imagem em blocos adjacentes (codificadores baseados na DCT). A principal vantagem é a eliminação do efeito de bloco dos artefatos especiais que ocorrem nas fronteiras entre os blocos codificados. Na codificação em sub-bandas, os filtros de análise e síntese são desenhados de forma que a sua resposta em freqüência não seja sobreposta (*nonoverlapping criteria*); como conseqüência, as sub-bandas resultantes não estão correlacionadas (um teorema já provado diz que processos aleatórios cuja banda de freqüências não sejam sobrepostas, não estão correlacionados (TREES, 1968) o que satisfaz um dos principais objetivos de uma transformada em um sistema de codificação de vídeo ou imagem. Os filtros são organizados em bandas com oitavas, tal como é ilustrado na **figura 2.13**; esta organização corresponde a dividir o espectro completo (W) por dois, voltar a dividir a banda de freqüência mais baixa por 2 e assim sucessivamente. Com esta organização é possível uma adaptação mais adequada às características do sistema visual humano, uma vez que as freqüências mais baixas da imagem (onde existe uma maior energia) ficam mais descorrelacionadas e podem ser quantificadas de uma forma independente das outras sub-bandas.

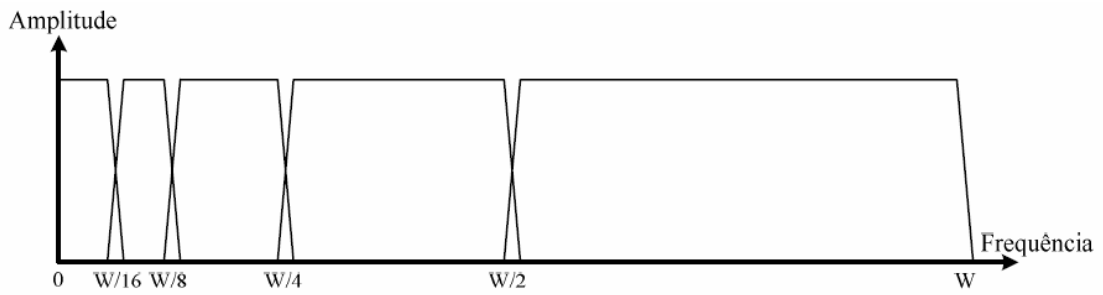


Figura 2.13: Banco de filtros passa-banda.

Na ausência de erros de quantificação, a imagem reconstruída deve ser uma cópia perfeita da imagem de entrada. Para permitir esta característica, é necessário que os filtros de análise ocupem todo o espectro sem se sobreporem na freqüência, o que implica regiões de transição infinitamente abruptas que não podem ser realizadas na prática. Como alternativa, os filtros de análise tem que possuir regiões de transição finitas que se sobrepõem na freqüência, tal como ilustrado na **figura 2.13**, o que significa que distorções podem ocorrer, devido ao efeito de *aliasing*. Para cancelar este efeito, é necessário a utilização de uma certa classe de filtros, denominados de filtros ortogonais que permitem a eliminação das componentes de distorção.

Em Vetterli, (1984) sugeriu que a transformada DWT (*Discrete Wavelet Transform*) fosse aplicada a codificação de imagens, através da filtragem sucessiva nas direções horizontais e verticais. Um exemplo deste tipo de codificação é ilustrado na **figura 2.14**.

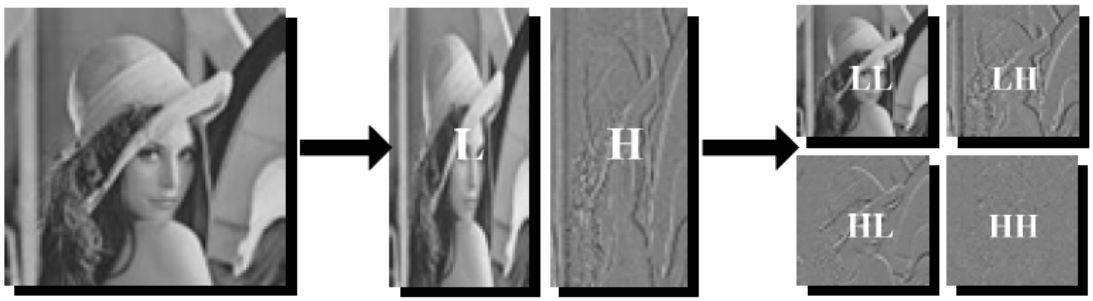


Figura 2.14: Decomposição bidimensional da freqüência de uma imagem.

Em primeiro lugar, aplicam-se os filtros horizontais, que resultam duas imagens distintas, referidas com a sub-banda L e a sub-banda H; em seguida, são aplicados os filtros verticais a cada uma delas, no que resulta uma imagem LL e três bandas LH, HL e HH que, quando somadas, permitem a reconstrução da imagem original. Devido às operações de sub-amostragem, o número total de amostras não é alterado.

Como se pode reparar, a estrutura em camadas está implícita no esquema de codificação, podendo cada banda ser codificada e transmitida independentemente; a escalabilidade é inerente ao próprio esquema de codificação. Esquemas de codificação bidimensionais com múltiplas bandas podem ser desenvolvidos a partir da estrutura apresentada na **figura 2.12**, ou seja, através da aplicação recursiva da decomposição em duas bandas. Uma transformada DWT de sete bandas é apresentada na **figura 2.15**, onde a divisão de bandas é aplicada alternadamente nas direções horizontal e vertical.

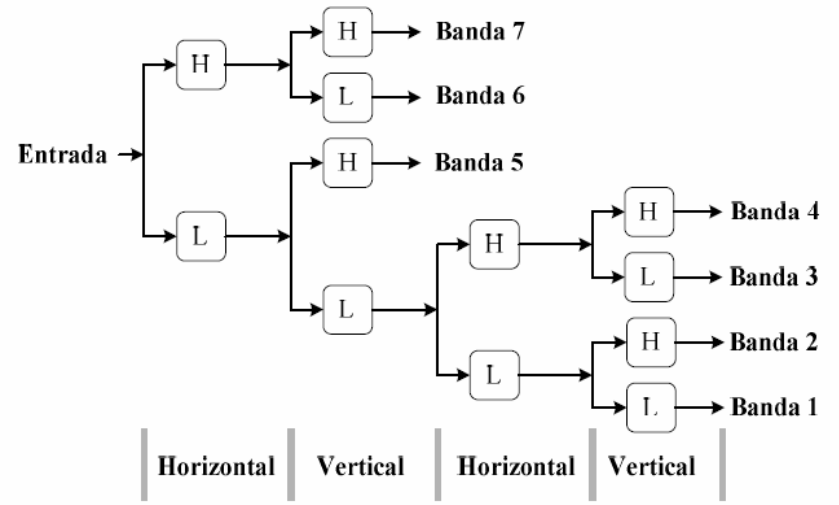


Figura 2.15: Decomposição DWT multi-banda.

A hierarquia que resulta da aplicação de uma transformada deste tipo é apresentada na **figura 2.16**. No lado esquerdo apresentam-se as imagens correspondentes às sete bandas, devidamente amplificadas para permitir a visualização dos detalhes que contém. A direita, apresenta-se a disposição das sete bandas com a respectiva correspondência em relação à decomposição apresentada na **figura 2.15**

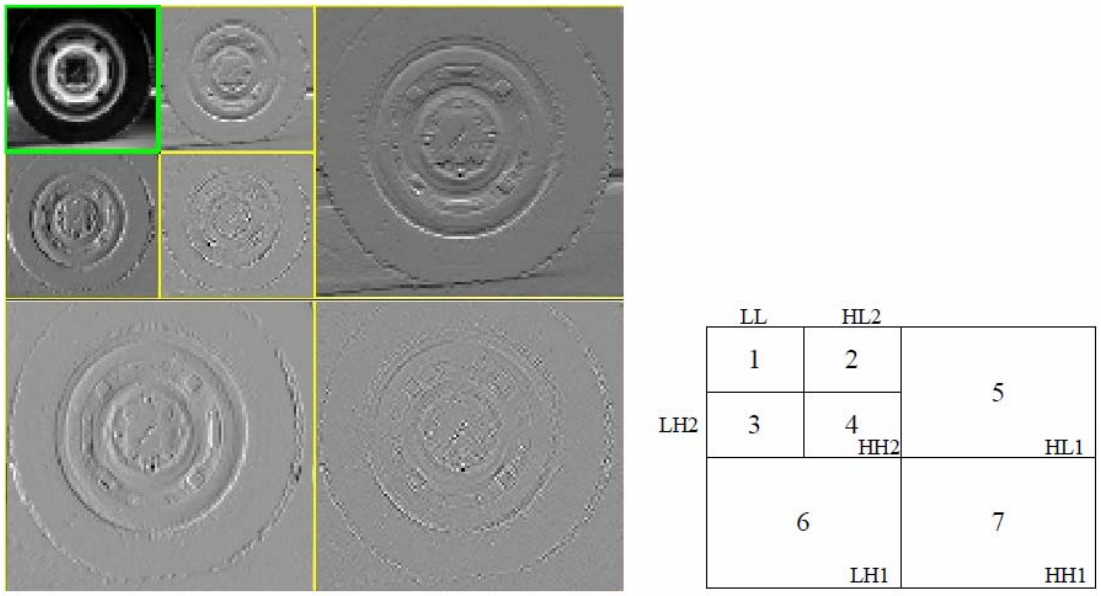


Figura 2.16: Sete bandas geradas pelo codificador da figura 2.16.

As vantagens desta transformada em relação aos codificadores baseados na DCT são a ausência do efeito de bloco, o bom desempenho em termos de eficiência, especialmente para imagens fixas, e a geração de uma representação escalável. Por estes motivos, esta técnica foi adotada pelas normas MPEG-4 Visual (para imagens fixas) e JPEG2000. Várias propostas interessantes para a codificação de vídeo com base nesta técnica foram recentemente feitas ao MPEG no contexto da atividade dedicada ao estudo da codificação escalável de vídeo.

2.3.2.1 Codificação de imagens em sub-bandas

As técnicas de codificação em sub-bandas descritas até aqui permitem transformar o sinal do domínio do tempo para o domínio da freqüência, mas não o codificam. Uma das vantagens da representação em sub-bandas de uma imagem é permitir usar algoritmos de compressão perceptual eficientes já que a estrutura hierárquica em escalas (ou resoluções) é adequada às características do sistema visual humano. Uma das técnicas de codificação em sub-bandas mais simples (VETTERLI, 1984) consiste em processar cada uma das bandas numa ordem de varredura definida, quantificar cada coeficiente de acordo com a importância de cada banda (são atribuídos mais bits às bandas com freqüências espaciais mais baixas) e codificar os coeficientes quantificados com um codificador entrópico adequado à estatística do sinal.

Um dos objetivos da análise em sub-bandas consiste na decomposição de uma imagem em um conjunto descorrelacionado (na freqüência) de bandas e isolar os detalhes da imagem em diferentes escalas; no entanto, a representação em sub-bandas de imagens naturais possui uma forte correlação espacial entre bandas. Tal como se pode observar na **figura 2.16**, existe uma correlação entre a localização de regiões em diferentes sub-bandas. Muitas vezes estas regiões correspondem a descontinuidades ou contornos na imagem original; freqüências altas que necessitam da contribuição de todas as sub-bandas para serem representadas. Outra característica importante é que existem muitos coeficientes em uma determinada sub-banda que tem valores muito perto de zero. Assim, parece conveniente selecionar um conjunto reduzido de coeficientes que minimize o erro quadrático médio da codificação para um dado *bit rate*

e codificar apenas um conjunto limitado de coeficientes na zona de mais alta freqüência. Contudo este método exige que o codificador envie informação sobre a posição dos coeficientes escolhidos bem como a respectiva amplitude, para que os dados possam ser decodificados corretamente. Dependendo do método usado, a informação sobre a posição dos coeficientes pode requerer um número de bits mais ou menos significativos, penalizando mais ou menos a eficiência de codificação.

Um grande avanço na codificação de imagens baseada na DWT, surgiu quando Shapiro introduziu o algoritmo EZW (*Embedded Zero-tree Wavelet*) (SHAPIRO, 1993) que codifica implicitamente a posição dos coeficientes. O algoritmo apresentado tem um desempenho superior em relação aos codificadores baseados na DCT (SAENZ, 1999) e marcou uma nova era para os codificadores DWT. Este algoritmo gera um *bitstream*, que pode ser cortado em qualquer ponto e ainda obter uma representação útil da imagem; à medida que mais bits são decodificados, maior é a qualidade da imagem decodificada. O codificador EZW baseia-se em duas observações importantes:

- As imagens naturais tendem a possuir maior energia nas freqüências mais baixas. Quando a transformada DWT é aplicada à imagem, os coeficientes resultantes irão ter uma amplitude decrescente à medida que a sub-banda tem uma resolução espacial superior (o que corresponde a freqüências mais altas).
- Coeficientes com amplitudes maiores são mais importantes que pequenos coeficientes.

A última observação é explorada através da codificação dos coeficientes DWT em várias varreduras. O algoritmo EZW implementa uma aproximação sucessiva dos coeficientes através de limiares sucessivamente decrescentes. O limiar é inicializado com o valor do coeficiente com maior amplitude e em cada varredura este valor é dividido por dois. Este limiar é utilizado para sinalizar ao decodificador se os valores de cada coeficiente são maiores ou menores que o limiar. Este esquema é equivalente a enviar um bit mais significativo de cada coeficiente primeiro e refinar sucessivamente o valor de cada coeficiente através do envio dos restantes bits nas varreduras seguintes. Paralelamente, este algoritmo oferece também uma solução elegante para o problema da correlação entre sub-bandas e elimina a necessidade de enviar a posição dos coeficientes DWT através da definição de uma estrutura em árvore, referida como *quad-tree*. Um coeficiente em uma sub-banda baixa (que corresponde a uma banda de freqüência baixa) possui quatro descendentes na próxima sub-banda mais alta, **figura 2.17**. Os quatro descendentes por sua vez também possuem quatro descendentes na próxima sub-banda mais alta e assim sucessivamente até todas as sub-bandas serem varridas; emerge assim uma árvore *quad-tree* onde cada nó (ou coeficiente) da árvore possui sempre quatro descendentes. Esta estrutura permite uma representação compacta dos coeficientes, em sub-bandas diferentes, mas espacialmente correlacionados.

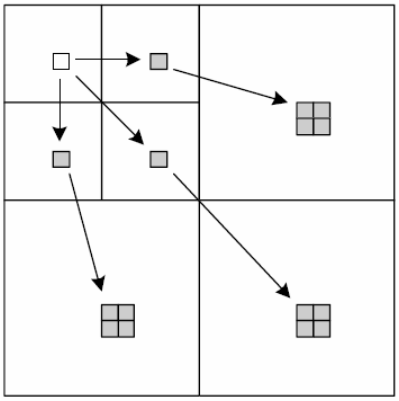


Figura 2.17: Relações entre os coeficientes DWT em sub-bandas diferentes.

O algoritmo EZW tira partido desta estrutura através da definição de um tipo específico de árvore: a *zero-tree*. Uma *zero-tree* é uma *quad-tree* na qual todos os nós da árvore são iguais ou menores que a raiz da árvore (SHAPIRO, 1993). A árvore é codificada com um único símbolo e de acordo com o princípio da aproximação sucessiva, a raiz da árvore tem de ser menor que o limiar estabelecido para a varredura em questão. O codificador explora a ocorrência de *zero-trees* baseado na primeira observação estabelecida anteriormente, ou seja, que os coeficientes DWT decrescem de amplitude com o aumento da resolução da sub-banda. Se a imagem for varrida com uma ordem pré-definida, começando na sub-banda com menor resolução e evoluindo para a maior resolução, muitas posições são implicitamente codificadas através de símbolos que correspondem à *zero-trees*. Para este efeito, o algoritmo EZW mantém duas listas: uma lista dominante que contém a localização de todos os coeficientes não significativos (menores que o valor do limiar) encontrados em varreduras anteriores e uma lista subordinada que contém a magnitude de todos os coeficientes significativos (maiores que o valor do limiar) encontrados em varreduras anteriores. Em Usevitch (2001) pode se encontrar uma descrição mais detalhada deste algoritmo, com exemplos e estudos de desempenho.

Um grande número de métodos de codificação foi proposto desde a introdução do algoritmo EZW(TAUBMAN, 2000) (XIONG, 1997) (SAID, 1996) tendo como característica comum a utilização dos conceitos fundamentais introduzidos pelo algoritmo EZW. Um dos métodos mais populares é o algoritmo SPIHT (SAID, 1996) (*Set Partitioning in Hierarchical Trees*) que melhora o desempenho do EZW e evita a utilização do codificador entrópico. Este algoritmo utiliza estruturas em árvore diferentes e permite que os símbolos *zero-tree* sejam gerados em mais casos, o que permite a combinação em paralelo de *zero-trees*. Este algoritmo define regras para dividir e varrer o conjunto de árvores e coeficientes compartilhados pelo codificador e decodificador. Outra técnica é o algoritmo EBOCOT (TAUBMAN, 2000) (*Embedded Block Coding with Optimized Truncation*) que foi escolhido como base para a norma JPEG2000.

2.3.2.2 Codificação de vídeo em sub-bandas

Todas as técnicas apresentadas até o momento, são exemplos de algoritmos de codificação de imagens fixas, só funcionam em duas dimensões e não se aplicam diretamente a seqüências de vídeo. Uma extensão trivial da codificação em sub-bandas de imagens fixas é a codificação de cada quadro de vídeo independentemente, tal como no formato MJPEG (*Motion* JPEG) onde cada quadro da seqüência de vídeo é

codificado independentemente com a norma de compressão de imagem JPEG. No entanto, este esquema resulta em uma eficiência de codificação baixa, uma vez que não se explora a redundância temporal. Por este motivo, várias técnicas de codificação de vídeo utilizando *wavelets* foram propostas.

Uma das arquiteturas possíveis consiste em combinar a transformada DWT com a transformada DCT em um esquema de codificação híbrido (SUN, 2001). A **figura 2.18** mostra o diagrama de blocos do codificador. Cada quadro do vídeo de entrada é transformado em N sub-bandas através da transformada DWT. A sub-banda LL é codificada usando um codificador baseado na DCT; neste caso, segundo a norma MPEG01 Vídeo. As sub-bandas restantes são compensadas de movimento e codificadas com a técnica EZW. As três sub-bandas HL, LH e HH, como possuem a mesma resolução espacial, sofrem a mesma compensação de movimento que a sub-banda LL, utilizando os mesmos vetores de movimento que são usados na codificação MPEG-1 da banda LL. Para as sub-bandas com resolução espacial superior, são necessários vetores de movimento com uma maior precisão. Para este efeito, a imagem reconstruída a partir das sub-bandas LL, HL, LH e HH é utilizada para calcular novos vetores de movimento. Todas as sub-bandas no nível de resolução espacial superior são compensadas utilizando estes novos vetores de movimento. Este processo é repetido até que todas as sub-bandas sejam codificadas com o método EZW.

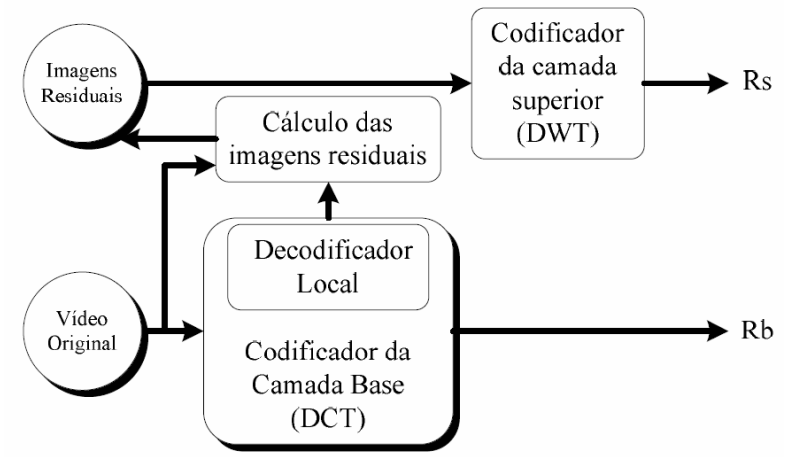


Figura 2.18: Esquema de codificação híbrido DCT/DWT.

O codificador MPEG-1 Vídeo codifica apenas a sub-banda LL que possui a menor resolução espacial e a menor qualidade (apenas as freqüências mais baixas estão presentes) e o *bitstream* gerado corresponde a camada base. O algoritmo EZW gera uma representação escalável de elevada granularidade e permite extrair do *bitstream* (camada superior) vários níveis de qualidade e resolução espacial, dependendo da forma como as *zero-trees* são quantificadas e varridas. Por exemplo, se for desejado vários níveis de qualidade é necessário estabelecer um conjunto de limiares (aproximações sucessivas) com valores decrescentes. Se for desejado vários níveis de resolução espacial, é necessário percorrer primeiro todos os coeficientes das *zero-trees* pertencentes às sub-bandas com resolução espacial inferior e depois percorrer todos os coeficientes das *zero-trees* de sub-bandas com resolução espacial superior e assim sucessivamente.

Outro tipo de técnica concentra-se na codificação da diferença entre as imagens decodificadas localmente pelo codificador e as imagens originais. Esta diferença corresponde a um sinal residual, que contém toda a informação necessária para representar o sinal de vídeo com a qualidade máxima. A estrutura do codificador é

apresentada na **figura 2.18**. Este codificador possui uma estrutura híbrida em termos de escalabilidade (tal como a técnica anterior), pois gera duas camadas: uma camada base que tem de ser totalmente decodificada e uma camada superior que permite uma granularidade elevada, podendo ser truncada em qualquer ponto do *bit rate*. Em Radha (1999), apresenta-se um esquema em que a camada base é gerada por um codificador baseado na DCT (MPEG-4 Visual) e a camada superior é gerada por um codificador baseado na transformada DWT, permitindo oferecer uma escalabilidade muito fina. A codificação dos coeficientes resultantes da DWT pode ser efetuada por qualquer um dos métodos de codificação já apresentados ou um novo método desenvolvido para o efeito. Esta técnica foi uma das candidatas para o modo de escalabilidade fina desenvolvido para a norma MPEG-4 e o modo de codificação foi o SPIHT.

Karlsson e Vetterli foram os primeiros, em 1989, a aplicar as técnicas de codificação em sub-bandas em vídeo através da introdução da pirâmide *wavelet* espaço-temporal (KARLSSON, 1989). Neste modelo, a decomposição em sub-bandas é efetuada espacialmente em cada quadro, mas também temporalmente entre quadros consecutivos. Em vez de codificar cada quadro independentemente, a decomposição em sub-bandas cria dois novos quadros que representam a média e a diferença dos quadros originais. Se existe pouco movimento, a diferença entre quadros é aproximadamente zero e logo é comprimido utilizando poucos bits. Caso contrário, se existe muito movimento, a média captura o movimento de forma difusa e a diferença contém a informação de melhoria necessária para recuperar o detalhe original. Este esquema pode ser generalizado para um número arbitrário de quadros através de filtros temporais com uma maior dimensão e/ou com várias iterações. A principal desvantagem deste modo vem do fato de não possuir módulos de estimação e compensação de movimento, explorando assim de forma pouco eficiente a correlação temporal.

Em Domanski (2000) é apresentada uma solução com o objetivo de melhorar o modo de escalabilidade espacial e temporal da norma MPEG-2 Vídeo através da análise em sub-bandas. A filtragem temporal é efetuada através de dois filtros lineares (ver **figura 2.19**) o que resulta em dois quadros. Estes dois quadros são decompostos em oito sub-bandas espaço-temporais. Três sub-bandas do quadro diferença são eliminadas por corresponderem a informação menos relevante para o sistema visual humano. A sub-banda LL do quadro médio é codificado através da DCT e enviado na camada base. As sub-bandas restantes são também codificadas através da DCT mas enviadas na camada superior. Neste sistema, explora-se a correlação temporal entre sub-bandas da mesma camada, através de módulos de estimação e compensação de movimento.

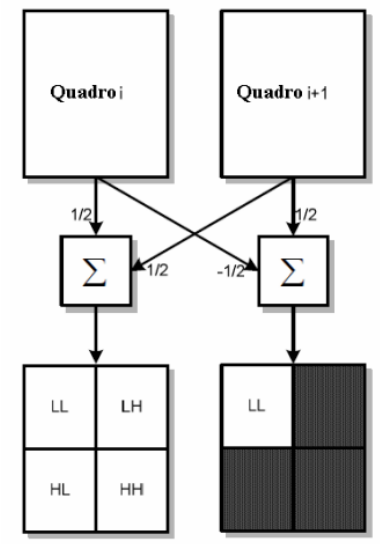


Figura 2.19: Análise em sub-bandas – filtro espaço-temporal

A codificação tridimensional (horizontal, vertical, temporal) é uma abordagem alternativa aos sistemas de codificação híbrida, baseados na DCT e compensação de movimento, utilizados hoje em dia nas normas de codificação de vídeo. Este tipo de técnicas permite oferecer múltiplas resoluções espaciais e temporais e uma gama fina de *bit rates*. O diagrama de blocos do codificador é apresentado na **figura 2.20**. O vídeo é processado por um módulo de estimação e compensação de movimento e por técnicas de decomposição e codificação em sub-bandas de forma a explorar a correlação espacial e temporal do vídeo.

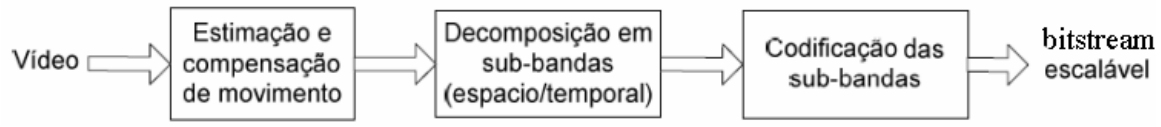


Figura 2.20: Estrutura do codificador de vídeo usando bandas 3D.

As sub-bandas de alta freqüência contém a informação sobre os contornos presentes na imagem. Quando estes contornos estão em movimento, a filtragem temporal os torna pouco nítidos o que leva a uma diminuição de eficiência de codificação. Para evitar este efeito, é necessário a utilização de um esquema de compensação de movimento de forma a apresentar ao sistema de codificação em sub-bandas uma seqüência de imagens com o mínimo de movimento possível. Em Taubman (1994), uma técnica de compensação de movimento global elimina movimento de câmera translacional e, em Tham (1999), um modelo de movimento mais complexo efetua compensação de movimento quando ocorrem ampliações (*zoom*) ou movimentos de objetos. Em seguida, as imagens compensadas são decompostas em sub-bandas, através da transformada DWT. Dependendo das resoluções espaciais e temporais que se pretende obter, vários filtros temporais e espaciais são utilizados. Em Taubman (1994), são utilizadas estruturas em que uma decomposição temporal em duas sub-bandas (freqüências altas e baixas) é aplicada a cada sub-banda espacial. Cada sub-banda temporal correspondente às freqüências baixas pode ainda ser dividida em mais sub-bandas temporais para se construir uma hierarquia temporal; deste modo uma combinação de 34 modos de resolução espacial e temporal são suportados. Em Tham (1999), um conjunto de quadros é agrupado (este conjunto é constituído por um número par de quadros) e

decomposto temporalmente em sub-bandas; em seguida, uma decomposição espacial é efetuada para cada sub-banda temporal. Esta estrutura permite obter entre 0 a 30 sub-bandas que correspondem a um conjunto de resoluções espaciais e temporais. Por fim, a última etapa do codificador consiste na codificação de cada sub-banda, uma das etapas mais importantes. Para que o desempenho seja mais elevado possível, é necessário explorar a correlação entre sub-bandas, tanto espacialmente como temporalmente. Em Taubman (1994), uma combinação de esquemas de codificação baseados nas técnicas PCD, DPCM (*Diferential Pulse Code Modulation*) e RLE (*Run Length Encoding*) é utilizada; em Tham (1999), é usado uma extensão 3D do algoritmo EZW; e finalmente, em Kim (2000) uma extensão 3D do algoritmo SPIHT foi desenvolvida para o efeito. Este tipo de técnicas apresentam um bom desempenho, mas necessitam de múltiplos quadros em memória para serem processados simultaneamente, o que significa que as exigências de memória do codificador e decodificador são elevadas e o tempo de atraso na comunicação é maior quando comparado com os algoritmos de codificação baseados na DCT e compensação de movimento.

Em conclusão, as principais dificuldades na utilização de esquemas de codificação em sub-bandas para vídeo estão associadas a forma de modelar o movimento e a elevada complexidade computacional. Por estes motivos, as técnicas baseadas na DWT não foram ainda adotadas em nenhuma norma de codificação de vídeo (MPEG). Por enquanto, a melhoria no desempenho não parece ser significativa ou as exigências em termos computacionais, de memória ou atraso são ainda muito elevadas.

2.3.3 Transformada DCT

A transformada discreta do coseno (DCT) é utilizada por várias normas de codificação de vídeo (ex: JPEG, H.261, MPEG-1, MPEG-2, etc). Esta técnica é aplicada a blocos da imagem e está normalmente associada a técnicas de compensação de movimento, no popular esquema híbrido de codificação de vídeo DPCM/DCT. Neste esquema, vários tipos de imagens codificadas podem coexistir se houver requisitos de acesso aleatório: imagens do tipo I que não possuem referências a quadros passados e/ou futuros, imagens do tipo P que dependem do quadro I ou P anterior e imagens do tipo B que dependem dos quadros I ou P anterior e seguinte. A primeira técnica apresentada explora esta estrutura; técnicas mais simples poderão possuir apenas codificação do tipo P como a norma H.261.

Ao contrário das técnicas baseadas na transformada DWT, a transformada DCT não produz uma representação em camadas e por este motivo, foi necessário o desenvolvimento de técnicas que permitisse uma codificação eficiente e progressiva dos coeficientes DCT. Algumas das técnicas aqui apresentadas podem ser aplicadas a outras transformadas ou utilizadas em esquemas de transcodificação (ou filtragem); no entanto, são aqui apresentadas devido à sua popularidade nos esquemas de codificação escalável baseados na DCT:

- Seleção de quadros: Uma representação escalável do vídeo pode ser obtida através da seleção de quadros para diferentes camadas. No caso de vídeo codificado com normas MPEG-2 Vídeo, MPEG-4 Visual ou ITU-T H.26x, a abordagem mais comum é estabelecer duas camadas: quadros I e P são transmitidos na primeira camada (camada base) e os quadros B são transmitidos na camada de melhoria (camada superior). Com uma estrutura temporal apropriada, é possível obter uma duplicação ou triplicação da freqüência do quadro com a camada superior e assim oferecer escalabilidade

temporal. Este esquema de codificação encontra-se normalizado na norma H.263+, mas pode ser aplicado a qualquer representação não escalável de vídeo (ex: Chang e Zakhor (1994), os quais implementaram escalabilidade temporal na norma MPEG-1 Vídeo através da seleção e armazenamento de quadros de um GOP (*Group of Pictures*) em uma ordem específica).

- Seleção de componentes: As imagens ou seqüências de imagens a codificar possuem já por si uma representação escalável uma vez que são representadas por um espaço de cores que possui mais do que uma componente. O espaço de cores YUV é normalmente usado (nas normas de codificação de imagem e vídeo) e é constituído por uma componente Y que corresponde a luminância e duas componentes U e V que correspondem às crominâncias. Deste modo, é possível desenvolver esquemas de codificação escaláveis que tiram partido desta característica. Vários esquemas foram propostos, por exemplo, enviar em uma camada apenas os coeficientes DCT da luminância e em outra os coeficientes DCT das crominâncias, ou enviar em uma camada os coeficientes DCT da luminância e o coeficiente DC da crominância e em outra camada enviar os coeficientes AC das crominâncias. Este esquema é apropriado para o caso dos terminais possuírem diferentes características em termos de resolução de cor, quando coexistirem terminais móveis em que o *display* apenas suporta tons de cinza e outros que suportam vários níveis de cor. Este esquema é suportado pela norma JPEG.

- Seleção de coeficientes: Uma característica importante da transformada DCT é que esta técnica é aplicada a pequenos blocos da imagem (em quase todas as normas, a dimensão dos blocos é de 8x8 *pixels*). Outra característica importante é que a transformada DCT está relacionada com a transformada de Fourier discreta e os coeficientes DCT têm uma interpretação no domínio da freqüência. Os coeficientes DCT são normalmente varridos para um vetor numa seqüência pré-definida (zig-zag), das freqüências mais baixas para as mais altas, tal como é ilustrado na **figura 2.21**. Deste modo, os coeficientes DCT nas posições mais baixas do vetor correspondem a freqüências espaciais baixas no bloco da imagem; por outro lado, os coeficientes DCT em posições altas correspondem a freqüências altas. O coeficiente na posição [0,0] do bloco é referido como coeficiente DC e corresponde à intensidade média em um bloco da imagem (luminância ou crominância); os restantes coeficientes são referidos como coeficientes AC. Várias técnicas de codificação escalável tiram partido destas características, através da implementação de filtros no domínio da freqüência. A **figura 2.21** exemplifica o caso da filtragem para duas camadas, uma passa-baixo e outra passa-alto: todos os coeficientes com ordem de varredura zig-zag a um dado valor (coordenada T) são codificados na camada base (a camada com maior importância) e todos os restantes coeficientes são codificados na camada superior. Este esquema pode ser generalizado para um maior número de camadas através de filtros passa-banda, tendo sido adotado pelo modo de separação de dados da norma MPEG-2 Vídeo e pela norma JPEG.

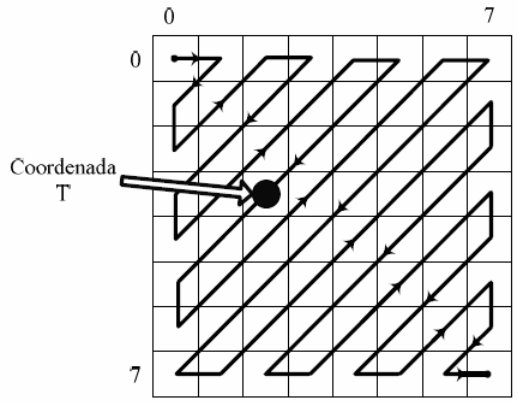


Figura 2.21: Geração de duas camadas de coeficientes através da filtragem passa-baixo no domínio da freqüência.

- Re-quantificação: Este esquema é baseado na aplicação de diferentes passos de quantificação para cada camada do vídeo ou da imagem a codificar. A **figura 2.22** ilustra este tipo de escalabilidade. Os coeficientes DCT são quantificados pelo passo *a* e em seguida inversamente quantificados com o mesmo passo. Os coeficientes DCT originais e os coeficientes DCT inversamente quantificados são subtraídos, gerando o erro de quantificação. Este erro pode ser re-quantificado com um passo de quantificação diferente (menor) e processado da forma já descrita. Esta estrutura pode ser generalizada para um número arbitrário de camadas. Através do uso de passos de quantificação adequados, é possível distribuir o *bit rate* (e consequentemente a qualidade) para cada camada, de acordo com os requisitos da aplicação. Normalmente, a medida que o número da camada diminui, o passo de quantificação diminui ($c < b < a$). Este esquema de codificação é suportado pelo modo de escalabilidade SNR da norma MPEG-2 Vídeo.

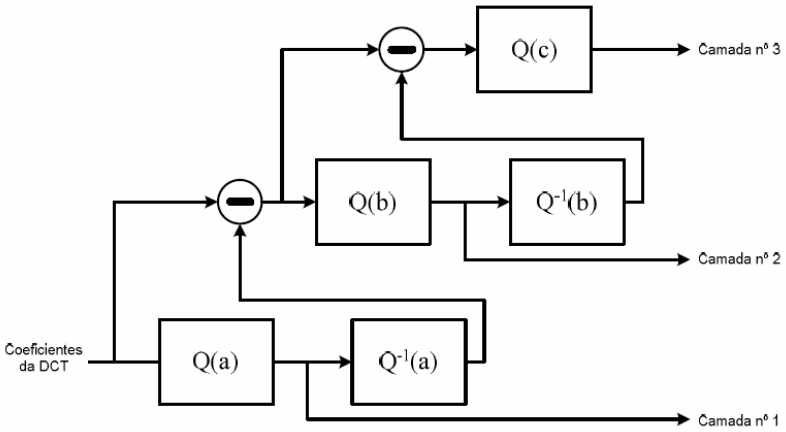


Figura 2.22: Re-quantificação dos coeficientes da DCT em 3 camadas.

- Codificação em planos de bit: Na codificação convencional baseada na DCT, os coeficientes DCT quantificados são codificados através da técnica RLE (*Run Length Encoding*). Com esta técnica, o número de zeros consecutivos antes de um coeficiente diferente de zero na ordem de varredura zig-zag é referido como número de ocorrências (*run*). O valor absoluto do coeficiente

DCT quantificado, diferente de zero, é referido como nível (*level*). Em seguida, os símbolos bidimensionais (nº de ocorrências, nível) ou (*run,level*) são codificados usando uma tabela VLC (*Variable Length Codes*). Um símbolo *eob* (*End of Block*) é utilizado para definir o fim do bloco, ou seja, o fato de não haver mais coeficientes para codificar no bloco em questão. O método de codificação em planos de bit é diferente, pois considera cada coeficiente DCT como um número binário com vários planos de bits em vez de um inteiro com determinado valor (LI, 1997). A **figura 2.23** ilustra esta técnica. Para cada bloco de 8x8 coeficientes DCT, os 64 valores absolutos são estruturados num cubo em que a altura corresponde ao número máximo de bits necessários para representar todos os coeficientes. Os coeficientes são varridos em zig-zag e codificados com símbolos (nº de ocorrências, EOP) no qual EOP (*End of Plane*) vale 1 se tiver chegado ao fim do plano de bit e 0 no caso contrário. Esta técnica pode também ser aplicada a coeficientes resultantes da aplicação de outro tipo de transformada. Este esquema é utilizado pela norma JPEG, JPEG2000 e pelo modo de escalabilidade fina da norma MPEG-4 Visual.

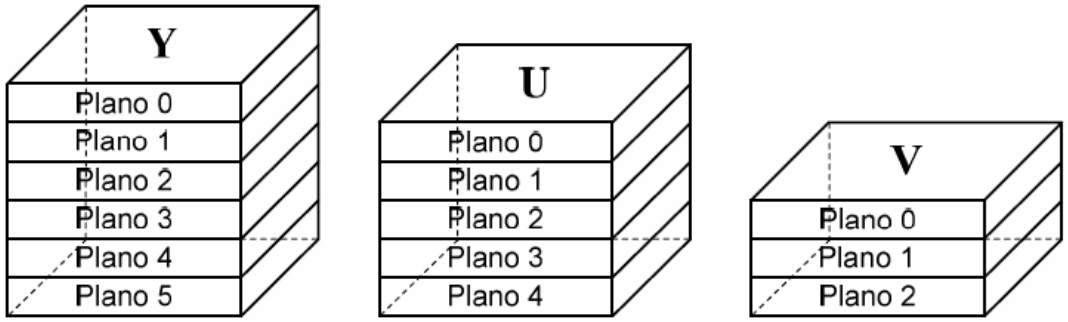


Figura 2.23: Codificação dos coeficientes DCT em planos de bit.

- DCT Piramidal: Esta técnica tira partido de uma característica da DCT: a capacidade de se efetuar sub-amostragem e sobre-amostragem no domínio da freqüência. Deste modo, é possível combinar a estrutura piramidal apresentada anteriormente e gerar uma pirâmide de coeficientes DCT que são posteriormente quantificados e codificados entropicamente (TAN, 1995). Para se sub-amostrar uma imagem (no domínio da freqüência) por um fator de M/N (com M < N), é necessário aplicar a transformada DCT direta em blocos de dimensão NxN e aplicar a transformada IDCT em blocos MxM utilizando os coeficientes das freqüências espaciais mais baixas. Este processo permite decompor a imagem em duas componentes distintas: uma imagem sub-amostrada filtrada passa-baixo e um conjunto de coeficientes DCT que correspondem às freqüências mais altas da imagem. A iteração deste método permite a geração de um conjunto de camadas, tal como é ilustrado na **figura 2.24**. Em cada nível da pirâmide, cada bloco de NxN *pixels* é codificado com a DCT. O canto superior esquerdo de MxM coeficientes é inversamente transformado com a IDCT para formar a nova imagem com uma resolução inferior. Os restantes coeficientes são quantificados e codificados. A imagem no último nível, que corresponde a imagem com resolução inferior, é simplesmente codificada com a DCT. Para se efetuar a decodificação, a operação inversa é realizada, tendo como

partida a imagem com resolução mais baixa. Esta técnica permite um desempenho superior à pirâmide laplaciana (SUN, 2001).

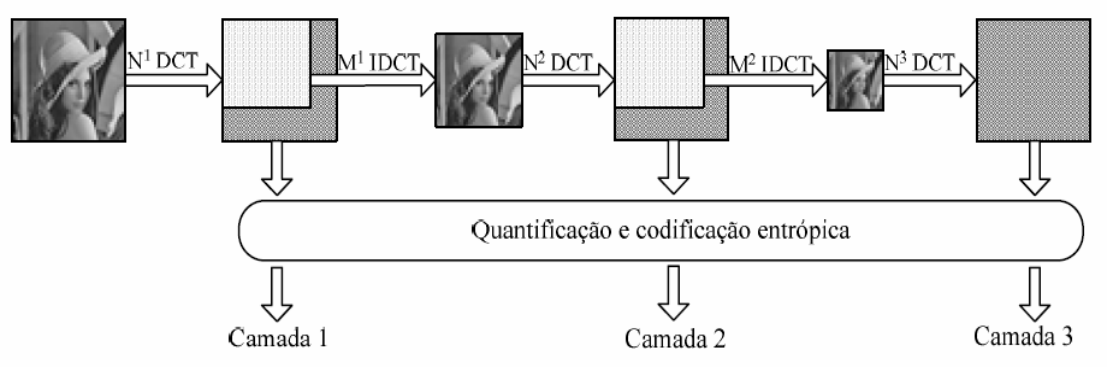


Figura 2.24: Pirâmide DCT com três camadas.

Muitas das técnicas de escalabilidade aqui apresentadas podem ser combinadas para que o sistema completo cumpra determinados requisitos; por exemplo, a norma JPEG combina as técnicas de filtragem e codificação em planos de bit para permitir uma maior granularidade e escalabilidade.

2.3.4 *Matching Pursuits*

Um dos componentes mais importantes de um codificador de vídeo com estimação e compensação de movimento é o algoritmo que se utiliza para codificar o erro de predição ou imagem residual, a diferença entre a imagem compensada a partir da anterior e a imagem atual. Os sistemas atrás descritos utilizam a transformada DCT ou a transformada DWT seguidas de quantificação e codificação entrópica (códigos de *Huffman*). Uma técnica alternativa para a codificação do erro de predição é o algoritmo de *matching pursuits* (NEFF, 1997) que permite uma representação escalável do vídeo. O algoritmo de *matching pursuits* (MP) utiliza, tal como a DCT, um conjunto de funções base para representar o sinal de vídeo; no entanto, o número de funções base definidas é superior (permitindo assim uma maior flexibilidade) em comparação com a transformada DCT. A utilização de um conjunto maior de funções base (*overcomplete basis set*) contém estruturas mais variadas que as funções de base da DCT permitindo representar o sinal residual com um conjunto menor coeficientes. A **figura 2.25** apresenta as funções base da DCT e as funções base do algoritmo MP apresentado em (NEFF, 1997) com 400 funções base (a DCT usa apenas 64), obtidas a partir de funções de Gabor bidimensionais (2D). No contexto deste algoritmo, um conjunto de funções base é referido como dicionário.

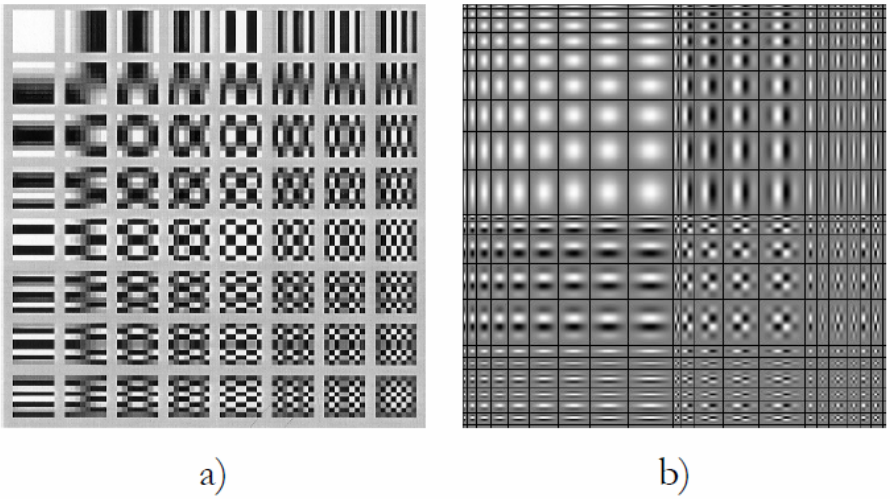


Figura 2.25: a) Funções base da DCT; b) dicionário de Gabor 2D com 400 funções base (NEFF, 1997).

A primeira etapa do codificador de *matching pursuits* é a divisão em blocos da imagem que corresponde ao erro de predição. Após isso, o codificador mede a energia de cada bloco através da soma quadrada dos valores dos *pixels* residuais e o centro do bloco que possui a maior energia é utilizado como estimativa inicial para um processo de busca iterativo. O primeiro passo consiste em definir uma janela de busca SxS ao redor da estimativa inicial de uma forma exaustiva (percorrendo todas as posições da janela de busca) encontrando uma posição (x,y) e uma função do dicionário adequada à representação de uma região da imagem (de mesma dimensão que a função do dicionário escolhida). Mais detalhadamente, o processo de busca consiste em centrar cada função do dicionário (com dimensão NxN) em todas as posições (x,y) contidas na janela de busca SxS e calcular o produto interno entre a estrutura do dicionário e a correspondente região da imagem NxN.

O maior produto interno, uma referencia para a estrutura do dicionário utilizada (posição x e y) e a respectiva localização na imagem (posição x e y) formam o conjunto de cinco parâmetros a se codificar. Este conjunto de parâmetros é referido como um átomo e é codificado entropicamente. Quando um átomo é determinado, é subtraído da imagem e o processo de busca é repetido. A decomposição em átomos do erro de predição é ilustrada na **figura 2.26**. Na **figura 2.26a** é apresentado um quadro da seqüência original cujo resíduo é ilustrado em b); em c), apresenta-se a localização dos primeiros cinco átomos. Como se pode ver, as características visuais mais relevantes são codificadas primeiro. As **figuras 2.26** d e e) mostram a representação da imagem residual com 30 e 64 átomos, respectivamente:

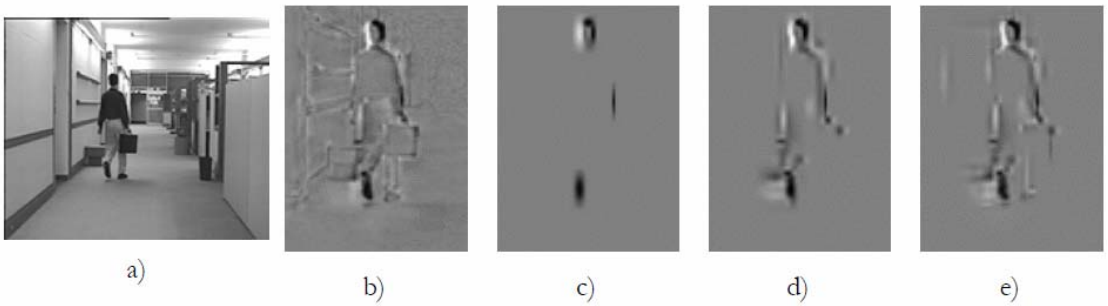


Figura 2.26: Algoritmo de *matching pursuits*: a) quadro 60 da seqüência *Hall Monitor*; b) erro de predição – os primeiros; c) 5 átomos; d) 30 átomos; e) 64 átomos.

Quando se codifica a baixos *bit rates*, a escolha das funções de base é extremamente importante uma vez que o codificador deve representar cada imagem residual com poucos coeficientes. Os codificadores baseados na DCT produzem imagens em que muitas vezes são visíveis efeitos de bloco e ruído de alta freqüência em torno dos contornos dos objetos em movimento. Ao utilizar um conjunto amplo de funções de base (ex: 400 como em (NEFF, 1997)) com uma variedade de escalas e com localizações arbitrárias na imagem, é possível a eliminação destes efeitos. Este codificador apresenta uma eficiência de codificação superior quando comparado com o codificador H.263+, especialmente para baixos *bit rates* (10 a 25 kbit/s); no entanto, possui uma complexidade computacional superior o que limitou as suas possibilidades de sucesso no contexto da norma MPEG-4 (AL-SHAYKH, 1999) (NEFF, 1997).

Este algoritmo permite que a representação do vídeo seja escalável como uma elevada granularidade, uma vez que cada imagem residual é representada por um conjunto de átomos e cada átomo pode ser independentemente decodificável, oferecendo um acréscimo de qualidade. No entanto, a codificação entrópica dos 5 parâmetros dos átomos vai influenciar o desempenho e o número de camadas de escalabilidade possíveis; normalmente, para se obter um desempenho superior, os átomos são codificados em grupos de N átomos, permitindo assim que a respectiva localização na imagem residual seja codificada de uma forma mais eficiente. Em Al-Shaykh (1999), é utilizada a codificação entrópica de *Huffman* para codificar cada conjunto de átomos e conclui-se que o desempenho do algoritmo de *matching pursuits* melhore a medida que o número de átomos codificados em conjunto aumentem, existe uma troca entre eficiência de codificação e a granularidade do *bitstream.* Vários métodos de codificação de pequenos grupos de átomos são apresentados em Al-Shaykh (1999), incluindo estudos sobre o seu desempenho e a sua granularidade. Esta técnica foi uma das propostas apresentadas ao grupo MPEG como método de codificação de vídeo com elevada granularidade (CHEN, 1998) a ser incorporado na norma MPEG-4 Visual, mas não foi escolhida.

3 CODIFICAÇÃO ESCALÁVEL DE VÍDEO NA NORMA MPEG-4

No mundo das telecomunicações, a largura de banda é um bem escasso e como tal, a codificação de vídeo tem um papel importante na redução da quantidade de informação a transmitir. O impacto das novas tecnologias digitais, devido a sua importância econômica e a necessidade de garantir a interoperabilidade entre terminais, levou ao desenvolvimento de normas de compressão de vídeo digital, segundo determinados requisitos. Um dos principais objetivos consistia em alcançar a melhor qualidade visual para um determinado *bit rate*. Para desenvolver este tipo de norma, o grupo MPEG (*Motion Picture Experts Group*) foi fundado em 1988, sob o patrocínio conjunto da ISO (*International Standards Organization*) e IEC (*International Engeneering Consortium*). A primeira norma desenvolvida por este grupo, a norma MPEG-1, permite a codificação de informação audiovisual com *bit rates* na ordem de 1.5 Mbit/s e foi desenvolvida com o objetivo principal da gravação de informação em CD-ROM. A norma seguinte, a MPEG-2, é mais abrangente, pois permite uma gama mais ampla de *bit rates* e resoluções espaciais para o vídeo e um maior número de canais de áudio. Esta norma é hoje largamente utilizada em televisão digital e gravação audiovisual DVD (*Digital Versatile Disc*). As normas MPEG-1 e MPEG-2 foram pioneiras em termos de representação digital de seqüências de vídeo, com vários benefícios: uma maior eficiência em termos de largura de banda, qualidade acrescida, fácil processamento da informação, etc. No entanto, as formas de consumo de conteúdo pelos usuários mantiveram-se fundamentalmente as mesmas por não mudar o modelo de dados utilizado na representação. Mais concretamente, as aplicações implementadas com base nestas normas oferecem ao usuário essencialmente as mesmas funcionalidades que aquelas já oferecidas com base no vídeo analógico, uma vez que o conteúdo visual continua a ser representado segundo o mesmo modelo de dados – uma seqüência periódica de quadros retangulares – mas agora em um formato digital, ou seja, com cada quadro constituído por um dado número de *pixels* organizados em uma matriz.

Contudo, a crescente difusão das tecnologias digitais, tem colocado cada vez mais em relevo as limitações do modelo de vídeo baseado em quadros, mostrando que novas funcionalidades podem ser oferecidas aos usuários (ex: em termos de acesso e manipulação da informação), se um novo modelo de dados baseado na composição de uma cena por vários objetos independentes for adotado. A norma MPEG-4 veio disponibilizar e normalizar tecnologias de representação audiovisual que permitem oferecer novas funcionalidades como codificação baseada no conteúdo, interatividade e a combinação eficiente na mesma cena de objetos com origem natural sintética. Por outro lado, um grande esforço tem sido feito para permitir a transmissão de vídeo a *bit*

rates muito baixos, especialmente para terminais móveis. A norma MPEG-4 Visual apresenta uma eficiente compressão de vídeo para *bit rates* entre 5 kbit/s e 1 Gbit/s e para resoluções espaciais entre sub-QCIF (128 x 96 *pixels* de luminância) e alta resolução (4k x 4k *pixels* de luminância). Esta norma inclui também novas ferramentas de resiliência a erros de forma a minimizar o impacto dos erros de transmissão na qualidade do vídeo visualizado na recepção. Estas ferramentas são indicadas para uma grande variedade de canais de transmissão (incluindo redes móveis) e para uma ampla gama de condições, sem causar um impacto significativo na diminuição da eficiência de codificação.

3.1 A norma MPEG-4

O modelo de representação baseado na composição de objetos audiovisuais está na base das novas funcionalidades oferecidas pela norma MPEG-4 e constitui a maior diferença conceitual relativa às normas anteriores (MPEG-1 e MPEG-2). Na **figura 3.1** está representada uma versão simplificada da arquitetura de um sistema MPEG-4. No lado do emissor, os vários objetos de áudio e de vídeo e ainda a informação de composição da cena são codificados separadamente. Os *bitstreams* elementares gerados são dois multiplexados, formando um único *bitstream* que é enviado para o canal. No receptor, o *bitstream* recebido é desmultiplexado para se obterem os *bitstreams* elementares correspondentes aos vários objetos de áudio e vídeo e a informação de composição de cena. Os *bitstreams* elementares são então decodificados por decodificadores adequados e o compositor compõe a cena com base na informação de composição de cena recebida e decodificada.

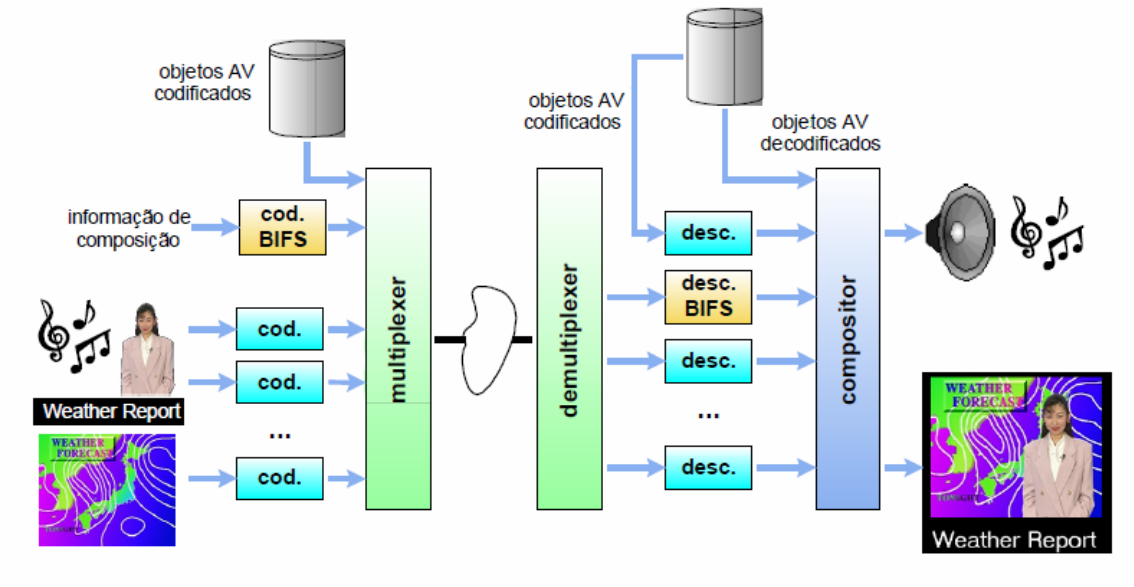


Figura 3.1: Arquitetura de um sistema MPEG-4

O fato de uma cena audiovisual ser modelada como uma composição de objetos, habitualmente como valor semântico, ou seja, significado relevante no contexto da aplicação em questão apresenta grandes potencialidades técnicas e funcionais, entre elas:

- **Processamento e codificação seletiva dos objetos:** O modelo baseado em objetos permite que tipos diferentes de objetos sejam processados e codificados de maneira diferente e adequada a cada um deles. Por exemplo,

texto e vídeo não são codificados utilizando as mesmas ferramentas de codificação porque são tipos de dados com características muito diferentes e que, como tal, beneficiarão em ser codificados com ferramentas ajustadas às suas características.

- **Reutilização de objetos**: Com a norma MPEG-4, qualquer objeto que seja colocado na cena permanece individualmente acessível, já que o *bitstream* codificado é independente para cada objeto, podendo deste modo cada objeto ser facilmente reutilizado em outras cenas.
- **Integração de conteúdos com origem natural e sintética:** A norma MPEG-4 permite integrar, numa mesma cena e com elevada eficiência, objetos com origem natural ou sintética. Por exemplo, é possível integrar na mesma cena um personagem de desenho animado com um ator real, permanecendo ambos individualmente acessíveis. Por outro lado, um ator pode fazer parte de um cenário virtual. Com o MPEG-4, conteúdos naturais e sintéticos "habitam" harmoniosamente na mesma cena sem discriminação como acontecia no passado.
- **Interação com e entre objetos**: Como os objetos são independentemente acessíveis, o usuário pode interagir com a cena de várias formas, alterando a dimensão espacial de um objeto ou ascendendo a informação (metadados) sobre ele. Além disso, os objetos podem interagir entre si: por exemplo, um objeto pode alterar a sua posição ou aparência ou a de outro objeto caso ao se moverem exista uma colisão entre eles.

Devido aos objetivos ambiciosos com os quais foi concebida, a norma MPEG-4 especifica um numeroso conjunto de ferramentas relacionadas com várias áreas tecnológicas. Para facilitar o seu uso da implementação de produtos de acordo com a norma, a mesma está organizada em várias partes, que podem ser utilizadas em conjunto ou separadamente ainda que o máximo benefício deva resultar quando as tecnologias MPEG-4 são usadas em sinergia. Em seguida, são apresentadas as partes mais significativas:

- **Parte 1 (Sistema)** (ISO:14496-1, 2001): Especifica a arquitetura global de um terminal MPEG-4, bem como a multiplexagem e sincronização dos *bitstreams*, tal como na norma MPEG-2 Sistema. No entanto, para permitir um conjunto de novas funcionalidades, também se especifica o formato de descrição/composição de cena BIFS (*Binary Format for Scenes*), informação descritiva do conteúdo OCI (*Object Content Information*), ferramentas visando permitir o uso de técnicas para proteger a propriedade intelectual IPMP (*Intellectual Property Management and Protection*), o formato de arquivo mp4 e interfaces visando o controle programático do terminal (MPEG-J), entre outras coisas.
- **Parte 2 (Visual)** (ISO:14496-2, 2001): Especifica a sintaxe e semântica do *bitstream* comprimido da informação visual de origem sintética (gerada por computador) bem como para objetos de vídeo com forma retangular ou arbitrária. Para além da sintaxe e semântica do *bitstream*, são também especificadas as ferramentas correspondentes de decodificação, de forma, movimento e textura, de resiliência a erros, escalabilidade e *sprites* (vista panorâmica de uma cena de vídeo), indicadas para objetos de origem natural.

Para objetos de origem sintética, esta parte da norma especifica o *bitstream* e a decodificação de modelos 2D e 3D, como faces e corpos, bem como a sua animação de modo eficiente mas realista.

- **Parte 3 (Áudio)** (ISO:14496-3, 2001): Especifica a representação codificada dos objetos de áudio naturais (fala e música) e sintéticos. Esta parte inclui várias ferramentas de decodificação, composição de objetos de áudio sintéticos e naturais, escalabilidade no *bit rate*, representação de áudio estruturado (sintético), uma interface para conversão de texto em fala TTS (*Text to Speech*), decodificação de fala CELP (*Coded Excited Linear Prediction*) e HVXC (*Harmonic Vector eXitation Coding*).
- **Parte 4 (Teste de Conformidade**) (ISO:14496-4, 2000): Esta parte da norma define um conjunto de testes concebidos para verificar se os *bitstreams* e os decodificadores cumprem as especificações definidas nas partes 1, 2 e 3 da norma de modo a garantir interoperabilidade. Através destes testes é possível verificar se um dado codificador produz *bitstreams* válidos (ainda que a codificação propriamente dita não seja normativa).
- **Parte 5 (Software de referência)** (ISO:14496-5, 2001): Inclui programas que implementam as ferramentas especificadas nas partes 1, 2, 3 e 6 da norma. Ainda que várias implementações de codificadores e decodificadores segundo a norma MPEG-4 não tenham de usar estes programas, o seu comportamento deve ser semelhante, e tal como descrito nas partes 1, 2, 3 e 6. Vale lembrar que estes programas estão livres de *copyright* (mas não de patentes) quando usados na implementação de produtos comerciais em conformidade com a norma MPEG-4.
- **Parte 6 (DMIF – *Delivery Multimedia Integration Framework*)** (ISO:14496-6, 2000): A parte Sistema da norma MPEG-4 não especifica a forma como o *bitstream* codificado é transportado nas diferentes redes de acesso (ao contrário da norma MPEG-2 Sistema). Esta parte da norma MPEG-4 especifica uma interface entre a representação codificada do conteúdo e um conjunto de protocolos de transporte. Para este efeito, é definido um protocolo de sessão (em termos OSI) que permite uma distribuição sincronizada em tempo-real de conteúdo MPEG-4.
- **Parte 7 (Software otimizado)** (VÍDEO, 2001): Esta parte inclui programas otimizados para alguns algoritmos relevantes para a implementação da parte visual da norma (ex: algoritmos de estimação de movimento).
- **Parte 8 (MPEG-4 em redes IP)** (SUB, 2001): Esta parte especifica uma arquitetura para o transporte de conteúdo MPEG-4 em redes IP. Também inclui regras (na forma de recomendações) para os formatos de transporte do conteúdo MPEG-4 em diversos protocolos IP, como as regras de fragmentação e concatenação para o protocolo RTP, regras de uso do SDP (*Session Description Protocol*), definições MIME (*Multi-Purpose Internet Mail Extensions*) e considerações sobre segurança e ligações ponto-multiponto (*multicast*).
- **Parte 9 (Hardware otimizado)** (ISO:14496-2, 2001): Esta parte inclui as descrições de circuitos integrados correspondentes a ferramentas MPEG-4 (ex: DCT / IDCT, estimação de movimento, etc). Estas descrições utilizam a

linguagem VHDL (*Very High speed integrated circuit hardware Description Language*) e tem como principal objetivo facilitar o desenvolvimento de implementações em hardware ou hardware/software de codificadores e decodificadores MPEG-4.

- **Parte 10 (AVC – *Advanced Vídeo Coding*)** (WIEGAND, 2002): Esta parte especifica a sintaxe de uma nova norma de codificação de vídeo que permite uma maior eficiência de codificação e robustez a redes com erros. Esta parte foi desenvolvida pelo grupo JVT (*Joint Vídeo Team*), formado a partir do grupo VCEG (*Vídeo Coding Experts Group*) da ITU-T e o grupo MPEG do ISO/IEC.

3.2 A norma MPEG-4 Visual

A norma MPEG-4 Visual consiste em um conjunto de ferramentas que permite a codificação de objetos de vídeo e imagens fixas (especificando-se a sintaxe e decodificação para cada uma destas ferramentas) bem como de objetos sintéticos tais como malhas 2D e modelos 3D, com especial relevância para os modelos faciais e do corpo humano. Nesta seção, será dada uma ênfase à codificação de objetos de imagem e vídeo (naturais). Enquanto a codificação de vídeo é baseada na transformada DCT (*Discrete Cosine Transform*) e na compensação de movimento, a codificação de imagens fixas é baseada na transformada DWT (*Discrete Wavelet Transform*) e na codificação com *zero-trees*. Para ambos os casos, foi necessário definir ferramentas que permitissem a codificação eficiente, de modo a realizar o modelo de representação baseado em objetos de forma arbitrária. As ferramentas MPEG-4 de codificação de informação de vídeo proporciona o seguinte conjunto de funcionalidades (PEREIRA, 2000):

- **Codificação de objetos de forma arbitrária:** Em conseqüência do novo modelo de dados, baseado na composição de objetos, a norma MPEG-4 permite a codificação de objetos com forma arbitrária (**figura 3.2**). Esta funcionalidade permite que o usuário interaja com o próprio conteúdo visual, por exemplo, alterando o tamanho de um objeto para melhor visualização, reposicionando o objeto espacialmente e/ou temporalmente para personalização da composição da cena, determinando a evolução de uma história conforme o objeto selecionado ou até mesmo reutilizando o objeto em outra cena. O usuário pode também obter informação sobre um dado objeto de vídeo, como o nome de um personagem em um filme, ou dados bibliográficos do ator selecionado.

Figura 3.2: Exemplo de composição de uma cena com 3 objetos distintos.

- **Eficiência de compressão para um conjunto amplo de *bit rates*:** A eficiência de compressão foi o grande objetivo das normas MPEG-1 e MPEG-2, o que permitiu o desenvolvimento da televisão digital e do DVD, entre outras aplicações. A norma MPEG-4 introduz novas ferramentas de codificação mais eficientes, especialmente para os *bit rates* mais baixos, mas também para os mais elevados (*studio profile*). Além disso, é possível escolher para cada objeto as ferramentas de codificação mais ajustadas às suas características, aumentando a eficiência de codificação. Por exemplo, o texto é codificado com ferramentas adequadas, ao contrário do que acontece nas normas anteriores MPEG-1 e MPEG-2 onde o texto é codificado como textura.

- **Escalabilidade de conteúdo, espacial, temporal e de qualidade:** O MPEG-4 normalizou um conjunto de ferramentas para garantir que decodificadores com diferentes capacidades computacionais ou diferentes larguras de banda possam escolher um *bitstream* adequando às suas características. A norma MPEG-4 Visual oferece escalabilidade espacial, temporal e de qualidade com baixa e elevada granularidade, tanto para o vídeo como para imagens fixas. Devido ao novo modelo baseado em objetos, também permite escalabilidade de objeto ou de conteúdo, ou seja, o decodificador pode consumir mais ou menos objetos (começando pelos mais prioritários) conforme os recursos disponíveis. As principais aplicações que a norma pretende possibilitar com estas ferramentas são: a transmissão de vídeo através da Internet ou de redes móveis, serviços de vídeo com qualidade variável entre os vários consumidores e a procura de vídeo em bases de dados com diferentes resoluções ou qualidades.

- **Transmissão robusta em canais com erros:** para permitir a transmissão de vídeo em canais (móveis ou fixos) com erros, a norma MPEG-4 inclui ferramentas visando aumentar a resiliência (robustez) a erros do *bitstream*. Dependendo das características do canal de transmissão, estas ferramentas podem substituir ou complementar a codificação de canal de maneira a maximizar a qualidade do vídeo recebido pelo cliente, por exemplo, através do aumento da capacidade de ressincronismo ou de ocultação dos erros. Sendo o seu principal objetivo a maximização da qualidade visual subjetiva na presença de erros, ou seja, a minimização do impacto negativo dos erros,

é necessário salientar que a norma não especifica qual o comportamento dos decodificadores na presença dos erros (ou seja, o comportamento de decodificação é sempre especificado para condições ideais em termos de erros), mas apenas a sintaxe semântica do *bitstream* corresponde a estas ferramentas.

Tal como acontece com as normas anteriores (MPEG-1 e MPEG-2), a norma MPEG-4 não especifica o codificador, mas apenas o decodificador e a sintaxe e semântica do *bitstream* uma vez que isso é suficiente para garantir a interoperabilidade. O fato do processo de codificação não precisar ser normalizado, incentiva a competição e inovação por parte da indústria, com resultados evidentes na obtenção de melhores e mais eficientes algoritmos para o codificador. É também este o caso para os algoritmos de segmentação de seqüências de vídeo, fundamentais para se obterem objetos de vídeo com uma forma arbitrária a partir de cenas obtidas com uma câmera de vídeo; estes algoritmos podem continuar a evoluir por muito tempo ou pode até fazer-se a segmentação com auxílio humano, se a aplicação permitir (conteúdo criado *off-line*).

3.2.1 Estrutura e sintaxe do *bitstream* codificado

O conceito mais importante definido no MPEG-4 é o de objeto; este conceito permite uma representação adequada a aplicações interativas e permite um acesso direto ao conteúdo de cada cena. No contexto desta dissertação só se consideram objetos de vídeo, mas os mesmos princípios permanecem válidos para os outros tipos de objetos. Um objeto de vídeo pode ser constituído por uma ou mais camadas escaláveis em termos espaciais, temporais ou de qualidade. Para atingir este objetivo, um codificador MPEG-4 possui a sua disposição três tipos de ferramentas que permitem:

- **Representação escalável de vídeo com baixa granularidade:** Permite que um objeto seja decodificado a partir de uma camada base a qual são acrescentadas outras camadas (em número bastante limitado) que melhoram sucessivamente a resolução espacial ou temporal do objeto (ISO:14496-2, 2001). Cada camada tem de ser decodificada por completo, para se obter uma melhoria na resolução espacial ou temporal.
- **Representação escalável de vídeo com elevada granularidade:** Para os casos onde seja necessária uma granularidade mais elevada em termos de escalabilidade, a norma MPEG-4 permite que um objeto retangular seja codificado em duas camadas, uma camada base não escalável e uma camada superior codificada de uma forma progressiva (com uma elevada granularidade) para obter uma representação escalável em termos de qualidade (SNR). Este modo é referido como MPEG-4 FGS (*Fine Granularity Scalability*) (ISO:14496-2, 2001).
- **Representação escalável na textura com elevada granularidade:** Para o caso de texturas ou imagens fixas (ex: fundo de cena), a norma MPEG-4 Visual define um modo de codificação especial, designado por VTC (*Visual Texture Coding*), baseado na transformada DWT (ISO:14496-2, 2001). Neste modo, é possível obter uma representação escalável da textura com elevada granularidade, tanto em termos espaciais como de qualidade.

A codificação permite que, para cada objeto, seja gerado um conjunto eficiente de *bitstreams* úteis para uma ampla gama de larguras de banda e características dos terminais (ex: dimensão do monitor, complexidade computacional, etc).

Uma cena de vídeo MPEG-4 pode ser constituída por um ou mais objetos. Cada objeto de vídeo é caracterizado através de informação espacial e temporal, referente a três características principais: forma, movimento e textura. Para as aplicações que não necessitam de objetos de vídeo de forma arbitrária, a norma MPEG-4 permite a codificação de objetos retangulares, que constituem assim um caso particular de objetos de forma arbitrária. Os *bitstreams* resultantes da codificação de cada um dos objetos numa dada cena seguem uma estrutura hierárquica, como mostrado na **figura 3.3**.

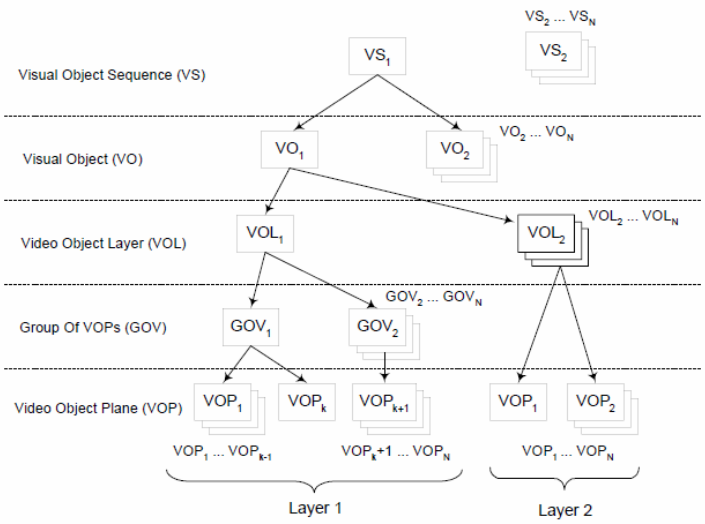


Figura 3.3: Estrutura hierárquica do *bitstream* de vídeo MPEG-4.

Os níveis hierárquicos que descrevem os objetos visuais numa cena MPEG-4 são os seguintes:

- **Visual Object Sequence (VS):** É a estrutura sintática que está no topo da hierarquia do *bitstream* codificado. Inclui objetos visuais 2D ou 3D, naturais ou sintéticos, e as respectivas camadas de melhoramento. Este nível hierárquico indica quais os objetos visuais presentes na cena e qual o perfil e o nível dessa cena.

- **Visual Object (VO):** O objeto visual é a mínima entidade presente na cena que o usuário pode manipular. Este objeto pode ter origem natural ou sintética, ou seja, pode ser uma seqüência de quadros retangulares, uma seqüência de formas arbitrárias, uma textura estática, uma malha 2D ou 3D genérica ou um objeto facial 3D. Habitualmente, um objeto visual corresponde a um objeto com significado semântico (que pode merecer ser manipulado ou fundo de cena). Este nível hierárquico serve para indicar qual o tipo de objeto visual em questão e quais as camadas escaláveis que o constituem.

- **Vídeo Object Layer (VOL):** A sintaxe de codificação de vídeo na norma MPEG-4 inclui codificação escalável (várias camadas) e não escalável (uma única camada). Cada camada de codificação de um objeto de vídeo é indicado pelo VOL. A escalabilidade permite a reconstrução de um objeto partindo da camada base (que pode ser decodificada independentemente) a qual são adicionadas camadas superiores de melhoria. Como já foi referido, um dado objeto de vídeo pode ser codificado utilizando escalabilidade temporal, espacial ou de qualidade, combinadas ou não. Este nível (VOL)

contém toda a informação referente a uma dada camada de um objeto de vídeo, como os parâmetros de codificação e *bitstream*.

- **Group of Vídeo Object Planes (GOV):** Os GOVs agrupam os VOPs e são essenciais para o acesso aleatório à informação de vídeo. No início de cada GOV, o VOP é codificado de modo independente (codificação Intra), sem se "pendurar" em quadros anteriores ou posteriores, permitindo deste modo que existam pontos de acesso aleatório para que se possa decodificar só uma dada parte do *bitstream*.
- **Vídeo Object Plane (VOP):** Uma instância de um objeto de vídeo num determinado instante de tempo é denominado VOP. O processo de codificação gera representações codificadas de VOPs, juntamente com informação de composição que permite compô-los em uma cena. A norma MPGE-4 Visual suporta 3 modos de codificação de VOPs, ilustrados na **figura 3.4**:
  - *Intra* VOP ou I-VOP: O VOP é codificado independentemente de qualquer outro VOP.
  - *Predicted* VOP ou P-VOP: O VOP é codificado utilizando o VOP I ou P mais recentemente decodificado (sempre no passado), através de técnicas de estimação e compensação de movimento.
  - *Bidirectional* VOP ou B-VOP: O VOP pode ser codificado utilizando como predição VOPs I ou P passados e/ou futuros (sempre os mais próximos), em termos de apresentação / visualização.

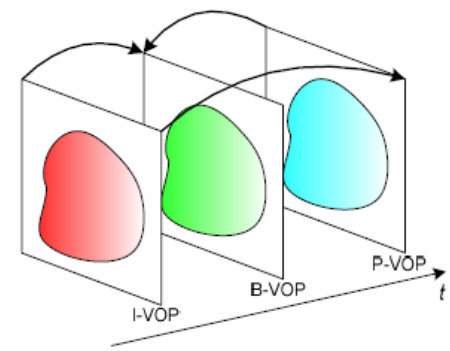


Figura 3.4: Modos de codificação de um VOP.

Apesar da norma MPEG-4 utilizar um modelo de representação baseado em objetos, o processamento dos VOPs é realizado ao nível do bloco, sendo cada VOP dividido em blocos quadrados, cujo conteúdo é então codificado. Deste modo, cada VOP é codificado a partir do retângulo de menor área, e com dimensões múltiplas de 16 *pixels*, que engloba completamente o objeto nesse instante, designado por *VOP Bounding Box; a bounding box* é então sub-dividida em blocos quadrados de dimensão 16x16 *pixels* (luminância), chamados macroblocos.

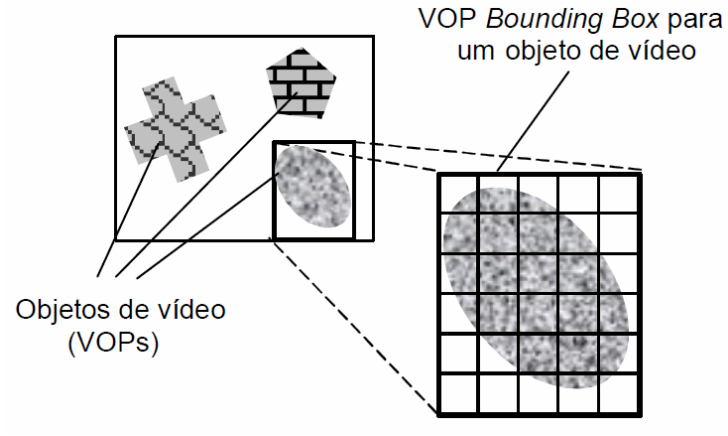


Figura 3.5: *Bounding box* para um objeto e divisão em macroblocos.

Cada um destes macroblocos é dividido em blocos de dimensão 8x8 *pixels* que contém componentes de luminância e de crominância, podendo a crominância ser sub-amostrada espacialmente em relação à luminância. Assim, para o formato 4:2:0 (crominâncias com metade da resolução da luminância nas linhas e nas colunas), cada macrobloco contém 4 blocos de luminância (16x16 *pixels*) e 2 blocos de crominância (8x8 *pixels*) (uma para cada uma das crominâncias). Três tipos de macroblocos podem ser distinguidos para cada VOP, em uma arquitetura de representação como a adotada pela norma MPEG-4:

- **Transparentes:** Macroblocos que estão completamente fora do VOP, mas dentro da sua *bounding box*. Para estes macroblocos, nenhuma informação de textura (YUV) é codificada, uma vez que através da informação de forma do objeto em questão o receptor sabe que estes macroblocos são transparentes (não sendo visíveis na cena decodificada).
- **Opacos:** Macroblocos que se encontram totalmente dentro do VOP. Estes macroblocos são codificados da mesma forma que os macroblocos em objetos retangulares, ou são codificados no modo Intra através da utilização da informação de textura naquele instante ou são codificados no modo Inter através de técnicas de estimação e compensação de movimento e codificação do erro de predição (P ou B-VOPs).
- **Fronteira:** Macroblocos que se encontram na fronteira do VOP, ou seja, que incluem uma parte opaca e uma parte transparente. Estes macroblocos são codificados através de ferramentas específicas para a codificação de objetos com uma forma arbitrária.

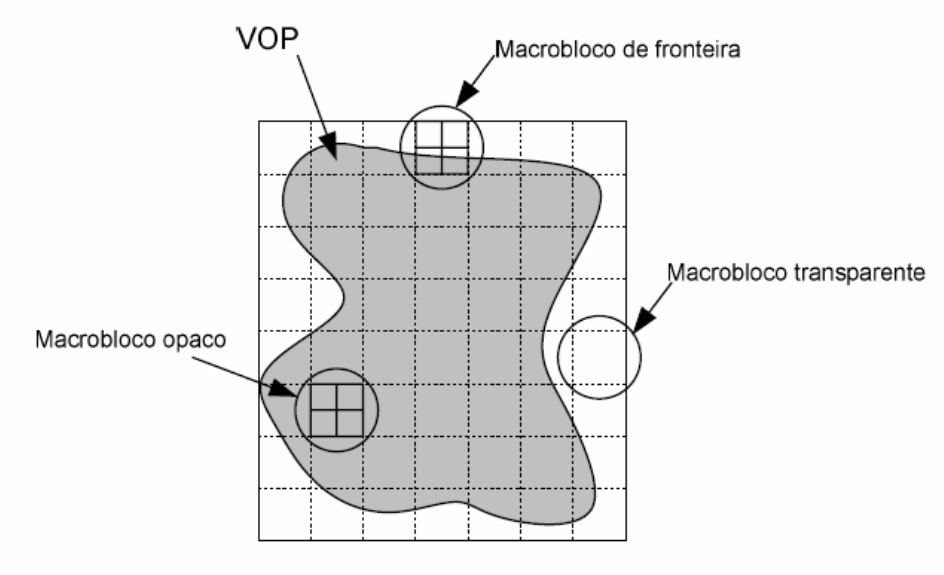


Figura 3.6: Tipos de macroblocos usados para a codificação de um VOP com forma arbitrária.

3.2.2 Arquitetura da codificação de vídeo

Um dos princípios adotados para o processo de normalização do grupo ISO/MPEG é o de que se deve especificar o número mínimo de ferramentas para garantir a interoperabilidade, deixando o máximo de liberdade e espaço para competição nas ferramentas cuja especificação não é essencial para garantir a interoperabilidade. Este princípio faz com que o codificador nunca seja normalizado nas normas MPEG (áudio e/ou vídeo) mas apenas o *bitstream* e o decodificador. Desta forma, os algoritmos de codificação podem ser otimizados segundo certos critérios relevantes para a aplicação em questão; eficiência de codificação ou complexidade computacional, estimulando a competitividade entre diversas implementações do mesmo codificador. Por exemplo, os codificadores MPEG-2 Vídeo (MainProfile@MainLevel) conseguem hoje garantir com 2Mbit/s a qualidade que no início só conseguiam com 6 a 8 Mbit/s devido à evolução das técnicas utilizadas na codificação.

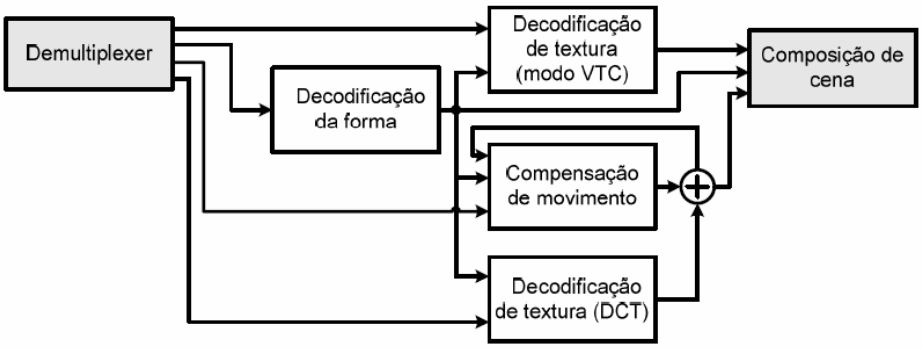


Figura 3.7: Resumo das ferramentas de decodificação de vídeo especificadas na norma MPEG-4 Visual.

Na **figura 3.7** são apresentadas ferramentas de decodificação que são especificadas na norma MPEG-4 Visual para o caso de imagens VCT e vídeo. Os módulos em cinza não fazem parte da norma MPEG-4 Visual (enquanto que a multiplexagem está definida na parte de Sistema, a composição de objetos de vídeo não é normalizada no contexto

da norma MPEG-4) e por este motivo não são aqui apresentados. A grande inovação da norma MPEG-4 em termos de codificação de vídeo é o módulo de decodificação de forma, necessário para se obter uma representação baseada em objetos de forma arbitrária. A informação de forma é também utilizada pelos módulos de compensação de movimento e decodificação de textura. Caso se pretenda decodificar um vídeo baseado em quadros retangulares, o módulo de decodificação de forma não é necessário e o decodificador apresentará a estrutura do já conhecido esquema híbrido como DCT e compensação de movimento. Apesar de esta estrutura básica ser a mesma das normas anteriores, vários algoritmos de codificação foram melhorados ou introduzidos pela primeira vez em uma norma que tinha também o objetivo de aumentar a eficiência de codificação e a robustez a erros.

Tendo sido introduzida a arquitetura geral da codificação de imagem e vídeo MPEG-4, serão apresentados a seguir as principais ferramentas de codificação de vídeo e imagens fixas, cuja descrição mais detalhada pode ser encontrada em (VÍDEO, 2001-2) (EBRAHIMI, 2000) (EBRAHIMI, 1997). Devido ao objetivo desta dissertação, optou-se por dar uma maior ênfase às ferramentas de codificação escalável de vídeo. No entanto, apresentam-se primeiro as ferramentas não-escaláveis de vídeo presentes na norma MPEG-4, organizadas segundo o tipo de dados que processam: forma, movimento e textura. A norma MPEG-4 possui ainda outras ferramentas que, por razões de espaço, não serão aqui apresentadas; as técnicas de resiliência a erros e a codificação *sprites*. Para mais detalhes, pode-se consultar (TALLURI, 1998) (LU, 2001).

3.2.3 Codificação de forma

A norma MPEG-4 é a primeira a incluir objetos de vídeo com forma arbitrária no seu modelo de representação e, por isso, inclui ferramentas para a codificação da informação de forma. Cada modelo VOP de um objeto de forma arbitrária é representado por 4 matrizes com os valores correspondentes às variáveis Y, U, V e A (*alpha*). Enquanto as 3 primeiras matrizes definem a textura de um objeto (luminância e crominâncias), os valores *alpha* definem a sua forma e o nível de transparência de cada *pixel* do VOP.

Na norma MPEG-4 Visual existem dois tipos de máscaras *alpha*: a máscara binária e a máscara multi-nível. No caso multi-nível, a gama de valores possíveis encontra-se entre 0 e 255 (inteiro de 8 bits); no caso binário, apenas dois valores são permitidos: *pixel* completamente transparente ('0') ou *pixel* completamente opaco ('1'), não havendo transparências intermediárias. O VOP apenas está definido para os *pixels* cujo valor *alpha* seja superior a zero (sendo transparente o restante dos *pixels* da sua *bounding box*).

3.2.3.1 Codificação de forma binária

A codificação de forma é feita ao nível do macrobloco, sendo pra cada macrobloco, os valores *alpha* independentemente codificados. Os macroblocos com valores *alpha* binários são denominados como BAB (*Binary Alpha Blocks*). A informação de forma binária é codificada através de uma técnica denominada como *Context Based Arithmetic Encoding* (CAE) (BRADY, 1997) (BRADY, 1999), baseada na codificação Inter e Intra de blocos BAB; esta codificação pode ser feita com ou sem perdas. Neste algoritmo, é calculado um contexto para cada *pixel*, baseado nos *pixels* vizinhos, que podem pertencer ao BAB atual ou ao BAB anterior, depois da compensação de movimento. Para BABs codificados no modo Intra, um contexto de 10 bits é construído para cada

pixel usando *pixels* pertencentes apenas ao BAB atual, tal como apresentado na **figura 3.8a**, onde ck = 0 para *pixels* transparentes e ck = 1 para *pixels* opacos.

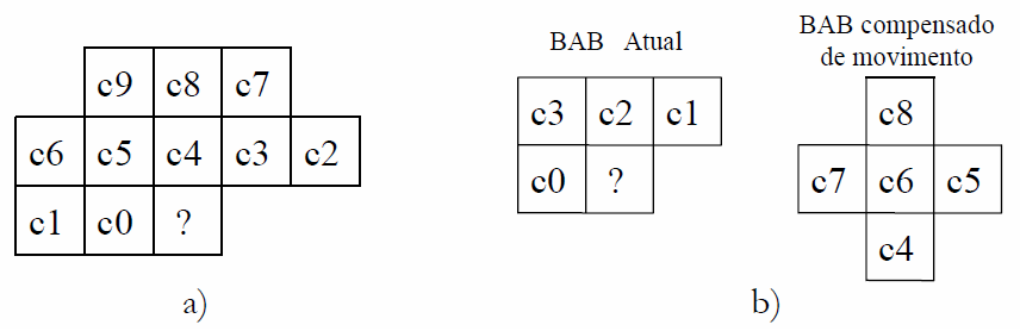


Figura 3.8: Definição do contexto utilizado pela técnica CAE: modo Intra (a) e modo Inter (b).

Para BABs codificados no modo Inter, a redundância temporal é explorada utilizando *pixels* que pertencem ao BAB atual e ao BAB anterior depois da compensação de movimento (**figura 3.8 b**). O contexto obtido serve para indexar uma tabela de probabilidades definida na norma que estima a probabilidade do *pixel* a se codificar ser 0 ou 1. O valor de probabilidade obtido e o contexto em questão são utilizados como parâmetros para o codificador aritmético, que gera os bits de saída.

3.2.3.2 Codificação de forma multi-nível

A codificação de informação de forma multi-nível segue a mesma estrutura de codificação da informação de forma binária, mas cada elemento da máscara *alpha* pode ter agora valores compreendidos entre 0 e 255 que representam o grau de transparência desse *pixel*. O primeiro passo da codificação consiste na extração da máscara binária a partir da máscara multi-nível. Cada valor da máscara binária é colocado a '0'se o valor correspondente da máscara multi-nível for também 0 e é colocado a '1'se o valor correspondente for diferente de 0. Esta máscara binária é codificada com ferramentas acima descritas para a codificação de forma binária. O valor da transparência de cada *pixel* é codificado utilizando a transformada DCT. A arquitetura deste tipo de codificação é ilustrada na **figura 3.9**.

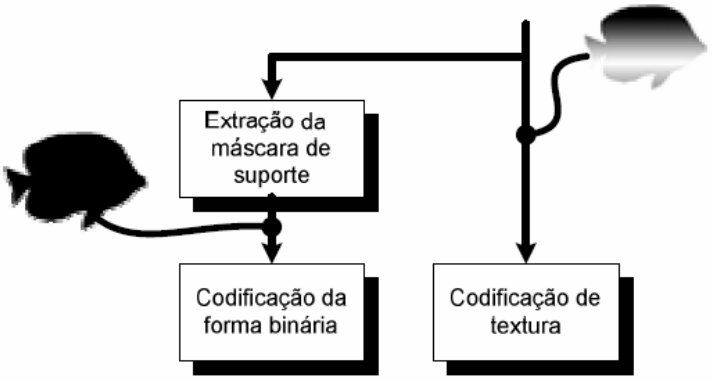


Figura 3.9: Diagrama de blocos da codificação de forma multi-nível.

Para mais detalhes sobre o processo de codificação e decodificação da informação de forma na norma MPEG-4 consultar (BRADY, 1999).

3.2.4 Estimação e compensação de movimento

Tal como nas normas anteriores, as técnicas de estimação e compensação de movimento também são utilizadas na codificação de textura no MPEG-4 e destinam-se a comprimir a informação de vídeo através da exploração da redundância temporal entre quadros ou VOPs. As principais diferenças vem do fato de técnicas típicas de estimação e compensação de movimento terem de ser adaptadas para lidar com os objetos de forma arbitrária, representados por seqüências de VOPs.

A estimação de movimento só é necessária para P-VOPs e B-VOPs, pois são estes tipos de VOPs que necessitam de predição temporal. Para os macroblocos transparentes não é efetuada estimação de movimento e para macroblocos opacos a estimação de movimento é efetuada de maneira semelhante às normas anteriores, ou seja, através do emparelhamento de macroblocos de dimensão 16x16 (na luminância e dos correspondentes blocos de dimensão 8x8 nas crominâncias), resultando um vetor de movimento por macrobloco, ou através do emparelhamento de blocos de dimensão 8x8 (na luminância e correspondentes crominâncias), resultando um vetor de movimento por cada um dos quatro blocos de luminância que fazem parte do macrobloco. Tal como as anteriores normas MPEG, o MPEG-4 não normaliza a técnica de estimação de movimento, nem o critério de emparelhamento, por isso não ser necessário para garantir a interoperabilidade. No entanto a norma MPEG-4 Visual introduziu novas e melhoradas ferramentas para compensação de movimento em relação às anteriores normas MPEG, entre elas:

- **Compensação de movimento a ¼ de *pixel*:** Algoritmo de compensação de movimento com uma resolução de ¼ *pixel*, em vez da resolução ½ *pixel* utilizada nas normas MPEG-1 e MPEG-2 Vídeo. Este algoritmo permite uma melhoria na precisão dos vetores de movimento e deste modo um decréscimo no erro de predição.
- **Compensação de movimento global GMC (*Global Motion Compensation*):** Apenas um conjunto de parâmetros de movimento é transmitido para o VOP a princípio, representando o movimento global do VOP. Estes parâmetros podem ser utilizados como alternativa ou complemento aos vetores de movimento estimados localmente para cada macrobloco. Em seqüências com um movimento global constante em grande parte da imagem (ex: translações), observa-se um decréscimo do *bit rate* usado para a informação local de movimento.
- **Modo direto:** Este algoritmo permite uma melhoria da eficiência de codificação para os quadros B, tal como são definidas na norma MPGE-2 Vídeo, baseada na abordagem da norma ITU-T H.263 para os quadros PB. Na compensação de movimento, dois VOPs podem ser utilizados como predição: o VOP imediatamente anterior do tipo I ou P, que corresponde a um instante de apresentação no passado (referência no passado), e o VOP imediatamente posterior do tipo I ou P, que corresponde a uns instantes de apresentação no futuro (referência no futuro). A norma MPEG-4 começa por incluir os 3 modos de predição definidos nas normas MPEG-1 e 2 ou seja os modos I, P e B, atrás definidos. Além destes 3 modos, o modo direto da norma H.263 é também definido na norma MPEG-4. No modo direto, os vetores de movimento para um dado quadro ou VOP são calculados a partir dos vetores de movimento do P-VOP de referência no futuro através de uma

operação de interpolação linear levando em conta o tempo decorrido entre os vários VOPs em questão, e do envio de um termo de correção ΔMV, tal como é mostrado na **figura 3.10**. Esta técnica permite reduzir o *bit rate* necessário para enviar os vetores de movimento associados a um B-VOP.

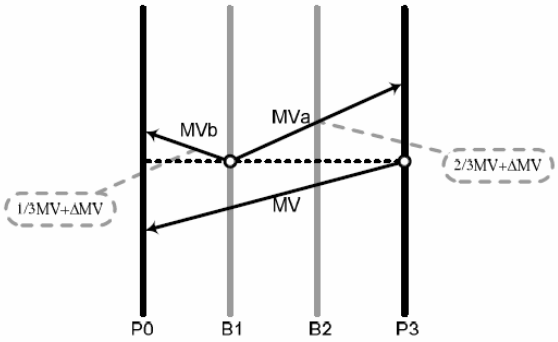


Figura 3.10: Modo direto da predição bidirecional.

- **Codificação preditiva dos vetores de movimento:** A codificação dos vetores de movimento para os P-VOPs e B-VOPs é diferencial, tomando-se como predição a mediana de 3 vetores de movimento vizinhos já transmitidos (dos macroblocos em cima, à esquerda e na diagonal superior esquerda do macrobloco corrente), utilizam-se nesse caso quatro vetores de movimento por macrobloco (um por cada bloco) para a compensação de movimento.

Para o caso de VOPs de forma arbitrária, o VOP de referência utilizado para a compensação de movimento possui uma forma arbitrária. Como alguns dos *pixels* no macrobloco de referência podem estar fora do VOP de referência (ex: o macrobloco contém *pixels* transparentes), é utilizada uma técnica de preenchimento (*padding*) para extrapolar o valor destes *pixels* a partir dos *pixels* que estão dentro do VOP de referência. Estes algoritmos de preenchimento tem de ser normativos, de forma a garantir que cada decodificador gere macroblocos de referência / predição idênticos. Os macroblocos opacos são codificados pelas ferramentas acima descritas; para os macroblocos fronteira e transparentes foram desenvolvidas ferramentas de preenchimento descritas a seguir.

3.2.4.1 Padding para macroblocos de fronteira

Os *pixels* transparentes dos macroblocos fronteira são preenchidos através de uma concatenação de operações de preenchimento (*padding*) horizontal e vertical, nesta ordem. A **figura 3.11** ilustra este processo para um bloco de 8x8 *pixels*. Os *pixels* na fronteira do VOP tem os valores “A”, “B”, “C”, etc. Em primeiro lugar, os *pixels* na fronteira são repetidos horizontalmente, para esquerda ou para direita. No caso de existir 2 valores na fronteira, é calculada uma média e preenchidos os *pixels* transparentes com esse valor (ilustrado na **figura 3.11** para a terceira linha de baixo para cima). O mesmo processo é repetido verticalmente, tendo em conta os *pixels* que foram preenchidos no passo anterior. No final, todos os *pixels* foram preenchidos e não existe nenhum *pixel* transparente.

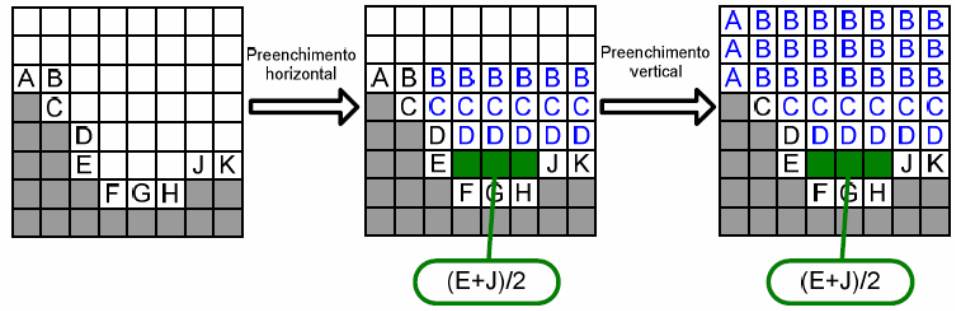


Figura 3.11: Processo de preenchimento (*padding*) para macroblocos fronteira.

3.2.4.2 Padding para macroblocos transparentes

Os macroblocos transparentes (que estão completamente fora do VOP) são preenchidos de uma forma diferente, com uma técnica denominada *extended padding*. Estes macroblocos são preenchidos através da repetição dos valores na fronteira de um dos macroblocos de fronteira vizinhos. Se existe mais do que um macrobloco fronteira, apenas um deles é escolhido, de acordo com as prioridades definidas na **figura 3.12**.

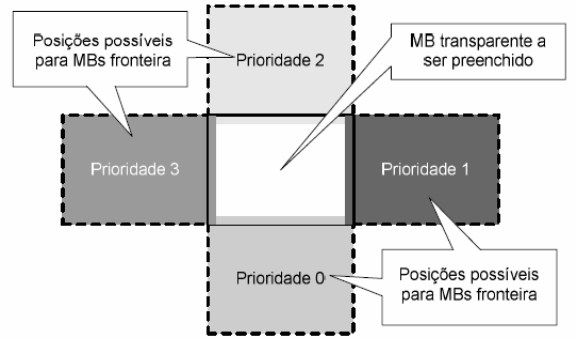


Figura 3.12: Processo de preenchimento para macroblocos transparentes.

No caso de existir mais de um macrobloco fronteira na vizinhança, o macrobloco transparente é preenchido repetindo os valores dos pixels na fronteira horizontal ou vertical de acordo com a prioridade estabelecida.

3.2.4.3 Compensação de movimento com sobreposição

A norma MPEG-4 Visual também suporta compensação de movimento com sobreposição (como na norma H.263) (OBMC – *Overlapped Block Motion Compensation*). Neste caso, para cada bloco de um macrobloco, são utilizados para a compensação de movimento 3 vetores de movimento (do próprio bloco ou macrobloco) e dos seus 2 vizinhos mais próximos. A predição para o bloco (ou macrobloco) em questão é obtida através da média das predições dadas pelos vários vetores candidatos, ponderando cada parcela de acordo com pesos pré-definidos e que depende da posição de cada *pixel* dentro do bloco (ou macrobloco) com os pesos pré-definidos e que depende da posição de cada *pixel* dentro do bloco (ou macrobloco). No entanto, esta ferramenta não faz parte de nenhum "tipo de objeto" e logo perfil da norma MPEG-4 Visual e por este motivo não pode ser utilizada neste momento por nenhuma aplicação em conformidade com a norma.

3.2.5 Codificação de textura

A informação de textura de um VOP é constituída por uma componente de luminância (Y) e duas componentes de crominância (U e V). No caso de um I-VOP, a informação de textura é diretamente codificada, enquanto que no caso de um P ou B-VOP a informação de textura reside no erro de predição depois de se efetuar a compensação de movimento. Em ambos os casos, o processo de codificação é ilustrado na **figura 3.13**. Para codificar a informação de textura é utilizada a transformada DCT (*Discrete Cosine Transform*) aplicada sobre blocos de 8x8 *pixels*, já conhecida das normas anteriores, mas adaptada de forma a lidar também com os objetos de forma arbitrária.

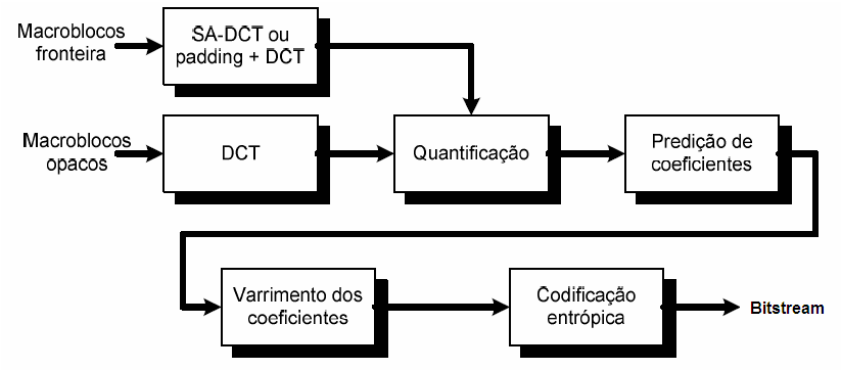


Figura 3.13: Processo de codificação da textura de um VOP.

Como se pode observar na **figura 3.13**, a norma MPEG-4 Visual trata de forma diferente os macroblocos fronteira e os macroblocos opacos (os macroblocos transparentes não tem informação de textura). No caso dos macroblocos opacos, a DCT é aplicada diretamente às três componentes de textura (YUV); no caso dos macroblocos de fronteira, dois métodos são permitidos:

- **Preenchimento (*padding*) + DCT:** Os *pixels* transparentes dentro do macrobloco fronteira são preenchidos com valores de textura de acordo com alguma regra ou algoritmo de preenchimento. Qualquer valor é permitido, uma vez que depois da decodificação da forma, o decodificado possui a informação sobre quais *pixels* são transparentes ou não. Depois deste processo de preenchimento, os macroblocos fronteira são codificados do mesmo modo que os macroblocos opacos, ou seja, usando a DCT. A forma como se preenchem os valores transparentes não é normativa e é deixada ao critério do codificador. Existem contudo várias estratégias (SHEN, 1999) (VÍDEO, 2001) que tentam minimizar o *bit rate* para codificar o conteúdo dos macroblocos fronteira, tanto para o modo Intra como para o modo Inter.

- **SA-DCT (*Shape Adaptive DCT*)**: Este método codifica e transmite apenas os valores dos *pixels* opacos dentro dos macroblocos de fronteira. Este algoritmo é baseado em um conjunto pré-definido de funções base da DCT (SIKORA, 1995) e é constituído por 4 passos: 1) deslocamento vertical para o topo; 2) DCT unidimensional na vertical; 3) deslocamento horizontal; e 4) DCT unidimensional na horizontal. A **figura 3.14** mostra um exemplo de aplicação da AS-DCT para um bloco de 8x8 *pixels*. Após isto, os coeficientes são varridos em zig-zag, omitindo os coeficientes não definidos, e finalmente codificados entropicamente. Uma variação deste algoritmo, referido como ΔAS-DCT (KAUFF, 1997), pode também ser utilizada; esta

variação consiste em adicionar alguns passos de pré e pós-processamento ao algoritmo AS-DCT melhorando assim sua eficiência.

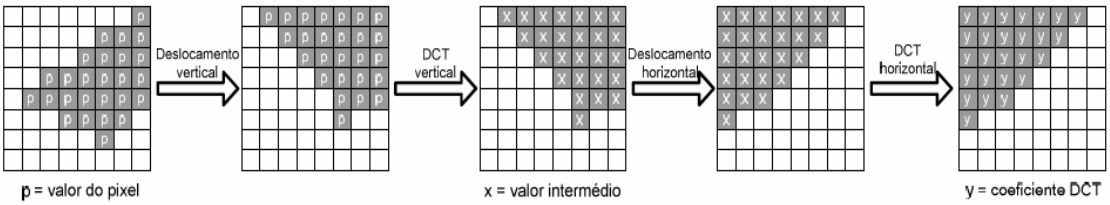


Figura 3.14: Exemplo de aplicação da transformada SA-DCT.

Para reduzir o *bit rate* e eliminar a informação irrelevante, os coeficientes DCT obtidos são quantificados. Existem dois métodos de quantificação para vídeo especificados na norma MPEG-4 Visual:

- **Quantificação MPEG-2:** O primeiro método é muito semelhante ao especificado na norma MPEG-2 Vídeo, utilizando uma de duas matrizes de quantificação (uma para o modo Intra e outra para o Inter) para definir o passo de quantificação para cada coeficiente DCT. O codificador pode utilizar as matrizes previamente definidas na norma ou transmitir novas matrizes ao decodificador. Este método permite que o codificador tenha em conta as características do sistema visual humano, pois cada passo de quantificação (ou peso) na matriz pode ser ajustado individualmente.
- **Quantificação H.263:** O segundo método é o especificado na norma H.263. Este método é menos complexo e mais fácil de implementar, pois utiliza o mesmo passo de quantificação para todos os coeficientes.

O codificado escolhe qual dos dois métodos de quantificação pretende utilizar para a codificação do objeto de vídeo em questão e sinaliza essa opção ao decodificador. O coeficiente DC de um bloco 8x8 codificado no modo Intra (representa a luminância ou crominância média) é tratado de uma forma diferente, sendo sempre quantificado com um passo de quantificação fixo (normalmente 8).

Para alguns dos coeficientes DC e AC pertencentes a blocos vizinhos existe uma forte dependência estatística, o valor de um coeficiente pode ser predito a partir do valor do coeficiente da mesma posição espacial mas pertence a um bloco vizinho. Este fato é explorado na codificação de textura da norma MPEG-4 Visual através do módulo mostrado na **figura 3.13** como "Predição de coeficientes". No entanto, este tipo de predição só é utilizado em macroblocos Intra. O processo é ilustrado na **figura 3.15**. Para o coeficiente DC, do bloco X, a predição pode ser feita a partir dos coeficientes DC dos blocos A ou C. Os coeficientes AC na primeira linha são preditos a partir dos coeficientes correspondentes do bloco C e os coeficientes AC na primeira coluna são preditos a partir dos coeficientes correspondentes do bloco a esquerda (A).

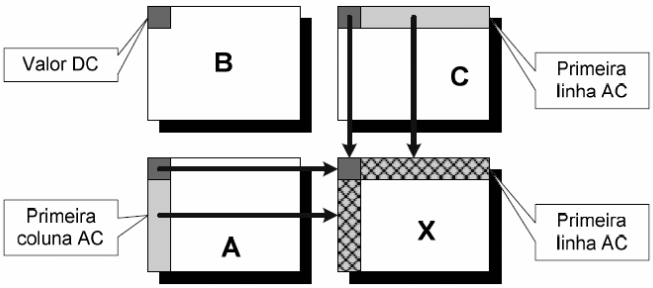


Figura 3.15: Coeficientes candidatos para a predição dos coeficientes AC e DC.

Em seguida, os coeficientes DCT preditos são convertidos em um vetor unidimensional com o objetivo de gerar símbolos para o codificador entrópico. A seqüência de varredura assegura sempre a transmissão prioritária dos coeficientes mais relevantes, independentemente do número de coeficientes enviados, pois a maior parte a energia está concentrada no canto superior esquerdo (baixas freqüências) de cada bloco. A norma MPEG-4 suporta além da popular varredura em zig-zag utilizado em normas anteriores mais dois modos de varredura adicionais para estruturas de imagem em que exista uma predominância de freqüências horizontais ou verticais. O resultado da varredura consiste em um vetor em que os coeficientes mais significativos estão no princípio e os zeros no fim. Esta característica e a estatística do sinal são exploradas pela ultima etapa de processamento: o codificador entrópico de Huffman.

3.2.6 Perfis e níveis

A norma MPEG-4 contém um grande número de ferramentas associadas às inúmeras funcionalidades desejadas, o que a torna bastante complexa e difícil de implementar na totalidade. Por este motivo, existe a necessidade de limitar a complexidade dos decodificadores que estão em conformidade com a norma, pois não se deve esperar que um decodificador tenha que implementar todas as ferramentas da norma com as respectivas capacidades máximas, por exemplo, em termos de *bit rate* e/ou resolução, devido a grande complexidade de implementação e também por muitas ferramentas serem desnecessárias em algumas classes de aplicações.

A necessidade de limitar a complexidade dos decodificadores levou a definição de subconjuntos de ferramentas e a definição de limites para os parâmetros de codificação. Para minimizar a complexidade, garantindo ainda assim a interoperabilidade entre terminais em um dado domínio de aplicação, a norma MPEG-4 regulamentou determinados subconjuntos relevantes de ferramentas, para que os codificadores possam produzir *bitstreams* de acordo com esta especificação e assim encontrar decodificadores menos complexos, mas mesmo assim com as capacidades necessárias para decodificação.

Para limitar a complexidade de implementação, a norma MPEG-4 utiliza um mecanismo baseado em 3 conceitos principais – tipos de objeto, perfis e níveis – de acordo com as seguintes definições:

- **Tipo de objeto:** Define a sintaxe do *bitstream* para um dado objeto que representa uma entidade com significado na cena audiovisual. Estabelece uma lista de ferramentas de codificação que podem ser utilizadas para a codificação deste objeto. Definem-se tipos de objeto para objetos de áudio e vídeo.
- **Perfil:** Define o conjunto de ferramentas que podem ser utilizadas num determinado terminal, por exemplo, para decodificar uma cena MPEG-4. Na norma MPEG-4, existem perfis de áudio, visuais, gráficos de descrição de cena, de descritores de objeto e MPEG-J. Os perfis de áudio e visuais são definidos com base nos tipos de objeto, especificando quais os tipos de objeto que podem ser usados para codificar os objetos de uma cena codificada de modo conforme com um dado perfil. Os perfis destinam-se a limitar o conjunto de ferramentas que é necessário implementar no

decodificador para garantir interoperabilidade entre terminais que só utilizem uma parte das ferramentas especificadas na norma.

- **Nível:** Especifica as restrições impostas aos perfis acima descritos, ou seja, as ferramentas por eles utilizadas, através de limitações impostas a alguns parâmetros relevantes. Os níveis especificam limites para a complexidade computacional que é exigida definindo limites máximos para os codificadores ao produzir os *bitstreams* e limites mínimos para os decodificadores.

Resumindo, uma dada combinação de perfil e nível (denominada por perfil@nivel) estabelece um limite superior para a complexidade do *bitstream* criado no codificador e um limite inferior para as capacidades do decodificador. Como a norma MPEG-4 é baseada em objetos, as cenas são compostas por vários objetos audiovisuais e um determinado perfil@nivel especifica a complexidade máxima para a totalidade dos objetos presentes na cena e não para cada objeto individualmente.

3.3 Codificação escalável de textura (VCT)

Nos últimos anos, as indústrias multimídias, de telecomunicações e de animação computadorizada assistiram a um aumento do interesse pelos serviços multimídia interativos. A eficiência dos esquemas de codificação utilizados para a compressão de conteúdo multimídia, a capacidade de suportar vários níveis de transparência e a flexibilidade para codificar múltiplos níveis de detalhe para uma cena em um único *bitstream* são essenciais para o desenvolvimento destes novos serviços. Para responder a essas necessidades, o grupo ISO MPEG desenvolveu um método de codificação de textura denominado VTC (*Visual Texture Coding*), incorporado na norma MPEG-4 Visual, para suportar aplicações deste tipo (codificação de texturas para mapear em modelos 3D). O método VTC baseia-se na transformada DWT e na codificação das correspondentes sub-bandas com *zero-trees*. Esta ferramenta oferece as seguintes capacidades (KOENEN, 2001):

- **Compressão eficiente para uma ampla gama de *bit rates*:** A capacidade de comprimir eficientemente texturas, imagens fixas e documentos em um conjunto amplo de *bit rates* utilizando uma única ferramenta permite simplificar o processo de criação de conteúdos multimídia.

- **Codificação de objetos com forma arbitrária**: Uma vez que um objeto de vídeo MPEG-4 pode possuir uma forma arbitrária, a ferramenta SA-DWT (*Shape Adaptative Discrete Wavelet Transform*) foi normalizada para garantir uma codificação eficiente de texturas com forma arbitrária, tal como o faz a técnica SA-DCT para objetos de vídeo.

- **Escalabilidade espacial e de SNR (qualidade) com elevada granularidade:** Esta capacidade torna possível a criação de texturas com múltiplas resoluções e qualidades, essenciais para aplicações de mapeamento 2D e 3D, comércio eletrônico, documentos compostos, entre outras, a partir de um único *bitstream*.

- **Transmissão robusta em canais sujeitos a erros:** Para permitir a transmissão em canais com erros, o método de codificação VTC adotou ferramentas de resiliência a erros. Dependendo do canal de transmissão,

estas ferramentas podem ser utilizadas para substituir ou complementar a codificação de canal de forma a melhorar a qualidade da imagem visualizada.

Para que as capacidades acima descritas pudessem ser alcançadas, o método VTC adotou uma estrutura de codificação baseada nas seguintes técnicas:

- Transformada DWT bi-ortogonal e o algoritmo de codificação baseado em *zero-trees*.
- Esquema de codificação SA-DWT para codificação de objetos com forma arbitrária.
- Três modos de quantificação e dois modos de varredura dos coeficientes da DWT para permitir diferentes níveis de granularidade para a escalabilidade espacial e de qualidade.
- Um esquema de empacotamento capaz de oferecer robustez a erros de transmissão.
- Um esquema de divisão da imagem (*tiling*) para reduzir os requisitos de memória do codificador e decodificador.

A arquitetura do codificador é apresentada na **figura 3.16** assumindo que os dados de entrada são três objetos visuais com forma arbitrária (duas crianças e uma bola). A forma e a textura dos objetos são codificados separadamente: no módulo de codificação escalável da forma e no módulo de codificação escalável da textura para objetos com forma arbitrária, respectivamente. A informação de saída de ambos os módulo é combinada com três *bitstreams* elementares que corresponde a cada objeto presente na cena. Por fim, os *bitstreams* são multiplexados de modo formando o *bitstream* final enviado para o canal.

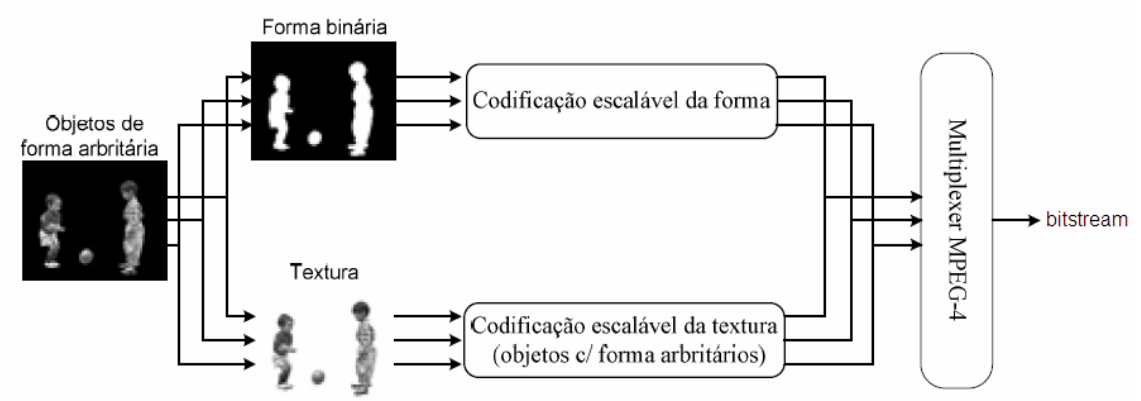


Figura 3.16: Diagrama de blocos do codificador MPEG-4 VTC.

3.3.1 Codificação de textura com forma retangular

O sistema de codificação MPEG-4 VTC baseia-se na transformada DWT e na codificação em *zero-trees*. Tal como é ilustrado na **figura 3.17**, a codificação VCT inclui três módulos principais: a transformada DWT, o módulo de quantificação e codificação em *zero-trees* e o codificador entrópico.

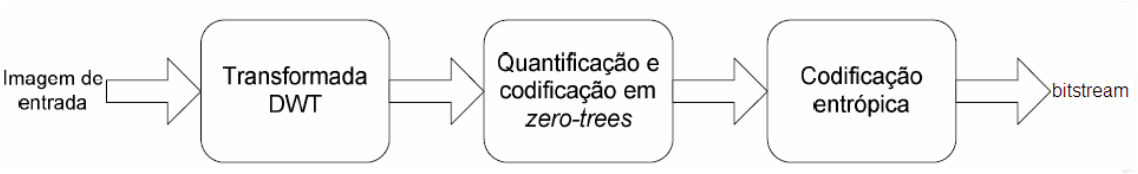


Figura 3.17: Codificação VTC para objetos retangulares.

A transformada DWT é aplicada a imagem de entrada para se obter uma representação em sub-bandas com várias resoluções. Na norma MPEG-4 Visual, uma transformada DWT bi-ortogonal (PURI, 2000) foi definida para ser usada por omissão; no entanto, a sintaxe da norma suporta qualquer transformada DWT, o codificador pode definir os filtros a serem utilizados pelo decodificador. Os filtros definidos podem possuir coeficientes inteiros, permitindo que o mesmo sistema de codificação seja utilizado para a codificação sem perdas. O *bitstream* pode ser escalável desde *bit rates* muito baixos até a codificação sem perdas, sempre de uma forma contínua.

A sub-banda com a resolução espacial mais baixa (normalmente referida como sub-banda DC) é codificada separadamente das restantes sub-bandas (AC). Os coeficientes DC são uniformemente quantificados e adaptativamente preditos a partir dos coeficientes vizinhos. Tal como é ilustrado na **figura 3.18a**, o coeficiente X é diferentemente codificado enviando um erro de predição calculado a partir de três coeficientes vizinhos já quantificados (A, B e C) da seguinte forma: se |A-B| for menor que |B-C| então X' = X-C; caso contrário X' = X-A. Finalmente, o erro de predição é codificado com um codificador aritmético adaptativo.

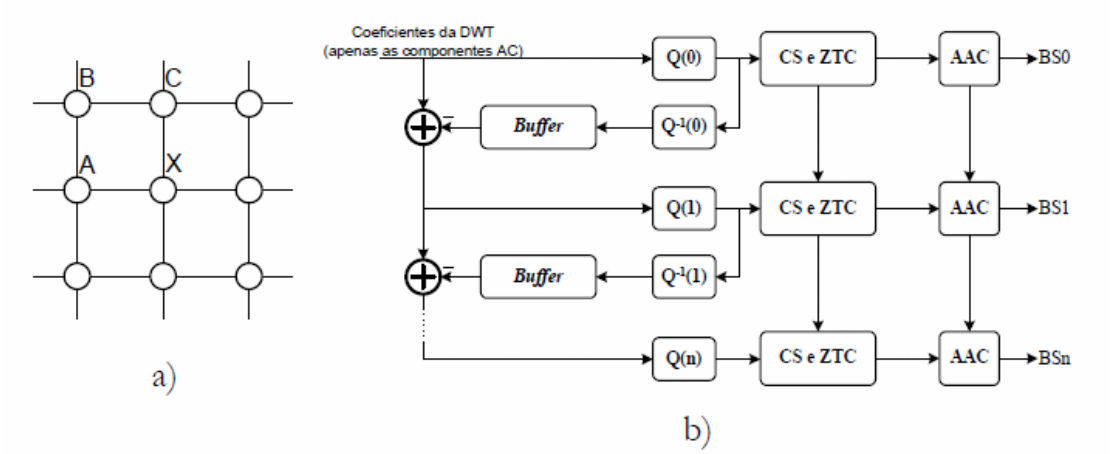


Figura 3.18: a) Predição de coeficientes DC; b) Modo com múltiplos passos de quantificação.

As sub-bandas AC são codificadas utilizando uma combinação de algoritmos, variantes do algoritmo EZW (*Embedded Zero-Tree Wavelet*) original (SHAPIRO, 1993): o algoritmo ZTE (*Zero-Tree Entropy*) (MARTUCCI, 1997), o algoritmo MZTE (*Multiscale Zero-Tree Entropy*) (SODOGAR, 1999) e o algoritmo PEZW (*Predictive Embedded Zero-Tree Wavelet*) (LIANG, 1997). Todos os algoritmos possuem um quantificador (implícito ou explícito), varredura dos coeficientes e codificador aritmético.

3.3.1.1 Quantificação

Baseando-se nestes três algoritmos, e garantindo uma flexibilidade acrescida em termos de eficiência e complexidade, o método MPEG-4 VTC suporta três modos de quantificação:

- **Passo de quantificação único:** É utilizado um único passo de quantificação para todos os coeficientes DWT.
- **Múltiplos passos de quantificação:** São utilizados múltiplos passos de quantificação de Q (0) a Q (n) nas várias etapas (**figura 3.18**). Os coeficientes DWT são primeiramente quantificados com passo de quantificação Q(0), logo após isso são varridos pelo módulo CS (*Coefficient Scanning*), codificados em *zero-trees* pelo módulo ZTC (*Zero-Tree Coding*) e codificados aritmeticamente pelo módulo AAC (*Adaptative Arithmetic Coding*). Os coeficientes quantificados são também desquantificados e subtraídos dos coeficientes originais. O erro de quantificação é quantificado novamente e processado pelos quantificadores restantes (Q(1) a Q(n)). O *bitstream* irá consistir em uma combinação dos *bitstreams* de cada etapa (BS0 a BSn na **figura 3.18b**) e fornece $n + 1$ camadas de escalabilidade de qualidade.
- **Quantificação bi-nível:** Igual ao modo de quantificação do algoritmo EZW, pois utiliza uma aproximação sucessiva dos coeficientes através de limiares sucessivamente decrescentes. O limiar é inicializado com o valor do coeficiente com amplitude maior e, em cada varredura, este valor é dividido por dois. Por este motivo, este modo é também referido como modo de quantificação implícita.

3.3.1.2 Varredura

O próximo passo de codificação consiste na varredura dos coeficientes DWT, disponibilizando a norma MPEG-4 Visual duas ordens de varredura:

- **Árvore:** Ordem igual a utilizada no algoritmo EZW, onde todos os coeficientes de uma árvore são codificados antes de se codificar a próxima árvore (**figura 3.19**).

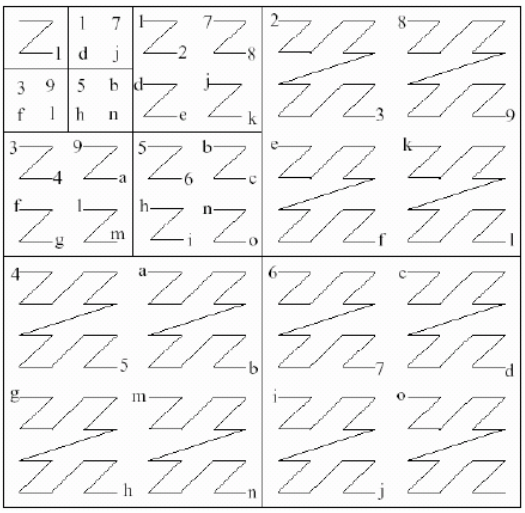


Figura 3.19: Ordem de varredura em árvore

- **Sub-banda a sub-banda:** Todos os coeficientes de uma sub-banda são codificador antes de se codificar a próxima sub-banda (**figura 3.20**). As sub-bandas são varridas em uma ordem de resolução espacial crescente, da sub-banda com resolução espacial mais baixa para a mais alta.

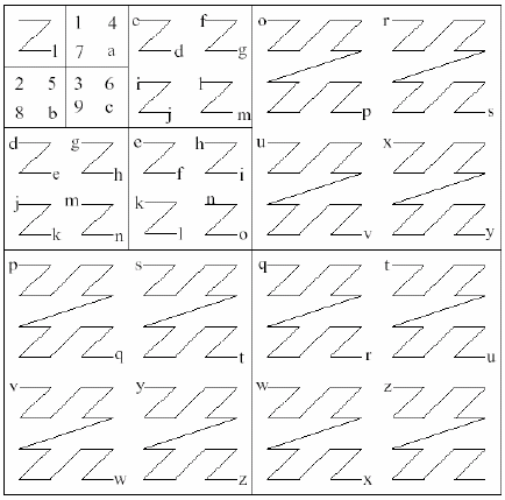


Figura 3.20: Ordem de varredura em sub-banda (a ordem é indicada pela seqüência 1...9, a...z)

As duas ordens de varredura servem para cumprir vários objetivos. Por exemplo, a varredura em árvore requer menos memória para a codificação e decodificação da imagem; por outro lado, a varredura sub-banda a sub-banda pode codificar os coeficientes com um menor atraso.

3.3.1.3 Codificação com zero-trees

O passo seguinte consiste na codificação com *zero-trees*. A estrutura em *zero-trees* explora a correlação entre um coeficiente em uma escala "grosseira" e seus descendentes em escalas mais finas (**figura 3.21**), através de uma estrutura em árvore (*quad-trees*). O conjunto de símbolos utilizado pela codificação com *zero-trees* é muito semelhante (mas não igual) ao utilizado pelo codificador EZW:

- **ZTR (*zero-tree root*):** Representa um nó da árvore cujo coeficiente é zero e todos os seus descendentes são zero.
- **VZTR (*valued zero-tree root*):** Representa um nó da árvore cujo coeficiente é diferente de zero e todos os seus descendentes são zero.
- **IZ (*isolated zero*):** Representa um nó da árvore cujo coeficiente é zero mas nem todos os seus descendentes são zero.
- **VAL (*isolated non-zero value*):** Representa um nó da árvore cujo coeficiente é diferente de zero mas nem todos os seus descendentes são zero.

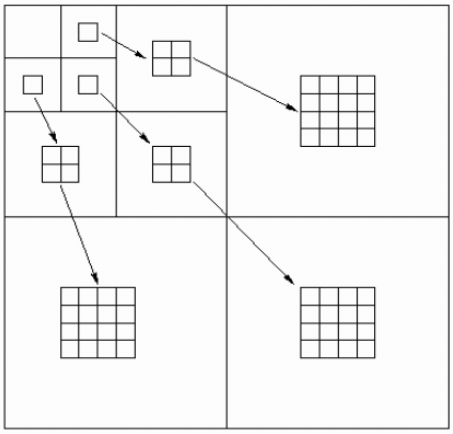


Figura 3.21: Exemplo de estrutura em *zero-trees*

A principal diferença em relação ao codificador EZW é a introdução do símbolo VZTR que permite representar eficientemente os descendentes zero de um nó diferente de zero. No algoritmo original EZW é necessário enviar seis símbolos para codificar este tipo de árvore. Este novo conjunto de símbolos permite uma melhoria na eficiência de codificação (maiores detalhes em Liang (1997)). Além da diferença, a codificação diferencial é utilizada no modo de quantificação com múltiplos passos. Os coeficientes quantificados com um determinado passo de quantificação (ex: Q(1)) são refinados a partir dos coeficientes do nível anterior (Q(0)) e só depois é que são codificados com *zero-trees* (ver (SODOGAR, 1999) para mais detalhes).

A ultima etapa é a codificação entrópica dos símbolos gerados. No método MPEG-4 VTC utiliza-se um codificador aritmético adaptativo para codificar os símbolos gerados pelo codificador de *zero-trees*, os valores dos coeficientes diferentes de zero quantificados e os valores refinados no modo de quantificação com múltiplos passos (detalhes em (PURI, 2000)).

3.3.2 Codificação de textura com forma arbitrária

A codificação SA-DWT (KATATA, 1997) (*Shape Adaptative Discrete Wavelet Transform*) é utilizada para a codificação de texturas com forma arbitrária. A única diferença em relação a codificação com *zero-trees* apresentada anteriormente é o tratamento das regiões que se encontram na fronteira das texturas com forma arbitrária. A codificação SA-DWT garante que o número de coeficientes para codificar seja igual ao número de *pixels* opacos da região com forma arbitrária. Além disso, esta transformada mantém a correlação espacial e a semelhança entre coeficientes de sub-bandas diferentes, tal como a transformada DWT. Para lidar com objetos de forma arbitrária, foi também necessário definir algumas extensões ao algoritmo de codificação com *zero-trees*, para garantir uma eficiência de codificação alta nas sub-bandas AC que possuem coeficientes a ignorar, que não contribuem fortemente para a textura do objeto.

Na codificação SA-DWT, a transformada DWT é aplicada a cada segmento de linha e coluna de *pixels* consecutivos, de uma forma semelhante à transformada SA-DCT. Extensões simétricas de estratégias de sub-amostragem foram incorporadas de maneira a ter em consideração o comprimento e posição de começo do segmento de linha ou de coluna a ser transformado (mais detalhes podem ser encontrados em (LI, 2000)).

3.3.3 Escalabilidade espacial e de qualidade

A ordem de varredura e o modo de quantificação estão intimamente ligados com as propriedades escaláveis do *bitstream*. Por exemplo, quando se utiliza um passo de quantificação único em conjunto com o modo de varredura em árvore, o *bitstream* gerado não é escalável, nem em qualidade (SNR), nem espacialmente. A granularidade do *bitstream* depende também do método de quantificação utilizado: o modo de quantificação bi-nível permite uma granularidade muito fina na qualidade enquanto no modo de quantificação com múltiplos passos a granularidade depende do número de passos utilizados. Como geralmente não se utiliza um número de passos muito elevado (no método MPEG-4 VTC o número máximo de passos é 31), o *bitstream* possui uma granularidade menor em relação ao modo de quantificação bi-nível. A **tabela 3.1** ilustra o número de níveis de granularidade que o codificador pode selecionar, através da combinação de diferentes técnicas de quantificação e varredura dos coeficientes. *Max_wavelet* é o número de sub-bandas geradas pela transformada DWT e *max_snr* é o número máximo de divisões por 2 (ou equivalentemente o número máximo de planos de bits) que cada coeficiente resultante da transformada DWT pode sofrer; este valor depende do passo de quantificação utilizado.

Tabela 3.1: Níveis de escalabilidade espacial e de qualidade possíveis com o método MPEG-4 VTC.

Modos de quantificação	***Ordem de varredura***	***Níveis de escalalilidade espacial***	***Níveis de escalabilidade de qualidade***
Passo de quantificação único	Árvore	1	1
	Sub-banda a sub-banda	max_wavelet	1
Múltiplos passos de quantificação	Árvore	1	[1,31]
	Sub-banda a sub-banda	max_wavelet	[1,31]
Quantificação bi-nível	Árvore	1	[1, max_snr]
	Sub-banda a sub-banda	max_wavelet	[1, max_snr]

A norma MPEG-4 Visual optou por normalizar três modos de quantificação VTC, não só para permitir vários níveis de escalabilidade, mas também para garantir uma maior flexibilidade entre três fatores: eficiência, escalabilidade e complexidade computacional. O modo do passo de quantificação único (1) possui uma complexidade baixa e uma alta eficiência, mais possui uma escalabilidade limitada. O método de quantificação bi-nível (2) possui flexibilidade máxima, ou seja, muitos níveis de escalabilidade em múltiplas resoluções; no entanto, possui uma complexidade elevada. O método com múltiplos passos de quantificação é um compromisso entre (1) e (2), pois possui um nível pré-determinado de escalabilidade, mas possui uma complexidade inferior ao método (2).

Em relação ao tipo de escalabilidade, o método MPEG-4 VTC suporta dois tipos: escalabilidade espacial e de qualidade. A escalabilidade espacial e de qualidade podem ser aplicadas a objetos com forma arbitrária através da combinação de diferentes técnicas de quantificação e ordens de varredura. A **figura 3.22** ilustra os dois modos de escalabilidade suportados pelas ferramentas MPEG-4 VTC.

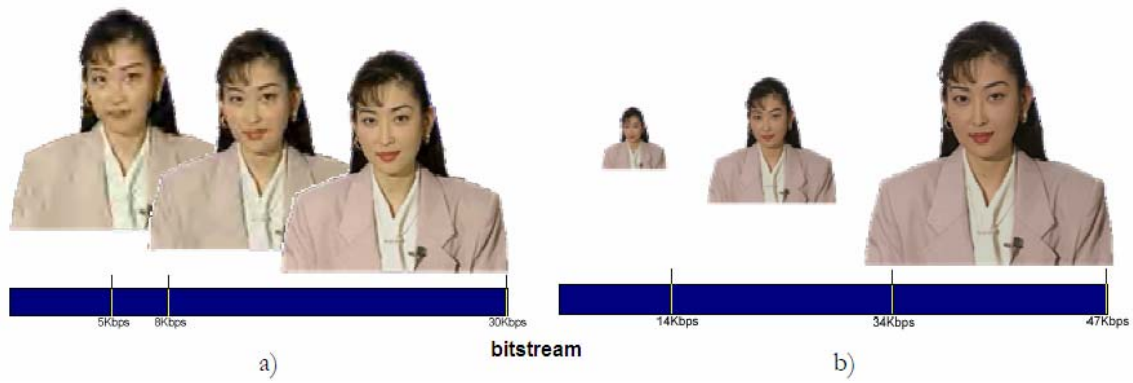


Figura 3.22: Codificação escalável de textura: a) SNR; b) espacial.

Se a ordem de varredura dos coeficientes for árvore, pode obter-se um *bitstream* escalável na qualidade, mas se a ordem de varredura dos coeficientes for de sub-banda em sub-banda, o *bitstream* é escalável em termos de resolução espacial. A combinação da escalabilidade espacial e de qualidade pode ser obtida mantendo a ordem de varredura em sub-bandas, mas escolhendo um dos dois modos de quantificação: a quantificação em múltiplos passos ou a quantificação bi-nível, com este último modo permitindo uma granularidade mais fina. Quando se utilizam os métodos de quantificação com um único ou múltiplos passos de quantificação, é importante salientar que o(s) passo(s) de quantificação definidos pelo codificador podem ser alterados entre níveis de escalabilidade, permitindo assim definir o *bit rate* que cada camada de escalabilidade possui.

Para oferecer escalabilidade de qualidade para objetos com forma arbitrária, o método SA-DWT explicado anteriormente pode ser utilizado para a codificação de objetos com a forma retangular. Contudo, a informação de forma não escalável e para oferecer escalabilidade espacial, a informação de forma necessita ser também escalável espacialmente. O método de codificação escalável de forma baseia-se no algoritmo SISC (*Scan Interleaving based Shape Coding*) semelhante ao utilizado para a escalabilidade espacial em seqüências de vídeo e usa ainda um codificador aritmético.

3.4 Codificação escalável de vídeo com baixa granularidade

A norma MPEG-4 Visual suporta um modo de codificação escalável de vídeo com baixa granularidade. Para se criar um *bitstream* escalável neste modo, cada objeto é codificado em duas ou mais camadas: uma camada base codificada de modo independente e uma ou mais camadas superiores, codificadas como melhoramentos em relação as camadas anteriores. Para permitir um fácil acesso a cada camada de codificação do objeto, a norma MPEG-4 Visual define uma sintaxe que permite uma fácil identificação de cada camada, através do nível hierárquico VOL (*Vídeo Object Layer*).

Neste modo escalável, dois tipos de escalabilidade são suportados: a escalabilidade espacial e a escalabilidade temporal, o que permite obter várias resoluções espaciais e

temporais para o mesmo objeto de vídeo, sem que isso implique a codificação repetida do mesmo conteúdo. Como os objetos de vídeo podem ser utilizados de uma forma independente, existe a possibilidade de decodificar um número de objetos limitado do *bitstream* total correspondente a cena: este tipo de escalabilidade é conhecido por escalabilidade de conteúdo ou objeto. Por exemplo, o receptor pode escolher só receber os objetos presentes na cena a se visualizar com maior importância, eliminando o fundo e os objetos menos importantes até atingir o *bit rate* desejado. Outra forma de escalabilidade de conteúdo consiste em selecionar as camadas VOL de cada objeto de maneira que o *bit rate* total de todos os objetos da cena, cumpra determinados requisitos. As ferramentas de escalabilidade espacial e temporal da norma MPEG-4 Visual são semelhantes as ferramentas correspondentes da norma MPEG-2 Vídeo. No entanto, o modo SNR de elevada granularidade (FGS) utiliza uma estratégia diferente.

3.4.1 Escalabilidade espacial

Uma das diferenças entre a norma MPEG-2 Vídeo e a norma MPEG-4 Visual é que a primeira permite escalabilidade espacial para vídeo entrelaçado e inclui um processo de desentrelaçamento no filtro de sobre-amostragem para permitir vídeo entrelaçado na camada base e vídeo progressivo na camada superior, enquanto que a norma MPEG-4 apenas permite a codificação escalável de um vídeo progressivo. Como a norma MPEG-4 permite codificar objetos de vídeo com uma forma arbitrária, a escalabilidade espacial pode ser aplicada a quadros retangulares e a objetos de vídeo com forma arbitrária. Neste caso são necessários dois tipos de escalabilidade:

- **Escalabilidade da forma:** A codificação da forma é feita de acordo com o algoritmo SISC (*Scan Interleaving based Shape Coding*) aplicado a informação de forma binária. De acordo com este algoritmo, a forma é decomposta em duas ou mais camadas através de um processo de sub-amostragem e seleção das linhas a codificar. A camada base é codificada de acordo com a codificação não escalável da forma. As camadas superiores são codificadas com um codificador aritmético baseado em contextos (CAE) que explora a redundância temporal (tal como o codificador da camada base) e a redundância espacial entre as diversas camadas espaciais (SON, 2000).
- **Escalabilidade de textura:** A textura da camada base tem de ser preenchida (*padded*) antes de ser filtrada (para obter a sobre-amostragem). O processo de preenchimento é igual ao para o caso de estimação e compensação de movimento e o processo de filtragem é igual ao da norma MPEG-2 Vídeo.

Ao contrário da norma MPEG-2 em que é utilizada uma predição espaço-temporal pesada, na norma MPEG-4 define-se um conjunto de regras para a codificação de I, P e B-VOPs na camada superior:

- **I-VOPs:** Nenhuma predição espacial é realizada para I-VOPs que pertencem as camadas superiores. Os I-VOPs são codificados em qualquer referência a outro VOP, são codificados como se pertencessem a camada base.
- **P-VOPs:** Os P-VOPs são preditos apenas a partir do VOP correspondente na camada base filtrada sem qualquer uso de predição temporal dentro da camada superior.

- **B-VOPs:** Os B-VOPs possuem duas referências temporais. Uma delas é o VOP correspondente da camada base sobre-amostrado (filtrado) e a outra é o VOP mais recentemente decodificado na camada superior.

3.4.2 Escalabilidade temporal

A escalabilidade temporal da norma MPEG-4 Visual possui muitas semelhanças em relação a escalabilidade temporal definida na norma MPEG-2 Vídeo.

Na norma MPEG-4 Visual, a codificação de VOPs do tipo I, P ou B da camada superior realiza-se da mesma forma que a codificação de quadros I, P ou B da camada superior da norma MPEG-2 Vídeo. A única exceção é que a norma MPEG-4 não oferece o modo de escalabilidade temporal entrelaçado / progressivo da norma MPEG-2 Vídeo.

No entanto, existem algumas diferenças entre a codificação dos B-VOPs em camadas superiores e os B-VOPs da camada base na norma MPEG-4: o modo direto da estimação de movimento só pode ser utilizado para os B-VOPs da camada superior. Tal como na norma MPEG-2 Vídeo, os B-VOPs da camada superior podem ser utilizados como referência para outros VOPs da camada superior; no entanto, não é permitido utilizar B-VOPs da camada base como referência. Para o caso de objetos com forma arbitrária, a decodificação da forma realiza-se no mesmo modo que para o caso não escalável uma vez que a dimensão do objeto não é alterada.

3.5 Codificação escalável de vídeo com elevada granularidade

A informação audiovisual é hoje transmitida em um número cada vez maior de tipos de redes diferentes. Com a massificação da Internet, os usuários querem ter acesso a áudio e vídeo com exigências cada vez maiores de qualidade. A transmissão de vídeo através da Internet assume um papel cada vez mais importante, como demonstra número cada vez maior de *websites* que incluem conteúdo multimídia (ex: noticiários, filmes, concertos ao vivo, etc.), especialmente codificado para a distribuição na Internet. A quantidade de informação audiovisual e o conjunto de aplicações que permitem a distribuição audiovisual na Internet tem aumentado consideravelmente nos últimos anos. No entanto, a qualidade do conteúdo multimídia distribuído, em particular do vídeo, ainda necessita de melhorias significativas para que seja aceito pelos usuários como uma alternativa confiável e viável, como por exemplo, a televisão. Por outro lado, a mobilidade das comunicações é um dado adquirido, como comprova a explosão do número de celulares, cada vez mais sofisticados em termos de funcionalidades. Com o aparecimento de novas redes móveis, as comunicações móveis não estarão limitadas a voz e dados, mas irão incluir também informação multimídia. Atento a importância destes dois ambientes – Internet e redes móveis – o grupo MPEG incluiu na norma MPEG-4 um conjunto de ferramentas que permitissem uma melhor resiliência a erros e uma maior eficiência de codificação, dois dos requisitos mais característicos destes ambientes. No entanto, estas ferramentas não bastam e a escalabilidade do *bitstream*, em termos de qualidade, resolução espacial e temporal, tem um papel crucial na obtenção de melhor qualidade visual em redes com largura de banda variável. A variação ao longo da comunicação da largura de banda disponível é uma das características mais determinantes em ambientes deste tipo. A escalabilidade de vídeo codificado permite que a aplicação adapte a qualidade de vídeo transmitido a variações nas características da rede. Uma solução de codificação escalável de vídeo para redes

deste tipo de possuir uma arquitetura simples e flexível para a distribuição de vídeo e deve cumprir os seguintes requisitos (RADHA, 2001):

- **Processamento mínimo no servidor:** O servidor de vídeo, responsável pelo controle de *bit rates*, deve efetuar o menor processamento possível para controlar um grande número de ligações simultaneamente.
- **Heterogeneidade das tecnologias de transporte:** A própria representação escalável do vídeo deve permitir uma fácil adaptação do conteúdo a diferentes tipos de redes e a alterações nas suas características.
- **Decodificação de baixa complexidade:** O decodificador deve possuir baixa complexidade e baixos requisitos de memória para permitir que o maior número de terminais possam ser capazes de decodificar o conteúdo desejado.
- **Suporte de aplicações ponto a ponto (*unicast*) e ponto-multiponto (*multicast*):** Este requisito elimina a necessidade de codificar o conteúdo em múltiplos formatos para diferentes tipos de aplicação.
- **Resiliência com degradação suave (*graceful*) a perda de pacotes:** Sendo a perda de pacotes bastante comum na Internet, a própria representação do vídeo deve permitir uma degradação suave da qualidade quando este tipo de erro ocorre.

Para que estes requisitos sejam cumpridos, é necessário que o *bitstream* de vídeo possa ser decodificado a qualquer *bit rate* e não apenas ao *bit rate* total que lhe corresponde. A **figura 3.23** ilustra este aspecto: o eixo horizontal indica a largura de banda do canal, o eixo vertical a qualidade de vídeo recebida pelo usuário e a curva de distorção / *bit rate* indica a qualidade máxima possível para qualquer técnica de codificação a um dado *bit rate*. Apesar de uma técnica de codificação não escalável alcançar um desempenho ótimo para um determinado *bit rate*, um *bitstream* pré-codificado não pode ser transmitido se a largura de banda disponível na rede é inferior ao *bit rate* que se utilizou para codificar o vídeo. Por outro lado, a qualidade do vídeo não pode aumentar se a largura de banda disponível for superior. Na **figura 3.23**, também se apresenta a curva distorção / *bit rate* ótima que indica o limite máximo de qualidade para qualquer técnica de codificação para um determinado *bit rate*. O objetivo da codificação escalável de vídeo com elevada granularidade é gerar o *bitstream* que possa ser decodificado a qualquer *bit rate*, eventualmente com uma curva de distorção / *bit rate* ligeiramente inferior a curva ótima.

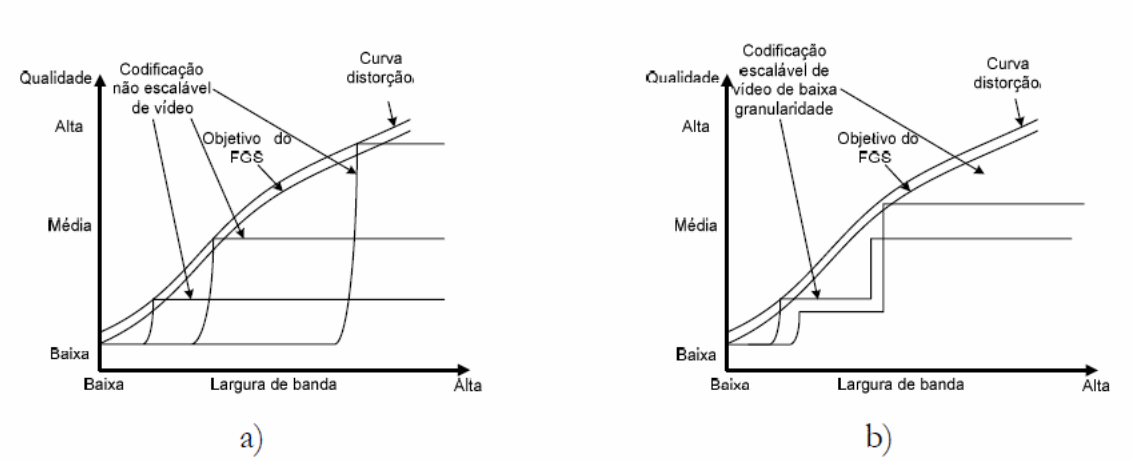


Figura 3.23: Objetivo da escalabilidade FGS em rede com largura de banda variável: a) comparando com a codificação escalável; b) com a codificação de baixa granularidade.

Para ultrapassar este problema, uma estratégia de codificação com múltiplos *bit rates* SSL (*Switched Single Layer*) tornou-se muito popular, especialmente na Internet, onde um número arbitrário de *bitstreams* com diferentes *bit rates* é armazenado no servidor e transmitido de acordo com as condições da rede ou as preferências do usuário (como os ter *bitstreams* não escaláveis, ilustrados na **figura 3.23a**). No entanto, esta estratégia é inferior ao FGS em termos de desempenho distorção / *bit rate* se a largura de banda disponível na rede não for igual ao *bit rate* de codificação. Uma técnica de codificação escalável, mesmo que tenha um desempenho inferior a codificação não escalável para um dado *bit rate*, possui um desempenho superior ao da solução SSL, uma vez que entre os pontos de operação do SSL irá apresentar um desempenho bem superior.

Outra estratégia de codificação é a utilização de técnicas de escalabilidade com baixa granularidade (presentes nas normas MPEG-2 Vídeo, H.263+ e MPEG-4 Visual). Tal como é ilustrado na **figura 3.23b**, estas técnicas de escalabilidade apenas transformam a curva não escalável com um único degrau em uma curva escalável com dois ou mais degraus. O *bit rate* da camada base determina o primeiro degrau e o *bit rate* total determina o segundo degrau se apenas duas camadas de codificação forem usadas.

Deste modo, o principal objetivo da codificação de vídeo na distribuição de vídeo em canais com largura de banda variável é obter uma curva contínua paralela à curva de distorção / *bit rate* ótima usando um único *bitstream*. Deste modo, obter-se-á um uso mais eficiente da largura de banda e uma degradação suave da qualidade do vídeo com a diminuição do *bit rate* disponível, ao contrário das técnicas SSL e da codificação não escalável com baixa granularidade onde as degradações são bruscas.

Este objetivo justificou que o grupo MPEG normalizasse uma tecnologia de codificação escalável de vídeo com elevada granularidade denominada *Fine Granularity Scalability* (FGS). Inicialmente, três tipos de técnicas foram propostos para alcançar a funcionalidade FGS: codificação em planos de bits dos coeficientes DCT [ASCENSO, 30 – CAP3], codificação através da transformada DWT (SCHUSTER, 1998) (CHEN, 1998) (LIANG, 1998) e codificação usando *matching pursuits* (CHEUNG, 1998) (BENETIERE, 1998), a codificação em planos de bit foi escolhida para inclusão na norma MPEG-4 devido a sua baixa complexidade e elevada eficiência e simplicidade de implementação.

3.5.1 Estrutura de escalabilidade

Para cumprir os requisitos definidos anteriormente, a codificação MPEG-4 FGS foi desenvolvida tendo em vista cobrir uma ampla gama de larguras de banda e mantendo uma estrutura de escalabilidade simples. Tal com é ilustrado na **figura 3.24a**, a estrutura de codificação consiste em apenas duas camadas: uma camada base codificada com um *bit rate* R_b e uma única camada superior codificada com elevada granularidade e com um *bit rate* máximo R_{max}. O codificador apenas necessita conhecer a gama de variação do *bit rate* no canal [R_b, R_{max}] e não necessita conhecer o valor do *bit rate* efetivo que irá ser utilizado em cada momento para distribuir o conteúdo. Deste modo, o processo de codificação é totalmente independente das condições de distribuição, das características da rede em que o conteúdo irá ser distribuído em cada instante, permitindo uma abstração entre o processo de codificação e de distribuição. Por outro lado, o servidor possui liberdade para enviar qualquer coisa da parte da camada superior em simultâneo com a camada base. Na **figura 3.24b**, apresenta-se este processo onde a linha vermelha corresponde ao *bit rate* disponível no canal de transmissão e logo a ser transmitido pelo servidor. Este, independentemente dos parâmetros de codificação de vídeo, é capaz de adaptar o conteúdo as características da rede em um determinado momento, através da seleção da quantidade de informação que deseja enviar. Deste modo, apenas é necessário é necessário cortar o *bitstream* de cada quadro com um número arbitrário de bits e, se durante o processo de transmissão não ocorrerem erros, a qualidade de vídeo no cliente será sempre proporcional ao número de bits enviados, ou seja, mais bits mais qualidade e vice-versa (**figura 3.24c**).

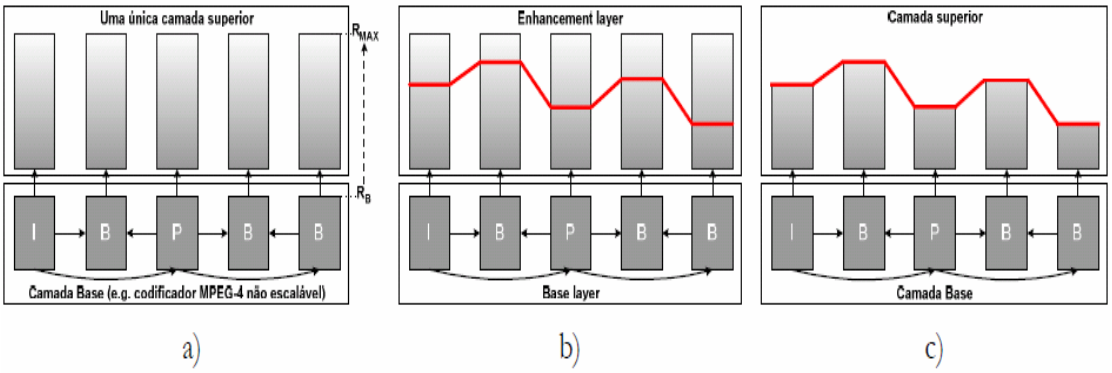


Figura 3.24: Exemplos da estrutura de escalabilidade FGS em uma aplicação *unicast*. a) no codificador; b) no servidor; c) no cliente.

Por outro lado, como o conteúdo é codificado uma única vez e adaptado quantas vezes desejar, evita-se a utilização de algoritmos de controle de *bit rate* com uma complexidade elevada. Como o vídeo já se encontra codificado, o servidor de vídeo é capaz de manter um grande número de ligações ponto a ponto (*unicast*) simultaneamente e adaptar o *bit rate* para cada uma das ligações individualmente e em tempo-real, independentemente da complexidade do codificador (este até pode não funcionar em tempo-real e normalmente não funciona). No cliente, o decodificador FGS possui requisitos de memória e processamento comparáveis aos de um decodificador MPEG-4 Visual do perfil *Advanced Simple* (ISO:14496-2, 2001) e a qualidade do vídeo recebido é proporcional ao *bit rate* que a conexão oferece.

Para conexões ponto-multiponto (*multicast*), a codificação FGS oferece uma arquitetura adequada à codificação, distribuição e decodificação de vídeo (SCHUSTER, 1999). Tal como no caso *unicast*, o vídeo é codificado para uma gama de variação do *bit*

rate [R_b, R_{max}]. Deste modo, o mesmo *bitstream* pode ser utilizado tanto para aplicações *unicast* como *multicast*, mas ainda que o processo de distribuição seja diferente. O servidor de vídeo divide a camada superior em um número arbitrário de partições que correspondem a diferentes canais *multicast* em que cada um tem um *bit rate* diferente (**figura 3.25**). O cliente subscreve um número arbitrário de canais, de acordo com o *bit rate* disponível ou com a sua capacidade de processamento. Uma restrição importante é que todos os clientes tem que receber a camada base, enviada em um sinal *multicast* separado. A codificação FGS oferece a flexibilidade necessária para este tipo de situação, pois deixa ao cliente a possibilidade de definir o *bit rate* que pretende receber e o servidor apenas necessita transmitir um único *bitstream* (contudo, a solução mais popular ainda é transmitir vários *bitstreams* com *bit rates* diferentes).

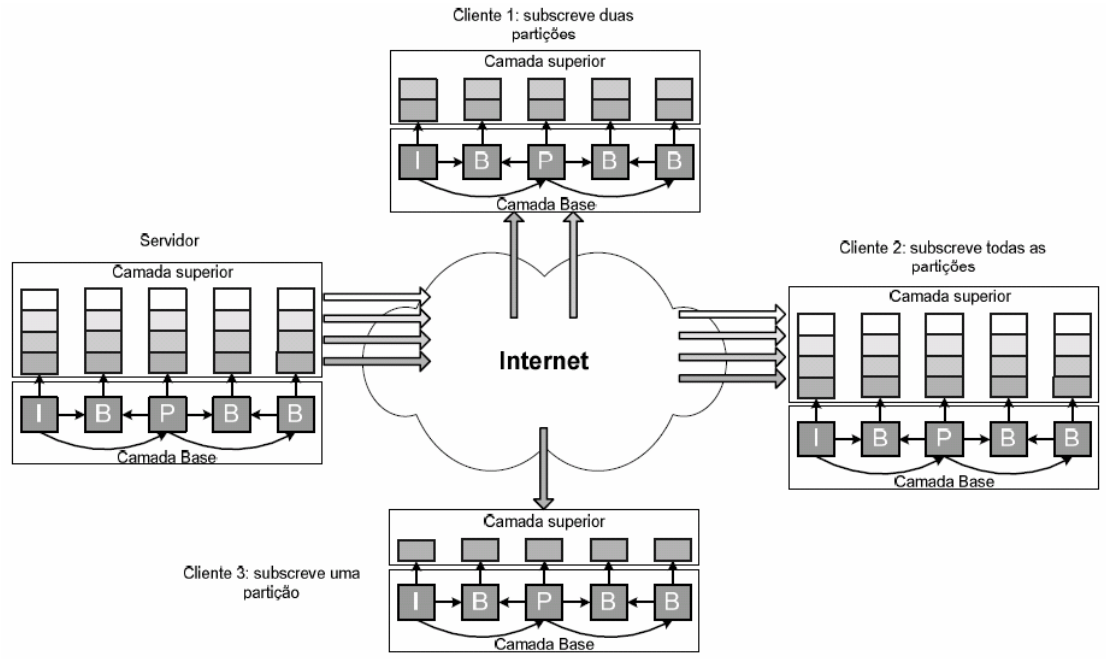


Figura 3.25: Exemplos da estrutura de escalabilidade FGS em uma aplicação *multicast*.

A solução *multicast* é suportada pela popular rede IP Multicast BackBONE (MBONE) (ERIKSSON, 1994). Como os protocolos de transporte e de controle para redes deste tipo estão definidos, apenas é necessária uma representação escalável do vídeo adequada para obter um sistema de distribuição audiovisual completo. A codificação MPEG-4 FGS permite alcançar este objetivo.

Na norma MPEG-4 FGS não existe nenhuma dependência entre quadros da camada superior, pois estes são sempre codificados no modo Intra; o que penaliza a eficiência de codificação, apesar de um esquema de codificação com compensação de movimento ser utilizado na camada base. A codificação Intra possui uma vantagem inerente importante: a robustez a erros de transmissão. Uma vez que as imagens da camada base são codificadas tanto em modo Intra como Inter, esta pode ser distribuída com uma elevada robustez e proteção usando técnicas de codificação de canal (GALLANT, 2001) ou de retransmissão da informação (se houver tempo para isso) (RHEE, 1998) uma vez que a sua recepção é essencial. Por outro lado, a camada superior pode ser distribuída com menos ou mesmo sem qualquer tipo de proteção uma vez que os erros de transmissão não se propagam de quadro para quadro. A **figura 3.26** exemplifica a ocorrência de erros de transmissão na camada superior para dois tipos de codificação: FGS e codificação escalável com baixa granularidade descrita anteriormente. Para FGS,

um erro em um quadro da camada superior provoca apenas um decréscimo de qualidade no quadro em que ocorreu o erro (**figura 3.26**), enquanto que na codificação escalável com baixa granularidade um erro em um quadro P irá propagar-se para os quadros que dependem deste, para quadros P seguintes e para os quadros B que o tem como referência (para 2 quadros B na **figura 3.26b**).

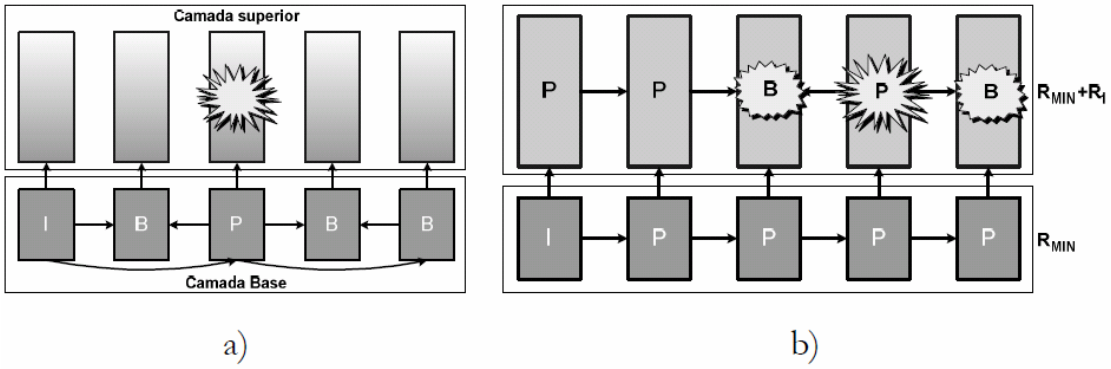


Figura 3.26: Robustez a perda de pacotes: a) FGS; b) codificação com baixa granularidade.

Descreve-se a seguir a técnica de codificação em planos de bit; esta técnica pode ser utilizada por qualquer codificador híbrido e possui uma importância vital no contexto da especificação MPEG-4 FGS.

3.5.2 Codificação em planos de bit

Na codificação DCT convencional, os coeficientes DCT quantificados são codificados com a técnica *Run Lenght Encoding* (RLE). Com esta técnica, o número de zeros consecutivos antes de um coeficiente DCT diferente de zero é referido como *run* e o valor absoluto do coeficiente DCT quantificado diferente de zero é referido com *level*. Os pares (*run, level*) são codificados usando uma tabela VLC bidimensional e um símbolo *eob* é utilizado para assinalar o fim do bloco da DCT, o fato de não existirem mais coeficientes DCT diferentes de zero para codificar (esta é a solução usada na recomendação ITU-T H.263). A principal diferença entre o método de codificação em planos de bit e o método RLE é que o primeiro considera cada coeficiente DCT quantificado como um número binário com vários bits, em vez de um valor inteiro com um determinado valor (WLI, 1997) (LING, 1999). Na codificação em planos de bit, cada bloco de 8x8 coeficientes DCT, os 64 valores são varridos em zig-zag para um vetor. Cada plano de bit do bloco de coeficientes DCT é definido como um vetor de 64 bits de comprimento, em que os seus valores (bits '0' ou '1') correspondem a uma dada posição significativa e são extraídos a partir dos valores absolutos em binário dos coeficientes DCT quantificados. Para cada plano de bit de cada bloco, símbolos (*run, eop*) calculados e codificados entropicamente. Começando pelo plano de bit mais significativo (MSB) os símbolos são gerados em dois componentes:

- *Run*: Número de zeros consecutivos antes de um bit com o valor '1'.
- *Eop*: Indica se existem mais bits com o valor '1' ou não nesse plano de bit; se um plano de bit só contém valores '0', um símbolo especial designado por ALL_ZERO representa esse plano.

O exemplo na **figura 3.27** ilustra esta técnica. Na **figura 3.27a**, os valores absolutos de cada coeficiente DCT quantificado e os bits de sinal correspondentes são

apresentados. O valor máximo dos coeficientes DCT neste bloco é 10 e o número máximo de bits necessário para representa-lo é 4 (10 = 1010). Escrevendo cada valor no formato binário, quatro planos de bit são gerados (**figura 3.27b**). Utilizando a técnica acima descrita, convertem-se os quatro planos de bit em símbolos (*run, eop*), tal como é ilustrado na **figura 3.27c**. Deste modo, são obtidos 10 símbolos que irão ser codificados entropicamente simultaneamente com o bit de sinal.

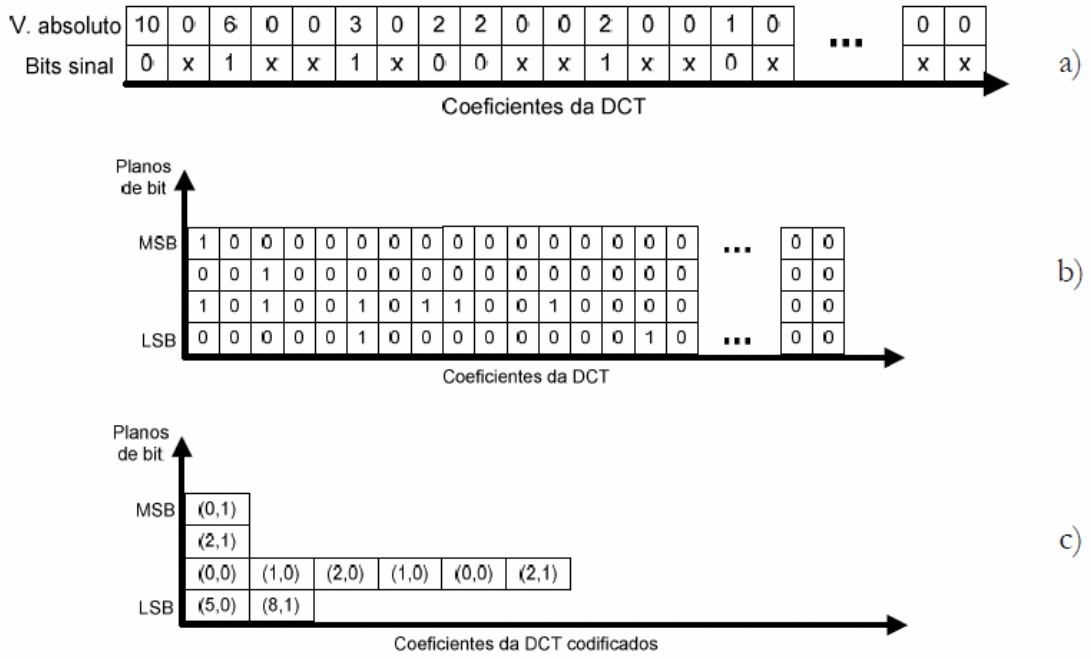


Figura 3.27: Codificação em plano de bits: a) coeficientes da DCT; b) matriz de planos de bit; c) codificação em pares (*run, eop*).

Cada bit de sinal é colocado no *bitstream* apenas uma vez (para cada coeficiente DCT), depois do par (*run, eop*) que contém o MSB do valor absoluto associado ao bit de sinal. A **figura 3.28** ilustra este processo, para os mesmos coeficientes da DCT da **figura 3.27**. Como exemplo, considere-se o coeficiente da DCT com valor 10 que contém 4 bits, dois dos quais a '1'. O código VLC usado para codificar o bit mais significativo deste coeficiente é o código VLC (0,1), colocando-se o bit '0'a seguir para indicar o sinal positivo. Um bit a '1' indicaria um sinal negativo. No entanto, quando se codifica outro plano de bit do mesmo coeficiente (1010) não se deve colocar novamente o bit de sinal, no MSB-2 a seguir VLC (0,0) não existe nenhum bit de sinal.

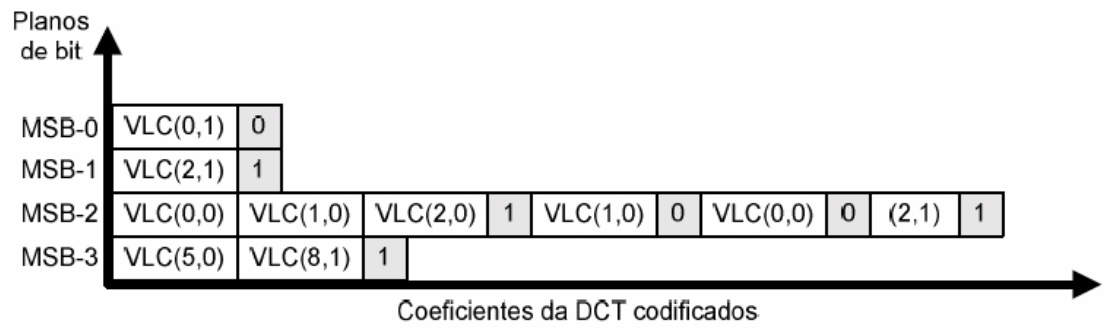


Figura 3.28: Codificação em planos de bit com inserção dos bits de sinal.

Para avaliar a eficiência da codificação em planos de bit, deve-se substituir o módulo de codificação RLE de um codificador não escalável, pelo módulo de codificação em planos de bit descrito. Várias experiências mostram que a codificação em planos de bits é mais eficiente que a codificação RLE (WLI, 2001). A principal razão para esta melhoria da eficiência de codificação é o fato das estatísticas para cada plano de bit serem independentes do valor Qp utilizado para a quantificação dos coeficientes DCT, as tabelas VLC definidas para o método RLE são um compromisso para todos os valores Qp possíveis enquanto na codificação em planos de bit as estatísticas dos vários planos de bit são independentes do valor Qp.

3.5.3 Arquitetura de codificação FGS

Como é ilustrado na **figura 3.29**, a arquitetura do codificador MPEG-4 FGS necessita de dois andares de codificação, um para a camada base e outro para a camada superior. A camada base pode ser codificada através de qualquer codificador de vídeo baseado na transformada DCT e na compensação de movimento. Naturalmente, a norma MPEG-4 Visual oferece vários candidatos válidos (os vários perfis) para o codificador da camada base devido a sua elevada eficiência, especialmente para os *bit rates* baixos. Como a norma MPEG-4 Visual contém inúmeras ferramentas de codificação de vídeo, foi necessário definir o subconjunto das ferramentas de codificação a ser utilizado pelo codificador da camada base e pelo codificador da camada superior, definir os perfis visuais correspondentes as duas camadas (REQ, 2000) (ISO:14496-2, 2001). O perfil visual adotado para a camada base foi o *Advanced Simple Profile* (ASP), por ser aquele que oferecia uma elevada eficiência de codificação da norma MPEG-4 Visual (LUTHRA, 2001), para uma ampla gama de *bit rates*, ainda que apenas para objetos retangulares. Este perfil inclui ferramentas de codificação para P e B-VOPs, predição dos coeficientes DC e AC, quatro vetores de movimento por bloco, vetores de movimento sem restrições, dois métodos de quantificação, ferramentas de codificação de vídeo entrelaçado, ferramentas de resiliência a erros e compensação de movimento global e com precisão de ¼ *pixel*. Além disso, este perfil permite compatibilidade direta com a norma H.263 através da opção *short headers* (ou seja, um decodificador MPEG-4 com este perfil pode decodificar um *bitstream* H.263). Devido ao tipo de ferramentas incluídas nos perfis da camada base e da camada superior, a codificação MPEG-4 FGS apenas suporta objetos retangulares (e não objetos com forma arbitrária). O *bitstream* gerado por este codificador pode ser cortado em qualquer ponto da camada superior (devido a limitações do *bit rate* da rede), mesmo depois do processo de codificação estar completo, devendo o decodificador ser capaz de decodificar qualquer *bitstream* cortado. Como é natural, a qualidade do vídeo visualizado pelo usuário depende do número de bits decodificados para cada quadro, mas existe a garantia que todos os bits recebidos são aproveitados para melhorar esta qualidade. Para o codificador são apresentadas na **figura 3.29** duas estruturas: uma vez que o codificador apresenta na **figura 3.29**, a imagem residual a codificar pela camada superior pode ser calculada de duas formas: no domínio do tempo (em verde na **figura 3.29**) ou no domínio da freqüência (azul na **figura 3.29**). Se a imagem residual for calculada no domínio do tempo, o codificador da camada superior deve calcular a diferença entre a imagem original e a imagem decodificada para o mesmo instante de tempo correspondente a camada base.

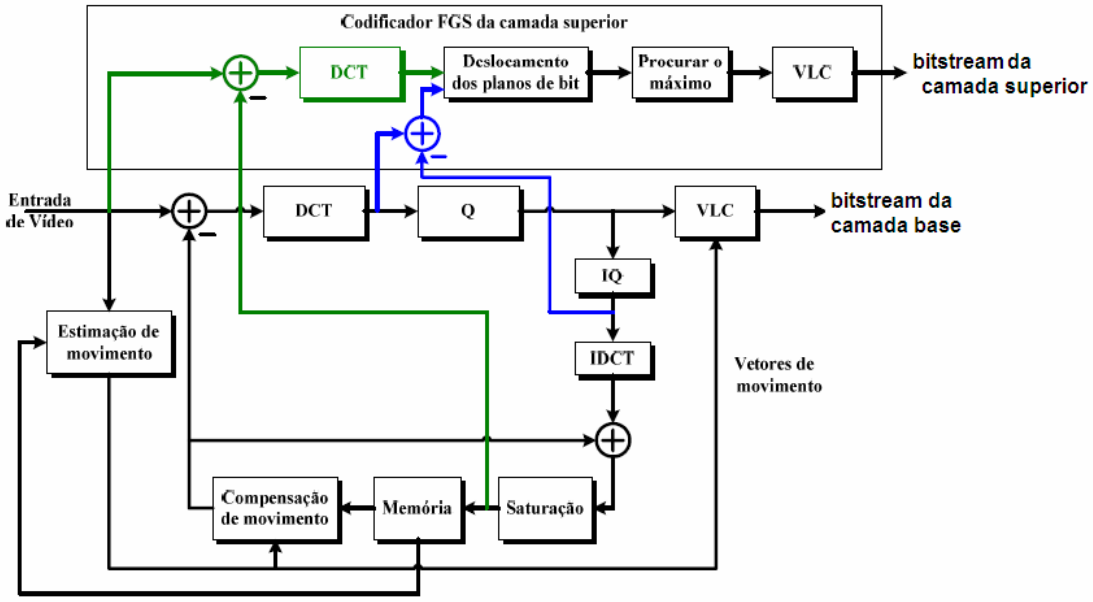


Figura 3.29: Arquitetura do codificador MPEG-4 FGS.

A estrutura normalizada do decodificador MPEG-4 FGS é apresentada na **figura 3.30**. Na arquitetura do decodificador, a operação inversa é calculada, decodifica a camada base e a camada superior separadamente e, no fim, adiciona-se a imagem da camada base com a imagem residual correspondente. Esta separação dos decodificadores das camadas superiores e de base permite uma implementação prática e eficiente, especialmente se o decodificar da camada base já estiver disponível.

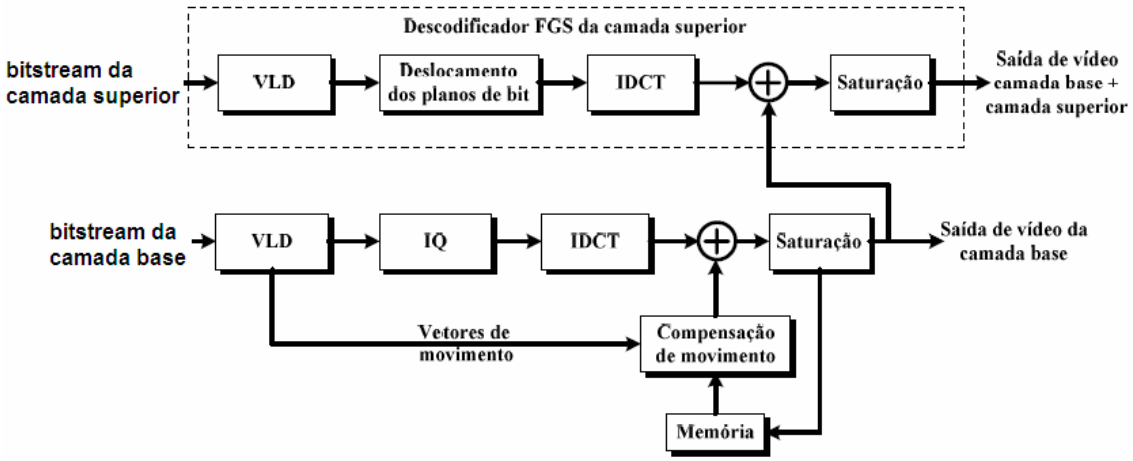


Figura 3.30: Arquitetura do decodificador MPEG-4 FGS

Outra forma de calcular a imagem residual no codificador é no domínio da freqüência (azul na **figura 3.29**); neste caso, o módulo DCT da camada superior já não é utilizado. Esta arquitetura tira partido de uma propriedade da transformada DCT: a linearidade. No entanto, o decodificador normalizado possui um módulo não linear, o módulo de saturação (*clipping*). A saturação tem como objetivo colocar a 0 qualquer *pixel* com valor inferior a 0 e a 255 qualquer *pixel* com valor superior a 255. Esta operação é necessária devido aos erros que se introduzem quando se calcula a DCT e em seguida a IDCT, com uma precisão finita.

Mesmo com esta não linearidade, e uma vez que a estrutura do codificar não é normalizada o cálculo do resíduo pode continuar a ser efetuado no domínio da freqüência, evitando o módulo da DCT no codificador da camada superior. No entanto, como a estrutura do decodificador é normalizada, a imagem da camada base será somada a imagem residual da camada superior depois de se efetuar a saturação o que causa uma diferença (*mismatch*) entre as imagens decodificadas no codificador e no decodificador. Como as imagens decodificadas da camada superior não são utilizadas para predição, esta diferença apenas afeta as imagens individualmente, pois não existe o problema da propagação de erros de uma imagem para outra(s). Várias experiências foram realizadas para avaliar o impacto desta diferença (JIANG, 2000) (WLI, 2000-2), concluindo que os erros grandes ocorrem poucas vezes e não são visualmente importantes. Desta maneira, quando se implementa um codificador FGS pode-se eliminar o módulo DCT na camada superior, ou seja, adotar a solução azul da **figura 3.29**.

Outro módulo presente na camada superior é o "deslocamento de plano de bit" que corresponde as funcionalidades "seleção de freqüências" e "melhoria seletiva"; estas funcionalidades tem como principal objetivo melhorar a qualidade objetiva e subjetiva do vídeo transmitido.

O módulo "procurar o máximo" tem como objetivo encontrar o número máximo de planos de bit necessários para representar um quadro, uma vez que as três componentes de cor (Y, U e V) podem ser representadas por um número arbitrário de planos de bit, para um determinado quadro. Os três valores (*maximum_level_y, maximum_level_u e maximum_level_v*) são codificados no cabeçalho de cada quadro FGS e indicam ao decodificador o número máximo de planos de bit para as componentes Y, U e V, respectivamente. O passo seguinte consiste em efetuar a varredura em "zig-zag" dos planos de bit, começando pelo plano de bit mais significativo BP(1) e acabando no plano de bit menos significativo BP(N), tal como é ilustrado na **figura 3.31** para a componente de luminância de um macrobloco (para a crominância a ordem de varredura é igual).

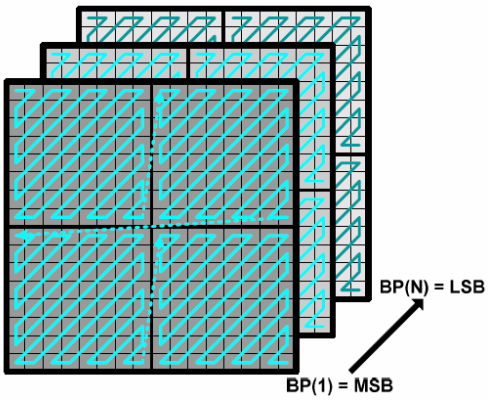


Figura 3.31: Ordem de varredura dos coeficientes DCT em um macrobloco para os vários planos de bit.

Depois de se efetuar a varredura dos planos de bit dos coeficientes da DCT, se obtém um vetor constituído por elementos (bits) com o valor "0" ou "1", determinando-se então os símbolos (*run, eop*), tal com descrito anteriormente.

3.5.3.1 Codificação entrópica

Este módulo tem como principal objetivo codificar entropicamente os símbolos (*run, eop*). Quatro tabelas VLC foram definidas com este objetivo, correspondendo ao plano de bit mais significativo (MSB), ao plano de bit MSB-2, ao plano de bit MSB-2 e aos restantes planos de bit. Nota-se que, no contexto da utilização das tabelas VLC, o plano de bit mais significativo é definido ao nível do bloco. O plano de bit MSB de cada bloco é o primeiro plano de bit que não possui todos os seus elementos a zero (plano de bit ALL_ZERO) e pode variar de bloco para bloco. Uma vez que existem 64 bits em cada plano de bit (correspondentes aos 64 coeficientes), o valor do *run* pode variar entre 0 e 62 para *eop* = 0 e entre 0 e 63 para *eop* = 1. Nota-se que não existes o caso de 63 zeros consecutivos com *eop* = 0 pois isso significaria que existiam mais bits no plano de bit do que é possível. Deste modo, cada tabela VLC deve conter 128 símbolos (63+63+1), incluindo o símbolo ALL_ZERO. No entanto, uma vez que a probabilidade para valores grandes do símbolo *run* são reduzidas, um código ESCAPE é utilizado em cada tabela VLC para assinalar um símbolo com um valor *run* a partir de 77 para o MSB; 66 para o MSB-1; 53 para MSB-2; e 37 para MSB-3. Depois do código ESCAPE, seis bits são utilizados para codificar o valor de um *run* e um bit para codificar o valor de *eop*.

No lado do receptor, o *bitstream* FGS é decodificado entropicamente pelo módulo VLD (*Variable Lenght Decoder*), tal como é ilustrado na **figura 3.30**. Devido a estrutura do *bitstream*, o VLD começa por decodificar primeiro os planos de bit mais significativos até chegar aos menos significativos. Além disso, o tipo de varredura utilizado pelo codificador FGS (**figura 3.31**) permite que o decodificador não receba todos os blocos que pertencem a um determinado plano de bit sem que isso cause problemas irreversíveis. Qualquer bloco não recebido (devido a erros de transmissão) pode ser preenchido pelo decodificador com valores iguais a zero. O resíduo recebido é inversamente transformado pela IDCT para gerar a imagem residual que irá ser somada a saída do decodificador da camada base e obter uma imagem com a máxima qualidade possível para o conjunto de bits recebidos.

Em uma aplicação típica da codificação FGS, o *bitstream* na entrada do decodificador FGS é uma versão truncada da saída do codificador FGS. Isto significa que no fim de cada quadro FGS, e antes do próximo quadro, apenas parte da informação correspondente a este quadro FGS está disponível na entrada do decodificador, devido ao corte do quadro FGS, pelo servidor de vídeo. A forma como se decodifica um *bitstream* FGS truncado não é especificado na norma MPEG-4 FGS. Um dos métodos possíveis para decodificar um quadro FGS cortado é ler os 32 bits em cada posição alinhada ao byte no *bitstream* e verificar se esses 32 bits correspondem ao começo de um novo quadro (se são iguais a *fgs_vop_start_code*), uma vez que a palavra *fgs_vop_start_code* tem 32 bits e esta alinhada ao byte. Se surgir o início de um novo quadro, o decodificador pode completar a decodificação até o *fgs_vop_start_code* ou desprezar os bits antes do *fgs_vop_start_code*. No caso de não corresponder ao início de um novo quadro, os primeiros 8 bits dos 32 bits correspondem a informação útil de textura para ser decodificada. Em seguida, o decodificador deve continuar a verificar se os próximos 32 bits começando no próximo byte são iguais ao *fgs_vop_start_code* ou não e assim sucessivamente.

3.5.4 Escalabilidade híbrida qualidade / temporal

Na estrutura de escalabilidade já descrita, a freqüência de quadro da camada superior é sempre igual a freqüência de quadro da camada base, independentemente do *bit rate*

disponível. No entanto, um dos principais objetivos da codificação FGS é abranger uma gama ampla de *bit rates*, especialmente em redes IP. Consequentemente surgiu a necessidade de combinar a escalabilidade de qualidade (SNR) do FGS com a escalabilidade temporal, em uma arquitetura que permita flexibilidade entre a suavidade do movimento e a qualidade espacial da imagem (SCHAAR, 2001-2).

Nas normas H.263, MPEG-2 e MPEG-4, a escalabilidade temporal é alcançada através da variação da freqüência de quadro da seqüência de vídeo, codificando a camada base com uma freqüência de quadro f_B e introduzindo quadros adicionais na camada superior até uma freqüência de quadro total f_E. A seqüência de vídeo é visualizada com uma freqüência de quadro adequada ao *bit rate* disponível, a capacidade computacional do decodificador ou as preferências do usuário, com freqüência de quadro f_B ou $f_B + f_E$. No entanto, na codificação FGS é desejável uma abordagem que proporcione uma escalabilidade de elevada granularidade em termos da qualidade dos quadros da camada superior que proporcionam um aumento da freqüência do quadro.

Uma das soluções possíveis consiste em separar as camadas SNR e temporais, tal como é ilustrado na **figura 3.32a**. Neste caso, a camada FGS é codificada no topo de duas camadas: a camada base e a camada temporal, conseguindo-se deste modo a desejável escalabilidade híbrida SNR / temporal. No entanto, devido a ausência de escalabilidade na camada temporal este esquema possui algumas desvantagens. O *bit rate* da camada temporal tem de ser conhecido quando esta é codificada e, para melhorar a resolução temporal da seqüência decodificada, é necessária a decodificação completa da camada temporal. Outra desvantagem consiste no aumento da complexidade computacional do decodificador, uma vez que é necessário efetuar estimação / compensação de movimento em duas camadas (FGS e temporal).

Outra estrutura possível é apresentada na **figura 3.32b**. Além dos quadros FGS que proporcionam escalabilidade na qualidade, esta estrutura inclui quadros residuais na camada superior; quadros FGS temporais (FGST). Tal como é ilustrado na **figura 3.32b**, estes quadros FGST são preditos a partir dos quadros da camada base que estão temporalmente antes e depois do quadro FGST, o que proporciona a desejada escalabilidade temporal. Uma vez que a predição temporal só pode basear-se nos quadros da camada base, a qualidade dos quadros FGST não afeta a qualidade de outros quadros, o que é desejável em ambientes onde ocorram erros de transmissão ou quando é necessário envia ou decodificar apenas uma parte do quadro FGST. A estrutura de escalabilidade temporal baseada em quadros FGST foi a adotada pela norma MPEG-4 Visual.

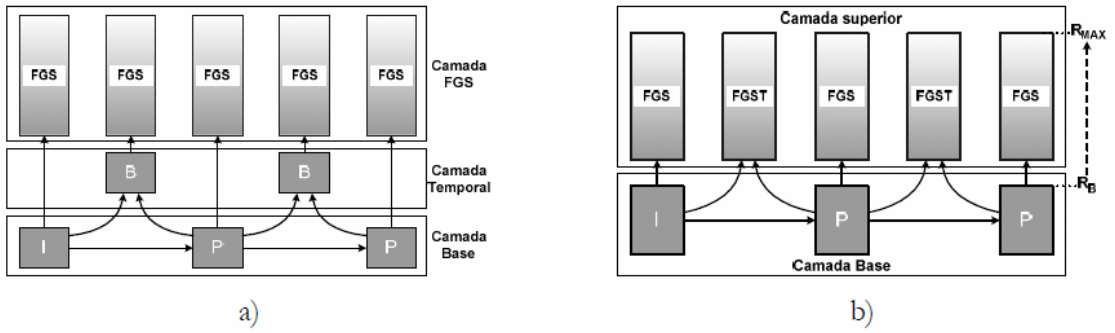


Figura 3.32: Estruturas de escalabilidade temporal: a) com camada temporal; b) com quadros FGST.

Os quadros FGST são constituídos por dois tipos de informação: vetores de movimento, calculados em relação aos quadros da camada base temporalmente adjacentes, e dados de textura que representam o quadro residual codificador com o mesmo método que os quadros FGS, com elevada granularidade. Estes dois tipos de informação são codificados e transmitidos através de uma estratégia de separação de dados. Ao contrário da camada base, onde os vetores de movimento e os dados de textura são multiplexados ao nível de macrobloco, nos quadros FGST todos os vetores de movimento são agrupados e transmitidos primeiro e só depois todos os planos de bit que representam o quadro residual; ou seja, a multiplexagem dos dois tipos de informação faz-se ao nível do quadro. Este método é uma ferramenta útil de resiliência a erros porque permite que os vetores de movimento tenham uma prioridade superior a informação de textura, reduzindo o impacto negativo das perdas de informação em quadros FGST.

No entanto, este esquema de codificação coloca questões importantes: o desempenho associado a codificação em planos de bit dos quadros FGST e o acréscimo de complexidade correspondente a codificação dos quadros FGST. Para esclarecer estas questões, um conjunto de testes foi efetuado (SCHAAR, 2000), tendo-se demonstrado que apesar das diferenças conceituais, os sinais FGS e os sinais FGST possuem uma estatística muito semelhante. Além disso, o desempenho é idêntico quando se utiliza a codificação em planos de bit para o sinal residual (quadros FGS) em comparação com os quadros FGST (SCHAAR, 2001-2). Quanto a complexidade, este esquema de codificação não necessita de uma alteração significativa da arquitetura do codificador e do decodificador FGS, mas apenas de um simples controle do fluxo de dados que tire partido do fato do codificador não comprimir um quadro da camada base e um quadro da camada superior no mesmo instante temporal. Desta maneira, todos os módulos disponíveis para o cálculo da camada base podem ser utilizados quando for necessário codificar um quadro FGST, a complexidade computacional é semelhante a do codificador FGS quando este funciona na mesma freqüência do quadro que o codificador FGST.

Esta arquitetura proporciona um novo nível de abstração entre o codificador e o servidor de vídeo através do suporte simultâneo da escalabilidade temporal e de qualidade (SNR) em uma única camada superior. Esta abstração é muito importante uma vez que o *bit rate* disponível e/ou as preferências do usuário não são conhecidas quando o vídeo é codificado. A arquitetura de codificação adotada permite o servidor de vídeo decidir que tipo de escalabilidade deve ser utilizada e qual a qualidade (SNR) que cada quadro FGS e/ou FGST deve ter. Resumindo, esta estrutura de escalabilidade permite:

- Escalabilidade de qualidade (SNR) mantendo a mesma freqüência do quadro.
- Escalabilidade temporal aumentando a freqüência do quadro.
- Escalabilidade híbrida de qualidade e temporal.

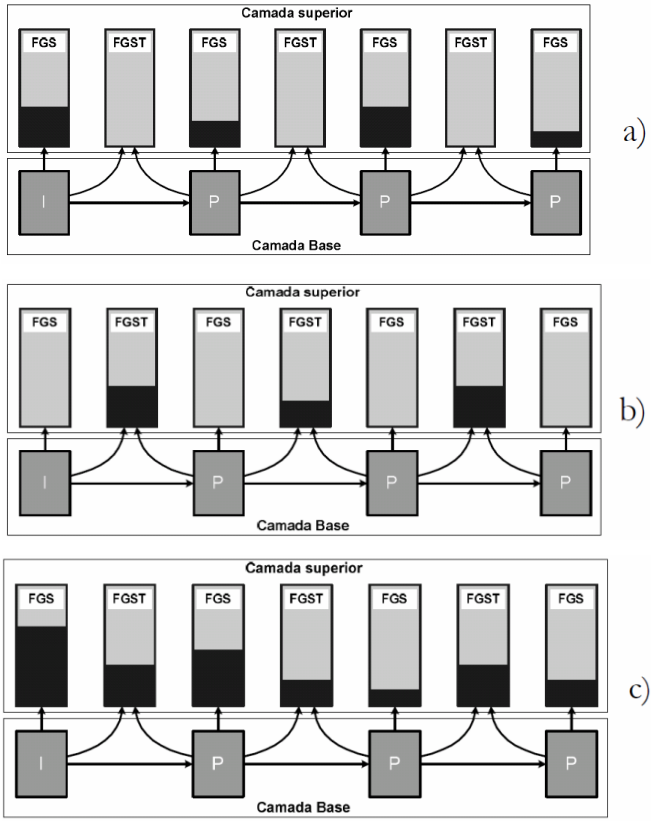


Figura 3.33: Exemplos de escalabilidade híbrida (▇ indica a quantidade transmitida da camada superior).

Dependendo do conteúdo do vídeo e do *bit rate* disponível, o servidor pode decidir melhorar a qualidade da camada base de diversas formas: por exemplo, enviando apenas os quadros FGS que melhoram a qualidade dos quadros da camada base e mantendo a mesma freqüência de quadros (**figura 3.33a**). Como alternativa pode decidir melhorar a suavidade do movimento, enviando apenas os quadros FGST, para que a seqüência de vídeo seja visualizada com uma freqüência de quadro superior (**figura 3.33b**); se ainda existir *bit rate* disponível, a qualidade SNR dos quadros da camada base pode ser melhorada, enviando os correspondentes quadros FGS (**figura 3.33c**). É importante salientar que este esquema permite uma troca entre a resolução temporal e a qualidade da seqüência de vídeo na transmissão do vídeo e não na codificação.

3.5.5 Quantificação adaptativa

Para melhorar a qualidade visual do vídeo codificado de acordo com a norma MPEG-4 FGS, duas funcionalidades foram introduzidas: seleção de freqüências e melhoria seletiva. Estas duas funcionalidades estão intimamente relacionadas com as técnicas de quantificação adaptativa utilizadas pelo codificador não escalável. Dos dois métodos de quantificação definidos pela norma MPEG-4 Visual, um deles permite o ajuste individual do passo de quantificação para cada coeficiente DCT, através de uma matriz de quantificação (como na norma MPEG-4 Vídeo) enquanto que o outro método adota um passo de quantificação constante para todos os coeficientes da DCT; este passo de quantificação pode ser ajustado ao nível do macrobloco, através da sintaxe de vídeo definida. No entanto, nenhuma destas técnicas pode ser utilizada para a codificação MPEG-4 FGS, uma vez que o *bit rate* não é conhecido, o módulo de quantificação não é utilizado na arquitetura do codificador. Para se efetuar quantificação

ao fazer codificação MPEG-4 FGS, outro tipo de técnicas são utilizadas. Cada plano de bits contém os bits mais significativos de cada coeficiente da DCT, o segundo plano contém os bits MSB-1 e assim sucessivamente. A idéia indicada por quantificação adaptativa consiste em atribuir um peso (ou importância) maior a determinados coeficientes da DCT, a enviá-los primeiro que os restantes. Assim, quando um determinado coeficiente é multiplicado por um peso, os bits que o representam são deslocados para um plano de bit superior. Quando o *bitstream* for cortado, os coeficientes DCT com maior importância são representados com um número de bits maior, com uma maior exatidão, e consequentemente uma maior qualidade. O processo de codificação é semelhante com e sem quantificação adaptativa; no entanto, a forma de organizar os bits no *bitstream* é diferente. Para oferecer a mesma flexibilidade que o método de quantificação adaptativa do codificador não escalável, foram definidas duas ferramentas que podem ser utilizadas individualmente ou em conjunto:

- **Seleção de freqüências:** Corresponde a seleção de coeficientes DCT pertencentes a um bloco de 8x8 *pixels*. A seleção de freqüências permite a utilização de diferentes pesos para diferentes componentes de freqüência, de maneira que os bits associados com freqüências diferentes visualmente importantes sejam colocados no *bitstream* antes de outras componentes de freqüência. A **figura 3.34a** ilustra este processo para um macrobloco em que os coeficientes DCT com freqüências horizontais e verticais baixas (4x4) são deslocados do plano de bit N para o plano de bit N'.
- **Melhoria seletiva:** Corresponde a seleção de macroblocos pertencentes a um quadro FGS. A melhoria seletiva permite que diferentes tipos de pesos sejam utilizados em diferentes localizações da imagem (normalmente aquelas subjetivamente mais importantes), de forma que os coeficientes que correspondem a determinadas zonas de uma imagem sejam amplificados em relação a outros coeficientes. A **figura 3.34b** ilustra este processo para um macrobloco, em que os coeficientes DCT do bloco no topo e a esquerda são deslocados do plano de bit N para N'.

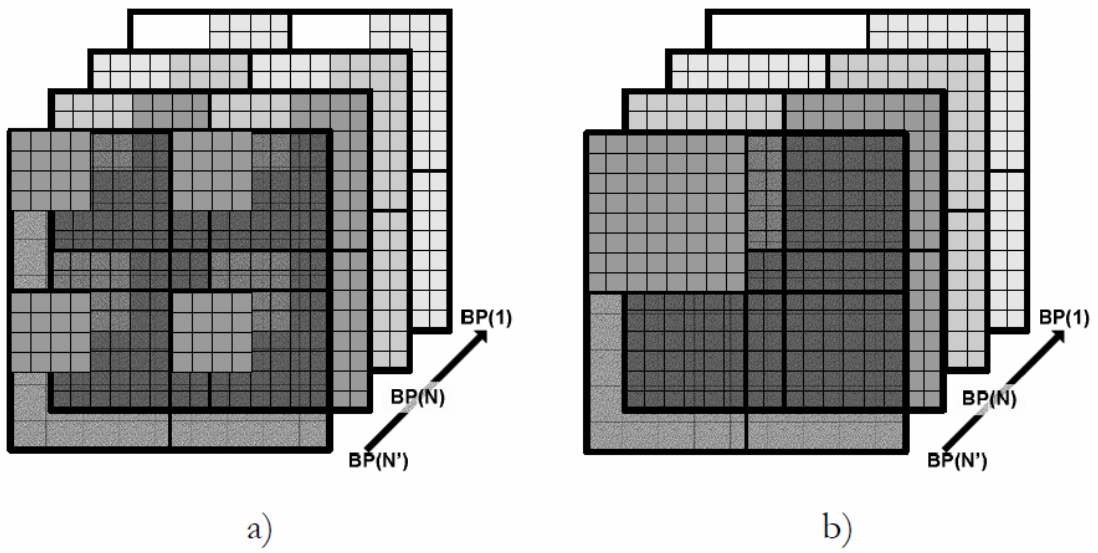


Figura 3.34: Exemplo de quantificação adaptativa para MPEG-4 FGS: a) seleção de freqüências; b) melhoria seletiva (BP(1) corresponde ao plano de bit menos significativo).

3.5.5.1 Seleção de freqüências

Um fato largamente conhecido é que os coeficientes DCT de baixa freqüência são visualmente mais importantes que os coeficientes DCT de alta freqüência. Assim sendo, a qualidade visual da seqüência de vídeo é melhorada se os bits que correspondem as componentes de baixa freqüência da imagem forem amplificados em relação aos restantes. Quando o *bitstream* da camada superior for truncado, há uma maior exatidão para os coeficientes DCT de baixa freqüência recebidos, uma vez que os bits mais significativos que o representem foram colocados antes no *bitstream*. A técnica de seleção de freqüências foi incorporada no FGS (WLI, 1999) (JIANG, 1999) para alcançar este objetivo, consistindo em elevar estes coeficientes para um plano de bit mais elevado. Esta operação é equivalente a multiplicar um conjunto específico de coeficientes DCT por uma potência de dois antes de serem transmitidos e dividir esses coeficientes pela mesma potência depois de serem recebidos. Para isso, define-se uma matriz de pesos de freqüência em que cada elemento indica o número de planos de bit que cada coeficiente DCT deve ser elevado, o expoente da potencia de dois. O codificador pode definir esta matriz e transmiti-la ao decodificador. Esta matriz equivale a matriz de quantificação utilizada pelo codificador não escalável. Se a matriz de quantificação for inserida no *bitstream*, esta é constituída por uma lista de 2 a 64 inteiros de três bits sem sinal, em ordem zig-zag, onde o valor zero indica que mais nenhum valor da matriz será enviado porque os restantes são zero. Esta matriz pode ser definida ao nível da seqüência ou ao nível do quadro.

No entanto, ao efetuar esta operação, as estatísticas de cada plano de bit são alteradas porque os planos de bit mais significativos contém menos coeficientes e estes coeficientes obedecem a distribuição de bits dos coeficientes de baixa freqüência. Por exemplo, verifica-se que existem mais valores pequenos de *run* com *eop* = 1, uma vez que os coeficientes de baixa freqüência estão no inicio do vetor de coeficientes (depois da varredura zig-zag). Assim, para que o codificador entrópico esteja otimizado, definiram-se mais duas tabelas VLC (ISO:14496-2, 2001) quando se utiliza a seleção de freqüências.

A utilização desta técnica permite reduzir os efeitos de bloco típicos da utilização da transformada DCT na codificação de vídeo através da utilização de uma matriz que realce as baixas freqüências. A avaliação subjetiva da qualidade visual indica que estes artefatos podem ser reduzidos e a qualidade visual melhora especialmente para *bit rates* baixos. No entanto, esta melhoria é alcançada porque existe uma atenuação das altas freqüências da imagem, tal como na técnica de quantificação adaptativa da norma MPEG-2 Vídeo. Esta técnica pode dar origem a valores mais baixos de PSNR (*Peak Signal Noise Ratio*) para um dado *bit rate* (**figura 3.35**) porque os coeficientes AC possuem uma menor exatidão (quando o *bitstream* for cortado) em comparação com os coeficientes DC para um dado *bit rate*, mas em princípio a qualidade subjetiva é maior.

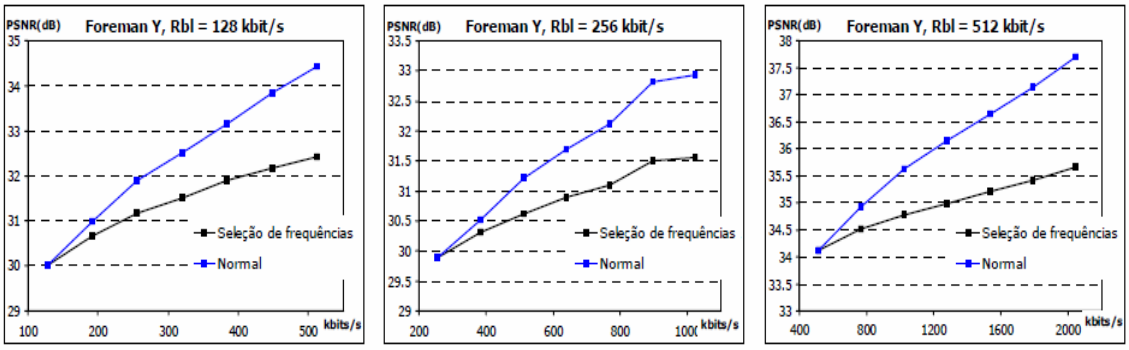


Figura 3.35: Comparação do PSNR para a seqüência "Foreman" com e sem seleção de freqüências para vários *bit rates* (em kbits/s) (JIANG, 1999).

3.5.5.2 Melhoria seletiva

Para alguns quadros da seqüência de vídeo, algumas regiões da imagem podem ser visualmente mais importantes que outras. Para permitir explorar esta característica, a codificação FGS define uma técnica (apresentada com mais detalhe em (SCHAAR, 1999)) que permite privilegiar algumas regiões da imagem através da colocação dos planos de bit dos macroblocos de interesse antes de outros no *bitstream*. Assim quando o *bitstream* da camada superior for truncado, estes macroblocos possuem uma maior qualidade (exatidão) que os restantes. Em um codificador não escalável, esta funcionalidade é alcançada através do controle, ao nível do macrobloco, do passo de quantificação. No codificador FGS, utiliza-se a mesma técnica já usada para a seleção de freqüências anteriormente apresentada, elevar os planos de bit pertencentes a um macrobloco para um plano de bit superior o que é equivalente a multiplicá-los por uma potência de dois. Um elemento da sintaxe (*shifted_bit_planes*) é utilizado para especificar o fator de deslocamento dos planos de bit selecionados. O fator de deslocamento máximo permitido na codificação MPEG-4 FGS é cinco, pois proporciona ao codificador flexibilidade suficiente da melhoria de regiões de interesse.

É importante salientar que apenas um número limitado de macroblocos deve ser selecionado para ser privilegiado, de forma a obter uma melhoria observável na qualidade da imagem. Além disso, o desempenho objetivo de um codificador FGS que utilize esta técnica pode ser globalmente menor se o número de fatores de deslocamento usado for muito significativo. Nota-se que o principal objetivo desta técnica não é melhorar a eficiência da codificação em geral, mas sim melhorar a qualidade subjetiva do vídeo decodificado, privilegiando a qualidade das zonas subjetivamente mais importantes.

Para se obter maior qualidade visual a qualquer *bit rate*, este algoritmo deve ser combinado com um método de segmentação que identifique as regiões visualmente mais importantes de uma seqüência. Este método, a ser combinado com o codificador FGS, deve possuir baixa uma baixa complexidade, especialmente para aplicações em tempo real. Em Schaar (2001) é proposto um sistema de detecção de faces, combinado com o método de melhoria seletiva do codificador FGS. Como se pode observar na **figura 3.36**, este sistema permite uma melhoria da qualidade na região da face a custa de alguma degradação da qualidade no fundo da imagem.

Figura 3.36: Impacto da melhoria seletiva a 250 kbit/s – a) sem melhoria seletiva; b) com melhoria seletiva (SCHAAR, 2001).

Resumindo, a utilização das ferramentas de quantificação adaptativa apresentadas permite melhorar a qualidade subjetiva do vídeo decodificado. Esta melhoria pode ser alcançada através de algoritmos eficientes de deslocamento dos macroblocos (usando memória seletiva). Desta maneira, o desafio na otimização do codificador do FGS é o desenvolvimento de algoritmos que se adaptem a diferentes seqüências de vídeo, diferentes cenas na mesma seqüência e diferentes regiões em um quadro de vídeo; estes algoritmos, residindo no codificador, não são normativos, e como tal podem evoluir em resultado da investigação e competição entre implementações.

3.5.6 Resiliência a erros

Nas aplicações de distribuição de vídeo em canais com erros, a codificação MPEG-4 FGS possui ferramentas adequadas para uma transmissão robusta de vídeo. Em primeiro lugar, a própria representação escalável do vídeo codificado permite que o decodificador facilmente possa recuperar-se de erros que possam ocorrer na camada superior, uma vez que não existe dependência entre quadros consecutivos na camada superior. Em segundo lugar, a estrutura em camadas permite a atribuição de diferentes prioridades a informação codificada, facilitando a atribuição de diferentes níveis de proteção ao vídeo codificado, através de técnicas de codificação de canal. Na estrutura de codificação FGS, a camada base possui uma sensibilidade elevada a erros de codificação. Quaisquer tipos de erros podem levar o decodificador a perda de sincronismo e os erros se propaguem até o próximo GOP. A norma MPEG-4 Visual inclui algumas técnicas de resiliência a erros que facilitam a transmissão de vídeo comprimido em canais com erros, as mais conhecidas são a sintaxe de partição de dados, resincronização, códigos RVLC (*Reversible Variable Lenght Codes*), introdução de HECs (*Header Extension Code*) e técnicas NEWPRED (TALLURI, 1998) (WANG, 1998). Como a codificação da camada base do MPEG-4 FGS é feita de modo conforme com o perfil MPEG-4 Visual ASP e este inclui as técnicas acima referidas de resiliência a erros, então a camada base pode utilizar estas ferramentas. Para o codificador da camada superior, surgiu a necessidade de incluir algumas ferramentas de resiliência a erros de forma a melhorar a robustez do *bitstream* em canais com erros (ex: canais móveis (YAN, 2000)). Para se obter um compromisso adequado entre a informação a adicionar, a eficiência de codificação e a robustez aos erros do canal, a sintaxe da

camada superior inclui apenas a capacidade de introduzir marcas de sincronismo (*fgs_resync_marker*) tão frequentemente quanto o decodificador o desejar.

Devido a utilização de códigos VLC, quando ocorre um erro de transmissão, o decodificador normalmente perde o sincronismo como o codificador, uma vez que o comprimento dos códigos é variável e implícito. Se nenhum dos mecanismos de resiliência a erros forem introduzidos, os bits seguintes são decodificados incorretamente e eventualmente o decodificador irá detectar a ocorrência do erro (ex: através de um código VLC inválido ou de um parâmetro não permitido), tentando em seguida recuperar o sincronismo (ver **figura 3.37**). Nestas condições, uma parte significativa do *bitstream* não poderá ser utilizada, degradando a qualidade do vídeo de forma significativa.

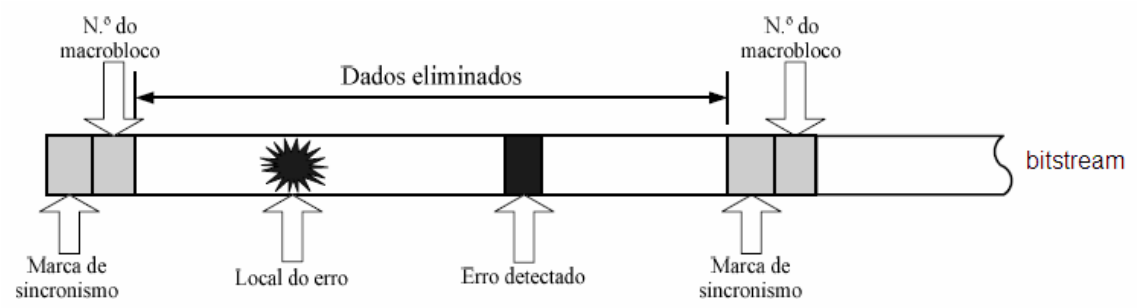


Figura 3.37: Exemplo (pessimista) do processo de decodificação com marcas de sincronismo.

As marcas de sincronismo também podem ajudar na detecção de erros, pois o decodificador pode determinar se um pacote de vídeo (dados entre marcas de sincronismo) foi corretamente decodificado ou não verificado se o número de macrobloco que se encontra a seguir a marca de sincronismo é válido ou não. Estas marcas são códigos únicos, uma seqüência de bits que não pode ser emulada pelo codificador por nenhum código ou combinação de códigos.

A **figura 3.38** ilustra a estrutura do *bitstream* FGS com parcas de sincronismo. Um elemento de sintaxe (*fgs_resync_marker_disable*) pode ser utilizado para ativar o desativar a utilização de marcas de sincronismo. No entanto, com ou sem marcas de sincronismo, existe sempre um elemento da sintaxe, *fgs_bp_start_code*, que serve para separar planos de bit pertencentes ao mesmo quadro. Este código possui duas finalidades:

- Funcionar como marca de sincronismo para tornar o *bitstream* mais robusto a erros.
- Permitir que o servidor e/ou o decodificador identifique o começo de um plano de bit sem ser necessário decodificar os dados por completo. Por exemplo, em um ambiente *multicast* o decodificador pode identificar e subscrever um número de planos de bit de acordo com a sua capacidade de processamento, *bit rate* disponível, etc.

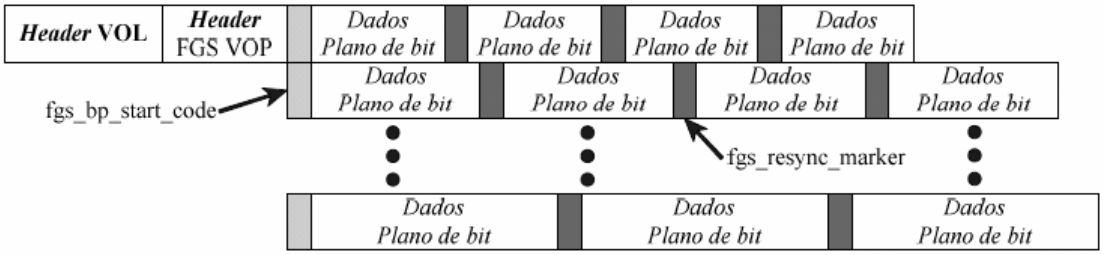


Figura 3.38: Estrutura do *bitstream* da camada superior com marcas de sincronismo.

O elemento de sintaxe *fgs_resync_maker* é constituído por uma palavra de 23 bits (22 bits a '0'e um bit a '1') e é seguido pelo número do macrobloco que corresponde a um código VLC com comprimento entre 1 e 14 bits. Esta técnica permite aumentar até 3 vezes o número de bits corretamente decodificados, dependendo do *bit rate* utilizado e da taxa de erros (YAN, 2000).

4 H.264 / MPEG-4 AVC

O principal objetivo da norma H.264/AVC é fornecer uma nova forma de codificação de vídeo que possua um elevado desempenho. Deste modo, as principais características definidas para esta norma são (CHIARIGLIONE, 2001):

- Desempenho elevado: Redução em cerca de 50% do *bit rate* para a mesma qualidade em relação às normas H.263++ ou MPEG-4 ASP (*Advanced Simple Profile*), para qualquer nível de qualidade.
- Máxima simplificação: Adoção de uma arquitetura simples, utilizando blocos conhecidos com uma complexidade reduzida.
- Adaptação a serviços em diferentes requisitos de atraso: Deve permitir serviços em tempo real ou com atraso reduzido (ex: videotelefonia), bem como serviço sem quaisquer restrições de atraso (ex: armazenamento e distribuição de vídeo).
- Resiliência a erros: Deve incluir ferramentas com intuito de minimizar o impacto resultante da perda de pacotes e erros de bit em redes móveis ou fixas.
- Escalabilidade da complexidade do codificador / decodificador: A assimetria entre a complexidade do codificador e do decodificador deve ser alta (de forma a haver um grande número de terminais capazes de decodificar conteúdo codificado com o H.264/AVC) e deve haver escalabilidade entre a quantidade de processamento do codificador e a qualidade alcançada.
- Adaptação às características da rede de transmissão (*network friendliness*): Deve haver um conjunto de mecanismos que facilitem o transporte do *bitstream* codificado em redes com características diferentes.

Outra função desta norma é a integração com normas já existentes, nomeadamente com a parte de Sistema do MPEG-4 (ex: MP4), com a parte de sistema da norma MPEG-2 e com as recomendações H.320 e H.323. Essa integração poderá requerer algumas alterações nas normas existentes (anexos), de forma a tirar melhor proveito te todas as potencialidades do H.264/AVC.

4.1 Arquitetura

A norma H.264/AVC especifica duas camadas principais de representação: uma camada de codificação de vídeo VCL (*Video Coding Layer*) que permite representar eficientemente o conteúdo de uma seqüência de imagens e uma camada de adaptação de rede NAL (*Network Adaptation Layer*) que formata essa representação para forma

adequada ao transporte em qualquer rede de comunicação ou meio de armazenamento. A **figura 4.1** mostra a relação entre as camadas VCL e NAL.

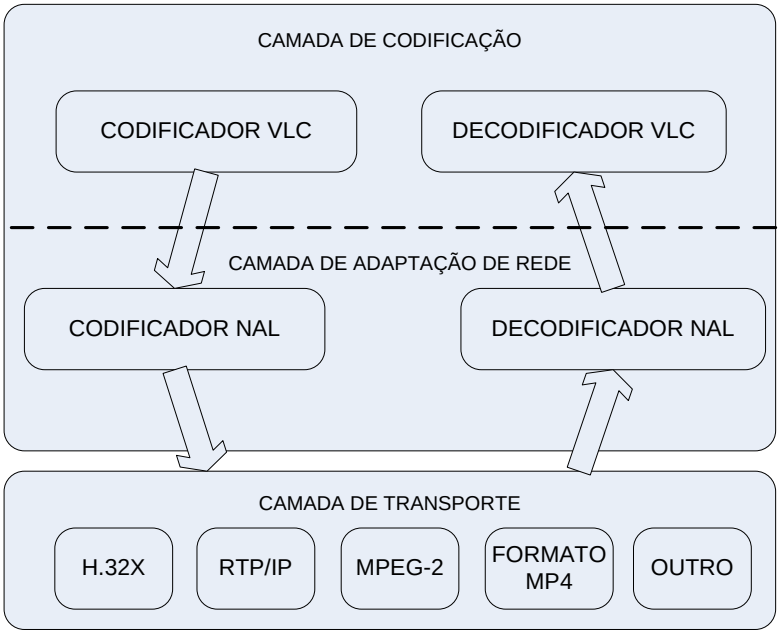


Figura 4.1: Arquitetura genérica da norma H.264/AVC.

De acordo com a figura, o transporte e encapsulamento dos dados codificados pela norma H.264/AVC nos diferentes sistemas de transporte, tais como H.32x, MPEG-2 Sistema e RTP/IP estão fora do âmbito da norma e devem ser especificados pelos órgãos internacionais responsáveis (IETF para o RTP/IP). O nível mais alto de abstração do VCL é o nível *slice* que é constituído por um conjunto de macroblocos pertencentes a uma imagem. A camada NAL funciona como uma abstração entre o VCL e a rede utilizada no transporte do *bitstream* codificado. Tanto o codificador VCL como a NAL podem conhecer as propriedades e características de uma rede de transporte, tais como a taxa de perda de pacotes esperada e efetiva, MTU (*Maximum Transfer Unit*) ou o *jitter* no atraso da transmissão. O codificador VCL pode explorar essas características através do ajuste de alguns parâmetros ou quando utiliza as técnicas de resiliência a erros.

4.1.1 Camada de adaptação de rede

Um conjunto bastante amplo de redes podem apresentar-se como candidatas válidas para transportar o *bitstream* segundo a norma H.264/AVC. As redes de transporte são hoje bastante heterogêneas uma vez que as suas características diferem-se significativamente, em termos de largura de banda, protocolos disponíveis, garantias QoS (*Quality of Service*), empacotamento, configuração interna, etc. A camada de adaptação de rede recebe um *slice* da camada de codificação de vídeo e possui um conjunto de mecanismos e interfaces para efetuar o mapeamento entre o *slice* codificado e a rede de transporte.

As unidades NAL são unidades elementares de transporte e são obrigatoriamente constituídas por um cabeçalho (1 *byte*) e dados codificados. As unidades NAL possuem as seguintes propriedades:

- São decodificadas independentemente. Uma unidade NAL não possui referência a outras.

- Podem ser diretamente mapeadas em sistemas baseados em rede de pacotes (através de pacotes RTP).
- O cabeçalho de uma unidade NAL indica o tipo de dados que possui (ex: *slice* do tipo Intra), a importância relativa da informação que transporta e um *flag* de erro que serve para indicar a ocorrência de erros de bits nos dados que transporta.

A camada de adaptação de rede também define um conjunto de funcionalidades incluídas para as unidades NAL (que podem ser suportadas ou não pela rede de transporte), a sintaxe e a semântica de informação adicional (ex: informação temporal) e um esquema de agregação e fragmentação de unidades NAL, necessário para segmentar unidade com uma dimensão elevada ou agrupar unidades NAL com uma dimensão reduzida.

No entanto, como nem todos os protocolos de transporte são baseados em pacotes, a norma H.264/AVC também define um formato de *bitstream* contínuo. Neste formato, os limites de uma unidade NAL são identificados através de códigos únicos orientados ao *byte*, de forma que o decodificador possa extrair as unidades NAL de uma forma simples e rápida.

Um dos principais problemas no transporte de vídeo em redes sujeitas a erros resulta da natureza estruturada da codificação de vídeo. A informação nos cabeçalhos dos *slices*, imagens, GOPs ou seqüências é transmitida uma única vez no começo de cada *slice*, imagem, GOP ou seqüência e é bastante importante para a decodificação correta do *bitstream* transmitido. Uma perda de um pacote ou um erro de *bit* num desses cabeçalhos possui um efeito desastroso, uma vez que todos os dados que dependem da informação codificada. Para resolver este problema o H.264/AVC utiliza um mecanismo chamado conjunto de parâmetros (*parameter set*). Como a maior parte dos parâmetros enviados ao nível da seqüência / GOP / imagem se mantém constantes durante a transmissão, esses são enviados de forma assíncrona (no início da transmissão) e de forma robusta (com mecanismo de retransmissão).

O codificador e o decodificador NAL mantém um ou mais conjuntos de parâmetros sempre sincronizados. Os parâmetros frequentemente alterados são enviados ao nível do *slice* em conjunto com uma referência que indica qual dos conjuntos de parâmetros disponíveis no decodificador deve ser utilizado para reconstruir um dado *slice*.

4.1.2 Camada de codificação de vídeo

A arquitetura de codificação de vídeo do H.264/AVC corresponde a um esquema híbrido baseado em blocos (**figura 4.2**), tal como qualquer das normas anteriores de codificação de vídeo ITU-T e MPEG. Dessa forma, continua-se utilizando predição entre imagens para reduzir a redundância temporal e uma transformada baseada em blocos para compactar a energia do sinal residual, permitindo assim explorar a redundância espacial. Os ganhos em termos de desempenho alcançados com esta norma não resultam de uma dada ferramenta específica, mas sim de novas formas de efetuar as principais operações de codificação no contexto de uma arquitetura híbrida que combinadas permitem um ganho significativo.

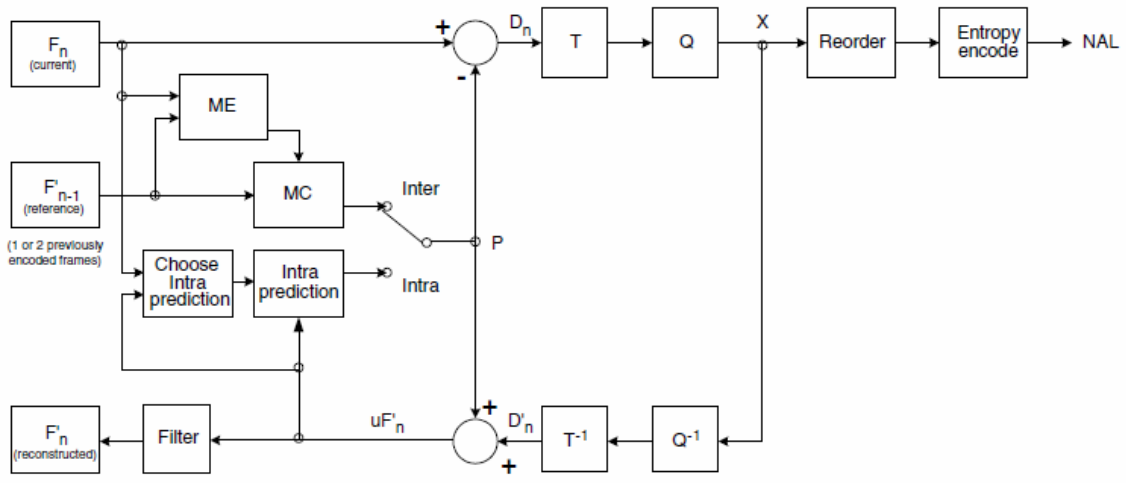


Figura 4.2: Arquitetura do codificador de vídeo H.264/AVC.

Abaixo é apresentada uma breve descrição dos principais módulos, salientando as alterações mais relevantes:

- Módulo de compensação de movimento: Este módulo é o principal responsável pelo acréscimo de desempenho do codificador H.264/AVC (SAPONARA, 2002). Foram introduzidas várias alterações como a compensação de movimento estruturada em árvore (com blocos hierárquicos de dimensão variável), múltiplas imagens de referência, novos filtros de interpolação com ¼ de *pixel* de precisão e imagens B generalizadas.

- Filtro de redução do efeito de bloco: Um dos mais conhecidos artefatos da arquitetura híbrida de codificação baseada em blocos é o efeito de bloco. O H.264/AVC propõe uma solução para este problema, introduzindo o filtro de *loop* de decodificação, antes da compensação de movimento, permitindo uma melhoria significativa da qualidade e uma redução de 5 a 10% (AU, 2002) do *bit rate* para a mesma qualidade (entre 26 dB e 34 dB).

- Transformada inteira: O H.264/AVC utiliza uma transformada baseada na DCT para converter um bloco de amostras para o domínio da freqüência. No entanto, foram introduzidas várias alterações: a transformada é agora definida com valores inteiros, evitando-se erros entre diferentes implementações da transformada (*mismatch*) e possui uma complexidade reduzida pois pode ser calculada sem utilizar multiplicações. Uma das principais inovações é o fato da transformada ser definida para blocos de 4x4 amostras, ao contrário das normas anteriores (8x8 amostras).

- Codificação entrópica: Existem dois modos de codificação entrópicas definidos no H.264/AVC. Um modo baseado em códigos VLC e outro em codificação aritmética. Ambos os códigos foram otimizados. O primeiro com introdução de códigos VLC Exp-Golomb e códigos VLC adaptativos CAVLC. O segundo, codificador aritmético CABAC (*Context Based Adaptative Binary Arithmetic Coding*), com um módulo de modelação de contextos que permite a adaptação dinâmica à estatística dos símbolos e um módulo de binarização para converter um valor não binário em uma seqüência de decisões binárias, referidas como *bins*. O CABAC apresenta melhorias significativas em relação ao codificador baseado nos códigos VLC

(MOCCAGATTA, 2002) (6,3 a 31% para seqüências como resoluções SD e HD TV).

- Novos modos de predição Intra: No MPEG-4 Visual, existem dois modos de predição Intra para o coeficiente DC e para os coeficientes AC na primeira linha e coluna de um bloco 8x8. O H.264/AVC estende este conceito utilizando predição para todos os valores das amostras contidos em um bloco ou macrobloco Intra. Além disso, o codificador pode escolher entre vários modos de predição de forma a obter um erro de predição mais baixo possível. A codificação Intra apresenta um desempenho comparável à recente norma JPEG2000 baseada em *wavelets* (HALBACH, 2002).

Por fim, um fato bastante importante no desempenho de codificação: o controle do codificador. Apesar da escolha dos parâmetros de codificação não ser normativa, o codificador tem ao seu dispor muitas opções e as decisões que toma influenciam fortemente no desempenho em termos de eficiência. Por exemplo, existem 9 modos de predição Intra 4x4, 4 modos de predição Inter para um macrobloco onde cada sub-macrobloco pode usar um dos 4 modos de predição. A utilização de múltiplas referências para a compensação de movimento permite mais um grau de liberdade na escolha da imagem de referência, como conseqüências tanto em termos de memória como em ternos de processamento. Torna-se difícil, senão impossível, testar todas as combinações de parâmetros possíveis. Algoritmos que otimizem as escolhas do codificador, minimizando a distorção e o tempo de processamento em relação ao *bit rate*, possuem uma importância vital, pois permitem explorar todas as potencialidades desta norma e maximizar seu desempenho.

4.1.3 Estrutura da sintaxe de codificação de vídeo

A nível de imagem, a norma H.264/AVC suporta a codificação de vídeo, com sub-amostragem 4:2:0, no formato progressivo ou entrelaçado ou mesmo em ambos os tipos, simultaneamente na mesma seqüência. Um quadro entrelaçado possui dois campos, um superior e um inferior.

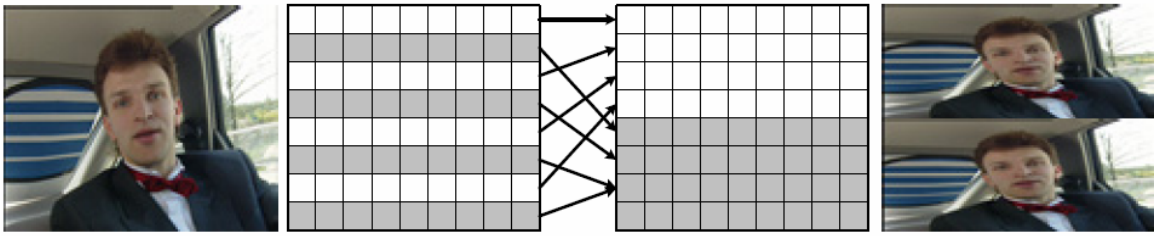


Figura 4.3: Codificação de um quadro entrelaçado em modo campo

Os dois campos de um quadro entrelaçado podem ser codificados separadamente ou em conjunto. O codificador pode escolher a forma como codifica os quadros entrelaçados, campos separados ou em conjunto, nível de seqüência, quadro ou macrobloco. Quando uma determinada cena de uma seqüência de vídeo contém detalhes significativos, mas o movimento é baixo, deve-se codificar ambos os campos em conjunto (modo quadro). Quando a cena de vídeo contém muito movimento, deve-se codificar os campos separadamente (modo campo), de maneira a que o segundo campo possa ser predito a partir do primeiro. Como pode haver regiões de uma cena de vídeo que são mais eficientemente codificadas em modo quadro e outras em modo campo, o

codificador pode escolher o tipo de codificação (quadro ou campo) utilizado ao nível macrobloco.

A nível de *slice*, um quadro pode ser dividido em um ou mais *slices*, correspondendo cada *slice* a uma dada área da imagem. Um *slice* é constituído por um número inteiro de macroblocos ou pares de macroblocos e pode ser decodificado independentemente dos restantes *slices*. Um macrobloco possui um tamanho fixo e cobre uma área retangular de 16x16 amostras da componente de luminância e 8x8 amostras de cada componente de crominância. Os macroblocos que pertencem a um *slice* podem depender uns dos outros em termos de codificação. Quando se codificam imagens entrelaçadas, cada *slice* tem de conter um número inteiro de pares de macroblocos, tal como é ilustrado na **figura 4.4a**. Dependendo do perfil, os *slices* pertencentes a uma imagem podem estar organizados no *bitstream* de uma forma arbitrária ou não, não tendo que obedecer necessariamente a ordem de varredura da direita para esquerda e de cima para baixo (*raster-scan*) (referida na norma como *Arbitrary Slice Order* - ASO). Os *slices* podem pertencer a uma estrutura denominada por grupo de *slices* que agrupa um ou mais *slices* de uma determinada imagem. Um *slice* contém sempre macroblocos ou pares de macroblocos contínuos e com varredura *raster-scan* dentro de um grupo de *slices*. No entanto, os macroblocos que pertence a um determinado grupo de *slices* não obedecem necessariamente a esta regra. A **figura 4.4b** apresenta um exemplo em que macroblocos que pertencem ao mesmo grupo de *slices* não são contínuos, uma vês que os *slices* pertencentes ao grupo 1 estão intercalados espacialmente com os *slices* que pertencem ao grupo 2.

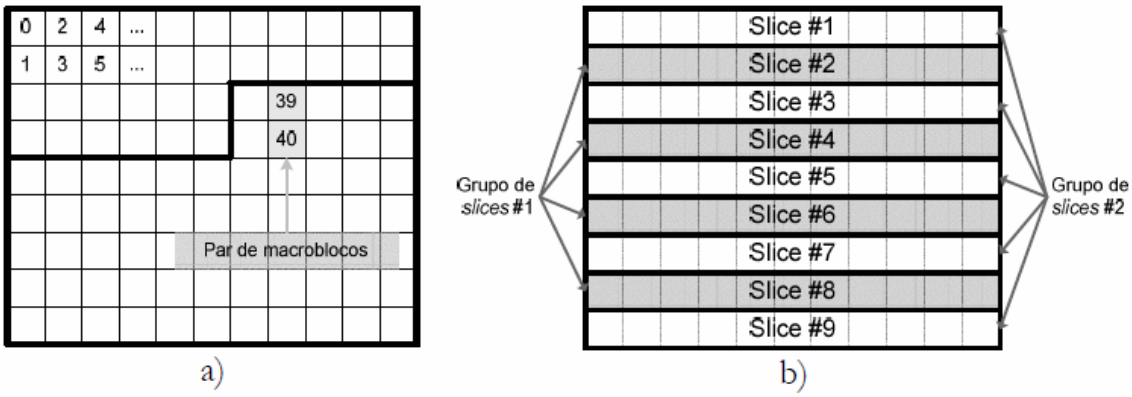


Figura 4.4: Divisão de uma imagem em: a) pares de macroblocos e b) *slices* e grupos de *slices*.

A ordem de transmissão dos macroblocos no *bitstream* depende do mapa de alocação de macroblocos *Macroblock Allocation Map* (MBAmp). O MBAmp constitui a forma do codificador indicar ao decodificador qual a ordem de transmissão dos macroblocos e consiste num inteiro por macrobloco que indica o grupo de *slices* a que este pertence. Este inteiro está entre 0 e 7, uma vez que não são permitidos mais de oito grupos de *slices* por quadro. O mapa de macroblocos pode obedecer uma estrutura regular, retangular ou não, ou a um padrão completamente aleatório.

Este mecanismo de estruturação de vídeo é referido como *Flexible Macroblock Ordering* (FMO) e possui várias vantagens quando combinado com o ASO.

Considerando o exemplo da **figura 4.5a** em que os macroblocos são agrupados em três grupos de *slices* retangulares e cada grupo de *slices* possui um único *slice*. Assume-se por simplicidade que cada um destes *slices* (ou grupo) possui o mesmo número de bits. Em redes IP, usando os protocolos UDP e RTP, os pacotes que transportam os *slices* podem ser recebidos fora de ordem. O decodificador pode decodificar qualquer *slice* independentemente da ordem que foi recebido, permitindo o caso de um dos *slices* sofrer atraso (devido ao *jitter* presente na rede de transmissão), o decodificador pode começar a processar outros *slices* recebidos e assim reduzir o atraso global de todo o sistema de distribuição.

Na **figura 4.5b** é apresentado um outro exemplo com dois grupos de *slices* em que os macroblocos pertencem alternadamente a um grupo ou outro (formando um tabuleiro de xadrez). Se um dos *slices* se perder devido a ocorrência de erros, os macroblocos ao redor daqueles que foram perdidos estão disponíveis para efetuar o cancelamento de erros. No entanto, o desempenho em termos de eficiência é afetado, uma vez que não se pode explorar a redundância espacial entre macroblocos vizinhos.

Na **figura 4.5c** é apresentado outra utilização para o mapa de macroblocos que faz uso desta estrutura para melhorar a qualidade de uma região de particular interesse na imagem quando ocorrem erros na transmissão, através da utilização de *slices* redundantes (no H.264/AVC os *slices* podem ocupar áreas sobrepostas). A idéia consiste em transmitir informação redundante para determinadas regiões da imagem e assim, quando ocorrerem erros durante a transmissão, o decodificador conseguirá reconstruir uma parte da imagem sem erros. No exemplo apresentado, existem três grupos de *slices*: o grupo 3 engloba toda a área da imagem, enquanto os grupos 1 e 2 são transmitidos ou decodificados, apenas quando ocorrerem erros de transmissão, de forma a melhorar a qualidade de uma determinada zona da imagem. Por exemplo, se ocorrerem erros durante a transmissão do grupo de *slices* 3 nas áreas cobertas pelo grupo de *slices* 1 e / ou 2, o decodificador pode substituir as áreas da imagem onde os erros ocorreram, decodificando os grupos 1 e 2 (assume-se que durante a transmissão não ocorreram erros na transmissão dos grupos 1 e 2 no mesmo local).

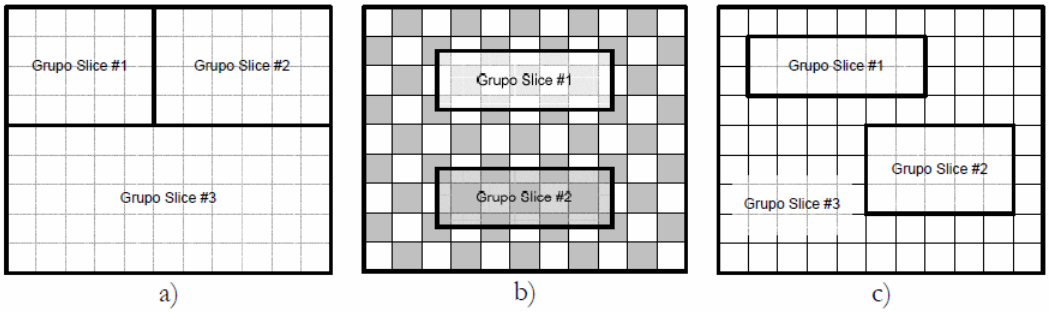


Figura 4.5: Exemplos de *slices* e grupos de *slices*: a) *slices* dispersos b) tabuleiro de xadrez c) *slices* redundantes.

Para todos os exemplos aqui apresentados, é necessário o envio de um mapa de macroblocos para o decodificador. Este mapa é e eficientemente codificado para o caso de *slices* retangulares, mas para outros tipos de mapas o desempenho pode ser penalizado, especialmente se este mapa for freqüentemente alterado por quadros sucessivos.

Dependendo do perfil, os dados codificados para um único *slice* podem ser divididos em três partições separadas. Esta ferramenta é referida pelo H.264/AVC como *Data Partitioning* (DP). As três partições definidas são as seguintes:

- DPA (*Data Partitioning A*): Contém a informação de cabeçalho do *slice* e do macrobloco e os vetores de movimento. Esta partição a mais importante.
- DPB (*Data Partitioning B*): Contém a informação de textura codificada no modo Intra. Esta partição elimina erros de predição.
- DPC (*Data Partitioning C*): Contém a informação de textura codificada no modo Inter.

Cada partição começa por uma palavra de código que indica o tipo de partição. Através deste campo é possível atribuir maior prioridade a uma maior proteção a um certo tipo de informação em perda de outra e, através deste mecanismo, melhorar a qualidade da imagem recebida. Numa rede baseada em pacotes (ex: H.324), estas partições do *slice* podem ser diretamente mapeadas num pacote e transmitidas, facilitando o cancelamento de erros. Por exemplo, se apenas a partição DPC for perdida ainda se consegue decodificar a parte Intra do *slice* e utilizar o vetores de movimento da partição DPA, para obter a textura dos macroblocos codificados no modo Inter. Esta ferramenta possui um impacto mínimo na complexidade do codificador, pois apenas é necessário estruturar a informação codificada de uma forma diferente (apenas a sintaxe é diferente).

Além do grupo de *slices* atrás apresentado, uma estrutura intermediária entre o nível da imagem e o nível de *slice*, no H.264/AVC foi introduzido um novo nível de dados entre o nível do macrobloco e do bloco, o sub-macrobloco. Nas normas anteriores, ao nível de macrobloco (16x16 amostras de luminância) efetuava-se a estimação e compensação de movimento e este continha normalmente 4 blocos (8x8 amostras) de luminância e 2 blocos de crominância. Ao nível do bloco era realizada a transformada DCT que transformava as amostras no domínio do tempo para o domínio da transformada. No entanto, no H.264/AVC, a estimação e compensação de movimento podem ser realizadas a nível de macrobloco ou a nível de uma partição do macrobloco, o sub-macrobloco, ou até ao nível do bloco. Desta maneira, na norma H.264/AVC, um macrobloco possui 16x16 amostras de luminância (e as 8x8 amostras de cada crominância correspondentes, isto para formatos 4:2:0), tal como as normas anteriores. Um sub-macrobloco possui um quarto das amostras de um macrobloco (8x8 amostras de luminância) alinhado com os contornos do macrobloco, e um bloco é a menor unidade de processamento possuindo a dimensão de 4x4 amostras. A **figura 4.6** exemplifica a divisão do macrobloco em sub-macroblocos e blocos.

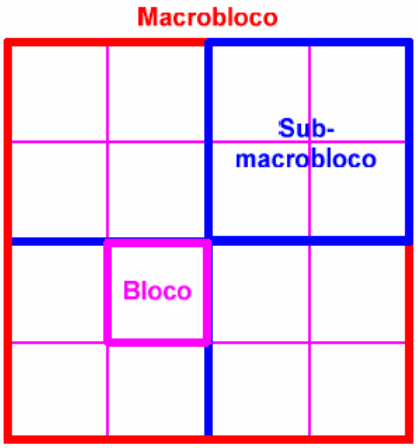


Figura 4.6: Divisão de um macrobloco em sub-macroblocos e blocos.

As amostras de um bloco, independentemente da sua dimensão, são convertidas para o domínio da transformada, tal como nas normas anteriores. Outra novidade do H.264/AVC ao nível da sintaxe de vídeo é a forma como os coeficientes da transformada são organizados no *bitstream*. A **figura 4.7** apresenta a ordem de varredura dos blocos de um macrobloco de 16 blocos 4x4.

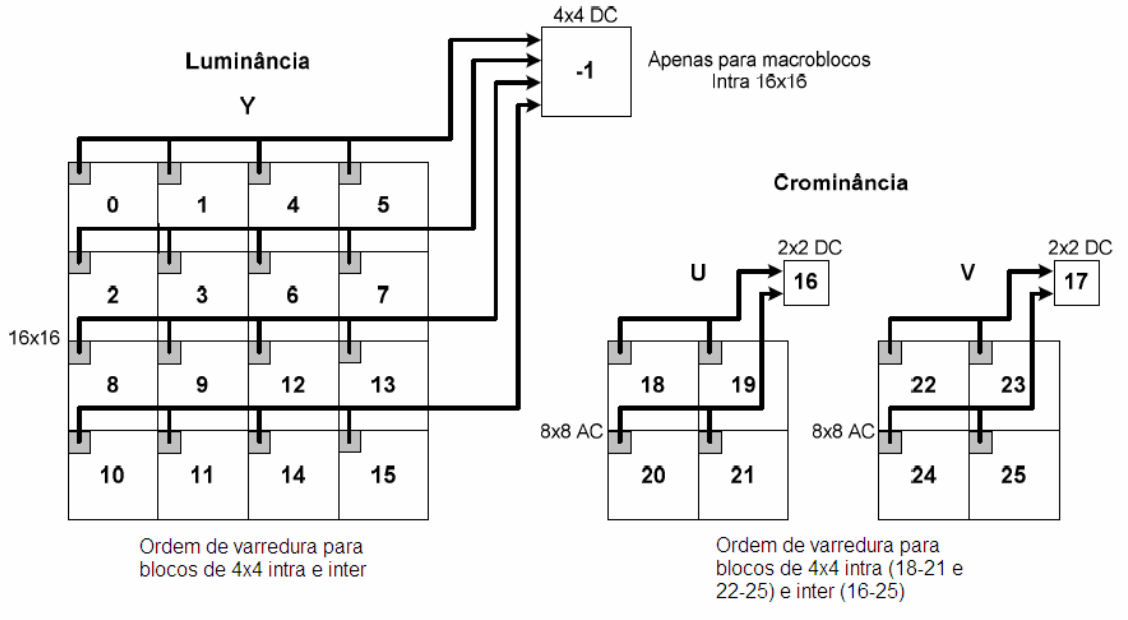


Figura 4.7: Ordem de varredura de um macrobloco.

Num macrobloco, todos os coeficientes de um determinado sub-macrobloco são enviados da seguinte forma: se o modo de codificação for o modo Intra 16x16 (sem predição temporal), os coeficientes DC da transformada são enviados primeiro, sendo enviados em seguida os blocos luma 0-15 na ordem indicada na **figura 4.7**, com o coeficiente DC a zero nos macroblocos Intra 16x16. Os blocos 16 e 17 contém cada um 4 coeficientes DC de cada uma das componentes de crominância. Finalmente, os blocos de crominância são enviados com os coeficientes DC a zero.

4.2 Ferramentas de codificação de vídeo

O H.264/AVC considera cinco tipos de *slices*: Em um *slice* tipo I (Intra), os macroblocos são codificados sem utilizar qualquer outra imagem da seqüência de vídeo, a predição Intra é sempre obtida a partir de macroblocos ou blocos que pertencem à mesma imagem. Por outro lado, os *slices* do tipo Inter (P ou B) podem utilizar uma ou mais imagens de referência, anteriormente decodificadas, para obter uma predição Inter para macroblocos do tipo P ou B. Os *slices* do tipo P podem conter macroblocos do tipo I ou P e os *slices* do tipo B podem conter macroblocos do tipo I ou B. Os dois restantes tipos de *slices*, SI e SP, foram definidos com vista a permitir a reconstrução de uma imagem quando diferentes imagens de referência são utilizadas ou quando estas não se encontram disponíveis.

Na norma H.264/AVC são definidos sete tipos de imagens que indicam quais os tipos de *slices* que podem ser utilizados para um determinado tipo de imagem. Os três primeiros são semelhantes as normas anteriores, a imagem do tipo I apenas contém *slices* I, a imagem do tipo P contém *slices* I ou P e a imagem do tipo B contém *slices* I, P ou B. Os tipos restantes permitem que os *slices* SI ou SP sejam utilizados em vários tipos de imagens, caracterizadas pelo tipo de *slices* que contém: a) SI; b) SI,SP; c) I, SI; d) I, SI, P, SP; e) I, SI, P, SP, B.

Depois de efetuar a predição Intra ou Inter, aplica-se a transformada 2D ao sinal residual, que o converte em um conjunto de coeficientes o mais descorrelacionado possível, de forma que a energia do sinal esteja contida em um número mínimo de coeficientes. Estes coeficientes são quantificados e codificados entropicamente para se obter o *bitstream* final.

4.2.1 Codificação e predição Intra

A codificação Intra refere-se ao caso em que apenas a redundância especial é explorada na codificação. Um macrobloco pode ser codificado no modo Intra em qualquer tipo de *slice*. Um *slice* que apenas contenha macroblocos do tipo Intra é referido como um *slice* do tipo I ou *slice* Intra. Este tipo de *slice* pode estar presente em qualquer tipo de imagem (I, P ou B). No entanto, uma imagem do tipo I (Intra) só pode conter *slices* Intra.

Como as imagens ou *slices* Intra não utilizam nenhum tipo de predição temporal, necessitam de muitos bits para as representar quando comparadas com outros tipos de imagem (para uma dada qualidade). Para aumentar a eficiência da codificação Intra, o H.264/AVC explora a correlação entre blocos e macroblocos adjacentes de uma imagem, uma vez que estes tem muitas vezes propriedades semelhantes (ex: uma imagem em que o fundo é uniforme ou gradiente).

Se um bloco ou macrobloco for codificado em modo Intra, o primeiro passo de codificação consiste em construir um bloco ou macrobloco de predição a partir dos blocos ou macroblocos que foram anteriormente codificados e reconstruídos. O bloco ou macrobloco de predição é subtraído ao bloco ou macrobloco que se pretende codificar. Nesta predição podem apenas utilizar as amostras que estejam contidas no mesmo *slice* para garantir a independência entre *slices*. A predição em modo Intra pode ser realizada a vários níveis:

- Predição única para todo o macrobloco (Intra 16x16): quatro modos (vertical, horizontal, DC e planar).

- Predições individuais para 16 amostras dos blocos 4x4 (Intra 4x4): nove modos (DC e 8 direcionais).
- Predição única para a crominância: quatro modos (vertical, horizontal, DC e planar).

A componente de luminância pode ser predita da mesma forma para todas as amostras pertencentes a um macrobloco 16x16. Estes modos são referidos como modos Intra 16x16 e são adequados a zonas da imagem que variam suavemente. Ao nível do macrobloco, existem quatro modos de predição que são apresentados na **figura 4.8**. O modo 0 utiliza as amostras reconstruídas do topo do macrobloco de forma a que as colunas do macrobloco de predição tenham o mesmo valor que a amostra que se encontra no topo (**figura 4.8a**). O modo 1 utiliza as amostras reconstruídas que se encontram a esquerda do macrobloco de forma que as linhas do macrobloco de predição tenham o mesmo valor que a amostra que se encontra mais a esquerda em cada linha (**figura 4.8b**). No modo 2, todas as amostras do macrobloco de predição possuem um valor igual a média das amostras reconstruídas do topo e a esquerda do macrobloco (**figura 4.8c**). O modo 3 calcula uma função linear que se adapta as amostras do topo e a esquerda do macrobloco para obter os valores das amostras do macrobloco de predição (**figura 4.8d**).

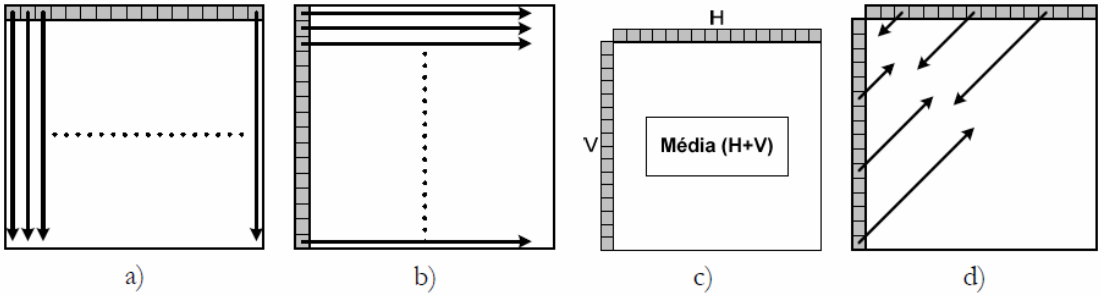


Figura 4.8: Predição Intra em blocos de 16x16 amostras: a) modo 0 – vertical b) modo 1 – horizontal c) modo 2 – DC d) modo 3 – planar.

O H.264/AVC oferece também nove modos de predição ao nível do bloco (**figura 4.9**), que são referidos como modos Intra 4x4. Os primeiros três modos são muito semelhantes aos utilizados ao nível do macrobloco enquanto que os restantes correspondem a direções consideradas importantes. Nos modos 3 e 4, as amostras são interpoladas com um ângulo de 45º. e nos modos 5, 6 e 7 com um ângulo de 26.6º.

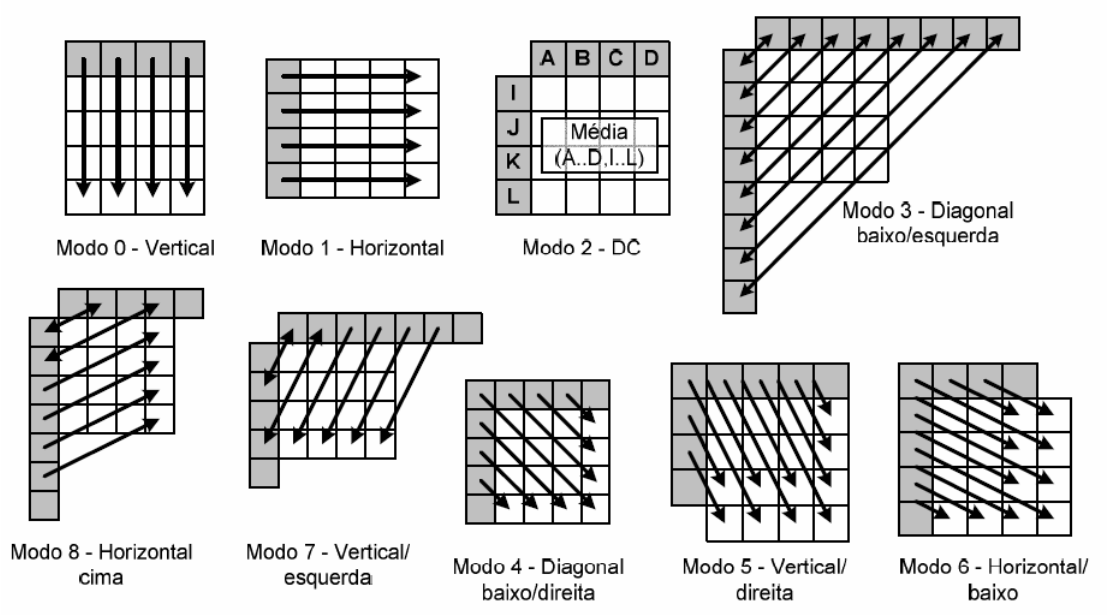


Figura 4.9: Modos de predição Intra para blocos 4x4.

O modo de predição escolhido pelo codificador corresponde normalmente ao modo que apresenta um menor erro de predição e tem de ser transmitido ao decodificador. Como para cada bloco 4x4 existem 9 blocos de predição, o número de bits necessário para o envio do modo é significativo. Felizmente, os modos Intra para os blocos 4x4 estão bastante correlacionados. Por omissão, o modo de predição de um bloco é sempre o modo que possui o menor valor (cada modo possui um valor de acordo com a **figura 4.9**) entre os modos de predição utilizados pelos blocos vizinhos, em cima e a esquerda. No caso de se pretender usar outro modo, é necessário indicar ao decodificador que há uma mudança de modo e enviar o novo modo.

É importante salientar que os modos de predição só podem ser usados quando todas as amostras a partir das quais se efetua a predição estão disponíveis no próprio *slice*. As amostras não estão disponíveis quando pertencem a outro *slice* ou quando o macrobloco a codificar se encontra nos limites da imagem. No entanto, uma exceção é aberta para o modo DC que utiliza apenas as amostras que estão disponíveis para calcular o bloco ou macrobloco de predição.

A crominância em macroblocos Intra é codificada de uma forma muito semelhante aos blocos da luminância de um macrobloco do tipo Intra 16x16. O mesmo modo de predição é comum a ambos os blocos de crominância (U e V), mas é independente do modo de predição utilizado para a luminância. Contudo, se algum bloco de luminância de um macrobloco for codificado no modo Intra, ambos os blocos da crominância são codificados em modo Intra.

4.2.2 Codificação e predição Inter

Em qualquer sistema de codificação de vídeo, cada amostra ou *pixel* pode ser predita a partir de uma ou mais amostras. O preditor Inter mais simples que se conhece é a imagem anteriormente transmitida. No entanto, todas as normas de codificação de vídeo utilizam técnicas mais complexas, de forma a reduzir a redundância temporal em quadros sucessivos, permitindo uma codificação mais eficiente das seqüências de vídeo. A estimação e compensação de movimento é uma das técnicas mais utilizadas. A estimação de movimento consiste em escolher o bloco numa imagem de referência (não

necessariamente a anterior e não necessariamente no passado) que apresente o menor resíduo em relação ao bloco a codificar, e a compensação de movimento consiste em subtrair o bloco encontrado da imagem a codificar. Quando a imagem de referência utilizada para um determinado macrobloco é uma imagem previamente codificada e temporalmente anterior, o macrobloco é referido como um macrobloco do tipo P. Os *slices* do tipo P podem conter macroblocos do tipo P ou I e as imagens do tipo P podem conter *slices* do tipo I ou P. Os *slices* ou imagens do tipo P podem ser preditos a partir de *slices* ou imagens do tipo I, P ou B anteriores. Quando são escolhidas duas imagens de referência para um macrobloco, uma antes e outra depois da imagem a codificar, o macrobloco é referido como um macrobloco do tipo B. Os macroblocos do tipo B apenas podem estar presentes em *slices* do tipo B e estes em imagens do tipo B. A compensação de movimento definida no H.264/AVC inclui a maior parte das características principais das normas anteriores, mas a sua eficiência é melhorada através de novas ferramentas.

4.2.2.1 Compensação de movimento estruturada em árvore

O H.264/AVC permite que um macrobloco seja dividido em partições de dimensão fixa, utilizadas para descrever o movimento. São definidos vários tipos de partições, desde 16x16 a 4x4 amostras de luminância, com muitas opções entre estas duas variantes. A componente de luminância de cada macrobloco (16x16 amostras) pode ser dividida de 4 formas, tal como ilustra a **figura 4.10a**, Inter 16x16, Inter 16x8, Inter 8x16, Inter 8x8, que correspondem a 4 modos de predição ao nível do macrobloco.

Se o modo Inter 8x8 for escolhido, cada um dos sub-macroblocos (com 8x8 amostras) pode ser dividido de novo, obtendo-se partições com dimensões de 8x8, 8x4, 4x8, 4x4, que correspondem a 4 modos de predição ao nível de sub-macrobloco. Este número elevado de modos de predição para cada macrobloco e sub-macrobloco da origem a um número elevado de partições possíveis para uma macrobloco, desde 1 a 16 partições. Este método é referido na literatura como compensação de movimento estruturada em árvore, no H.264/AVC o primeiro nível da árvore é o macrobloco, o segundo o sub-macrobloco e o terceiro o bloco.

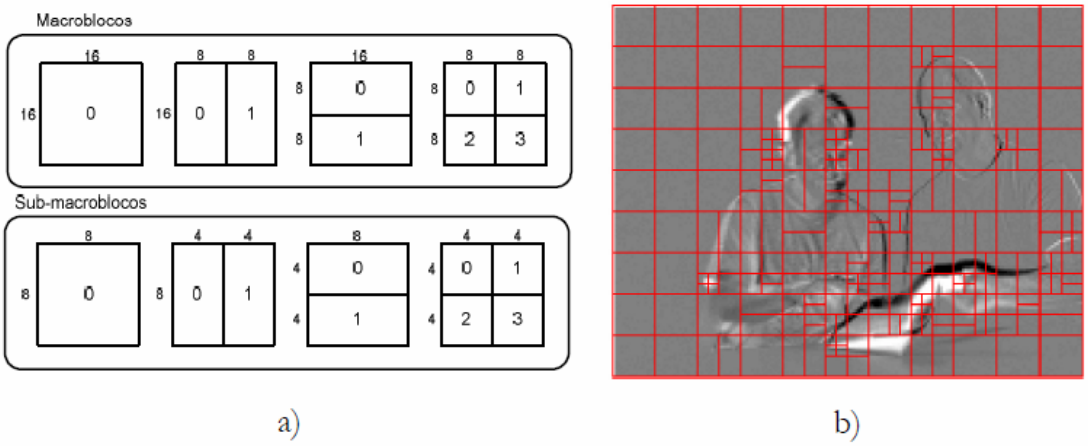


Figura 4.10: Macroblocos e sub-macroblocos: a) partições possíveis; b) escolha da partição conforme o conteúdo da imagem.

Para cada modo de codificação de um macrobloco e sub-macrobloco é necessário enviar um vetor de movimento. Cada vetor de movimento deve ser codificado e transmitido em conjunto como a escolha da partição. A escolha de uma partição como

dimensão elevada (16x16, 16x8 ou 8x16) implica que apenas um pequeno número de bits é necessário para indicar o tipo de partição e os vetores de movimento. No entanto, alguns casos, pode ser mais eficiente escolher partições menores (4x4, 4x8 ou 8x4), uma vez que apresentam um erro de predição menor, mesmo sendo necessário transmitir os vetores de movimento e a escolha do tipo de partição. Estas escolhas não são especificadas pela norma, no entanto, seu impacto no desempenho é considerável o que permite que codificadores de diferentes empresas ofereçam desempenhos bem diferentes. Geralmente, partições de dimensão elevada são escolhidas para zonas homogêneas da imagem e partições de dimensão reduzida escolhidas para áreas com maiores detalhes. A **figura 4.10b** mostra o resíduo de uma imagem com as decisões tomadas pelo software de referência da norma H.264/AVC (SUEHRING, 2006) em termos de partições para a compensação de movimento. O software de referência escolhe a dimensão da partição que minimiza o resíduo e os vetores de movimento codificados. As partições estão sobrepostas (vermelho) na imagem residual; as áreas com pouco movimento estão em cinza e as áreas com muito movimento estão em preto ou branco.

Como se usam formatos 4:2:0, a resolução de cada componente da crominância de um macrobloco (U e V) é metade da componente de luminância, tanto horizontal como verticalmente. Cada bloco de crominância (8x8 amostras) é dividido da mesma forma que a componente de luminância, só que a dimensão de cada partição corresponde a metade da resolução horizontal e vertical da luminância (ex: um bloco de 4x8 amostras de luminância corresponde um bloco 2x4 amostras de crominância). As componentes horizontais e verticais de cada vetor de movimento são divididas por dois quando aplicadas aos blocos de crominância.

4.2.2.2 Compensação de movimento com precisão de ¼ pixel

A precisão da compensação de movimento no H.264/AVC é de ¼ da distância entre amostras da imagem de referência. Nos casos que o vetor de movimento aponta para uma posição inteira na imagem de referência, a amostra da imagem compensada de movimento é igual a amostra que se encontra no local indicado pelo vetor de movimento na imagem de referência. No entanto, se o vetor de movimento aponta para uma posição a ½ ou a ¼ de *pixel*, as amostras da luminância e da crominância não existem na imagem de referência, sendo necessário um processo de interpolação para obtê-las. A compensação de movimento com precisão de ¼ de *pixel* pode aumentar significativamente o desempenho da codificação em comparação com a precisão ½ *pixel* (utilizada na norma MPEG-2 Vídeo), a custa de uma maior complexidade do codificador.

A primeira fase da interpolação consiste em obter as amostras a metade da distância entra amostras. Na **figura 4.11** as amostras em azul representam amostras em posições inteiras e as amostras em verde (a, b, c, d, ..., m) representam as amostras com ½ *pixel* de precisão, metade da distância entre duas amostras azuis). Para obter as amostras a ½ *pixel* de precisão, utiliza-se um filtro *Finite Impulse Response* (FIR) com 6 passos. Os coeficientes do filtro são (1/32, -5/32, 5/8, 5/8, -5/32, 1/32); por exemplo, a amostra c é calculada a partir das amostras E, F, G, H, I, J horizontais da seguinte forma:

$$c = round\left(\frac{E - 5F + 20G + 20H - 5I + J}{32}\right)$$

O h é calculado de forma semelhante a partir das amostras B, D, H, N, S e U verticais. Depois de todas as amostras com ½ *pixels* de precisão adjacentes as posições inteiras estarem calculadas, as amostras restantes (g na imagem) são calculadas a partir da interpolação das amostras já calculadas. Por exemplo, g é calculado a partir das amostras a, b, c, k, l e m. O resultado é o mesmo se g for calculado horizontalmente ou verticalmente. O filtro de interpolação de 6 passos é relativamente complexo (comparando com a interpolação bi linear utilizada no MPEG-4 Visual para o mesmo efeito) mas apresenta um melhor desempenho.

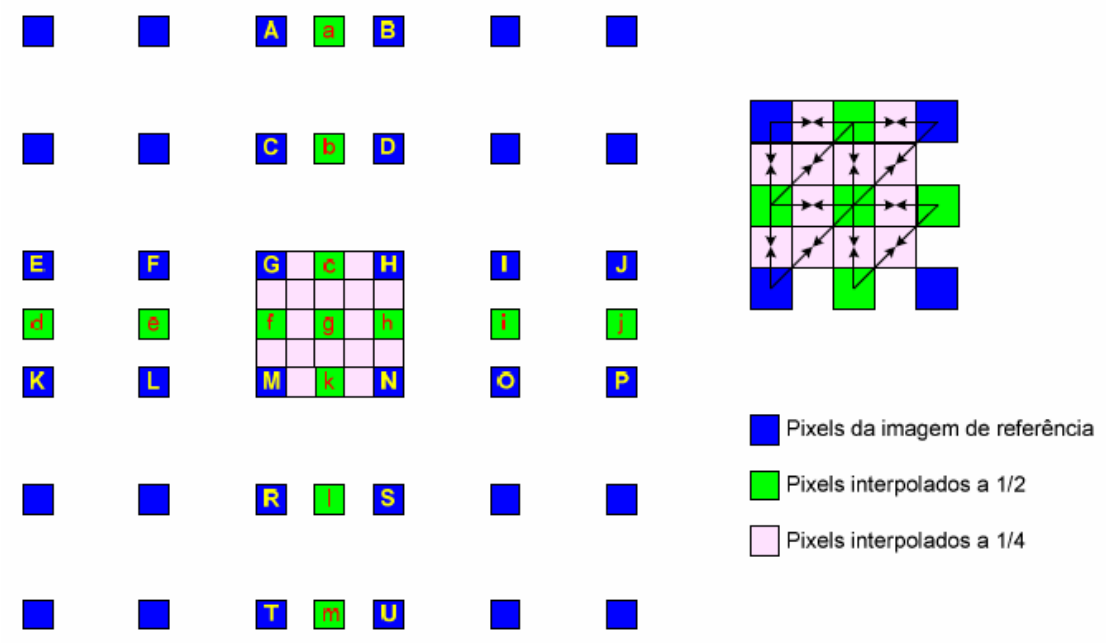


Figura 4.11: Interpolação das amostras com ¼ de *pixel* de precisão.

Depois de todas as amostras nas posições com ½ *pixel* de precisão estarem disponíveis, as posições com ¼ de *pixel* de precisão são obtidas por interpolação linear. As posições a ¼ de *pixel* de precisão são calculadas a partir das duas amostras adjacentes em posições inteiras ou a ½ *pixel* de precisão, tanto horizontalmente como verticalmente (**figura 4.11**).

Os vetores de movimento com ¼ de *pixel* de precisão para a componente de luminância necessitam de vetores de movimento com 1/8 de *pixel* de precisão para ambas as componentes de crominância (no formato 4:2:0). As amostras interpoladas a intervalos de 1/8 de *pixel* entre duas amostras da crominância são obtidas através de uma interpolação linear. Cada amostra na posição a é calculada como uma combinação linear das amostras nas posições A, B, C e D da **figura 4.12**:

$$a = round\left(\frac{(8-dx)(8-dy)A + dx(8-dy)dyC + dxdyD}{64}\right)$$

Os valores dx, dy e 8-dx e 8-dy são definidos na **figura 4.12**.

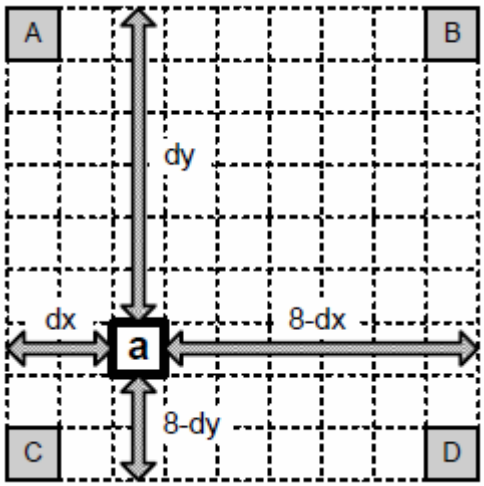


Figura 4.12: Interpolação da componente de crominância com 1/8 de *pixel* de precisão.

A sintaxe do H.264/AVC permite o uso de vetores de movimento sem restrições, os vetores de movimento podem apontar para fora da área da imagem. Neste caso, a imagem de referência é estendida para além dos limites da imagem, repetindo as amostras que se encontram no limite da imagem antes da interpolação. Para codificar um vetor de movimento para cada partição, pode ser necessário um número de bits significativo, especialmente se partições de dimensão reduzidas forem escolhidas. Contudo, vetores de movimento de partições vizinhas são muitas vezes altamente correlacionados entre si. Para explorar este fato, cada vetor de movimento é predito a partir dos vetores de movimento de partições vizinhas já codificadas e transmitidas. O vetor de movimento predito é obtido a partir dos vetores de movimento de partições que se encontram a sua esquerda, em cima, cima-direita e cima-esquerda. Um conjunto de regras que dependem da dimensão das partições vizinhas e do cálculo de simples operações aritméticas é definido em Wiegand (2002).

4.2.2.3 Múltiplas referências

A norma H.264/AVC suporta compensação de movimento com múltiplas referências, mais que uma imagem anteriormente codificada pode ser utilizada simultaneamente como referência para a compensação de movimento. A **figura 4.13** ilustra este conceito para os *slices* do tipo P, preditos a partir de uma ou mais imagens no passado; no entanto, este conceito é também utilizado para os *slices* do tipo B e preditos a partir de uma ou mais imagens no passado e no futuro.

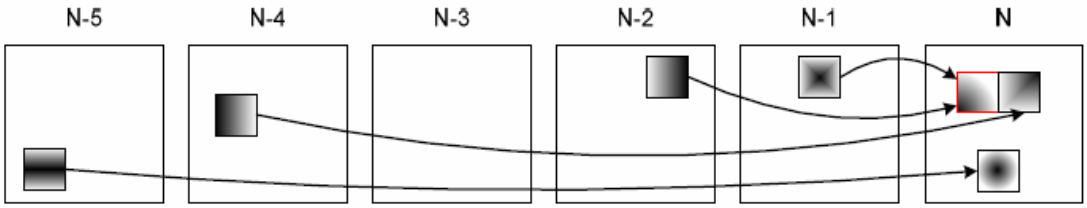


Figura 4.13: Compensação de movimento com múltiplas referências.

Tanto o codificador como o decodificador guardam as imagens de referência para utilizar em uma memória com múltiplas imagens. O decodificador possui as mesmas imagens na memória que o codificador, que através de operações de controle do conteúdo da memória são transmitidas no *bitstream* codificado. Se a dimensão da

memória permitir guardar mais do que uma imagem, o índice da imagem de referência é transmitido para cada partição 16x16, 8x16, 16x8 ou 8x8 de um macrobloco. O índice vai indicar ao decodificador qual é a imagem de referência que deve utilizar de todas as imagens de referência que estão disponíveis na memória.

Além disso, para cada partição de macrobloco é possível utilizar uma predição pesada obtida a partir de duas imagens de referência, r_1 e r_2, da memória (vermelho **figura 4.13**). Esta ferramenta é referida como predição pesada e só pode ser utilizada em alguns perfis. A predição pesada consiste em efetuar a compensação de movimento para as duas imagens de referência r_1 e r_2 e calcular uma soma pesada a partir de um conjunto de pesos w_1 e w_2 e das duas imagens compensadas de movimento, para se obter a predição final. Os pesos w_1 e w_2 são utilizados para as referências r_1 e r_2, respectivamente. Com esta ferramenta é possível codificar transições ou diferenças de intensidade (devido a alterações de iluminação) presentes na maior parte do conteúdo, de uma forma mais eficiente (TIANG, 2002).

A **figura 4.14b** mostra as decisões tomadas pelo software de referência da norma H.264/AVC (SUEHRING, 2006) para uma determinada imagem da seqüência de vídeo *Table Tennis*, em termos da imagem de referência escolhida. As zonas em branco indicam que o índice transmitido é 0 (corresponde a imagem anteriormente codificada, N-1), as zonas em preto indicam que o índice transmitido é 3 (corresponde a imagem N-4 da **figura 4.14b**) com valores intermediários que correspondem ao cinza claro e escuro. Neste exemplo, a memória tem capacidade para cinco imagens (do branco ao preto na **figura 4.14b**) e a predição pesada não é utilizada.

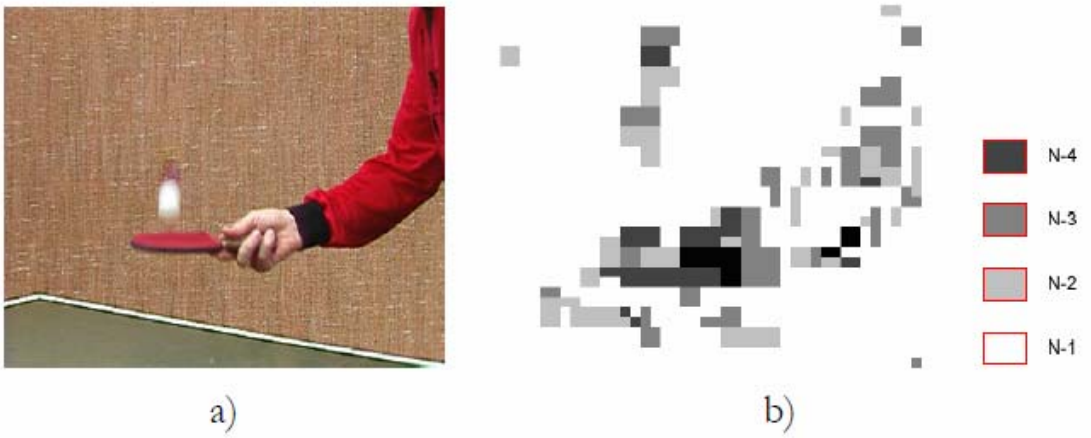


Figura 4.14: Compensação de movimento: a) imagem original; b) escolha da imagem de referência para cada partição.

Quando nenhum erro de predição, vetor de movimento ou índice de referência é codificado, o macrobloco é codificado com o modo especial referido como SKIP. Para macroblocos codificados com este modo, assume-se que o índice de referência é 0, a imagem de referência é a imagem anteriormente decodificada e o vetor de movimento é igual ao vetor de movimento predito (em algumas condições especiais, o vetor de movimento é zero).

As imagens de referência que a memória do decodificador possui são controladas através de um conjunto de instruções definidas no H.264/AVC. Por exemplo, o codificador pode indicar ao decodificador para marcar todas as imagens decodificadas

como não disponíveis o que implica que todas as imagens que irão ser posteriormente codificadas não podem utilizar como referência essas imagens ou imagens anteriores. Este tipo de imagem (I) da origem a um *Instantaneous Decoding Refresh* (IDR) e só pode conter *slices* do tipo I, evitando a propagação de erros. A memória contém duas listas de imagens: as imagens de longa duração e as imagens de curta duração. Uma imagem de longa duração possui sempre o mesmo índice, independentemente do número de imagens decodificadas até ser removida. Por outro lado, os índices das imagens de curta duração correspondem a imagens diferentes a medida que as imagens são decodificadas (através de uma estratégia FIFO). O H.264/AVC define várias operações de controle, marcação de imagens como "não utilizadas", remoção de uma imagem da lista de curta duração para colocar na lista de longa duração, etc. Finalmente, a norma impõe um limite máximo de 15 para o número de quadros de referência disponíveis no decodificador.

4.2.2.4 Slices do tipo B

Em comparação com as normas de codificação anteriores, o conceito de *slices* do tipo B é generalizado no H.264/AVC uma vez que agora uma imagem do tipo B pode utilizar como referência imagens do tipo B para a compensação de movimento, ou seja, a escolha das imagens de predição só depende da gestão de memória efetuada pelo codificador. Deste modo, os *slices* do tipo B são codificados de forma que para uns blocos ou macroblocos a predição corresponde a uma média pesada de dois valores distintos obtidos a partir de dois blocos ou macroblocos depois da compensação de movimento. Esta predição pesada permite obter uma predição Inter mais eficiente, com um menor erro de predição. As imagens do tipo B utilizam duas memórias de referência, referidas como a primeira e a segunda memória de imagens de referência. As imagens de referência que estão em cada uma destas memórias dependem de uma decisão que cabe ao codificador. Na **figura 4.15**, um exemplo de dependências entre imagens do tipo B é apresentado. Para cada imagem pode existir mais do que uma referência, uma vez que o codificador pode escolher quais as referências a utilizar ao nível de macrobloco.

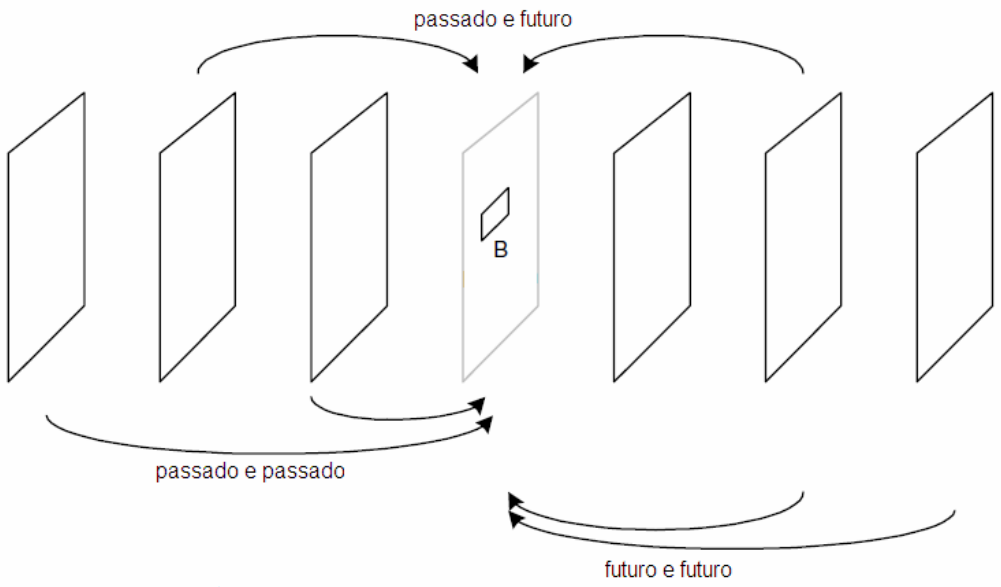


Figura 4.15: Dependências das imagens do tipo B.

Em *slices* do tipo B, quatro tipos de predição Inter são disponibilizados:

- Lista 0 (L0): A predição é calculada a partir de uma imagem de referência presente na primeira memória.
- Lista 1(L1): A predição é calculada a partir de uma imagem de referência presente na segunda memória.
- Bi-preditivas (Bi): A predição é calculada a partir da média pesada da imagem de referência na primeira memória e da imagem de referência na segunda memória.
- Predição direta (*Direct*): A predição é calculada a partir dos elementos de sintaxe previamente transmitidos e pode ser da lista 0, 1 ou bi-preditiva.

Os macroblocos do tipo B utilizam também uma divisão semelhante aos macroblocos do tipo P. Além dos modos Inter 16x16, Inter 16x8, Inter 8x16 e Inter 8x8, foi também definido um modo de predição direta. Adicionalmente, para cada partição de 16x16, 16x8, 8x16, 8x8, o tipo de predição (L0, L1, Bi ou *Direct*) pode ser escolhido separadamente. Se nenhum erro de predição for transmitido para um macrobloco codificado com o método de predição direta, este pode ser codificado de uma forma muito eficiente através do modo *skip*. A codificação dos vetores de movimento nos macroblocos do tipo B é muito semelhante a utilizada nos macroblocos do tipo P com as modificações resultantes de fato dos blocos vizinhos poderem ser codificados utilizando modos de predição diferentes. Finalmente, a predição pesada Inter Bi-preditiva, pode ser realizada com diferentes pesos sendo bastante eficiente na codificação de transições suaves (*cross-fades*) entre diferentes cenas de um vídeo (WIEGAND, 2002-2).

4.2.3 Filtro de bloco

Uma das principais características da arquitetura híbrida de codificação é o efeito de bloco ou seja, o fato da estrutura de blocos ser visível na imagem, normalmente para taxas de codificação mais baixas. As amostras nas fronteiras dos blocos são normalmente reconstruídas com menos precisão que as amostras interiores; este efeito de bloco é geralmente considerado como o artefato mais característico dos esquemas de codificação híbrida tal como usados nas normas de codificação anteriores. A norma H.264/AVC especifica a utilização de um filtro de bloco adaptativo que opera nas fronteiras dos blocos, tanto horizontalmente como verticalmente. Este filtro tem que estar presente no codificador e no decodificador uma vez que filtra os blocos depois destes serem decodificados. Este filtro possui duas vantagens principais:

- As fronteiras dos blocos são suavizadas, sem tornar a imagem difusa, melhorando a qualidade subjetiva da imagem.
- Os macroblocos filtrados são utilizados na compensação de movimento (filtro no *loop*), resultando um resíduo menor depois da predição, ou seja, reduzindo o *bit rate* para a mesma qualidade objetiva.

A filtragem é aplicada nas fronteiras verticais e horizontais dos blocos 4x4 de um macrobloco. O filtro é aplicado primeiro nas fronteiras verticais (ordem a,b,c,d na **figura 4.16a**) e em seguida nas fronteiras horizontais (ordem e,f,g,h na **figura 4.16a**). Cada operação de filtragem calcula apenas três amostras (em cinza na **figura 4.16b**), a partir de quatro amostras em cada lado da fronteira. Dependendo dos valores de vários elementos da sintaxe, várias hipóteses são possíveis, desde a) nenhuma amostra filtrada

a b) p0,p1,p2,q0,q1,q2 filtrados o que permite controlar a quantidade de filtragem aplicada para cada macrobloco.

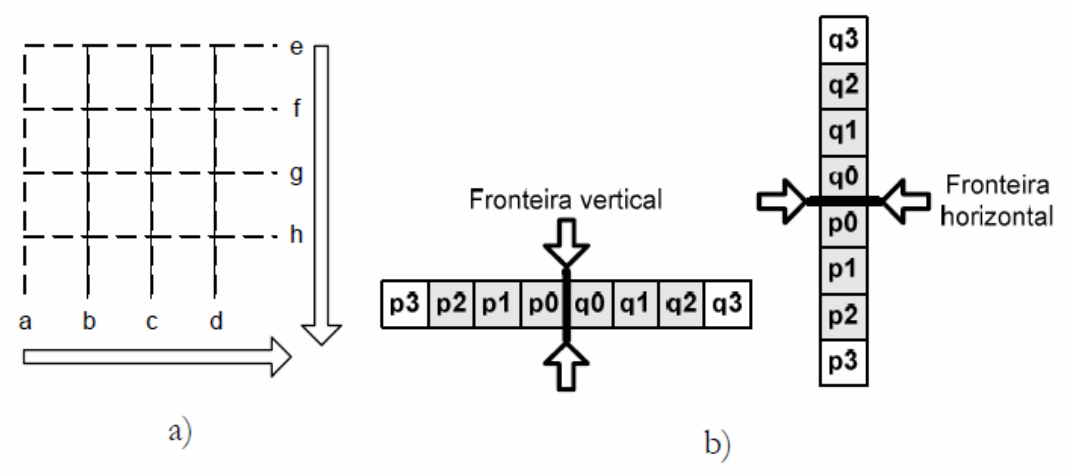


Figura 4.16: Filtro de bloco – a) ordem de filtragem; b) amostras adjacentes nas fronteiras horizontais e verticais.

O filtro adaptativo definido no H.264/AVC pode ser ajustado ao nível do *slice*, das fronteiras e das amostras. O filtro adaptativo é controlado por um parâmetro Bs $\in \{0,1,2,3,4\}$ que representa a força do filtro; para Bs = 0 nenhuma amostra é filtrada e para Bs = 4 o filtro reduz ao máximo o efeito de bloco. O filtro é mais "forte" nos lugares onde ocorre uma distorção mais significativa devido ao efeito de bloco. Ao nível do *slice*, a força do filtro depende do modo de macrobloco escolhido, dos vetores de movimento, da imagem de referência utilizada, e se os blocos na fronteira contêm coeficientes codificados ou não. O conjunto de regras completo para determinar a força do filtro (o valor de Bs) encontra-se especificado em Wiegand (2002). A filtragem é mais acentuada para macroblocos que irão sofrer uma maior distorção, tais como as fronteiras de um macrobloco Intra (Bs = 4), as fronteiras entre blocos de um macrobloco Intra (Bs = 3) e quando o macrobloco é Inter mas seus blocos contém coeficientes diferentes de zero (Bs = 2).

Ao nível de amostra, existem limiares dependentes do fator de quantificação (Qp) utilizado que podem desligar a filtragem para amostras individuais. O objetivo desta decisão é desligar o filtro quando existe uma mudança significativa (gradiente) na fronteira do bloco. Mais especificamente, se |p0-q0|, |p1-p0| e |q1-q0| (que estimam o gradiente no contorno) forem simultaneamente menores que um limiar estabelecido, uma ou mais amostras são filtradas (ver (WIEGAND, 2002-2) para mais detalhes). Para Qp baixos, gradientes com valores médios ou altos correspondem a características das imagem que devem ser preservadas; deste modo os limiares são mais baixos. Por outro lado, quando Qp é alto, a distorção no bloco é mais significativa e os limiares são mais elevados.

Na **figura 4.17** apresenta-se uma imagem decodificada com e sem filtro. Tal como se pode observar na imagem da esquerda o efeito de bloco é muito mais significativo, pois o filtro de bloco encontra-se desligado. Na imagem da direita, o filtro de bloco é utilizado, melhorando a qualidade subjetiva da imagem significativamente. Note-se que os contornos dos objetos são preservados pelo filtro enquanto as fronteiras dos blocos são suavizadas nas regiões mais suaves da imagem.

Figura 4.17: Codificação de um quadro *Foreman* – a) sem filtro de bloco; b) com filtro de bloco (WIEGAND, 2002).

4.2.4 Transformada de quantificação

Uma das inovações a transformada utilizada no H.264/AVC consiste na utilização de blocos de dimensão mais reduzida (4x4 ou 2x2 amostras) que os utilizados nas normas anteriores (8x8 amostras). Esta pequena dimensão é compatível com a mais fina compensação de movimento (4x4 amostras) e permite também uma redução significativa dos artefatos de codificação (normalmente do tipo *ringing*). Além disso, permite uma implementação exata para os decodificadores e codificadores, eliminando o problema de erros entre diversas implementações da DCT, tipicamente denominados erros de *mismatch*. A transformada DCT é uma boa aproximação da transformadas de Karhunen-Loève para um amplo conjunto de sinais (RAO, 1990). A norma H.264/AVC utiliza também transformadas baseadas na DCT com as diferenças abaixo resumidas:

- Usa uma transformada inteira, todas as operações podem ser efetuadas apenas com somas, subtrações e deslocamento de bits (*shifts*), sem perda de precisão.
- A transformada inversa é completamente especificada na norma H.264/AVC e se esta especificação for cumprida não existe nenhum erro entre diferentes implementações da transformada.
- O processo de quantificação e normalização encontra-se integrado com a transformada, sendo apenas necessário uma única operação de multiplicação por coeficiente. Todas as operações necessárias para transformar, normalizar e quantificar um bloco de coeficientes podem ser realizadas em aritmética de 16 bits, reduzindo a complexidade computacional.

4.2.4.1 Transformada inteira

A transformada é aplicada ao erro de predição resultante da predição Intra ou Inter, anteriormente descritas. O H.264/AVC utiliza três transformadas dependendo do tipo de dados a codificar:

1. Uma transformada “nuclear” 4x4 que é aplicada a todos os blocos. Esta transformada é baseada na DCT.

2. Uma transformada 4x4 para os coeficientes DC de luminância obtidos a partir da transformada nuclear 4x4 (utilizada apenas para os macroblocos Intra 16x16).
3. Uma transformada 2x2 para os coeficientes DC de crominância obtidos a partir da transformada nuclear 4x4.

As últimas duas transformadas são baseadas na transformada de Hadamard (CLARK, 1990). Na **figura 4.18** apresenta-se um diagrama de blocos que ilustra a relação entre as três transformadas acima apresentadas. Independentemente do modo de codificação escolhido, é sempre aplicada a transformada (1) "nuclear", Cf, de 4x4, para o modo Intra 16x16, é ainda aplicada a transformada (2) aos coeficientes DC da luminância resultantes da transformada Cf (4x4). Independentemente do modo de codificação escolhido os coeficientes DC da crominância são sempre transformados com uma transformada (3) (2x2). Todos os coeficientes são normalizados e quantificados num único passo, para a evitar a utilização de divisões e operações de vírgula flutuante e a incorporar as matrizes de normalização necessárias para reduzir a gama dinâmica dos coeficientes (apenas são necessários 16 bits para processar e representar os coeficientes de qualquer transformada). Para reconstruir as amostras, as operações inversas ilustradas na **figura 4.18** são efetuadas.

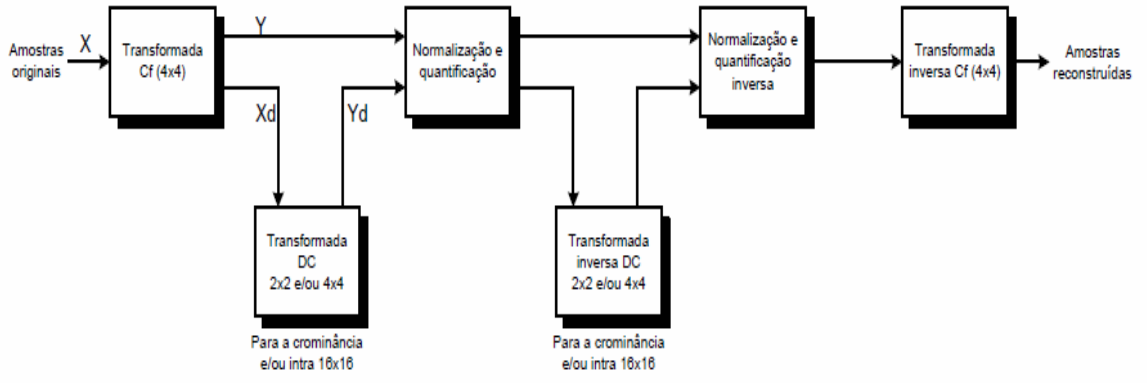


Figura 4.18: Transformada, quantificação, normalização e respectivas operações inversas.

A primeira transformada, *Cf*, é baseada na popular DCT. A transformada DCT transforma um vetor x num novo vetor X de coeficientes da transformada através da transformação linear de X = H x, onde cada elemento da coluna *k* e da linha *n* da matriz H é definido por (CLARK, 1990):

$$H_{kn} = H(k,n) = c_k \sqrt{\frac{2}{N}} \cos\left((n+1/2)\frac{k\pi}{N} \right)$$

com o índice de freqüência k = 0,1, ...,N-1, o índice de amostra n = 0,1,...,N-1 $c_0 = \sqrt{2}$ e $c_k = 1$ para k > 1. A matriz da DCT é ortogonal, x = $H^{-1}X = H^T X$. A principal desvantagem da DCT é que os elementos de H são números irracionais, o que significa que num sistema digital o cálculo da DCT requer aproximar estes números inteiros ou de vírgula flutuante. As diferentes estratégias para se obter esta aproximação levam a diferentes resultados na codificação e sobretudo na decodificação. A solução adotada no H.264/AVC consiste em aproximar a matriz H por uma matriz que contenha apenas

números inteiros o que preserva as propriedades da DCT. Deste modo, definiu-se a matriz H como (MALVAR, 2002):

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

Para se obter uma transformada inversa, é necessário calcular a matriz inversa de H. Se a matriz H fosse $H^T H = I$ apenas seria necessário calcular a transposta de H [ASCENSO, 6 CAP4]; como não é, existe a necessidade de normalizar os coeficientes da transformada H de uma forma adequada e definir a matriz inversa $\tilde{H}^{-1}$ como uma versão normalizada da matriz inversa de H (MALVAR, 2002) (o ~ representa a normalização):

$$\tilde{H}^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix}$$

e a relação entre $\tilde{H}^{-1}$ e H é a seguinte:

$$\tilde{H}^{-1}\, diag\{1/4,\ 1/5,\ ¼,\ 1/5\}\ H = I$$

As multiplicações por ½ de $\tilde{H}^{-1}$ são obrigatoriamente implementadas com deslocamentos de 1 bit a direita, de forma a que todos os decodificadores obtenha os mesmos resultados (se a matriz inversa fosse calculada diretamente a partir de H esta propriedade não seria possível). Se o bloco é codificado no modo Intra 16x16, cada bloco 4x4 é primeiro transformado utilizando a transformada 4x4 anteriormente descrita (Cf) e cada coeficiente DC de cada bloco 4x4 é transformado de novo com a transformada de Hadamard. Esta transformação é referida como uma transformada hierárquica (uma vez que tem dois níveis) e é adequada a zonas da imagem em que as amostras apresentem valores semelhantes dentro de um bloco 16x16 porque ainda existe correlação significativa entre os vários coeficientes DC. A matriz da transformada de Hadamard direta é definida como (MALVAR, 2002):

$$H = \tilde{H}^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

Os coeficientes DC da crominância também são transformados de forma idêntica. Componentes de crominância de um macrobloco são constituídas por blocos de 8x8 amostras que, transformados pela transformada 4x4 já descrita, dão origem a uma matriz de 2x2 coeficientes DC, que irá ser transformada através (MALVAR, 2002):

$$H = \tilde{H}^{-1} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

4.2.4.2 Quantificação

As transformadas de bloco, já descritas, por si só não permitem nenhuma compressão do sinal de vídeo, apenas representam a imagem num domínio mais adequado para a codificação. A quantificação remove componentes da imagem consideradas irrelevantes, permitindo obter uma redução muitas vezes substancial do *bit rate*. A operação de quantificação corresponde a dividir cada coeficiente por um fator de quantificação e a operação de quantificação inversa (reconstrução) corresponde a multiplicar cada coeficiente pelo mesmo fator. Todo este processo introduz perdas irrecuperáveis nos coeficientes resultantes das transformadas. Para quantificar os coeficientes, o H.264/AVC utiliza a quantificação escalar (CLARK, 1990) e o fator de quantificação é igual para todos os coeficientes da transformada. Um de 52 valores possíveis para o fator de quantificação (Qstep) é selecionado para cada macrobloco, indexado através do passo de quantificação (Qp), e de uma tabela que estabelece a relação entre cada Qp e Qstep. Os valores da tabela foram obtidos de forma a haver um aumento de aproximadamente 12,5% do *bit rate* para cada incremento de 1 do valor do passo de quantificação.

Para algumas aplicações, é desejável reduzir o fator de quantificação para valores muito baixos, para que o PSNR (e a qualidade) obtido seja elevado, podendo o mesmo corresponder a níveis visualmente considerados sem perdas. No entanto, para outros tipos de aplicações, são necessários fatores de quantificação elevados de forma que o *bit rate* seja o menor possível. O H.264/AVC estende a gama dos fatores de quantificação de 31 para 52, (nas normas anteriores a gama é 31), permitindo obter PSNRs com 50 dB ou até mais elevados.

Normalmente, o fator de quantificação utilizado para a crominância é igual ao utilizado para a luminância. No entanto, para evitar artefatos visíveis para a crominância a valores de Qp elevados, a norma limita o valor máximo de Qp para a crominância a 80% do valor máximo de Qp para luminância.

Ao nível da imagem, é especificado o valor inicial de Qp (menos 26) para cada macrobloco. O valor inicial pode ser modificado ao nível do *slice* e/ou ao nível do macrobloco, através da transmissão de um valor diferencial não igual a zero. Tanto ao nível do *slice* como ao nível do macrobloco, pode mudar-se para qualquer outro fator de quantificação (ex: de 0 a 51).

4.2.5 Codificação entrópica

O último passo do processo de codificação de vídeo é a codificação entrópica. O principal objetivo deste passo consiste em explorar a redundância estatística dos símbolos ou elementos de sintaxe a codificar. Para este efeito, o H.264/AVC utiliza três métodos, dois dos quais baseados em códigos VLC:

1. *Universal Variable Length Coding* (UVLC)

2. *Context Adaptative Variable Length Coding* (CAVLC)

3. *Context Adaptative Binary Arithmetic Coding* (CABAC), baseado em codificação aritmética.

Tanto a codificação baseada em códigos VLC como a codificação aritmética são já utilizadas por normas anteriores (MPEG-4 e H.263+); no entanto, os métodos aqui utilizados apresentam novidades significativas em relação aos seus predecessores. Os códigos VLC são baseados na atribuição de palavras de código com dimensão reduzida a símbolos com uma probabilidade de ocorrência elevada e palavras de código com dimensões superiores a símbolos com uma probabilidade de ocorrência baixa. Ao contrário da codificação baseada em códigos VLC (Huffman), a codificação aritmética permite que um número não inteiro de bits seja atribuído a cada símbolo, aproximando-se mais do limite teórico de máxima compressão determinado pela entropia (WIEGAND, 2002-2). No H.264/AVC, o método de codificação entrópica utilizado depende do elemento de sintaxe a codificar. Na **tabela 4.1** apresenta-se os principais elementos de sintaxe, a respectiva descrição semântica e ainda os métodos de codificação entrópica utilizados.

Tabela 4.1: Elementos de sintaxe e respectiva codificação entrópica

Elementos de sintaxe	**Descrição**	**Codificação**
Elementos de sintaxe de alto nível	Ao nível da seqüência	FLC/UVLC
	Ao nível do quadro/campo	FLC/UVLC
	Ao nível do *slice*	FLC/UVLC/CABAC
Tipo de macrobloco	Tipo de predição para cada macrobloco	UVLC/CABAC
Coded block pattern	Indica quais são os macroblocos que contém coeficientes	UVLC/CABAC
Passo de Quantificação (Qp)	Codificados diferencialmente	UVLC/CABAC
Índice de quadro de referência	Identifica os quadros de referência	UVLC/CABAC
Vetores de movimento	Codificados diferencialmente	UVLC/CABAC
Informação residual	Coeficientes para cada bloco 4x4 ou 2x2	CAVLC/CABAC

A informação de sintaxe de alto nível é sempre codificada utilizando códigos binários com comprimento fixo (*Fixed Length Coding* – FLC) ou comprimento variável (UVLC). A informação de sintaxe de alto nível ao nível do *slice* pode ser codificada com códigos FLC, VLC ou com o codificador aritmético CABAC. O tipo de macrobloco indica o tipo de predição (Intra, Inter) utilizado em cada macrobloco e o parâmetro CBP indica se os vários blocos pertencentes a um macrobloco contém coeficientes diferentes de zero ou não. O passo de quantificação é codificado diferencialmente em relação ao vetor de movimento predito. O índice que indica qual o quadro de referência utilizado para a predição é enviado ao nível de macrobloco. Todos estes parâmetros são codificados com o método UVLC ou CABAC, dependendo do modo de codificação entrópica escolhido. Por fim, os coeficientes quantificados resultantes da transformada direta podem ser codificados com os métodos de codificação entrópica CAVLC ou CABAC.

4.2.5.1 Codificação entrópica UVLC (códigos de Exp-Golomb)

Os códigos de Exp-Golomb (PERKIS, 2001) (códigos de Golomb exponenciais) são códigos de comprimento variável com uma construção regular. A **tabela 4.2** ilustra a estrutura destes códigos. Cada código é constituído por um sufixo e um prefixo que inclui o bit separador com o valor '1'. Os bits do prefixo possuem sempre o valor '0'e os bits do sufixo X_0 X_1 . . . X_n podem ser '0' ou '1' e são utilizados no cálculo da palavra de código. O número de bits do sufixo é igual ao número de bits do prefixo menos 1 para qualquer palavra de código.

Tabela 4.2: Codificação entrópica Exp-Golomb – a) estrutura do código; b) primeiras 9 palavras de código

a)

Elemento de sintaxe	***Palavra de código***
0	1
1-2	0 1 X0
3-6	0 0 1 X1 X0
7-14	0 0 0 1 X2 X1 X0
15-30	0 0 0 0 1 X3 X2 X1 X0
31-62	0 0 0 0 0 1 X4 X3 X2 X1 X0
...	...

b)

Elemento de sintaxe	***Palavra de código***
0	1
1	0 1 0
2	0 1 1
3	0 0 1 0 0
4	0 0 1 0 1
5	0 0 1 1 0
6	0 0 1 1 1
7	0 0 0 1 0 0 0
8	0 0 0 1 0 0 1
...	...

Deste modo, o processo de decodificação para os elementos da sintaxe usando este tipo de código é bastante simples. Basta ler todos os bits com o valor zero, a partir da posição atual, até encontrar um bit '1'. Em seguida basta ler o mesmo número de bits mais 1, interpretar o número lido como um inteiro como o bit mais significativo a esquerda e subtrair-lhe o valor 1 para obter o elemento de sintaxe que a cadeia de bits representa. Os elementos de sintaxe negativos (indicação do fator diferencial de quantificação ao nível do *slice*) são convertidos da forma apresentada na **tabela 4.3**.

Tabela 4.3: Mapeamento entre elementos de sintaxe.

Elemento de sintaxe positivo	Elemento de sintaxe negativo
0	0
1	1
2	-1
3	2
4	-2
5	3
6	-3
k	$(-1)^{k+1}$ Ceil(k/2) – retornando o menor inteiro maior ou igual a x

Aos elementos de sintaxe que possuem uma elevada probabilidade de ocorrência, foram atribuídas palavras de código com uma dimensão pequena e aos elementos de sintaxe com uma menor probabilidade de ocorrência foram atribuídos palavras de código com uma dimensão superior. Deste modo, em vez de desenhar uma tabela VLC diferente para cada elemento de sintaxe, apenas é necessário mapear cada elemento a palavra de código de acordo com as estatísticas de dados, sendo necessária apenas uma tabela VLC. A utilização de uma única tabela VLC é simples, mas possui uma desvantagem significativa: a tabela é calculada a partir de um modelo de probabilidade estático que ignora a correlação entre os elementos de sintaxe gerados pelo codificador.

4.2.5.2 Codificação adaptativa baseada em códigos VLC (CAVLC)

Para codificar entropicamente os coeficientes da transformada, um esquema mais eficiente denominado CAVLC é utilizado. Neste esquema, várias tabelas VLC são utilizadas e escolhidas adaptativamente, dependendo dos elementos de sintaxe anteriormente codificados. Uma vez que as tabelas VLC foram projetadas para serem adequadas para estatísticas condicionadas (onde a probabilidade de um símbolo depende dos elementos anteriores), o desempenho do módulo de codificação entrópica é melhorado em relação ao esquema ULVC anteriormente descrito. Este método é utilizado para codificar os coeficientes quantificados dos blocos de luminância e crominância 4x4 e 2x2 explora as seguintes características (BJONTEGAARD, 2002):

- Depois da predição, transformada e quantificação, os blocos contém um número significativo de zeros. O CAVLC utiliza codificação (*run, level*) para representar uma cadeia de zeros de uma forma compacta.
- Os coeficientes de alta freqüência diferentes de zero são muitas vezes seqüências de $\pm$ 1. O CAVLC indica o número de coeficientes de alta freqüência com o valor 1 ou -1 de uma forma compacta. Estes coeficientes são referidos como T1s (*Trailing 1s*).
- O número de coeficientes diferentes de zero em blocos vizinhos é muito correlacionado. O número de coeficientes é codificado através de várias tabelas. A escolha da tabela depende do número de coeficientes diferentes de zero nos blocos vizinhos.
- O nível de amplitude dos coeficientes diferentes de zero tende a ser mais alto perto do coeficiente DC e a decrescer a medida que se aproxima de freqüências mais elevadas. O CAVLC tira partido desta propriedade através da escolha das tabelas VLC para a amplitude dos coeficientes dependendo das amplitudes dos símbolos recentemente codificados.

A ordem de varredura dos coeficientes em zig-zag ainda é utilizada, mas a varredura é feita por ordem inversa, partindo dos coeficientes de alta freqüência para o coeficiente DC. Outra característica do CAVLC é a separação entre o comprimento (*run*) e o nível ou amplitude (*level*) que permite uma melhor adaptação e menor complexidade. O algoritmo de codificação CAVLC consiste em seis passos distintos (**figura 4.19** apresenta um exemplo):

1. Varrer os coeficientes do bloco em zig-zag, das altas e baixas freqüências, de forma a obter um vetor de coeficientes; a **figura 4.19** exemplifica este processo.

2. Codificar o número total de coeficientes (*Tcoeff*) e o número de *T1s*. Para obter o número de T1s é necessário varrer o vetor de coeficientes do menor para o maior índice, das freqüências mais altas para as mais baixas e contar o número de coeficientes com o valor de $\pm$ 1, até que seja encontrado um coeficientes diferente de zero e $\pm$ 1. O número total de coeficientes *Tcoeff* está entre 0 e 16 para um bloco de 4x4, e o número de T1s permitidos entre 0 e 3. Se existirem mais do que 3 T1s, apenas os últimos três são codificados neste passo. Os valores *Tcoeff* e *T1s* são codificados com um único símbolo e a tabela VLC escolhida depende do número de coeficientes codificados nos blocos vizinhos (adaptação ao contexto). Existem quatro tabelas, a primeira das quais é adequada a um número pequeno de coeficientes; valores baixos de *Tcoeff* possuem códigos pequenos e valores altos de *Tcoeff* códigos como um comprimento superior. A segunda tabela é adequada a um número médio de coeficientes *Tcoeff*; aos valores de *Tcoeff* entre 2 e 4 são atribuídos códigos pequenos. A terceira tabela é adequada a valores altos de *Tcoeff* e a quarta consiste em códigos de comprimento fixo para cada valor de *Tcoeff*. No exemplo a **figura 4.19**, a palavra de código corresponde a *Tcoeff* = 5 e *T1* = 3 é "0000100".
3. Codificar o sinal de cara T1 com um único bit (0 para + e 1 para -). Estes bits são enviados na ordem que estão presentes no vetor de coeficientes os sinais do T1s, do menor índice para o maior. Para o exemplo da **figura 4.19**, são necessários três bits ("011") que correspondem ao sinal dos coeficientes (1, -1 e -1).
4. Codificar os níveis dos coeficientes diferentes de zero restantes. A escolha da tabela para cada nível codificado depende dos símbolos anteriores (adaptação ao contexto). Existem 7 tabelas que podem ser escolhidas, sendo a primeira adequada a níveis baixos e a última adequada a níveis altos. O processo de escolha da tabela está definido através de limiares. Na **figura 4.19**, os níveis dos coeficientes 1 e 3 são codificados com as palavras de código '1' e '0010'.
5. Codificar o número tal de zeros antes do último coeficiente (*Tzeros*). O número total de zeros antes do último coeficiente é codificado através de uma tabela VLC. Como o número total de coeficientes já é conhecido, pode também saber-se o número total de zeros que o bloco possui o que evita codificar o comprimento (*run*) para o número de zeros presentes no fim do vetor. Uma de 15 tabelas VLC é escolhida, dependendo do número total de coeficientes (*Tcoeff*). Para o exemplo da **figura 4.19**, *Tzeros* = 3 o que corresponde a palavra de código '111'.
6. Codificar cada comprimento (*run*) de zeros. Depois de terem sido codificados os níveis dos coeficientes nos passos 2, 3 e 4, é necessário enviar o comprimento (*run*) para cada coeficiente diferente de zero, o número de zeros antes de qualquer coeficiente diferente de zero. Tal como nos passos anteriores, é necessário varrer o vetor dos coeficientes, do menor índice para o maior (das freqüências mais altas para as mais baixas) e enviar o número de zeros para cada coeficiente diferente de zero, com duas exceções:
    - Se não estiverem mais zeros para serem codificados, não é necessário codificar mais nenhum valor de comprimento;

- Não é necessário codificar o comprimento para o último coeficiente (freqüência mais baixa).

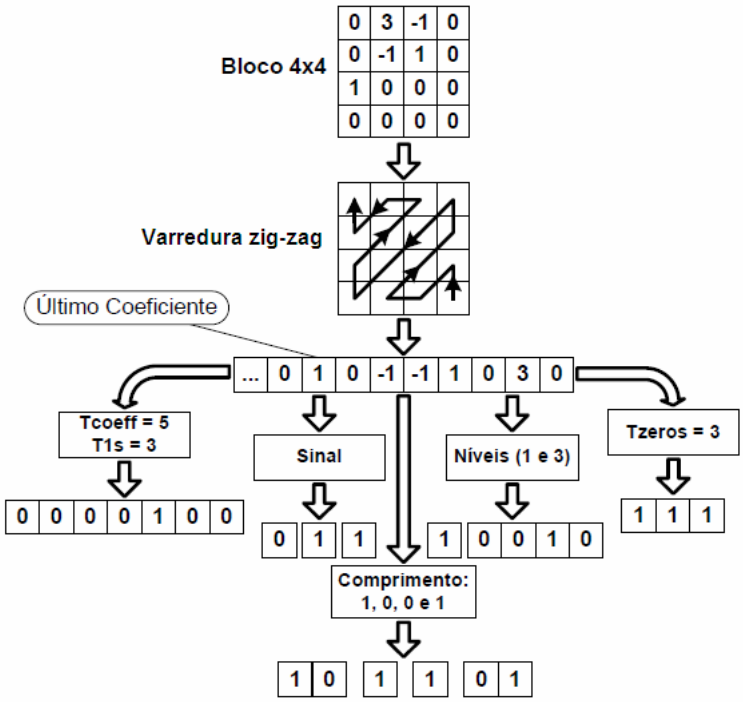


Figura 4.19: Exemplo de codificação entrópica CAVLC.

A escolha da tabela VLC depende do número de zeros que ainda falta codificar e do comprimento a codificar, uma vez que apenas um conjunto de comprimentos é permitido, dependendo do valor de *Tzeros* e dos comprimentos já codificados (se falharem codificar 2 zeros, o comprimento só pode tomar 3 valores: 0, 1 ou 2, e a palavra de código utilizada possui no máximo 2 bits). Para o exemplo da **figura 4.19**, são enviados 4 comprimentos, o primeiro para os coeficientes 1 e -1, o segundo para os coeficientes -1 e -1, o terceiro para os coeficientes -1 e 1 e o quarto para os coeficientes 1 e 3. A palavra de código que é necessária enviar para todos os coeficientes deste bloco é constituída pela concatenação de todas a palavras de código resultantes de cada passo realizado (‘000010001110010111101101’).

Apesar da complexidade acrescida do decodificador, esta técnica apresenta um desempenho superior em relação a codificação entrópica Exp-Golomb, especialmente para *bit rates* altos (com ganhos até de 18% para Qp = 4 (WIEGAND, 2002).

4.2.5.3 Codificação aritmética do CABAC

O codificador entrópico do H.264/AVC pode ser ainda mais eficiente com a utilização do codificador aritmético CABAC (MOCCAGATTA, 2002). Por um lado, a utilização de codificação aritmética permite atribuir um número não inteiro de bits para cada símbolo de um alfabeto, o que é adequado para probabilidades de símbolo maiores que 0,5. Por outro lado, a utilização de códigos adaptativos permite uma boa adaptação as estatísticas não estacionárias dos elementos de sintaxe a codificar. Por exemplo, a estatística das amplitudes dos vetores de movimento varia no tempo e no especo para diferentes seqüências de *bit rates*. Logo, um modelo adaptativo permite ter em conta as probabilidades dos vetores de movimento já codificados e consequentemente uma melhor adaptação dos códigos aritméticos a estatística do sinal.

A **figura 4.20** apresenta a arquitetura genérica do codificador entrópico CABAC constituído pelos seguintes módulos:

1. Binarização – O CABAC utiliza codificação binária aritmética o que significa que apenas elementos binários são codificados. Deste modo, é necessário converter um símbolo não binário (ex: um vetor de movimento) em uma seqüência binária antes deste ser codificado. Esta representação binária consiste num conjunto de decisões binárias, denominadas *bins* que correspondem a um bit da seqüência. Este processo é especificado para cada símbolo a codificar com um conjunto de tabelas e os códigos utilizados podem ser de comprimento variável (códigos VLC) ou fixo (códigos FLC).

2. Seleção de contextos – Um contexto é um modelo da probabilidade de ocorrência de um ou mais *bins* da seqüência binária. Este passo consiste na seleção de um contexto para cada *bin* de acordo com um conjunto de observações passadas (depende da estatística dos símbolos recentemente codificados). O contexto guarda a probabilidade de cada *bin* para símbolos relacionados, para todos os coeficientes AC da crominância.

3. Codificador aritmético adaptativo e binário – Finalmente, cada *bin* é codificado através de um codificador aritmético adaptativo que utiliza as estimativas de probabilidade de um determinado contexto, escolhido no passo anterior. Depois de codificar cada *bin*, as probabilidades de um contexto vão ser atualizadas, usando todos os símbolos binários já codificados. Deste modo, o contexto selecionado adapta-se a estatísticas que variam ao longo do tempo.

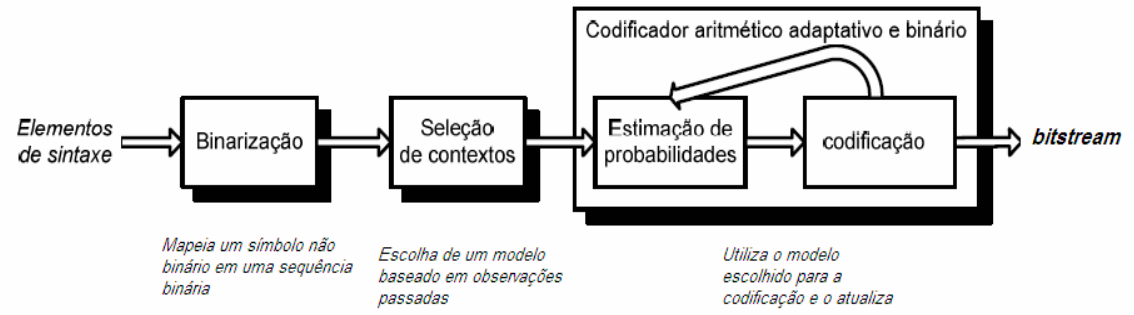


Figura 4.20: Arquitetura do codificador entrópico CABAC.

Esta arquitetura é válida para a codificação entrópica de vários símbolos (correspondentes a vários elementos de sintaxe), como o tipo de macrobloco, o CBP (*Coded Block Pattern*), os quadros de referência, os modos de predição Intra e Inter e os coeficientes quantificados.

O símbolo CBP indica quais são os blocos dentro do macrobloco que possuem coeficientes diferentes de zero. Se o símbolo CBP indicar que só existem coeficientes com o valor zero, mais nenhuma informação é transmitida para esse bloco. Se o símbolo CBP indicar que existem coeficientes diferentes de zero, um mapa de coeficientes (*significant map*) que especifica as posições dos coeficientes significativos (diferentes de zero) é codificado seguido da informação de amplitude e de sinal para cada coeficiente, em vez do popular esquema (*run, level*) utilizado pelo método ULVC. Para se construir o mapa de coeficientes, é necessário varrer todos os coeficientes em zig-zag (das baixas para as altas freqüências) e transmitir um símbolo de um bit, referido como SIG (que indica se um dado coeficiente é significativo ou não). Se um coeficiente for

diferente de zero, o valor de SIG é ‘1’; se o coeficiente for igual a zero, o valor de SIG é ‘0’. Para cada coeficiente significativo, é também necessário enviar um símbolo LAST que indica se este coeficiente é o último elemento significativo, só existem zeros depois deste coeficiente. A **figura 4.21** exemplifica este processo. O símbolo LAST é igual a ‘1’ para o último coeficiente significativo transmitido e a ‘0’ no caso contrário. O par (SIG, LAST) da última posição varrida em zig-zag de um bloco (posição 16) nunca é transmitido. Se a última posição tiver sido alcançada e o símbolo LAST não tiver sido transmitido, está claro que o último coeficiente tem de ser significativo.

Coeficientes	14	0	-5	3	0	0	-1	0	1	0	0	0	0	0	0	0
SIG	1	0	1	1	0	0	1	0	1							
LAST	0		0	0			0		1							

Coeficientes	18	-2	0	0	0	-5	1	-1	0	0	0	0	1	0	0	1
SIG	1	1	0	0	0	1	1	1	0	0	0	0	1	0	0	(1)
LAST	0	0				0	0	0					0			(1)

Figura 4.21: Dois exemplos de codificação do mapa de coeficientes (os símbolos em amarelo não são transmitidos)

O mapa de coeficientes indica a posição de todos os coeficientes quantificados do bloco. As amplitudes ou níveis de cada coeficiente são transmitidas em seguida através do símbolo ABS que representa o valor absoluto de cada coeficiente e do símbolo SIGN que representa o sinal de cada coeficiente. Os níveis são codificados utilizando uma ordem inversa a utilizada anteriormente, das altas freqüências para as baixas (tal como no método anterior).

No H.264/AVC existem 12 tipos de macroblocos diferentes para a luminância, para cada crominância, para cada tipo (Intra, Inter e Intra 16x16) e para cada componente (DC e AC). No entanto, para a maior parte das seqüências e condições de codificação, as estatísticas são muito semelhantes.

Para reduzir a dimensão do espaço de modelagem dos contextos, os tipos de blocos são classificados em 5 categorias: três para luminância (Luma-Intra16x16-DC, Luma-Intra16x16-AC, Luma-4x4) e duas para a crominância (Chroma-DC e Chroma-AC). Para cada uma destas categorias, um conjunto independente de contextos é utilizado. Por exemplo, para o símbolo CBP quatro contextos são utilizados para cada uma das cinco categorias e a escolha específica do modelo é baseada na forma como foram codificados os blocos vizinhos em cima e a esquerda. Para a codificação do mapa de coeficientes, até 15 contextos podem ser utilizados para os símbolos SIG e LAST. A escolha do contexto é determinada a partir posição do coeficiente e da escolha feita para o símbolo anterior. Para codificar o valor absoluto ABS, dois modelos são utilizados, um para o primeiro *bin* (ou bit) e outro para os restantes *bin*. A escolha do primeiro modelo depende do número de coeficientes sucessivos com o valor ‘1’ até o máximo de três, tirando partido da ocorrência de seqüências de $\pm$ 1 para os coeficientes de alta freqüência. Todos os restantes *bins* do valor absoluto ABS são codificado utilizando o mesmo contexto. Este é determinado pelo número de coeficientes com valor absoluto superior a 1 já transmitidos. Para o sinal dos coeficientes, apenas um contexto é utilizado para cada categoria de tipo de blocos. São utilizados 52 contextos para

codificar a informação de amplitude; no início de cada *slice*, os contextos são inicializados dependendo do valor inicial do passo de quantificação, uma vez que este possui um efeito significativo na probabilidade de ocorrência de cada elemento de sintaxe.

O motor de codificação e a estimação de probabilidades encontram-se descritos com mais detalhe na norma (WIEGAND, 2002-2) e possui três propriedades:

- A estimação de probabilidades é realizada através de uma máquina de estados com 64 estados respectivos das probabilidades e a transição entre estados é especificada através de uma tabela.
- O intervalo R que representa o estado atual do codificador aritmético é quantificado com uma pequena gama de valores pré-definidos e o cálculo da nova gama pode ser realizado através de tabelas (sem a utilização de multiplicações).
- Um processo simplificado para a codificação e decodificação de símbolos com uma distribuição de probabilidade uniforme.

A estimação de probabilidades e o motor de codificação são especificadas através de tabelas e deslocamentos (*shifts*), libertando o processador do cálculo de multiplicações.

4.2.6 *Slices* SP e SI

Os *slices Switching* Intra (SI) e *Switching Predicted* (SP) são *slices* codificados de uma forma diferente dos seus homólogos I e P para permitir o acesso aleatório mais eficiente, transferência entre *bitstreams* codificados, resiliência a erros, resincronização e avanço / recuo rápido. Os *slices* SI e SP podem pertencer a uma imagem que apenas contenha *slices* SI e SP, respectivamente (imagens SI e SP) ou imagens com *slices* I, P ou B. O método de codificação de imagens ou *slices* SP permite obter imagens reconstruídas idênticas as suas homólogas P, mesmo quando diferentes imagens de referência são utilizadas para a sua predição. Tal como os *slices* I, os *slices* SI não dependem das imagens anteriormente codificadas (apenas fazem uso da predição espacial) e os *slices* SP utilizam a compensação de movimento para explorar a redundância temporal de uma forma semelhante aos *slices* P.

4.2.6.1 Transferência entre bitstreams

Em algumas aplicações, um dos requisitos é que o decodificador possa escolher entre um de vários *bitstreams* disponíveis, codificados com diferentes *bit rates*, resoluções espaciais e/ou temporais, e que a transferência da decodificação de um *bitstream* para outro seja feita de forma elegante, sem atrasos significativos. Este requisito é muito comum em aplicações de distribuição de vídeo na Internet, onde uma seqüência é codificada com múltiplos *bit rates* e o decodificador pode escolher dinamicamente um deles, de acordo com as características de rede do usuário que requisitou.

Com as normas de codificação de vídeo atuais, o decodificador só pode interromper a decodificação de um *bitstream* e começar a decodificar outro em determinados locais, em imagens que não utilizem nenhuma predição temporal, ou seja, em imagens do tipo I. No entanto, esse tipo de imagens necessita de um maior número de bits para serem codificadas com a mesma qualidade em relação as imagens P. As imagens SP permitem a transferência entre *bitstreams* em qualquer altura, pois o método de codificação das

imagens do tipo SP permite obter imagens idênticas, mesmo quando as imagens de referência utilizadas como predição são diferentes. A **figura 4.22a** apresenta um exemplo em uso de imagens SI e SP para este tipo de aplicação. Neste caso existem dois *bitstreams* que correspondem a mesma seqüência, mas que usam *bit rates* e/ou resoluções espaciais / temporais diferentes. Ambos os *bitstrems* incluem imagens SP (com o nome SP1 e SP2) nos locais onde se pretende mudar de um *bitstream* para outro. A imagem S1,2 da **figura 4.22a** apenas é enviada no momento do "salto" para um outro *bitstream* no sentido indicado e é referida como a representação secundária das imagens SP. Esta imagem serve como "ponte" entre dois *bitstreams* codificados com parâmetros diferentes, sem uma perda muito significativa de eficiência (KURCEREN, 2002) em relação as imagens do tipo P.

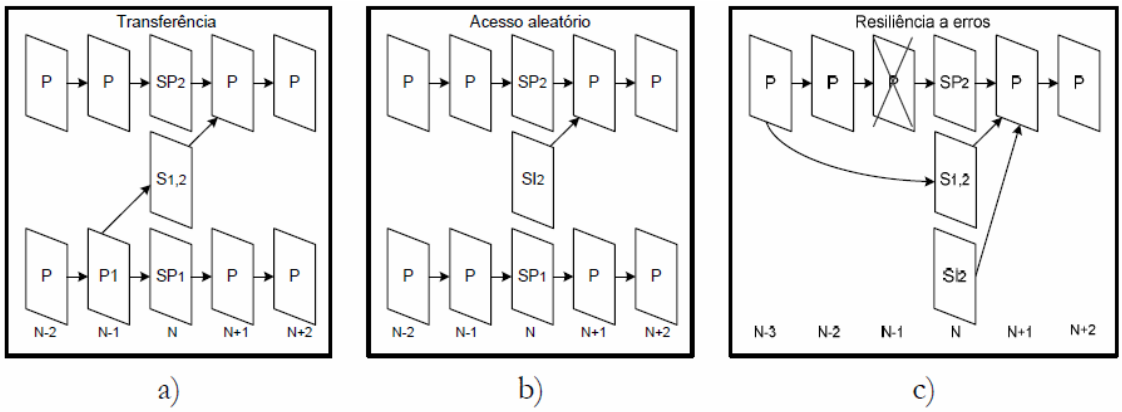


Figura 4.22: Cenários de utilização para as imagens do tipo SI e SP.

No exemplo da **figura 4.22a**, apenas é possível efetuar o "salto" num único sentido: do *bitstream* de baixo para o de cima; no entanto, também é possível efetuar um "salto" no sentido contrário, através da transmissão de outra representação secundária da imagem SP codificada de uma forma diferente, a imagem S2,1.

A imagem SP1 (do tipo SP) é codificada subtraindo a imagem anterior decodificada (P1) depois da compensação de movimento da imagem SP1, de uma forma semelhante às imagens do tipo P. No entanto para imagens do tipo SP, a subtração é efetuada no domínio da freqüência (depois da transformada). A imagem SP2 é codificada da mesma forma. O processo de codificação da imagem S1,2 é apresentado na **figura 4.23**. Nesta arquitetura, a imagem SP2 (que representa o fluxo de destino) é transformada e quantificada, e a partir da imagem P1 (que representa o fluxo de origem) obtém-se uma predição através do módulo de compensação de movimento. A respectiva estimação de movimento é realizada para cada bloco da imagem SP2, tendo como referência a imagem P1. Esta predição é transformada, quantificada e subtraída da imagem SP2 transformada e quantificada, obtendo-se deste modo um resíduo que explora as semelhanças entre imagens que pertencem a seqüências codificadas de forma diferente. Este método é apresentado com mais detalhes em Kurceren (2002).

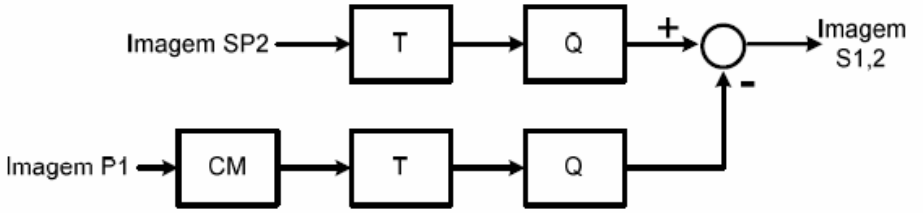


Figura 4.23: Codificação da imagem SP (simplificado).

4.2.6.2 Acesso Aleatório

Como mostrado anteriormente, se assume que os *bitstreams* pertencem a mesma seqüência. No entanto, a transferência entre *bitstreams* pode ser utilizada genericamente para escolher entre *bitstreams* que representam a mesma cena a partir de câmeras com diferentes perspectivas, inserção de anúncios comerciais, alternar entrada de programas diferentes, etc. Como neste caso as seqüências são diferentes entre si, não é eficiente explorar a redundância temporal entre as imagens que pertencem a seqüências diferentes. Neste caso, a utilização de predição espacial é mais eficiente, o que levou a introdução das imagens do tipo SI. Um exemplo é apresentado na **figura 4.22b**, onde a imagem SI2 utilizada não depende de nenhuma imagem anterior (ao contrário do cenário anterior). Esta imagem permite a reconstrução sem erros da imagem SP (S2), garantindo que as imagens seguintes são decodificadas corretamente. A introdução de uma imagem SI permite a criação de pontos de acesso numa seqüência de vídeo. A codificação das imagens SI é feita de uma forma semelhante das imagens SP, com a predição a ser obtida através do módulo de predição Intra 4x4 descrito anteriormente (em vez de ser obtida através do módulo de compensação de movimento).

4.2.6.3 Resiliência a erros

Múltiplas representações de uma única imagem SP permitem aumentar a resiliência a erros de um *bitstream*. Considerando o caso em que um *bitstream* é distribuído por um servidor de vídeo a um determinado cliente e ocorre uma perda de uma imagem (**figura 4.22c**). O cliente pode indicar ao servidor de vídeo a imagem perdida e o servidor pode enviar em resposta uma representação secundária do quadro SP (S12). Esta representação pode utilizar um ou mais quadros de referência já recebidos pelo cliente. No entanto, outra alternativa é enviar um quadro SI (SI2) que, como não depende de nenhum outro quadro, permite a reconstrução da imagem SP sem erros e consequentemente de todas as imagens posteriores. Neste caso, este processo pode ser realizado ao nível do *slice* (raramente ocorrem erros que levam a perda total de uma imagem), evitando o envio completo de uma imagem.

4.3 Perfis e níveis

Para gerir o elevado número de ferramentas de codificação incluídas na norma H.264/AVC e a máxima complexidade que o decodificador para um dado domínio de aplicação pode suportar, o conceito de Perfis e Níveis é utilizado de uma forma semelhante ao que já foi definido para a norma MPEG-4.

Deste modo, no H.264/AVC um perfil define um conjunto de ferramentas de codificação ou algoritmos que podem ser utilizados para gerar um *bitstream* normativo. Cada perfil é definido de forma a facilitar a interoperabilidade entre aplicações que possuem requisitos funcionais semelhantes. Um nível coloca restrições em alguns parâmetros do *bitstream*, limitando a complexidade que os decodificadores necessitam possuir estar conforme um determinado perfil e nível (memória e capacidade de cálculo). Todos os decodificadores que são conforme um determinado "ponto de conformidade" (combinação perfil@nível) tem que possuir as ferramentas definidas no perfil para serem capazes de operar dentro dos parâmetros definidos para o nível correspondente. Os codificadores não são obrigados a fazer uso de todas as ferramentas

de codificação definidas no perfil, nem de utilizar os limites máximos de um determinado nível, mas os decodificadores tem que estarem preparados para os casos extremos no contexto de um dado ponto de conformidade. A **figura 4.24** ilustra a solução adotada pela norma H.264/AVC em termos de perfis (WIEGAND, 2002-2):

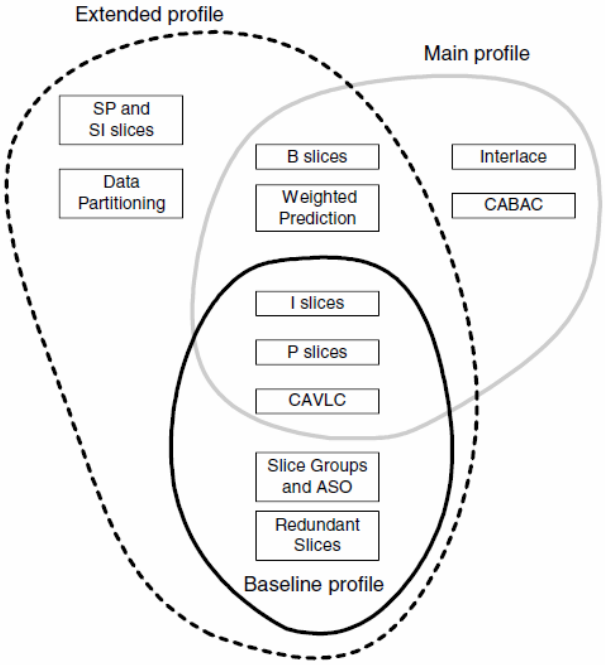


Figura 4.24: Estrutura dos perfis no H.264/AVC.

Os perfis definidos na norma H.264/AVC são os seguintes:

- Baseline: Este perfil foi definido para aplicações que possuem requisitos de atraso e complexidade reduzidos, videotelefonia e videoconferência. Neste perfil estão incluídas ferramentas de resiliência a erros, para que se possa otimizar o transporte de vídeo em canais com uma taxa de erros elevada.
- Main: Este perfil permite um desempenho em termos de eficiência mais elevado que o perfil Baseline devido a introdução de ferramentas com uma complexidade superior e atraso mais elevado (imagens do tipo B). É especialmente indicado para serviços que pretendam obter a melhor qualidade possível para um dado *bit rate*, sem requisitos críticos em termos de complexidades e atraso (serviços de difusão e armazenamento de vídeo). O canal de distribuição deve possuir uma taxa de erros baixa, devido ao tipo de ferramentas utilizadas (codificação aritmética) e a exclusão de todos os mecanismos de resiliência a erros.
- Extended: Este perfil possui um conjunto de ferramentas indicadas para a distribuição de vídeo em ambientes sujeitos a erros. Possui uma complexidade média e um atraso elevado, pois inclui todas as ferramentas do perfil Baseline e as imagens B que lhe permitem obter um maior desempenho. Fazem parte deste perfil todas as ferramentas de resiliência a erros presentes na norma H.264/AVC e as imagens / *slices* SI e SP que permitem funcionalidades úteis em ambientes deste tipo (ex. acesso aleatório, alternar entre *bitstreams*).

O conjunto de ferramentas suportados em cada um destes perfis esta mostrado na **tabela 4.4** (falta o nome da tabela)

Tabela 4.4: Definição de perfis da norma H.264/AVC.

Ferramentas de codificação		Perfis		
		Baseline	Main	Extended
Formatos de imagem	Imagens progressivas	X	X	X
	Imagens entrelaçadas	Nível2.1+	Nível2.1+	Nível2.1+
Tipos de imagem / *slice*	I e P	X	X	X
	B		X	X
	SI e SP			X
Compensação de movimento (CM)	CM estruturada em árvore	X	X	X
	CM com múltiplas imagens	X	X	X
	Precisão de ¼ pel na CM	X	X	X
	Predição pesada		X	X
Codificação entrópica	Baseada em códigos VLC	X	X	X
	Aritmética – CABAC		X	
Filtragem	Filtro de bloco	X	X	X
Ferramentas de resiliência a erros	Ordem arbitrária de *slices*	X		X
	Ordem flexível de macrobloco	X		X
	Slices redundantes	X		X
	Separação de dados			X

Na norma H.264/AVC, o mesmo conjunto de definições para os níveis é utilizado para todos os perfis. Foram definidos 11 níveis, através da especificação de limites superiores para a dimensão da imagem (em macroblocos), a taxa de processamento que o decodificador deve suportar (em macroblocos por segundo), a dimensão máxima para a memória que guarda as imagens de referência, o *bit rate* e a dimensão da memória de imagem codificada.

4.4 Comparação com as normas anteriores

Para conhecer o desempenho da norma H.264/AVC, apresenta-se um dos vários estudos de desempenho realizados e disponíveis na literatura [ASCENSO, 30 CAP4]. Neste estudo compara-se a norma H.264/AVC com normas anteriores de sucesso, como MPEG-2 Vídeo, H.263++ e o MPEG-4 Visual ASP para um conjunto conhecido de seqüências no formato QCIF (10 e 15 Hz) e CIF (15 e 30 Hz). As seqüências utilizadas em formato QCIF são: *Foreman, News, Container Ship* e *Tempete*; em formato CIF usaram-se as seqüências: *Bus, Flower Garden, Mobile and Calendar* e *Tempete*. Uma descrição das seqüências de teste *Foreman* e *Tempete*, encontra-se no Anexo B.

Para garantir uma comparação justa entre várias normas, utiliza-se sempre o mesmo controle do codificador em todos os codificadores utilizados neste teste. Para isso foi escolhida uma otimização RD com métodos Lagrangeanos (SULLIVAN, 1998), pois permite alcançar um desempenho superior através de decisões ótimas do ponto de vista *bit rate* / qualidade (a otimização RD encontra-se já integrada no software de referência da norma H.264/AVC).

O codificador MPEG-2 Vídeo utilizado produz *bitstreams* que obedecem as regras estabelecidas de compensação de movimento com precisão de ¼ de *pixel* e compensação de movimento global do perfil ASP. Para o MPEG-4 ASP, também é utilizado o filtro recomendado de redução de efeito de bloco e de artefatos do tipo *ringing* [ASCENSO, 15 – CAP4], como uma operação de pós processamento. O codificador H.263++ [ASCENSO, 32 – CAP4] utiliza todas as ferramentas definidas para o perfil HLP (*High Latency Profile*); este perfil permite a maior eficiência de codificação da norma H.263++ sendo adequado a aplicações que suportam um elevado atraso.

Para o H.264/AVC, utilizou-se o codificador JM 5.0 (SUEHRING, 2006), com todas as ferramentas do perfil Main ativadas. Para os codificadores H.263 e H.264 são utilizadas cinco imagens de referência, para quase todas as seqüências. A única exceção foi a seqüência de *News* para a qual se utilizam mais imagens de referência para explorar melhor a redundância temporal desta seqüência. Para todos os codificadores, apenas a primeira imagem de cada seqüência é codificada no modo Intra e duas imagens do tipo B são inseridas entre cada imagem do tipo P (ex: a estrutura do GOP é IPBBPBBP...). Para o H.264/AVC, as imagens do tipo B não são utilizadas como referência para outras imagens (apesar de isso ser permitido pela norma). Finalmente, a estimação de movimento em todos os codificadores é efetuada com um método exaustivo de procura para uma gama de 32 *pixels* e o passo de quantificação mantém-se constante para toda a seqüência de vídeo. Os *bit rates* desejados foram obtidos através do ajuste individual do passo de quantificação de cada seqüência, e não foi utilizado nenhum tipo de controle de *bit rate* para fazer variar o passo de quantificação ao longo do tempo de acordo com os bits gastos. A **figura 4.25** apresenta as curvas RD para os 4 codificadores para a seqüência de teste *Tempete*, com resolução espacial CIF e resolução temporal 30 Hz.

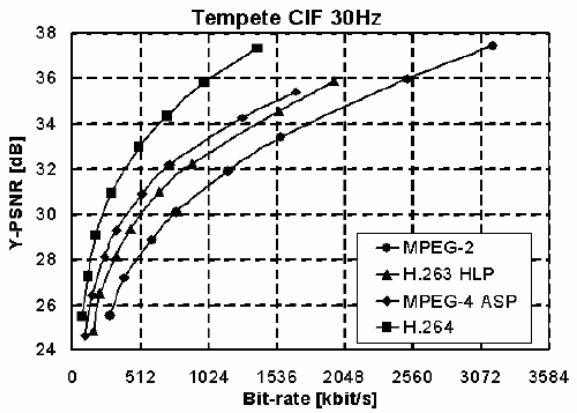


Figura 4.25: Curvas RD da seqüência *Tempete*, codificada segundo: a) H.264/AVC, b) H.263 HLP, c) MPEG-4 ASP e d) MPEG-2 Vídeo (WIEGAND, 2002).

Tal como é ilustrado, a seqüência *Tempete* apresenta uma qualidade superior quando é codificada com a norma H.264/AVC para toda a gama de *bit rates* em teste, com ganhos significativos para *bit rates* médios (entre 256 kbit/s e 512 kbit/s) e altos (> 512 kbit/s). O ganho de qualidade depende do *bit rate* utilizado para codificar a seqüência e aumenta a medida que o *bit rate* também aumenta (até o máximo de 3dB para 1.3 Mbit/s). Na **figura 4.26** apresenta-se o ganho em termos de *bit rate* para um dado nível de qualidade, da norma H.264/AVC, MPEG-4 ASP e H.263 HLP em relação a norma MPEG-2 Vídeo, para um conjunto amplo de qualidades (aproximadamente entre 25 dB e 37 dB). Tal como era esperado, a norma H.264/AVC apresenta um ganho sempre superior a 50% para as seqüências *Foreman e Tempete*, com um ganho mínimo de 55% e máximo de 75%. Os ganhos em eficiência do H.264/AVC são sempre superiores em relação o restante das normas em teste (entre 15 e 35% em relação ao segundo classificado MPEG-4 ASP).

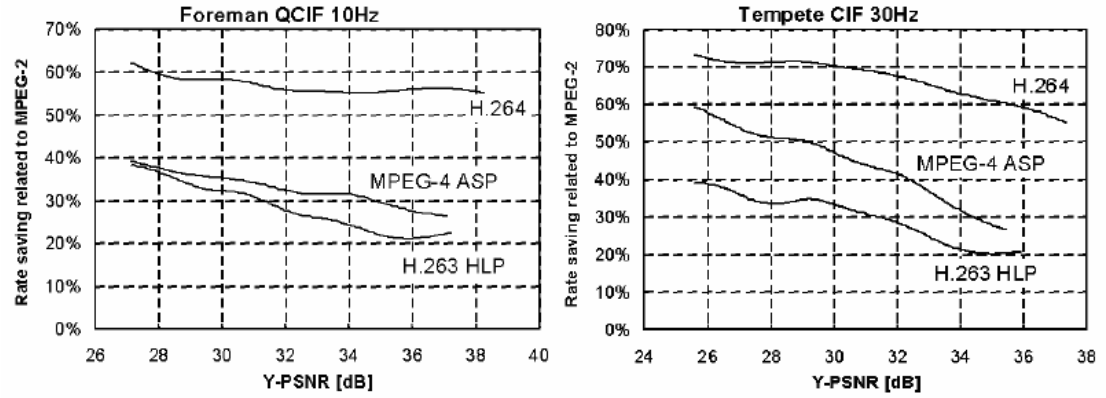


Figura 4.26: Ganho em *bit rate* em relação ao MPEG-2 para um dado nível de qualidade (seqüências *Foreman* e *Tempete*) (WIEGAND, 2002).

A diminuição média do *bit rate* entre todos os codificadores em testes, para todas as seqüências de teste para o intervalo de níveis de qualidade atrás indicado é apresentado na **tabela 5**. Tal como se pode ver, o H.264/AVC apresenta um ganho positivo em relação a todas as restantes normas. Os principais responsáveis por este desempenho superior são as ferramentas de estimação de movimento e a codificação aritmética CABAC.

Tabela 4.5: Diminuição média do *bit rate* entre os codificadores na vertical em relação aos da horizontal para todas as seqüências de teste (WIEGAND, 2002).

Codificado	MPEG-4 ASP)	H.263 (HLP)	MPEG-2 (MP@ML)
H.264/AVC (Main)	38,62%	48,80%	64,46%
MPEG-4 (ASP)	-	16,65%	42,95%
H.263++ (HLP)	-	-	30,61%

O H.264/AVC representa um passo importante na evolução das normas de codificação de vídeo, pois apresenta um desempenho mais do que duas vezes superior, redução do *bit rate* maior que 50% para o mesmo nível de qualidade, em comparação com a popular MPEG-2 Vídeo para um conjunto significativo de *bit rates* e qualidades. Com este incremento de eficiência, novas aplicações e serviços podem ser desenvolvidos como, por exemplo, novas utilizações para os sistemas de televisão digital DVB, DVD e vídeo sobre xDSL. No entanto, estudos de complexidade em relação a norma MPEG-2 Vídeo indicam uma complexidade 2 a 3 vezes superior para o decodificador e 4 a 5 vezes superior para o codificador (SCHAFER, 2003). O desenvolvimento tecnológico de novas memórias e processadores irá ditar o ritmo de adoção desta norma, mas esta é relativamente menos complexa que a norma MPEG-2 Vídeo. Outro fato bastante importante é que a norma H.264/AVC é pública e aberta o que permite que cada fabricante construa codificadores e decodificadores num mercado competitivo, para diferentes aplicações, domínios ou públicos alvos.

5 ESCALABILIDADE DE VÍDEO H.264-FGS

A norma H.264/AVC apresentada em detalhe anteriormente apresenta um desempenho em termos de compressão superior em relação a qualquer outra norma de codificação de vídeo anteriormente existente. Esta norma desenvolvida pelo grupo JVT faz também parte da norma MPEG-4 (parte 10), oferecendo assim a norma MPEG-4 uma ferramenta mais potente para codificação de textura, voltada em relação aos vários perfis de codificação de vídeo que esta norma já disponibiliza com base na tecnologia especificada na parte 2 (Visual). Para alcançar uma capacidade maior de compressão, a norma H.264/AVC utiliza soluções de codificação de vídeo mais complexas que as existentes nas normas anteriores, necessitando de uma maior capacidade de processamento e memória. No entanto, a introdução de processadores mais rápidos e de memórias com maior capacidade a custos cada vez mais reduzidos parece garantir um futuro promissor para a norma H.264/AVC. Nota-se que a introdução de uma nova norma não significa que as normas anteriores vão ser substituídas. Primeiro, porque ainda existem produtos conforme a norma H.264/AVC; segundo, porque os perfis SP e ASP da norma MPEG-4 Visual já foram adotados por outras normas ou especificações (ISMA ou 3GPP); terceiro, porque para alguns ambientes o aumento da complexidade é difícil de aceitar, como por exemplo, em terminais móveis onde o custo do terminal e a duração da bateria são fatores importantíssimos; e finalmente, o fato dos perfis SP e ASP da norma MPEG-4 Visual terem já resolvido as questões ligadas ao licenciamento da tecnologia pode ser também uma vantagem extremamente importante. Um dos exemplos mais paradigmáticos é a norma de codificação de áudio AAC (*Advanced Audio Coding*), incluída na norma MPEG-2 e MPEG-4 Áudio, que apesar de um desempenho superior (e uma maior complexidade) ainda não atingiu os níveis de popularidade do famoso mp3 (MPEG-1/2 Áudio *layer* III).

A escalabilidade com elevada granularidade oferecida pela norma MPEG-4 FGS é adequada à distribuição de vídeo para vários tipos de redes com diferentes características, pois suporta variações abruptas do *bit rate* do canal e a ocorrência de erros de uma forma robusta. A estrutura de codificação de vídeo mantém-se simples, flexível e com uma complexidade reduzida. A norma H.264/AVC pode beneficiar com a introdução de uma estrutura de codificação escalável semelhante e, uma vez que está inserida na norma MPEG-4, é desejável a sua integração com outras das ferramentas de codificação de vídeo já existentes nesta norma. No entanto, apesar destas vantagens, a escalabilidade com elevada granularidade ainda não foi incluída na norma H.264/AVC. Contudo, esta funcionalidade é considerada como um dos tópicos de trabalho mais importantes para uma fase de desenvolvimento posterior da norma H.264/AVC; o desenvolvimento de um codificador escalável baseado nesta norma tem suscitado um interesse por parte de indústria (LI, 2002-3) (WU, 2003), que assim vê uma forma de melhorar o desempenho do MPEG-4 FGS em termos de eficiência de codificação.

Um dos principais problemas da codificação escalável de vídeo com elevada granularidade (FGS) incluída na norma MPEG-4 Visual (parte 2) é a sua quebra de desempenho em comparação com a codificação não escalável (perfil ASP), tal como foi demonstrado anteriormente. O desempenho da camada base da norma MPEG-4 FGS influencia significativamente o desempenho do sistema FGS completo. Assim, a utilização de um codificador não escalável H.264/AVC na camada base do FGS deverá permitir um desempenho superior em comparação com o uso do perfil ASP da norma MPEG-4 Visual. No entanto, a implementação direta da camada superior do MPEG-4 FGS, sem qualquer modificação, sobre a norma H.264/AVC, acrescentando-lhe uma camada adicional, possui grandes desvantagens uma vez que a norma H.264/AVC utiliza ferramentas de codificação de vídeo diferentes do perfil MPEG-4 ASP, sendo assim necessária uma duplicação de ferramentas com as mesmas funcionalidades. Por exemplo, a utilização do MPEG-4 FGS sem modificação no topo do H.264/AVC, obrigaria a utilização da transformada DCT na camada superior (o que não aconteceria na camada base) o que aumentaria a complexidade do codificador e do decodificador desnecessariamente. Além disso, é desejável uma reutilização das ferramentas da camada base na camada superior, facilitando a implementação e tirando partido das boas características (ex: baixa complexidade da transformada (MALVAR, 2002-2)) das novas ferramentas de codificação de vídeo presentes na norma H.264/AVC.

O principal objetivo deste capítulo é o desenvolvimento de um novo codificador escalável, referido como H.264/FGS, integrando uma camada base baseada no H.264/AVC com uma camada superior baseada na codificação em planos de bit utilizada na norma MPEG-4 FGS. O H.264/FGS utiliza na camada superior as mesmas ferramentas básicas de codificação do H.264/AVC já utilizadas na camada base, como a transformada inteira e a técnica de codificação entrópica UVLC com códigos *exp-Golomb*, tal como são definidas na norma H.264/AVC. A sintaxe do *bitstream* da camada superior MPEG-4 FGS é modificada de maneira a suportar estas modificações. O H.264/FGS é um esquema de codificação escalável verdadeiramente baseado em H.264/AVC usando a técnica de codificação em planos de bit para alcançar a escalabilidade fina. Resumindo, este capítulo tem como principais objetivos:

- Desenvolvimento de um codificador escalável adotando uma estrutura de escalabilidade semelhante a utilizada no MPEG-4 FGS, mas utilizando com ferramentas de codificação as ferramentas do codificador não escalável H.264/AVC, quer na camada base, quer na camada superior, mantendo as principais funcionalidades (ex: simplicidade, adaptação ao *bit rate* do canal, robustez a erros, etc.) do MPEG-4 FGS intactas.
- Proposta de métodos alternativos para a codificação escalável de vídeo na camada superior que tirem partido das novas ferramentas de codificação presentes na norma H.264/AVC.
- Estudo do desempenho do codificador desenvolvido, H.264/FGS.

Este capítulo encontra-se organizado da seguinte forma: primeiro apresenta-se a arquitetura do sistema escalável desenvolvido, a codificação em planos de bit, um estudo estatístico da distribuição dos elementos de sintaxe a codificar para cada plano de bit e algumas considerações sobre a codificação entrópica na camada superior. Finalmente, apresenta-se uma descrição aprofundada da sintaxe e da semântica do *bitstream* do novo codificador H.264/FGS e faz-se a avaliação do desempenho do

codificador em relação ao codificador escalável MPEG-4 FGS e ao codificador não escalável H.264/AVC, perfil *Baseline*.

5.1 Arquitetura H.264/FGS

Como é ilustrado na **figura 5.1**, a arquitetura do codificador H.264/FGS necessita de dois andares de codificação, um para a camada base e outro para a camada superior. A camada base utiliza um codificador de vídeo H.264/AVC, tendo-se adotado o perfil *Baseline* por possuir uma boa relação qualidade/complexidade. Tal como na norma MPEG-4 FGS, depois do processo de codificação estar completo, o *bitstream* gerado pelo codificador H.264/FGS da camada superior pode ser cortado em qualquer ponto, sendo o decodificador capaz de decodificar qualquer *bitstream* cortado. No codificador H.264/FGS, a imagem residual a codificar pela camada superior é calculada no domínio do tempo, e corresponde à diferença entre a imagem original e a imagem decodificada para o mesmo instante de tempo (antes da compressão de movimento) pela camada base. No entanto, o cálculo da imagem residual também se poderia efetuar no domínio da freqüência, de uma forma semelhante ao que é permitido pelo MPEG-4 FGS.

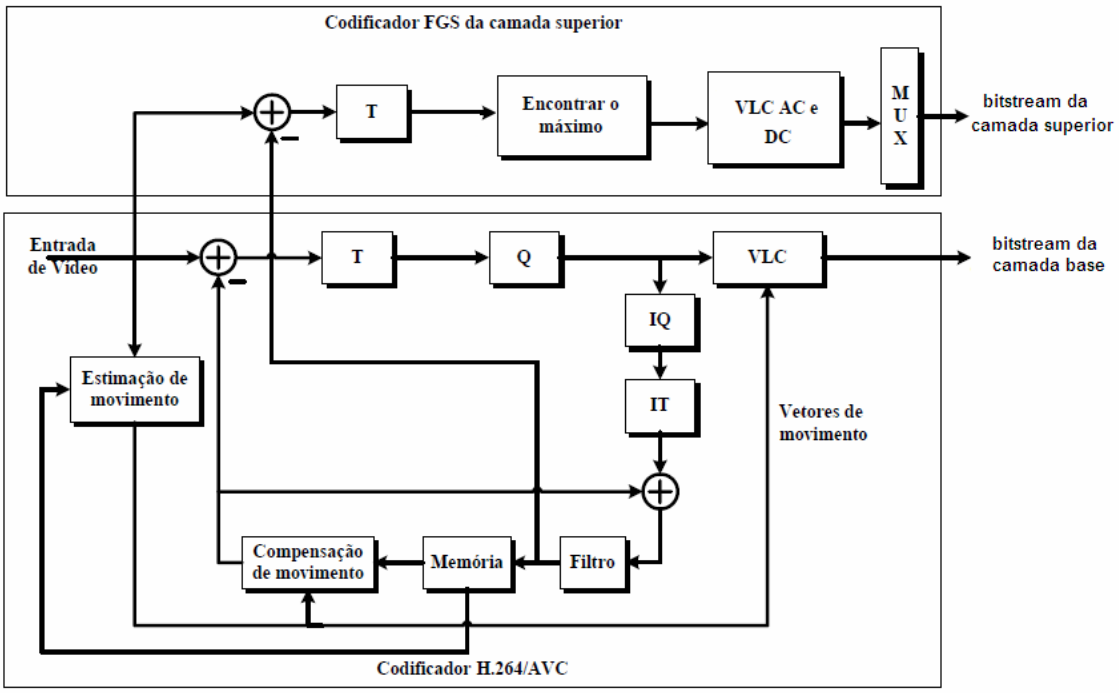


Figura 5.1: Arquitetura do codificador escalável H.264/FGS.

Tal como se pode verificar na **figura 5.1**, a arquitetura do codificador apresenta algumas novidades em relação ao MPEG-4 FGS, sendo as mais importantes as seguintes:

- **Transformada:** O codificador da camada superior utiliza a mesma transformada que o codificador da camada base utiliza para os macroblocos Intra 16x16. Esta transformada é hierárquica uma vez que utiliza a transformada de 4x4 baseada na DCT aplicada às amostras da imagem residual e a transformada de Hadamard aos coeficientes DC resultantes.

- **Codificação separada dos coeficientes DC da luminância:** Após a transformada, a imagem residual é convertida para o domínio da freqüência e obtém-se três tipos de blocos:
    - Blocos 4x4 de coeficientes DC da luminância (DCLum).
    - Blocos 2x2 de coeficientes DC das crominâncias (DCChr).
    - Blocos 4x4 de coeficientes AC das crominâncias (ACChr) e luminância (ACLum).

  Na camada superior, todos os blocos com coeficientes DCLum que pertencem a um plano de bit são agrupados e transmitidos em conjunto ao decodificador. Desta maneira, enviam-se primeiro todos os coeficientes DC perceptualmente mais importantes; os restantes coeficientes são enviados para cada macrobloco, segundo a ordem: ACLum (16 blocos), DCChr (2 blocos) e ACChr (8 blocos). Esta organização do *bitstream*, é diferente da utilizada no MPEG-4 FGS (mas semelhante a definida no H.264/AVC) que envia os coeficientes, independentemente da sua importância perceptual do decodificador, segundo a ordem Y, U e V.

- **Dois módulos de codificação entrópica:** A codificação entrópica é realizada através de códigos ULVC, de forma semelhante à usada na camada base do H.264/AVC (perfil *Baseline*). Esta solução foi adotada devido a sua reduzida complexidade. Para explorar a diferente distribuição estatística dos coeficientes DC em relação aos restantes coeficientes, utilizam-se dois módulos de codificação entrópica ULVC com tabelas diferentes. Finalmente, os coeficientes DC são multiplexados com os restantes coeficientes, no mesmo *bitstream*, através de um *multiplexer*.

No MPEG-4 FGS, o módulo "Encontrar o máximo" tem como objetivo encontrar o número máximo de planos de bit necessários para representar um quadro para as componentes Y, U e V. No H.264/FGS, este módulo tem uma função diferente, ou seja, encontrar o número máximo de planos de bit necessários para representar os coeficientes DC e AC (para a luminância e crominâncias), separadamente. Os quatro valores *maximum_level_y_dc, maximum_level_chr_dc, maximum_level_y_ac, maximum_level_chr_ac* são codificados no cabeçalho de cada quadro H.264/FGS e indicam ao decodificador o número máximo de planos de bit para os vários tipos de coeficientes da transformada. No exemplo da **figura 5.2**, codificar o primeiro plano de bit apenas envolve a codificação dos coeficientes DC da luminância, enquanto que codificar o segundo plano de bit envolve já a codificação dos coeficientes DC da luminância e crominância e os coeficientes AC da crominância, e assim sucessivamente.

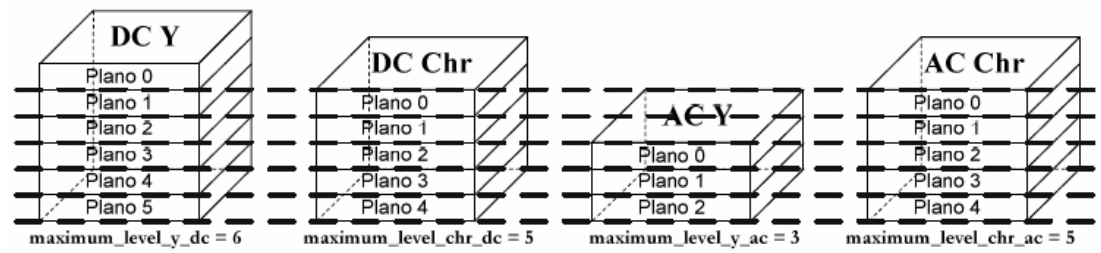


Figura 5.2: Exemplo de codificação H.264/FGS com um número de planos de bit diferente para cada tipo de coeficientes.

A arquitetura do decodificador H.264/FGS (**figura 5.3**) é semelhante a do decodificador MPEG-4 FGS, pois decodifica-se a camada base e a camada superior separadamente e no fim adiciona-se a imagem da camada base com a imagem residual obtida a partir da camada superior.

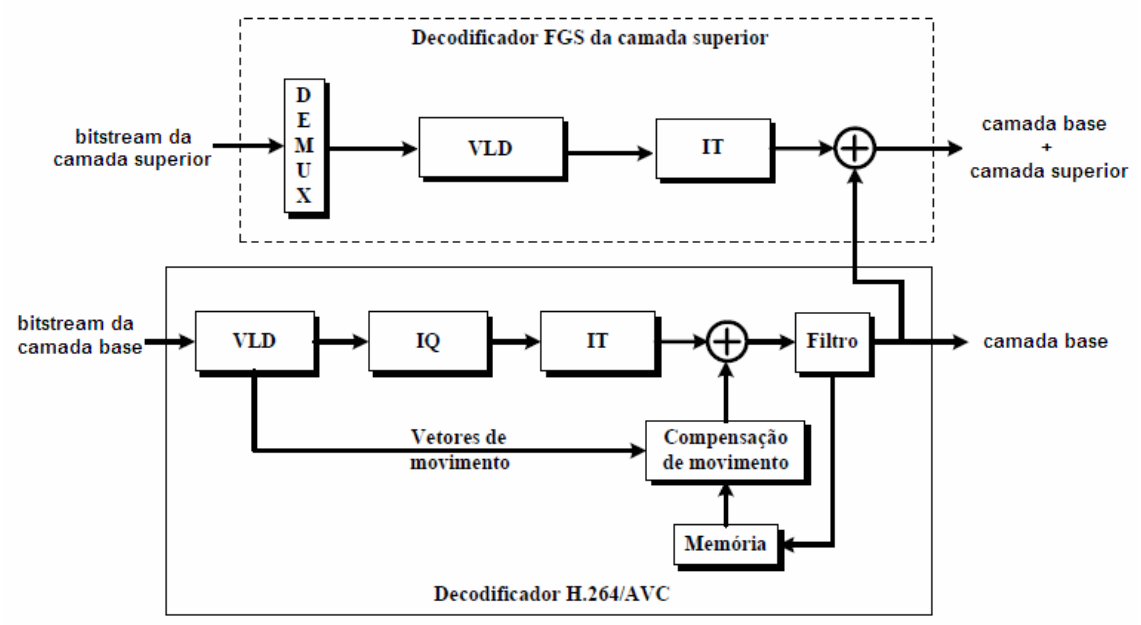


Figura 5.3: Arquitetura do decodificador escalável H.264/FGS.

No decodificador H.264/FGS, as mesmas operações (ou mais precisamente as operações inversas correspondentes) efetuadas no codificador são agora efetuadas por ordem inversa, começando-se por separar os coeficientes DCLum dos coeficientes DCChr e AC através de um *demultiplexer*. Todos os coeficientes são decodificados entropicamente (VLD), sendo os coeficientes obtidos convertidos para o domínio do tempo através da transformada inversa. Como a transformada inversa definida pela norma H.264/AVC integra a normalização com a quantificação, utilizam-se os fatores de normalização/quantificação definidos para o passo de quantificação máximo.

5.1.1 Melhoria seletiva dos coeficientes DC

Uma das novidades da arquitetura H.264/FGS, em relação ao MPEG-4 FGS consiste na codificação separada dos coeficientes DC da luminância em relação aos restantes coeficientes. É de conhecimento que alguns coeficientes da transformada, tais como os coeficientes DC e os coeficientes AC de mais baixa freqüência, possuem, tipicamente, um impacto visual mais significativo que outros coeficientes. No entanto, o método de codificação em planos de bit não distingue as diferentes componentes de freqüência associadas aos vários coeficientes da transformada. Deste modo, o MPEG-4 FGS definiu um mecanismo de seleção de freqüências, para o qual os coeficientes perceptualmente mais importantes (normalmente de baixa freqüência) são elevados para um plano de bit mais elevado. Quando o *bitstream* é cortado, os coeficientes da DCT com maior importância são representados com um número de bits maior, com uma maior exatidão.

No H.264/FGS, o objetivo é o mesmo; no entanto, realizado de uma forma diferente: primeiro, porque a transformada é realizada em blocos de menor dimensão (4x4), o que por si só já permite uma maior redução de artefatos; segundo porque no MPEG-4 FGS,

para se efetuar esta operação, utilizam-se tabelas VLC diferentes o que representa um acréscimo de complexidade e de memória; o que é desnecessário no H.264/FGS, uma vez que a transformada utilizada (e a própria arquitetura) já faz uma separação entre os coeficientes DC e AC.

Assim no H.264/FGS, definiu-se um mecanismo semelhante à seleção de freqüências do MPEG-4 FGS, referido como melhoria seletiva dos coeficientes DC que é realizada colocando um conjunto pré-definido (pelo codificador) de planos de bit dos coeficientes DCLum antes dos restantes no *bitstream*. Quando o *bitstream* for cortado, apenas os coeficientes DC da luminância irão possuir um maior número de bits a representá-los e consequentemente uma maior exatidão, uma vez que os bits mais significativos que os representam são colocados primeiro no *bitstream*. Como se codificam separadamente os coeficientes DC dos restantes, para realizar esta funcionalidade apenas é necessário organizar o *bitstream* de uma forma diferente, transmitindo os planos de bit mais significativos dos coeficientes DC antes de se começarem a transmitir os restantes. O número de planos de bit DCLum colocados antes dos restantes é indicado ao decodificador no começo do quadro da camada superior (no cabeçalho) através do elemento de sintaxe *fgs_vop_dc_enhancement*, permitindo ao decodificador realizar a operação inversa. O *multiplexer* é o principal responsável pela realização desta operação, pois apenas é necessário implementar uma reorganização dos dados codificados sendo o processamento adicional muito reduzido. A **figura 5.4** apresenta um exemplo de *bitstream* com e sem deslocamento dos planos de bit dos coeficientes DCLum, onde os planos de bit MSB-1 e MSB-2 (o MSB já se encontrava antes dos restantes) dos coeficientes DC da luminância são colocados antes dos restantes coeficientes DCChr, ACLum e ACChr.

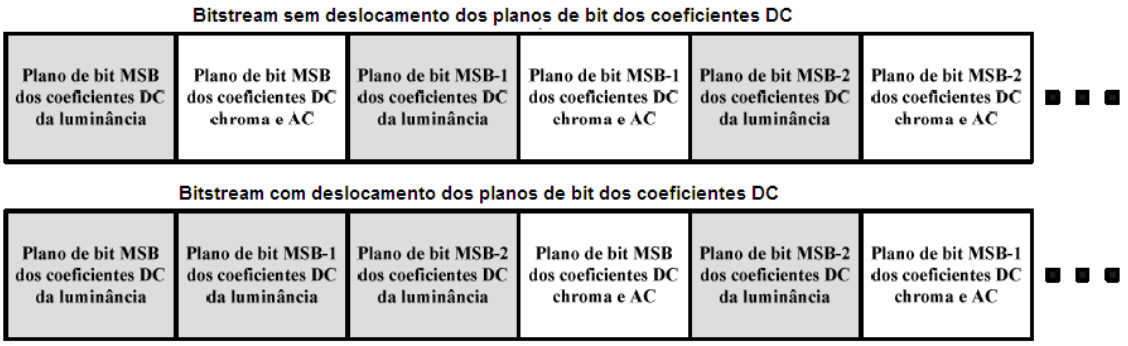


Figura 5.4: Exemplo de reorganização do *bitstream* para os planos de bit DC da luminância.

Além disso, esta organização do *bitstream* permite que os planos de bit que correspondem aos coeficientes DCLum possam ser de um nível de proteção mais elevado, através de técnicas de codificação de canal, uma vez que possuem um maior impacto na qualidade da imagem.

5.2 Codificação H.264/FGS sem planos de bit

Da mesma maneira que ocorre na norma MPEG-4 FGS, cada bloco 4x4 ou 2x2 com coeficientes da transformada é varrido em zig-zag para um vetor. Cada plano de bit de um bloco de coeficientes é definido com um vetor com 16 ou 4 bits de comprimento, onde os seus elementos (bits com o valor ‘0’ou ‘1’) são extraídos a partir dos valores

absolutos em binário dos coeficientes. Para cada plano de bit de cada bloco, símbolos (*run, EOP*) são calculados e entropicamente codificados.

Para os planos de bit MSB e MSB-1, um elemento de sintaxe ao nível do macrobloco é utilizado para indicar se existem elementos diferentes de 0 em um determinado plano de bit de um macrobloco de coeficientes, como ocorre no MPEG-4 FGS. Este elemento, chamado de *fgs_cbp* (*fgs coded block pattern*), permite a codificação eficiente de macroblocos que contenham um ou mais sub-macroblocos 8x8 com todos os elementos a 0; este caso é referenciado como ALL_ZERO. A idéia consiste em agrupar os sub-macroblocos de cada macrobloco e codificar os casos ALL_ZERO em conjunto. Vale lembrar que um sub-macrobloco contém 4 blocos (4x4) e tem sempre a dimensão de 8x8 na camada superior, tal como na camada base H.264/AVC.

O *fgs_cbd* é codificado entropicamente e indica quais os sub-macroblocos 8x8 – luminância (AC) e crominância (AC e DC) – de um plano de bit de um macrobloco possuem elementos diferentes de 0; como cada sub-macrobloco contém 4 blocos de 4x4 (aos quais é aplicada a transformada direta), o *fgs_cbd* indica se um ou mais blocos que pertencem ao mesmo sub-macrobloco contém elementos diferentes de 0.

Os 4 bits menos significativos do *fgs_cbd* contém informação sobre qual dos 4 sub-macroblocos 8x8 de luminância de um macrobloco contém elementos diferentes de 0 (*fgs_cbd_y*). Um valor '0' na posição *n* do *fgs_cbd* (representação binária) significa que o sub-macrobloco 8x8 correspondente não tem elementos diferentes de 0 enquanto o valor '1'significa que o sub-macrobloco 8x8 possui um ou mais elementos com o valor 1. Para as crominâncias, apenas dois bits *nc* são necessários para indicar se os sub-macroblocos 8x8 da crominância contém coeficientes (para um determinado plano de bit), de acordo com as seguintes regras:

- **nc = 0:** Nenhum coeficiente de crominância.
- **nc = 1:** Existe um ou mais coeficientes DC diferentes de 0 e todos os restantes coeficientes AC são iguais a 0. Não é enviada nenhuma informação para os coeficientes AC da crominância.
- **nc = 2:** Existe um ou mais coeficientes DC diferentes de 0 e pelo menos um coeficientes AC é diferente de 0. Neste caso é necessário enviar 10 EOPs (2 para os coeficientes DC e 2x4=8 para os 8 blocos 4x4) para a crominância de um macrobloco.

O valor final do *fgs_cbp* para um macrobloco é: $fgs\_cbp = fgs\_cbp\_y + (nc << 4)$ onde << representa a operação deslocamento para direita. Na **figura 5.5** é representado o *fgs_cbp* e a forma como este é construído a partir dos sub-macroblocos da luminância e crominância. É importante lembrar que o *fgs_cbp* apenas é enviado para os dois planos de bit mais significativos (MSB e MSB-1), uma vez que a ocorrência de sub-macroblocos ALL_ZERO é mais elevada nesses planos de bit.

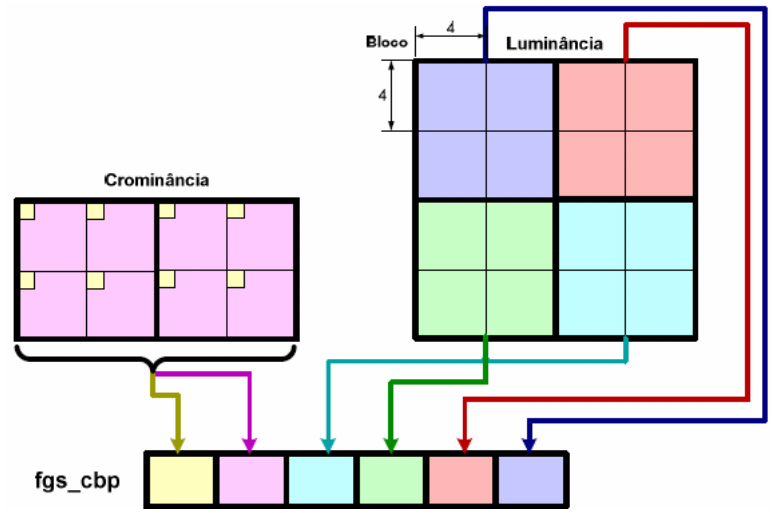


Figura 5.5: Construção do valor final do *fgs_cbp*.

Para os restantes planos de bit, é enviado um elemento de sintaxe de 1 bit ao nível do sub-macrobloco 8x8, referido com *fgs_msb_not_reached*. Este elemento de sintaxe indica (para os planos de bit MSB-n com n > 1) qual é o plano de bit mais significativo para cada sub-macrobloco. Se este valor for 1, indica que o sub-macrobloco contém todos os elementos a zero; se for 0, indica que existe um ou mais elementos com o valor 1, tendo sido alcançado o plano de bit mais significativo. Depois do elemento de sintaxe *fgs_msb_not_reached* possuir o valor 1, não é enviado mais nenhuma vez para o decodificador, mesmo que todos os elementos (de um sub-macrobloco) sejam iguais a zero; nesse caso, é necessário codificar entropicamente 4 pares (*run*=0, *EOP*=1), um para cada bloco.

5.3 Estudo estatístico para a camada superior

Os módulos VLC e VLD definidos na arquitetura efetuam a codificação e decodificação entrópica dos símbolos (*run, EOP*) e *fgs_cbp*, descritos anteriormente. O módulo VLC substitui cada símbolo por uma palavra de código com um comprimento variável de bits. As palavras de código são atribuídas em função da distribuição estatística dos símbolos a codificar. Aos símbolos que ocorrem mais frequentemente são atribuídas as palavras de código com menos bits (dimensão reduzida) e aos símbolos que ocorrem menos frequentemente são atribuídas palavras de código com número maior de bits.

Para que a atribuição das palavras de código a cada símbolo no codificador H.264/FGS seja adequada, foi efetuado um estudo estatístico exaustivo para todos os símbolos a se codificar na camada superior: *fgs_cbp* e os pares (*run, EOP*) obtidos a partir dos coeficientes DC e AC da luminância e crominâncias. As condições de treino utilizadas para este estudo são resumidas na **tabela 5.1**. Em primeiro lugar, um número bastante significativo de seqüências (10), representativas de vários tipos de conteúdo, foi escolhido. Para se efetuar o estudo estatístico, optou-se por utilizar um passo de quantificação constante para o codificador da camada base, sem controle de *bit rate*. Passos de quantificação baixos, dão origem a *bit rates* superiores (> 1 MBit/s) para o *bitstream* da camada base, sendo o resíduo a codificar pela camada superior representado por um menor número de planos de bit (ou menor energia). Por outro lado, passos de quantificação elevados dão origem a *bit rates* mais baixos (< 128 Kbit/s) para

o *bitstream* da camada base, sendo o resíduo a se codificar pela camada superior representado com um número maior de planos de bit (ou maior energia). Os passos de quantificação escolhidos representam uma ampla gama de *bit rates* para o codificador da camada base. O número de quadros de referência que, para uma dada imagem, o codificador da camada base pode escolher para efetuar a compressão de movimento foi limitado a 5, um compromisso entre a memória ocupada e o desempenho da camada base do H.264/FGS.

Tabela 5.1: Condições de treino para determinar a estatísticas dos símbolos H.264/FGS na camada superior.

Seqüências	Akiyo, Big_show, F1, Fair, Hall, Lts, Mobile, Novel, Letters, Stefan
Resolução espacial	QCIF, CIF
Passo de quantificação	16, 22, 28, 32, 35, 37, 39
Configuração	IPPPPP
Número de quadros de referência	5

As ferramentas restantes de codificação da camada base obedecem ao perfil H.264/AVC *Baseline* e a estrutura de codificação utilizada é do tipo IPP(P), o primeiro quadro é codificado com o modo Intra (I) e todos os quadros seguintes com o modo Inter (P). A resolução temporal utilizada é a mesma que a freqüência de quadro da seqüência original; a resolução temporal não possui uma influência significativa na distribuição estatística dos símbolos uma vez que não é utilizada predição entre quadros na camada superior. Os fatores que influenciam mais significativamente a distribuição estatística dos símbolos são a seqüência de treino utilizada e o passo de quantificação.

5.3.1 *Coded Block Pattern* (*fgs_cbp*)

O *fgs_cbp* é enviado ao nível de macrobloco, apenas para os planos de bit MSB e MSB-1, para indicar se os sub-macroblocos 8x8 contém elementos diferentes de zero ou não; a única exceção é o caso dos blocos 4x4 que contém coeficientes DC de luminância que são codificados separadamente dos restantes. O elemento de sintaxe *fgs_cbp* é codificado de uma forma diferente para os planos de bit MSB e MSB-1. Um caso especial é também contemplado para o plano de bit MSB-1, quando um sub-macrobloco 8x8 do plano de bit anterior (MSB) contém todos os bits com o valor zero (*Exceptcode* = 1). Este caso especial também existe na norma MPEG-4 FGS, permitindo aumentar a eficiência de codificação para o elemento de sintaxe *fgs_cbp*, uma vez que a distribuição estatística é diferente para o plano de bit MSB-1 com um ou mais elementos diferentes de zero ou com todos os elementos (de um sub-macrobloco 8x8) iguais a zero no plano de bit anterior (MSB).

No H.264/FGS, cada plano de bit é constituído por elementos com o valor 0 ou 1 e classificado de acordo com as componentes de luminância e crominância que contém. O elemento de sintaxe *fgs_cbp* é codificado diferentemente para cada plano de bit, de acordo com o tipo de informação que contém, se o plano de bit apenas possuir uma componente de crominância, o *fgs_cbp* não envia nenhuma informação de luminância. O plano de bit pode ser classificado em três tipos:

- **Luminância (YYYY):** O plano de bit apenas contém informação de luminância e o *fgs_cbp* é representado com 4 bits (é igual ao *fgs_cbp_y*) que

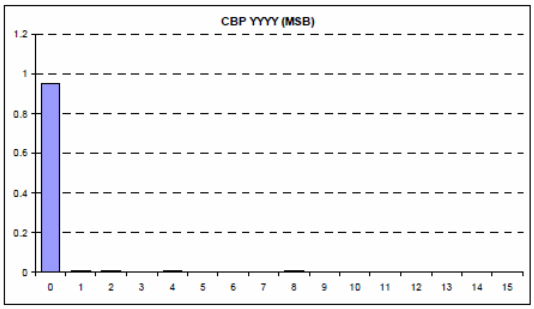
indicam quais são os sub-macroblocos de luminância que contém elementos diferentes de zero.

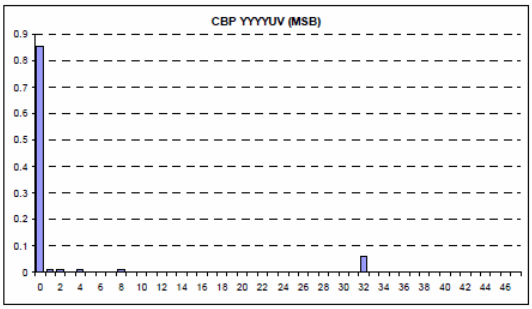
- **Crominância (UV):** O plano de bit apenas contém informação de crominância e o *fgs_cbp* é representado com 2 bits (é igual ao *nc*) que indicam quais são os sub-macroblocos da crominância que contém elementos diferentes de zero.
- **Luminância + Crominância (YYYYUV):** O plano de bit contém luminância e crominância e o *fgs_cbp* é representado com 6 bits (*fgs_cbp*) que indicam quais os sub-macroblocos de luminância e crominâncias que contém elementos diferentes de zero.

O codificador envia a classificação sobre o tipo de plano de bit para o decodificador através de um conjunto de campos (*maximum_level_chr_dc, maximum_level_y_ac, maximum_level_chr_ac*) ao nível do quadro, ao contrário do *fgs_cbp* que é enviado ao nível do macrobloco.

É apresentado a seguir um estudo estatístico do *fgs_cbp*, realizado no contexto desta dissertação, para os três tipos de planos de bit: YYYY, UV e YYYYUV e para os três níveis de plano de bit: MSB, MSB-1 e MSB-1 com *Exceptcode* =1, o que totaliza 9 combinações. Como é muito rara (< 1%) a ocorrência de um plano de bit UV, ou seja, de um plano de bit apenas com informação de crominância, representa-se na **figura 6.6** apenas a distribuição estatística para os planos de bit YYYY e YYYYUV, para os dois planos de bit mais significativos (MSB e MSB-1).

No eixo horizontal da **figura 5.6**, apresentam-se os valores que o *fgs_cbp* pode assumir (em decimal) e no eixo vertical a probabilidade de ocorrência desse valor (a soma de todas as probabilidades é 1). Para o plano de bit YYYY, a representação binária do *fgs_cbp* pode assumir 16 valores, desde ‘0000’ a ‘1111’; para o plano de bit YYYYUV, a representação binária do *fgs_cbp* pode assumir 47 valores, desde ‘000000’ a ‘101111’.





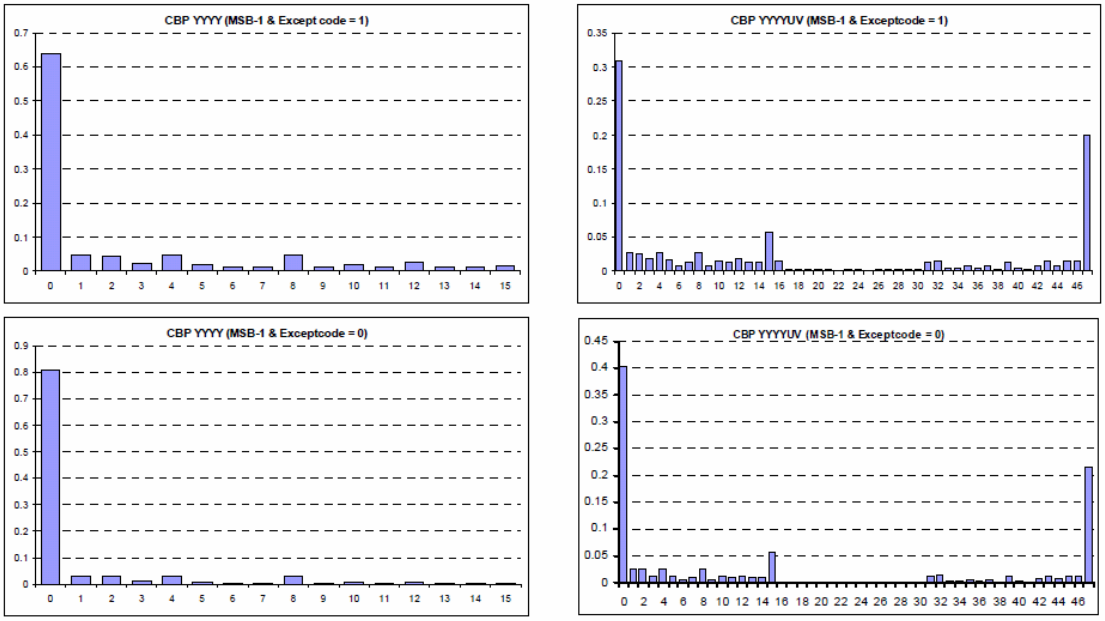


Figura 5.6: Distribuição estatística do *fgs_cbp* (*fgs coded block pattern*).

Tal com se pode verificar na **figura 5.6**, para qualquer um dos planos de bit estudados, o valor mais freqüente em qualquer situação é o valor 0. O valor 0 corresponde à situação em que todos os sub-macroblocos de um macrobloco são ALL_ZERO, todos os elementos de cada sub-macrobloco de um macrobloco são zero. Para o plano de bit MSB, a probabilidade de ocorrência do valor 0 é elevada: 95% para YYYY e 85% para YYYYUV; esta fato era esperado uma vez que corresponde a ter outros coeficientes com uma magnitude elevada. Para o plano de bit MSB-1 com *ExceptCode* = 1, a probabilidade de ocorrência do valor é menor: 63% para YYYY e 31% para YYYYUV. Para os planos de bit do tipo YYYYUV, também existe uma probabilidade elevada (20%) do valor 47, que corresponde a uma representação binária de 101111; todos os sub-macroblocos de um macrobloco contém elementos diferentes de zero: com o valor de *nc* = 2 (10) e *fgs_cbp_y* = 1 (1111). Para o plano de bit MSB-1, com um ou mais elementos diferentes de zero no plano de bit anterior (MSB), a probabilidade de ocorrência do valor 0 é 81% para YYYY e 41% para YYYYUV. Para esses casos, a utilização do elemento de sintaxe *fgs_cbp* ao nível do macrobloco é sempre vantajosa, uma vez que o valor com a maior probabilidade possui um código de 1 bit. Se o *fgs_cbp* fosse enviado ao nível do sub-macrobloco, seria necessário enviar 6 bits para um plano de bit YYYYUV e 4 para o plano de bit YYYY para assinalar o caso mais provável (todos os sub-macroblocos com o valor ALL_ZERO), para cada macrobloco.

Segundo o estudo efetuado para planos de bit MSB-2 e restantes, é rara (<1%) a ocorrência de um plano de bit apenas com informação de luminância (YYYY) e a ocorrência de um número significativo de sub-macroblocos com todos os elementos a zero já não é predominante (<20%) para os planos de bit YYYYUV. Como a principal vantagem na utilização do *fgs_cbp* consiste em codificar os casos ALL_ZERO em conjunto, para estes planos de bit não é vantajosa a utilização do *fgs_cbp*. Deste modo, para os planos de bit MSB-n com n > 1 é enviado ao decodificador o elemento de sintaxe *fgs_msb_not_reached* que indica se o plano de bit mais significativo foi alcançado para um determinado sub-macrobloco 8x8.

5.3.2 Coeficientes DC

A distribuição estatística dos coeficientes DC na camada superior é apresentada na **figura 5.7** para os planos de bit MSB, MSB-1 e MSB-n com n>1, isto para os coeficientes DC da luminância e crominâncias. No eixo vertical apresenta-se a probabilidade de ocorrência de um determinado par (*run*, *EOP*). Para os blocos DC 4x4 da luminância, o *run* pode assumir os valores de 0 a 15 para *EOP* = 0 e *EOP* = 1. De uma forma semelhante, para os blocos DC 2x2 da crominância, o *run* pode assumir os valores de 0 a 3 para *EOP* = 0 e *EOP* = 1. Deste modo, no eixo horizontal da **figura 5.7a**, **b e c**, os valores de 0 a 15 representam o *run* quando *EOP* = 0 e os valores de 16 a 31 o *run* quando *EOP* = 1. Na **figura 5.7d** os valores de 0 a 3 representam o *run* quando *EOP* = 0 e os valores de 4 a 7 o *run* quando *EOP* = 1. A **figura 5.7d** apresenta a distribuição estatística dos coeficientes DC da crominância, de uma forma condensada para os planos de bit MSB, MSB-1 e MSB-n com n > 1. Para os coeficientes DC da luminância, a **figura 5.7a** apresenta a distribuição estatística para o plano de bit MSB, a b) para o MSB-1 e a c) para o MSB-n com n>1.

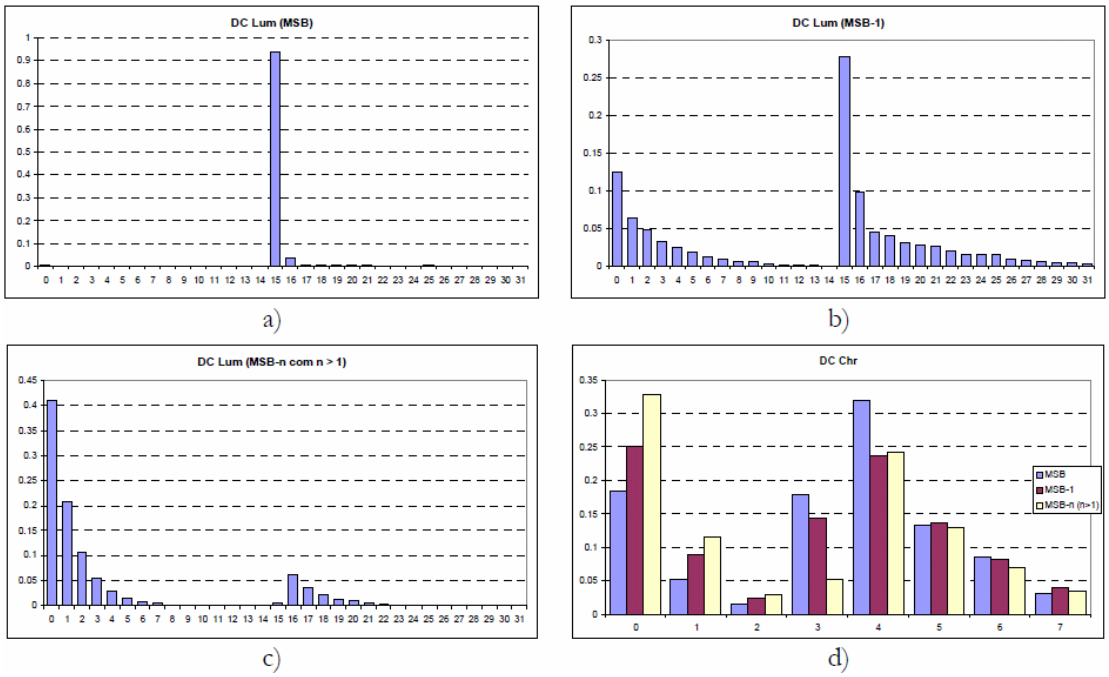


Figura 5.7: Distribuição estática dos coeficientes DC: Plano de bit da luminância a) MSB, b) MSB-1 e c) MSB-n com n>1; d) plano de bit da crominância.

Tal como mostra a **figura 5.7a**, para os planos de bit MSB e MSB-1, o par (*run*, *EOP*) mais freqüente é o par (15, 0) que corresponde a um bloco 4x4 com todos os elementos a zero, com probabilidade de ocorrência de 93% e 27%, respectivamente. Para o plano de bit MSB, o número de blocos com todos os elementos a zero é predominante, uma vez que o *fgs_cbp* não se aplica para os coeficientes DC da luminância e o número de coeficientes com uma grande magnitude é pequeno. Para o plano de bit MSB-n com n >1, o par (*run*, *EOP*) mais freqüente é o (0,0) que corresponde a um bloco com um elemento a 1 na posição (0,0).

A distribuição estatística para os elementos de um bloco apresenta a habitual característica que para um dado valor de *EOP* a probabilidade de ocorrência decrescer a medida que o valor de *run* cresce. Esta característica é esperada uma vez que os coeficientes DC da luminância (obtidos a partir da transformada inteira) são submetidos

à transformada de *Hadamaard* que descorrelaciona os coeficientes de um bloco e concentra a energia dos coeficientes de freqüência mais baixa. Os coeficientes DC da crominância apresentam uma distribuição estatística semelhante, com exceção do par (3,0) (todos os elementos de um bloco 2x2 a zero) que possui uma probabilidade de ocorrência superior ao par (2,0) para qualquer plano de bit.

5.3.3 Coeficientes AC

A distribuição estatística dos coeficientes AC para o plano de bit MSB, MSB-1 e MSB-n com n>1 é apresentada na **figura 5.8**. Tal como para os coeficientes DC, o eixo horizontal indica os pares (*run*, *EOP*) e o eixo vertical a respectiva probabilidade de ocorrência. Para o plano de bit MSB, o par (*run*, *EOP*) mais freqüente é o par (15,0) que corresponde a um bloco 4x4 com todos os elementos iguais a zero. É importante lembrar que esta probabilidade não reflete a probabilidade de ocorrência de todos os blocos com os elementos iguais a zero, uma vez que muitos são codificados através do elemento de sintaxe *fgs_cbp*. Para este valor de probabilidade apenas contribuem os sub-macroblocos 8x8 que possuem um ou mais elementos diferentes de zero e onde um ou mais blocos tem todos os elementos a zero. Para os planos de bit MSB-1 e MSB-n com n>1, a probabilidade de ocorrência do valor 15, decresce significativamente de 32% (MSB) para 5,7% (MSB-1) e 0,9 (MSB-n com n>1), pois a medida que o plano de bit é menos significativo há uma menor probabilidade de ocorrência de blocos com todos os elementos a zero.

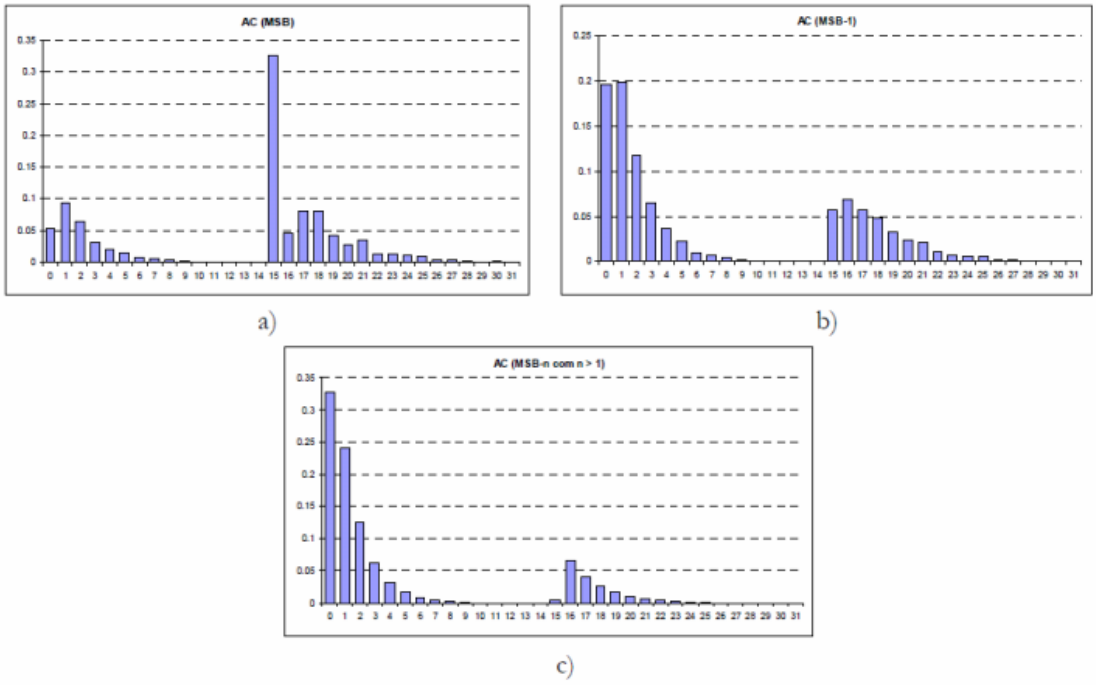


Figura 5.8: Distribuição dos coeficientes AC: Plano de bit a) MSB, b) MSB-1, e c) MSB-n com n>1.

Para os restantes planos de bit, valores pequenos de *run* com *EOP*=0 ocorrem mais frequentemente o que corresponde a blocos com um conjunto significativo de elementos a 1 e relativamente perto uns dos outros. Tal como os coeficientes DC, para um dado valor de *EOP*, a probabilidade de ocorrência decresce a medida que o valor de *run* cresce. No entanto, por inspeção visual, pode-se verificar que a distribuição estatística

dos coeficientes AC é diferente dos coeficientes DC de luminância, especialmente para o plano de bit MSB e MSB-1.

5.4 Codificação entrópica H.264/FGS

A norma H.264/AVC define três métodos de codificação entrópica: UVLC, CAVLC e CABAC; o ULVC e o CAVLC são utilizados no perfil *Baseline* e o CABAC e o ULVC no perfil *Main*. Os métodos CAVLC e CABAC possuem uma maior eficiência de codificação em relação ao ULVC à custa de um acréscimo de complexidade. O método UVLC utiliza um único conjunto de palavras de código para todos os elementos da sintaxe e com este método apenas é necessário especificar a correspondência entre os elementos de sintaxe a codificar e as palavras de código, de acordo com a distribuição estatística dos dados. Esta técnica elimina a necessidade de utilizar diferentes tabelas VLC para cada um dos elementos de sintaxe (como na norma MPEG-4 FGS) e possui uma baixa complexidade. A utilização de várias tabelas VLC na camada superior introduziria uma complexidade adicional ao sistema. Como uma das principais áreas de aplicação da codificação escalável de vídeo (FGS) é a transmissão de vídeo em redes móveis, para terminais com uma capacidade de processamento baixa, memória escassa e bateria limitada, o UVLC permite diminuir a complexidade global do sistema H.264/FGS. Estas vantagens levaram a escolha do método de codificação entrópica UVLC para o sistema H.264/FGS, apesar do decréscimo de desempenho que implica em comparação com os métodos CAVLC e CABAC. O método ULVC é bastante simples, uma vez que define as palavras de código de comprimento variável a utilizar através de uma estrutura regular. Para construir a tabela de codificação entrópica basta ordenar as probabilidades de ocorrência de cada símbolo e mapear cada símbolo na palavra de código *exp-Golomb* correspondente, para que sejam atribuídas palavras de código com uma dimensão menor aos símbolos com uma probabilidade maior de ocorrência e vice-versa. Na **tabela 5.2** apresenta-se uma parte da tabela de codificação UVLC utilizada na camada superior do H.264/FGS, para os pares (*run*, EOP) dos blocos (4x4) DC de luminância (DCLum). Os símbolos com valores entre 0 e 15 representam, para os coeficientes DCLum, o *run* quando EOP=0 e os valores de 16 a 31 o *run* quando EOP=1. A cada símbolo corresponde uma palavra de código (em que 0x representa o formato hexadecimal) de acordo com o tipo de dados que representa; nesta tabela DCLum MSB, DCLum MSB-1 ou DCLum MSB-n com n >1. A tabela UVLC completa possui mais 12 colunas: 6 para os coeficientes da transformada (3 para os coeficientes DC Chr e 2 para os coeficientes AC) e 6 para o elemento de sintaxe *fgs_cbp* (3 para o plano de bit YYYY e 3 para o plano de bit YYYYUV). Para o plano de bit UV, o *fgs_cbp* é codificado utilizando códigos de comprimento fixo (FLC), uma vez que a probabilidade de ocorrência do plano de bit UV é baixa (<1%). Um ponto importante a ressaltar é que o plano de bit MSB, no contexto da utilização da tabela UVLC, é definido ao nível do bloco (4x4 ou 2x2). Note-se que os vários blocos não têm necessariamente o mesmo número de bits; o plano MSB de cada bloco individual é o primeiro plano de bit que contém pelo menos um elemento a 1 (isso é, não é um bloco ALL_ZERO), indicado através do elemento de sintaxe *fgs_cbp* ou *fgs_msb_not_reached*.

Tabela 5.2: Parte da tabela UVLC para coeficientes DC de luminância: MSB e MSB-n (n>1).

Símbolos	Palavras de código		
	DCLum MSB	DCLum MSB-1	DCLum MSB-n
0	0x07	0x02	0x01
1	0x0E	0x04	0x02
2	0x011	0x05	0x03
3	0x015	0x08	0x05
4	0x016	0x0C	0x07
5	0x017	0x0E	0x0C
6	0x018	0x012	0x0F
7	0x019	0x014	0x011
8	0x0A	0x016	0x014
...	...	...	...

No MPEG-4 FGS são definidas quatro tabelas diferentes para os coeficientes da DC, uma para cada plano de bit: MSB, MSB-1, MSB-2 e MSB-n com *n* igual ao número máximo de planos de bit. Quando se utiliza seleção de freqüências, a distribuição estatística dos planos de bit é diferente, o que obriga a utilização de mais oito tabelas. Deste modo, o MPEG-4 FGS define 12 tabelas com uma dimensão variável, totalizando 588 códigos.

Como o número de elementos de um bloco na norma MPEG-4 FGS é 64, o número máximo de pares (*run*, *EOP*) possível é 128. Para evitar códigos de *Huffman* muito compridos, a norma MPEG-4 FGS define um código de ESCAPE para codificar símbolos com *run* elevado. Por outro lado, no H.264/FGS, o número total de elementos da tabela UVLC é 200: 3x32 (DCLum) + 3x32 (AC) + 3x8 (DCChr), o que resulta em um número inferior de códigos em relação ao MPEG-4 FGS, permitindo assim reduzir os requisitos de memória do codificador e decodificador.

Através de um estudo estatístico da eficiência teórica dos códigos UVLC (que não é aqui apresentado por razões de espaço), verificou-se que, no AVC_FGS, a distribuição estatística dos pares (*run*, *EOP*) para os planos de bit MSB-n com n > 1 é bastante semelhante à distribuição estatística individual dos planos de bit MSB-2, MSB-3, ... MSB-m com *m* igual ao número máximo de planos de bit. É importante lembrar que, na norma MPEG-4 FGS, os planos de bit MSB, MSB-1, MSB-2 e MSB-n com n > 2 são codificados separadamente através da utilização de diferentes tabelas, enquanto no H.264/FGS apenas se codificam os planos de bit MSB, MSB-1 e MSB-n com n > 1 separadamente, o que permite reduzir o número de entradas na tabela. Outra diferença entre a norma MPEG-4 FGS e o H.264/FGS é a codificação separada dos planos de bit dos coeficientes DC. Comparando as distribuições estatísticas dos coeficientes AC e dos coeficientes DC, nota-se por inspeção que as distribuições não são semelhantes; em conseqüência, a correspondência entre os códigos UVLC e os pares (*run*, *EOP*) depende do tipo de bloco a codificar: DCLum, DCChr ou AC. Esta separação do tipo de coeficientes a codificar permite uma maior adaptação da estatística do sinal a codificar e, consequentemente, um melhor desempenho.

5.5 Sintaxe e semântica do *bitstream* H.264/FGS

Nesta seção apresenta-se a sintaxe e a semântica do *bitstream* utilizada no sistema H.264/FGS proposto neste capítulo da dissertação. Esta sintaxe é semelhante à sintaxe do MPEG-4 FGS em todos os aspectos onde pode ser mantida a mesma solução para alcançar a mesma funcionalidade, como os códigos de sincronismo que assinalam o começo de um quadro e plano de bit, o tipo de quadro (I, P ou B), o número máximo de planos de bit para qualquer coeficiente, etc. Contudo, o resto da sintaxe é essencialmente diferente uma vez que tiveram de ser introduzidas alterações para suportar as novas ferramentas de codificação aqui propostas para a camada superior ou seja as ferramentas especificadas pela norma H.264/AVC. A sintaxe da camada superior do H.264/FGS foi estruturada em três níveis:

- **Nível de quadro (*FGSVideoObjectPlane*):** Para este nível é transmitido um conjunto de parâmetros necessários para decodificar os planos de bit da luminância e crominâncias. Estes parâmetros são utilizados para assinalar o começo de um quadro, a ordem dos quadros, o número de planos de bit necessários para representar os coeficientes DC e AC (luminância e crominâncias), etc.
- **Nível de plano de bit (*FGSDCLumBitplane* e *FGSDCChrACBitplane*):** Para este nível são transmitidos todos os bits correspondentes aos coeficientes AC e DC que pertencem a um determinado plano de bit. Para um determinado plano de bit, os coeficientes DC luminância (*FGSDCLumBitplane*) são codificados separadamente dos coeficientes DC da crominância e coeficientes AC (*FGSDCChrACBitplane*). Para o plano de bit *FGSDCLumBitplane*, são transmitidos os pares (*run*, *EOP*) codificados entropicamente e para o plano de bit *FGSDCChrACBitplane* é transmitido o elemento de sintaxe *fgs_cbp*, recorrendo ao próximo nível (*FGSBlock*) para transmitir os bits dos restantes coeficientes.
- **Nível de bloco (*FGSBlock*):** Neste nível são enviados pares (*run*, *EOP*) dos coeficientes DC da crominância e todos os coeficientes AC. Também é transmitido o elemento de sintaxe *fgs_msb_not_reached* (1 bit) para indicar se o plano de bit mais significativo de um bloco já foi alcançado.

É importante lembrar que, por razões de tempo, a sintaxe aqui proposta não suporta algumas características do MPEG-4 FGS como a melhoria seletiva de algumas regiões da imagem, a escalabilidade temporal e a codificação de material de vídeo no formato entrelaçado. No entanto, no caso de se pretender introduzir estas funcionalidades, as modificações necessárias são pequenas. A seguir é apresentada a sintaxe da camada superior do sistema H.264/FGS, na forma de quatro tabelas. Para cada elemento de sintaxe a se transmitir (mais carregados), indica-se o número de bits necessário para o representar; os restantes elementos correspondem a funções ou variáveis. Para cada elemento da sintaxe, apresenta-se também a sua semântica. A semântica de cada elemento de sintaxe é descrita, do ponto de vista do decodificador, da forma como o *bitstream* deve ser lido e interpretado, uma vez que é sempre o decodificador (e não o codificador) que dever normalizado para garantir interoperabilidade.

5.5.1 Sintaxe *FGSVideoObjectPlane*

Tabela 5.3: Sintaxe do elemento *FGSVideoObjectPlane*

FGSVideoObjectPlane() {	No. de bits
fgs_vop_start_code	32
fgs_vop_coding_type	2
vop_time_increment	8
fgs_vop_max_level_y_dc	5
fgs_vop_max_level_y_ac	5
fgs_vop_max_level_uv_dc	5
fgs_vop_max_level_uv_ac	5
maker_bit	1
fgs_vop_number_of_vop_bp_coded	5
fgs_vop_mc_bit_plane	5
fgs_vop_weight_bit_plane	5
fgs_vop_dc_enhancement	3
next_start_code()	
if (nextbits_bytealigned() == fgs_bp_start_code) {	
while (nextbits_bytealigned() != '000 0000 0000 0000 0000 0000' || next_bytealigned() == fgs_bp_start_code {	
fgs_bp_star_code	32
FGSCDLumBitplane()	
FGSDCChrACBitplane()	
}	
next_start_code()	
}	
}	

5.5.2 *Semântica do FGSVideoObjectPlane*

- ***fgs_vop_start_code*:** Assinala o começo de um VOP H.264/FGS. Consiste em um código fixo de 32 bits com valor ‘000001B9’ em hexadecimal.
- ***fgs_vop_coding_type*:** Identifica o tipo de quadro utilizado na camada superior, sendo do tipo Intra (I), Preditivo (P) ou Bi-preditivo (B). No H.264/FGS este elemento possui sempre o valor ‘0’ que corresponde ao tipo Intra.
- ***modulo_time_base*:** Indica a ordem dos quadros H.264/FGS. Este valor é incrementado de 1 a medida que se processam os quadros H.264/FGS. Quando o valor do *modulo_time_base* chega a 255, é colocado a zero no quadro seguinte.
- ***fgs_vop_max_level_y_dc*:** Especifica o número de planos de bit necessário para representar os coeficientes DC da luminância de um determinado quadro.
- ***fgs_vop_max_level_y_ac*:** Especifica o número de planos de bit necessário para representar os coeficientes AC da luminância de um determinado quadro.
- ***fgs_vop_max_level_uv_dc*:** Especifica o número máximo de planos de bit necessário para representar os coeficientes DC da crominância de um determinado quadro.

- ***fgs_vop_max_level_uv_ac*:** Especifica o número máximo de planos de bit necessário para representar os coeficientes AC da crominância de um determinado quadro.
- ***fgs_vop_number_of_vop_bp_coded*:** Especifica o número máximo de planos de bit para qualquer coeficiente.
- ***fgs_vop_dc_enhancement*:** Especifica o número de planos de bit DC da luminância que são colocados antes dos restantes no *bitstream*.
- ***next_start_code( ):*** Permite remover qualquer bit a zero de um conjunto de 0 a 7 bits a '1' utilizados para *stuffing* e localiza o próximo código (*start_code*) alinhado ao byte.
- ***nextbits_bytealigned( ):*** Retorna uma cadeia de 32 bits que começa na próxima posição alinhada ao byte. A posição atual do ponteiro de decodificação não é alterada por esta função.
- ***fgs_bp_start_code*:** Consiste em uma cadeia de 32 bits e assinala o começo de um plano de bit. Consiste em um código fixo de 27 bits com o valor '0000 0000 0000 0000 0000 0001 010' em binário e os últimos 5 bits representam um valor na gama de '00000' a '11111' em binário.

5.5.3 Sintaxe de *FGSDCLumBitplane, FGSACBitplane e FGSBlock*

Tabela 5.4: Sintaxe do elemento *FGSDCLumBitplane*.

FGSDCLumBitplane( ) {	No. de bits
if (start_decode_dc_lum == 1)	
for (i=0; i<mb_in_bitplane; i++) {	
while (eop == 0) {	
fgs_run_eop_code	1-11
if (coeff_msb_not_reached == 1)	
fgs_sign_bit	1
}	
}	
}	

Lembrando que a ordem de varredura dos coeficientes DC da crominância e todos os coeficientes AC é igual a ordem definida na norma H.264/AVC, primeiro seguem os coeficientes AC da luminância, depois os coeficientes DC da crominância e, finalmente os coeficientes AC da crominância.

Tabela 5.5: Sintaxe do elemento *FGSDCChrACBitplane*.

FGSDCChrACBitplane( ) {	No. de bits
for (i=0; i<mb_in_bitplane; i++) {	
if (fgs_vop_bp_id < 2)	
fgs_cbp	1-11
for (i=0; i<4; i++) {	
if (start_decode_ac_lum == 1)	
fgs_block (i)	
}	
for (i=0; i<2; i++) {	

if (start_decode_ac_chr == 1)	
fgs_block (i+4)	
}	
for (i=0; i<2; i++) {	
if (start_decode_ac_chr == 1)	
fgs_block (i+6)	
}	

Tabela 5.6: Sintaxe do elemento *FGSBlock.*

FGSBlock ( ) {	No. de bits
if (fgs_vop_bp_id > 1 && previous_fgs_msb_not_reached == 1) {	
fgs_msb_not_reached	1
if (fgs_msb_not_reached == 0) {	
while (eop == 0) {	
fgs_run_eop_code	1-11
if (coeff_msb_not_reached == 1)	
fgs_sign_bit	1
}	
}	
}	

5.5.4 Semântica do *FGSDCLumBitplane, FGSACBitplane e FGSBlock*

- ***mb_in_bitplane*:** Especifica o número de macroblocos que um plano de bit possui.
- ***eop*:** Indica se um bit com o valor '1' é o último bit a '1'de um bloco.
- ***fgs_run_eop_code*:** Código VLC para um par (RUN, EOP).
- ***coeff_msb_not_reached*:** Consiste em um valor interno (*flag*) que indica com o valor '1' que o MSB de um coeficiente associado com o *fgs_run_eop_code* foi alcançado. O valor '0' indica que o *fgs_run_eop_code* associado corresponde a um bit em um nível inferior (MSB-n com n>0) do coeficiente.
- ***fgs_sign_bit*:** Indica o valor do sinal de um coeficiente da transformada DCT inteira ou de Hadamard. O valor '0' indica o valor positivo e o valor '1' um valor negativo.
- ***fgs_vop_bp_id*:** Corresponde aos últimos 5 bits do código *fgs_bp_start_code*. O *fgs_vop_bd_id* identifica um plano de bit de uma forma única. O valor do *fgs_vop_bp_id* é '0' para o plano de bit mais significativo e é incrementado de 1 para cada plano de bit em um nível inferior.
- ***fgs_cbp*:** Código de comprimento variável com um comprimento entre 1 e 11 bits. Especifica o *coded bit pattern* do *fgs_msb_not_reached* de um macrobloco.

- ***start_decode_dc_lum*:** Consiste em um valor interno que indica se a decodificação de um *FGSBlock* com bits de coeficientes DC da luminância deve ser realizada ou não.
- ***start_decode_ac_lum*:** Consiste em um valor interno que indica se a decodificação de um *FGSBlock* com bits de coeficientes AC da luminância deve ser realizada ou não.
- ***start_decode_dc_chr*:** Consiste em um valor interno que indica se a decodificação de um *FGSBlock* com bits de coeficientes DC da crominância deve ser realizada ou não.
- ***start_decode_ac_chr*:** Consiste em um valor interno que indica se a decodificação de um *FGSBlock* com bits de coeficientes AC da crominância deve ser realizada ou não.

Os valores *start_decode_dc_lum, start_decode_ac_lum, start_decode_dc_chr e start_decode_ac_chr* indicam se um determinado plano de bit contém pelo menos um bit com o valor '1' e são calculados da seguinte forma:

```
start_decode_dc_lum = 0;
start_decode_ac_lum = 0;
start_decode_dc_chr = 0;
start_decode_ac_chr = 0;
if (maximum_level_y_ac >= maximum_level – fgs_vop_bp_id)
  start_decode_ac_lum = 1;
if (maximum_level_y_dc >= maximum_level – fgs_vop_bp_id)
  start_decode_dc_lum = 1;
if (maximum_level_chr_dc>= maximum_level–fgs_vop_bp_id)
  start_decode_dc_chr = 1;
if (maximum_level_chr_ac>= maximum_level–fgs_vop_bp_id)
  start_decode_ac_chr = 1;
maximum_level = MAX (maximum_level_y_ac,
 maximum_level_y_dc, maximum_level_chr_dc, maximum_level_chr_ac);
```

- ***fgs_msb_not_reached***: Este elemento de sintaxe é '1' se o plano de bit mais significativo de um *FGSBlock* não tiver sido alcançado, indica um *FGSBlock* com todos os elementos que o constituem a zero; caso contrário, é '0'.
- ***previous_fgs_msb_not_reached*:** Consiste no *fgs_msb_not_reached* decodificado no *FGSBlock* do plano de bit anterior para o mesmo bloco 4x4.

5.6 Estudo do desempenho do H.264/FGS

Para efetuar o estudo do desempenho do algoritmo H.264/FGS proposto neste trabalho foram desenvolvidos dois codificadores (e os respectivos decodificadores); ambos utilizam um codificador H.264/AVC na camada base. O primeiro codificador desenvolvido usa na camada superior uma solução conforme a norma MPEG-4 FGS e o segundo usa a solução proposta neste capítulo. Um codificador MPEG-4 FGS também será utilizado no decorrer dos testes.

O codificador H.264/AVC utilizado neste estudo é o modelo de referência (*Joint Model* – JM) versão 7.5 (SUEHRING, 2006) com controle de *bit rate* (MA, 2002) desenvolvido pelo grupo JVT e o perfil escolhido foi o perfil *Baseline* devido a sua boa relação qualidade/complexidade. Para permitir uma avaliação progressiva do desempenho do algoritmo aqui proposto, foram adotadas três configurações de teste apresentadas na **tabela 5.7**.

Tabela 5.7: Configurações de teste.

Configurações de teste	**Codificador 1**		**Codificador 2**	
	Camada base	Camada superior	Camada base	Camada superior
Teste 1	MPEG-4 ASP	MPEG-4 FGS	H.264/AVC	MPEG-4 FGS
Teste 2	H.264/AVC	MPEG-4 FGS	H.264/AVC	H.264/FGS
Teste 3	H.264/AVC	H.264/FGS	H.264/AVC não escalável	

Com a configuração de teste 1, pretende-se avaliar a melhoria de desempenho quando se utiliza o codificador não escalável H.264/AVC na camada base em substituição de um codificador MPEG-4 ASP. Com a configuração de teste 2, pretende-se avaliar o desempenho da solução escalável H.264/FGS em relação ao MPEG-4 FGS, usando sempre na camada base a norma H.264/AVC. Com a configuração de teste 3, pretende-se avaliar a quebra de desempenho do sistema H.264/FGS em relação ao codificador não escalável H.264/AVC (para o mesmo *bit rate*). Finalmente, também é efetuado um estudo de desempenho da técnica de melhoria seletiva dos coeficientes DC, aqui proposta, com o objetivo de avaliar o desempenho do H.264/FGS com e sem melhoria seletiva dos coeficientes DC.

Para a configuração de teste 1, o codificador 1 utilizado, ou seja o MPEG-4 FGS consiste no *software* de referência incluído na Parte 5 da norma MPEG-4 (ISO:14496-5, 2002). As condições de teste para as restantes configurações de teste e para o codificador / decodificador 2 da configuração de teste 1 são apresentadas na **tabela 5.8**.

Tabela 5.8: Condições de teste.

	Cenário1	Cenário2	Cenário3	Cenário4	Cenário5	Cenário6
Resolução espacial	QCIF	QCIF	QCIF	CIF	CIF	CIF
Freqüência de quadro (Hz)	5	10	10	10	30	30
Bit rate da camada base Rb (Kbit/s)	16	32	64	128	256	512

Bit rate máximo Rmax (Kbit/s)	64	128	256	512	1024	2048
Resultados PSNR a:	16,24,32 48,64	32,48,64, 96,128	64,96,128 192,256	128,192,256, 384,512	256,384,512 768,1024	512,768,1024, 1536,2048
Período dos quadros I	N=28	N=56	N=56	N=60	N=120	N=120
Número de quadros de referência	5	5	5	5	5	5
Amplitude dos vetores de movimento	16	16	16	32	32	32
Otimização RD	SIM	SIM	SIM	SIM	SIM	SIM
Controle de *bit rate*	TM5	TM5	TM5	TM5	TM5	TM5

O número de cenários, resoluções espacial e temporal, *bit rates* da camada base e *bit rates* máximos são iguais aos adotados nas condições de teste definidas pelo grupo MPEG-4 durante o desenvolvimento do FGS, permitem abranger uma gama variada de *bit rates*, freqüências temporais e resoluções espaciais. Os restantes parâmetros correspondem as novas formas de ferramentas de codificação da norma H.264/AVC, tendo-se adotado para a camada base:

- O número máximo de quadros de referência que pode ser utilizado é 5.
- A amplitude máxima dos vetores de movimento é 16 para as seqüências QCIF e 32 para as seqüências CIF.
- Como o perfil *Baseline* não permite a utilização de quadros do tipo B, ao contrário do perfil MPEG-4 ASP, o período dos quadros Intra (I) para o codificador da camada base H.264/AVC é o dobro do que foi definido para o codificador MPEG-4 ASP. Tenta-se desta forma que o desempenho do H.264/AVC não seja muito penalizado pela não utilização de imagens do tipo B.
- A otimização RD (*Rate / Distortion*) é utilizada para permitir ao codificador escolher um conjunto de parâmetros (ex: seleção de modo) que minimize o *bit rate* para um dado nível de qualidade.
- O controle de *bit rate* utilizado é baseado no modelo TM5, com algumas modificações de forma a suportar a otimização RD e a permitir um desempenho ligeiramente superior ao TM5 original (MA, 2002).

Para qualquer configuração de teste, utilizam-se as medidas de Bjontegaard (2001) para avaliar o desempenho dos codificadores em teste; obtém-se assim duas medidas: dPSNR e dRate que exprimem a diferença média de PSNR (em dB) para o intervalo de *bit rates* definido nas condições de teste e a diferença média do *bit rate* para um dado intervalo de PSNR (em %).

Finalmente, para este estudo do desempenho, e para limitar a quantidade de resultados, foram escolhidas cinco seqüências, o critério da escolha destas seqüências

foi a diferença entre o codificador escalável e o codificador não escalável estudados anteriormente. As seqüências *Boat* e *Rugby* representam os casos extremos, maior quebra de desempenho (entre 2 e 5 dB) e menor quebra de desempenho (entre -0,02 e 1,6 dB), respectivamente, as seqüências *Canoa*, *Stefan* e *Table Tennis* são representativas de vários tipos importantes de conteúdo e apresentam uma quebra de desempenho intermédia.

5.6.1 Resultados para a configuração de teste 1

A configuração de teste 1 utiliza sempre o MPEG-4 FGS na camada superior e usa o MPEG-4 ASP ou o H.264/AVC (perfil *Baseline*) na camada base. O principal objetivo deste estudo consiste em determinar qual é o ganho de eficiência quando se utiliza o H.264/AVC na camada base em vez do MPEG-4 ASP, tal como acontece atualmente no MPEG-4 FGS. Vale a pena lembrar que com este estudo, não se pretende comparar a norma H.264/AVC com o MPEG-4 ASP, e o ganho de eficiência do sistema FGS completo (camada base + camada superior) não é necessariamente igual a melhoria de desempenho do H.264/AVC em relação ao MPEG-4 ASP. Primeiro, porque não se utilizam perfis "equivalentes", o perfil ASP da norma MPEG-4 FGS permite a utilização de quadros do tipo B que possuem a maior eficiência de codificação enquanto o perfil *Baseline* do H.264/AVC não permite a utilização de quadros do tipo B. Segundo, porque o H.264/AVC alcança o desempenho superior através da utilização da otimização RD. Terceiro, porque a melhoria de desempenho da camada base não é refletida da mesa forma na camada superior.

Os resultados para esta configuração de teste são apresentados na **tabela 5.9**, apenas para a componente de luminância. Para qualquer cenário e seqüência de teste, o desempenho do codificador 2 (H.264/AVC na camada base) é superior. Tal como esperado, a camada base possui um impacto significativo no desempenho global do sistema FGS, com um ganho mais significativo para seqüências com uma correlação temporal entre quadros elevada, para a seqüência *Boat* e *Table Tennis*. Em relação ao *bit rate*, os ganhos de desempenho são mais significativos para *bit rates* médios e altos (para os cenários de 3 a 6) para as seqüências *Canoa, Rugby e Stefan* e para os *bit rates* baixos e médios para as seqüências *Boat* (cenários 1 a 3) e *Table* (cenários 1 a 4). Os ganhos são mais acentuados para as seqüências e *bit rates* onde a camada base apresenta uma melhoria de desempenho mais significativa, devido às novas ferramentas de compensação de movimento que exploram a correlação temporal de uma forma mais eficiente.

Tabela 5.9: Desempenho relativo do MPEG-4 FGS com H.264/AVC e MPEG-4 ASP na camada base.

	Boat		Canoa		Rugby		Stefan		Table Tennis	
	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate
cenário 1	4,767	99,98	***0,681***	***15,41***	***0,301***	***6,75***	***0,016***	***1,69***	2,445	47,43
cenário 2	4,017	99,95	***0,931***	***24,75***	***0,933***	***21,95***	***0,691***	***25,16***	2,660	62,54
cenário 3	6,465	99,83	3,086	54,63	3,027	52,24	3,011	61,07	3,914	77,75
cenário 4	2,025	71,56	***1,829***	***42,70***	***1,795***	***37,73***	2,446	56,92	3,083	69,37
cenário 5	3,850	99,99	***1,113***	***42,24***	***1,605***	***44,54***	2,494	71,40	2,341	77,81
cenário 6	2,258	75,05	2,212	45,64	2,523	47,60	2,311	51,64	2,047	61,21
Média	*3,897*	*91,06*	*2,649*	*50,14*	*2,775*	*49,92*	*2,566*	*60,26*	*2,748*	*66,02*

Para *bit rates* baixos (cenário 1), os ganhos de desempenho são mais reduzidos para a seqüência *Stefan* (0,016 dB para o PSNR e 1,69 % para o *bit rate*) e para a seqüência *Rugby* (0,301 dB para o PSNR e 6,75 % para o *bit rate*). O ganho é reduzido para estes cenários / seqüências porque o codificador MPEG-4 ASP da camada base não permite atingir o *bit rate* desejado (gasta mais bits), devido a uma menor eficiência de codificação e ao tipo de controle de *bit rate* utilizado (TM5), desenvolvido para o modelo de teste do MPEG-2 e adequado a *bit rates* superiores. Por exemplo, para a seqüência *Stefan* – cenário 1, o valor de *bit rate* que o codificador MPEG-4 ASP consegue alcançar é de 34,3 kbit/s (deveria ser de 16 kbit/s) enquanto o codificador H.264/AVC usa 15,99 kbit/s (menos de metade). Uma vez que a qualidade da camada base é bastante inferior para o codificador 2 (H.264/AVC na camada base), o desempenho do sistema FGS completo é influenciado negativamente. Na **tabela 5.9**, estes casos são assinalados em negrito e não são utilizados no cálculo da média para todos os cenários. Em conclusão, a utilização do codificador H.264/AVC na camada base em relação ao codificador MPEG-4 ASP, pois permite uma eficiência de codificação superior, para todos os cenários definidos, com um valor médio mínimo de 2,57 dB e máximo de 3.9 dB.

5.6.2 Resultados para a configuração de teste 2

Para a configuração de teste 2, comparam-se dois codificadores desenvolvidos no contexto desta dissertação, onde a camada base é sempre codificada com a norma H.264/AVC enquanto a camada superior usa o MPEG-4 FGS ou o H.264/FGS. Como se pode ver na **tabela 5.10**, a solução H.264/FGS apresenta um desempenho um pouco inferior em relação a combinação AVC+MPEG-4 FGS (componente de luminância), para todos os cenários e seqüências de teste. A quebra de desempenho do H.264/FGS é ligeiramente maior para as seqüências de teste com uma menor correlação entre quadros. Por exemplo, para a lenta seqüência *Boat*, o desempenho do H.264/FGS sofre uma quebra de desempenho de apenas 0,04 a 0,12 dB; no outro extremo, para a rápida seqüência de *Rugby*, a quebra de desempenho já é de 0,2 a 0,39 dB. Tal fato, deve-se a qualidade da camada base, que nas seqüências *Boat* e *Table Tennis* é mais elevada que as restantes; isto vai reduzir a energia do resíduo (menos planos de bit) a codificar na camada superior, o que é normalmente codificado de uma forma mais eficiente quando se usa a técnica de codificação entrópica ULVC.

Tabela 5.10: Desempenho do MPEG-4 FGS em relação ao H.264/FGS sempre com o H.264/AVC na camada base.

	Boat		Canoa		Rugby		Stefan		Table Tennis	
	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate
cenário 1	0,099	6,26	0,259	5,58	0,313	6,70	0,327	9,96	0,322	8,49
cenário 2	0,051	4,89	0,353	9,96	0,247	6,36	0,275	10,29	0,152	6,25
cenário 3	0,119	7,04	0,324	8,71	0,395	8,40	0,392	10,20	0,162	5,53
cenário 4	0,123	8,96	0,320	9,61	0,377	9,39	0,319	10,77	0,180	8,52
cenário 5	0,038	4,22	0,132	5,03	0,202	6,66	0,148	7,45	0,077	5,68
cenário 6	0,075	5,76	0,348	10,51	0,399	11,00	0,276	9,50	0,148	8,47
Média	**0,084**	**6,19**	**0,289**	**8,23**	**0,322**	**8,08**	**0,286**	**9,69**	**0,174**	**7,16**

Tal como se pode comprovar na **figura 5.9**, as quebras de desempenho do H.264/FGS são sistematicamente mais acentuadas para os *bit rates* mais elevados adotados para cada cenário. Nesta figura apresentam-se os gráficos RD (em azul o

H.264/FGS e em vermelho o MPEG-4 FGS) para a seqüência *Canoa* – cenário 1 (dPSNR = 0,26 dB) e para a seqüência *Rugby* – cenário 5 (dPSNR = 0,2 dB).

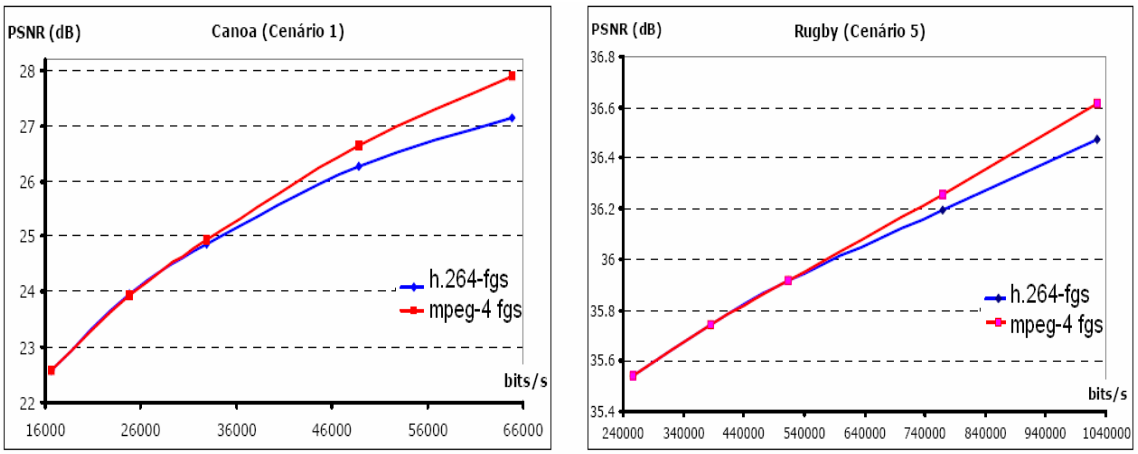


Figura 5.9: Alguns resultados da configuração de teste 2 para a luminância.

Tal como se pode ver, o desempenho é bastante semelhante para os *bit rates* mais baixos do intervalo [R_b, R_{max}] e, à medida que o *bit rate* aumenta, a quebra de desempenho do H.264/FGS é superior. Esta quebra de qualidade é devida aos códigos de *exp-Golomb* que apresentam tipicamente uma eficiência inferior em relação aos códigos de *Huffman* utilizados no MPEG-4 FGS (UGUR, 2003). Para os *bit rates* mais altos do intervalo de codificação, o número de símbolos a codificar aumenta e os códigos de *exp-Golomb* são responsáveis pelo desempenho inferior, uma vez que não modelam a distribuição estatística dos símbolos de uma forma adequada quando o número de símbolos a codificar é elevado. O desempenho dos códigos *exp-Golomb* pode ser melhorado uma vez que as palavras de código não foram desenhadas de acordo com as probabilidades de cada símbolo, mas apenas atribuídas de acordo com uma regra de construção fixa. Esta quebra de desempenho foi também observada pelo grupo VCEG (JEON, 2000) (KEROFSKY, 2000); de fato, estudos estatísticos efetuados concluíram que os códigos *exp-Golomb* utilizados podem ser melhorados de forma a aproximarem-se mais do limite máximo de eficiência estabelecido pela entropia, especialmente para passos de quantificação baixos (Qp < 13). Deste modo, no perfil *Baseline* da norma H.264/AVC os coeficientes da transformada são codificados com a ferramenta CAVLC, pois esta permite uma eficiência superior em relação ao UVLC (com códigos de *exp-Golomb*) à custa de uma maior complexidade (BJONTEGAARD, 2002). É importante lembrar que, na camada base, apenas os coeficientes da transformada são codificados com a ferramenta CAVLC; para os restantes elementos (ex: os vetores de movimento) é utilizada a ferramenta UVLC. Em conclusão, a utilização dos códigos de *exp-Golomb* permite diminuir a complexidade na camada superior do H.264/FGS (usando a ferramenta de codificação entrópica ULVC já disponível na camada base), à custa de uma pequena quebra de desempenho, com um valor médio mínimo de 0,08 dB e máximo de 0,32 dB; esta quebra é mais acentuada para os *bit rates* mais altos do intervalo de codificação.

5.6.3 Resultados para a configuração de teste 3

Na configuração de teste 3, compara-se o sistema H.264/FGS proposto com o codificador não escalável H.264/AVC. Com esta configuração de teste, pretende-se conhecer a quebra de eficiência do sistema H.264/FGS em relação a codificação não

escalável H.264/AVC de uma forma semelhante à avaliação de desempenho do MPEG-4 FGS em relação ao MPEG-4 ASP. Dito de outra forma pretende-se conhecer o "preço" da escalabilidade em termos de eficiência. Os resultados para a componente de luminância são apresentados na **tabela 5.11**.

Tabela 5.11: Desempenho do codificador não escalável H.264/AVC em relação ao H.264/FGS.

	Boat		Canoa		Rugby		Stefan		Table Tennis	
	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate	dPSNR	dRate
cenário 1	4,084	49,64	0,465	9,55	0,655	10,34	1,675	29,79	2,663	33,20
cenário 2	4,178	50,14	1,437	28,71	1,045	20,10	2,387	41,20	3,099	43,58
cenário 3	4,099	49,63	1,881	29,99	1,354	22,21	2,525	38,35	3,189	41,68
cenário 4	3,734	49,71	1,514	30,73	1,284	24,27	2,388	39,75	1,284	24,27
cenário 5	3,395	50,69	1,525	35,22	1,596	31,97	2,703	45,51	2,870	46,46
cenário 6	2,756	50,13	1,892	32,95	1,913	32,04	2,190	39,94	2,914	46,78
Média	**3,708**	**49,99**	**1,452**	**27,86**	**1,308**	**23,49**	**2,311**	**39,09**	**2,670**	**39,33**

Este estudo permite estabelecer qual é o limite teórico do desempenho para o sistema H.264/FGS. Para todos os cenários, a codificação não escalável possui um desempenho médio superior; além disso, as seqüências com uma elevada correlação entre quadros (ex: *Boat*) apresentam maior quebra de desempenho e as seqüências com um grau de movimento significativo (ex: *Rugby* e *Canoa*) apresentam a menor quebra de desempenho. Estes resultados são esperados uma vez que o H.264/FGS, tal como o MPEG-4 FGS, não possui um módulo de compensação de movimento na camada superior o que limita a sua capacidade de explorar redundância temporal. Comparativamente ao estudo MPEG-4 FGS versus MPEG-4 ASP, a quebra de desempenho é superior para um número significativo de cenários / seqüências, uma vez que o H.264/AVC possui novas ferramentas de compensação de movimento, que explorar a redundância temporal de uma forma mais eficiente que no perfil MPEG-4 ASP. Em conclusão, o sistema H.264/FGS proposto mantém um desempenho inferior em relação a codificação não escalável H.264/AVC, tal como o MPEG-4 FGS em relação ao MPEG-4 ASP, com um valor médio mínimo de 1,31 dB e máximo de 3,71 dB, para as várias seqüências nas condições de teste definidas.

5.6.4 Avaliação da melhoria seletiva dos coeficientes DC

Nesta seção, compara-se o sistema H.264/FGS proposto, com e sem melhoria seletiva dos coeficientes DC. No H.264/FGS, este método permite uma melhoria da componente DC e consequentemente uma reconstrução do vídeo do decodificador mais suave e uma menor ocorrência de artefatos do tipo *flickering*, freqüentes em um sistema de escalabilidade híbrida, pois a qualidade da imagem da camada base pode variar significativamente (especialmente para *bit rates* baixos). Apesar de não ter sido efetuado um estudo do aumento da qualidade subjetiva associada a esta técnica, é de se esperar uma melhoria na qualidade visual, tal como foi demonstrado para a técnica de seleção de freqüências do MPEG-4 FGS (semelhante a melhoria seletiva dos coeficientes DC) em (JIANG, 2003) e (LI, 1999-3).

No entanto, já no MPEG-4 FGS quando se utilizava a medida de desempenho PSNR, para avaliar a qualidade do vídeo decodificado, os resultados indicavam um desempenho em termos de qualidade objetiva inferior quando se usa a técnica de

seleção de freqüências, uma vez que os coeficientes AC possuem uma menor precisão em comparação com os coeficientes DC para um dado *bit rate*. Como se sabe, a medida PSNR não reflete necessariamente a qualidade visual (ex: quando ocorrem alguns tipos de artefatos), normalmente porque as componentes de freqüência visualmente mais importantes contribuem tanto para o PSNR como as componentes de freqüências visualmente menos importantes o que não é subjetivamente verdade. Para a seleção de freqüências adotada pela norma MPEG-4 FGS, a quebra de desempenho, em relação ao MPEG-4 FGS sem seleção de freqüências pode atingir 2 dB para a componente de luminância (LI, 1999-3).

Na **figura 5.10** apresentam-se alguns gráficos de PSNR que permitem avaliar a qualidade objetiva (PSNR) do vídeo quando se utiliza a técnica de melhoria seletiva dos coeficientes DC no H.264/FGS; contemplam-se três configurações:

- A curva RD azul representa o H.264/FGS sem a melhoria seletiva dos coeficientes DC ativada.
- A curva RD vermelho corresponde ao caso onde *fgs_vop_dc_enhancement*=1 o que significa que os coeficientes DCLum são deslocados de 1 plano de bit; a ordem de transmissão é: DCLum MSB, DCLum MSB-1 em seguida os restantes coeficientes.
- A curva RD verde correspondente ao caso onde *fgs_vop_dc_enhancement*=2 (ver **figura 5.4**) o que indica que os coeficientes DCLum sofrem um deslocamento de 2 planos de bit; a ordem de transmissão é: DCLum MSB, DCLum MSB-1, DCLum MSB-2 em seguida os restantes coeficientes.

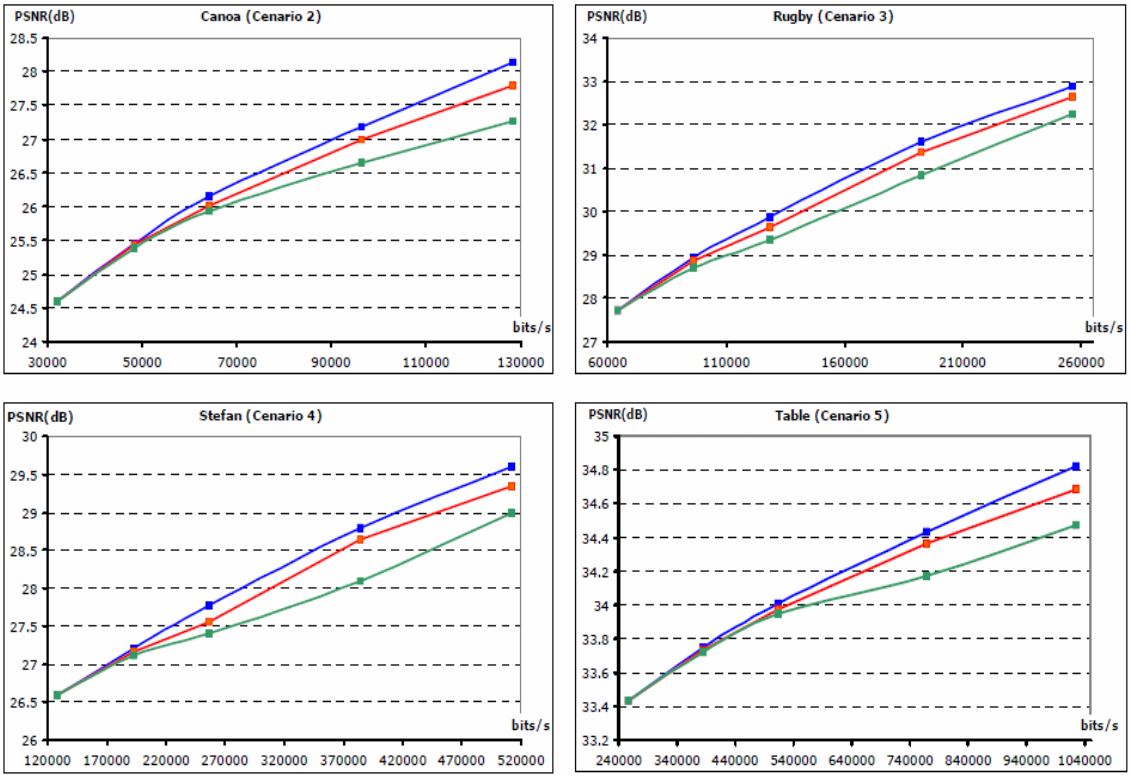


Figura 5.10: Desempenho da melhoria seletiva dos coeficientes DC: em azul o H.264/FGS sem melhoria seletiva, em vermelho com *fgs_vop_dc_enhancement* =1 e em verde com *fgs_vop_enhancement* = 2.

Tal como era esperado, uma das conseqüências na atribuição de uma maior importância aos coeficientes DC é a quebra de PSNR, especialmente para os *bit rates* mais altos do intervalo de codificação. Os valores do PSNR sofrem uma quebra entre 0,02 a 0,35 dB para *fgs_vop_dc_enhancement* = 1 e entre 0,34 a 0,87 dB para *fgs_vop_dc_enhancement* = 2. Este fato pode ser explicado através de um exemplo: considerem-se dois coeficientes C_{dc} e C_1 sem melhoria seletiva, os bits transmitidos são os seguintes (sem codificação entrópica): 10,11,10,00,01. Se o *bitstream* for truncado e apenas dois planos de bit decodificados, o valor reconstruído de $C_{dc} = (11000)_{binário} = 24$ e o valor reconstruído de $C_1 = (01000)_{binário} = 8$; o erro quadrático, utilizado no cálculo do PSNR é $(28\text{-}24)^2 + (9\text{-}8)^2 = 17$. Por outro lado, se for utilizado a melhoria seletiva do coeficiente DC, com *fgs_vop_dc_enhancement* = 2, os bits transmitidos são os seguintes: 1,1,10,01,00,00,00. Se apenas forem decodificados três planos de bit (de forma a manter o número de bits decodificados de 4) o valor reconstituído de $C_{dc} = (11100)_{binário} = 28$ e o valor reconstruído de $C_1 = (00000)_{binário} = 0$. É claro que o valor do coeficiente DC possui uma maior exatidão que C_1; no entanto, o valor do erro quadrático $(28\text{-}28)^2 + (9\text{-}0)^2 = 81$ é mais elevado, causando um valor de PSNR inferior. Desta forma, a medida que o *bit rate* aumenta as freqüências mais elevadas possuem valores menos exatos quando comparadas com o H.264/FGS sem melhoria seletiva dos coeficientes DC, levando a valores de PSNR inferiores. Tal como se tinha mencionado para o MPEG-4 FGS, uma avaliação subjetiva pode indica o contrário, ou seja, que a qualidade visual da seqüência decodificada superior.

É importante salientar as propriedades da técnica aqui apresentada em relação a melhoria seletiva do MPEG-4 FGS. Com a melhoria seletiva dos coeficientes DC incorporada ao H.264/FGS, não é necessário qualquer processamento adicional (apenas uma reorganização do *bitstream*), uma vez que os coeficientes DC são codificados separadamente dos restantes. Ao contrário, no MPEG-4 FGS é necessário elevar os coeficientes de baixa freqüência para um plano de bit superior, efetuar a varredura em *zig-zag* para todos os coeficientes que pertençam a um dado plano de bit e, finalmente, codificar entropicamente os símbolos (RUN, EOP) resultantes. A norma MPEG-4 FGS define mais oito tabelas para efetuar a codificação entrópica, pois a estatística dos símbolos é alterada quando se elevam os coeficientes de baixa freqüência para um determinado plano de bit. Como desvantagem, o método aqui proposto apenas suporta a melhoria seletiva dos coeficientes DC ao contrário do MPEG-4 FGS que permite a melhoria seletiva de qualquer número de coeficientes de baixa freqüência. Um estudo subjetivo mais detalhado seria necessário para determinar se é suficiente fazer apenas a melhoria seletiva dos coeficientes DC ou se mais coeficientes de baixa freqüência devem ser considerados.

6 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi apresentada uma solução para a codificação escalável de vídeo, incorporando algumas das ferramentas especificadas na norma H.264/AVC. O sistema escalável H.264/FGS proposto mantém as mesmas características que o sistema escalável MPEG-4 FGS, ou seja, adaptação fina ao *bit rate* disponível, robustez a erros, etc. Na camada superior da solução H.264/FGS são utilizadas as transformadas DCT inteira e de Hadamard em blocos de dimensão 4x4 e 2x2. Estas transformadas possuem baixa complexidade e eliminam o erro entre diferentes implementações da transformada (*mismatch error*). O codificador entrópico da camada superior utiliza o esquema UVLC com códigos *exp-Golomb* de comprimento variável e construção regular. Este esquema possui uma baixa complexidade e permite a utilização de uma única tabela VLC, responsável pela atribuição de cada elemento de sintaxe a palavra de código correspondente. Para obter esta tabela, realizou-se um estudo estatístico da distribuição dos símbolos (*run*, *EOP*) e do *fgs_cbp* para os vários planos de bit. Com base neste estudo, desenhou-se uma única tabela VLC com uma dimensão inferior ao conjunto de tabelas VLC da norma MPEG-4 FGS. Para projetar um sistema equivalente ao MPEG-4 FGS, também a solução H.264/FGS proposta neste trabalho não usa compensação de movimento na camada superior o que tem limitações evidentes em termos de eficiência de compressão.

Para avaliar o desempenho do sistema proposto, várias configurações de teste foram estudadas visando responder as seguintes questões:

- Qual é o ganho de desempenho resultante da introdução da codificação não escalável H.264/AVC na camada base substituindo a solução MPEG-4 ASP?
- Qual é o desempenho relativo do H.264/FGS em relação ao MPEG-4 FGS quando se usa o H.264/AVC na camada base?
- Qual é a quebra de desempenho do H.264/FGS em relação a codificação não escalável H.264/AVC para *bit rates* semelhantes?

Em relação a primeira questão, a introdução do H.264/AVC na camada base permitiu uma melhoria significativa da qualidade PSNR em todos os cenários e *bit rates*, com um valor médio mínimo de 2,57 dB e máximo de 3,9 dB. Em relação a segunda questão, a introdução do H.264/FGS possui um desempenho ligeiramente inferior em relação ao MPEG-4 FGS, devido ao tipo de codificação entrópica adotada (quebras de desempenho médias entre 0,08 e 0,32 dB). Deve-se, contudo lembrar que a complexidade do H.264/FGS é mais baixa em relação ao MPEG-4 FGS, tanto em termos da transformada utilizada como no esquema de codificação entrópica. Esta redução de complexidade irá facilitar a introdução de ferramentas de compensação de movimento na camada superior sem sacrificar demasiadamente a complexidade global

do sistema. A resposta da terceira questão mostrou qual a diferença de desempenho entre a solução escalável H.264/FGS e a solução não escalável H.264/AVC, com uma quebra de desempenho média entre 1,3 e 3,7 dB respectivamente.

Apesar dos esforços realizados no âmbito deste trabalho para melhorar a eficiência de codificação do MPEG-4 FGS, constituírem um avanço significativo, não esgotam o trabalho efetuado nesta área. Algumas possibilidades em termos da continuação do trabalho apresentado aqui são apresentadas:

Melhoria da eficiência de codificação entrópica na camada superior: O sistema de codificação entrópica (ULVC) desenvolvido no âmbito do codificador H.264/FGS apresenta uma quebra de desempenho em relação aos códigos de *Huffman* utilizados no MPEG-4 FGS. O uso de um esquema de codificação entrópica mais eficiente na camada superior deverá permitir uma melhoria do desempelho significativa para um conjunto amplo de condições de teste. Assim, é desejável a integração dos esquemas de codificação entrópica CAVLC e CABAC já definidis na norma H.264/AVC. Estes esquemas, apesar de sua complexidade, apresentam uma eficiência superior, essencialmente porque exploram informação sobre o contexto; a codificação de um elemento de sintaxe depende dos elementos de sintaxe anteriormente codificados. A técnica CABAC, apenas definida no perfil *Main* do H.264/AVC, permite alcançar um melho desempenho devido ao modelo de condificação aritmética utilizado. Para integrar qualquer um destes modelos de codificação entrópica na camada superior, será necessário conhecer as estatísticas condicionadas de cada elemento de sintaxe de acordo com o contexto e ter em consideração que os elementos a se codificar são bits (com valores 0 e 1) que pertencem a um dado plano de bit.

Codificação escalável universal: O objetivo principal de qualquer sistema de codificação escalável de vídeo é obter uma única representação codificada do conteúdo que possa servir o maior número de terminais e redes. Neste contexto, deve poder ser extraído deste fluxo às suas características; sendo necessário suportar um número elevado de níveis de resolução temporal espacial e de qualidade. A codificação escalável universal obriga ao suporte flexível de combinações de cada um destes tipos de escalabilidade. Um sistema deste tipo permitiria também ir ao encontro das preferências de cada usuário; alguns usuários preferem resolução temporal elevada, enquanto outros preferem uma melhoria da resolução espacial e uma melhor qualidade.

Escalabilidade de complexidade: Devido ao número cada vez maior de terminais, com diferentes capacidades de processamento e memória, é desejável que o sistema de codificação escalável de vídeo permita implementações de baixa complexidade tanto do codificador como do decodificador. Por exemplo, para certos tipos de terminais, tais como terminais móveis, é necessário que o sistema de codificação e decodificação de vídeo não seja exigente demais em termos de processamento ou memória, tendo em conta os recursos escassos que este tipo de terminal possui.

Escalabilidade de conteúdo: A escalabilidade de conteúdo é uma funcionalidade desejada por cada vez mais aplicações, pois permite ao usuário visualizar um ou mais objetos de vídeo com uma resolução temporal, espacial ou de qualidade diferentes, de acordo com a importância que o usuário lhe atribui. Por outro lado, os terminais com uma capacidade de processamento baixa ou acesso a um canal de comunicação com baixo *bit rate*, podem facilmente reduzir a qualidade de alguns objetos de vídeo menos importantes sem influenciar a qualidade de outrs objetos com maior relevância no contexto da aplicação em questão.

Elevada eficiência de codificação e elevada robustez a erros de transmissão: Estes dois tipos de requisitos são normalmente conflituosos, pois a medida que se alcança uma melhor eficiência de codificação, maior é o impacto dos erros de transmissão na qualidade do vídeo decodificado. No entanto, a escalabilidade de alta granularidade, devido a sua estrutura em camadas, permite uma degradação suave da qualidade da imagem em diferentes condições de transmissão, para diferentes tipos de erros e texas de erros. Sendo assim, é importante o desenvolvimento de sistemas de codificação escalável quem mantenha um elevado desempenho, mantendo uma elevada robustez a erros de transmissão.

Em conclusão, a escalabilidade de vídeo é um tópico de estudo bastante atual com muitos aspectos problemáticos que ainda necessitam ser investigados e resolvidos. Atualmente, existe uma iniciativa do grupo MPEG para desenvolver uma norma de codificação escalável com desempenho superior ao da norma MPEG-4 FGS e acrescentar também algumas novas funcionalidades exigidas pelo mercado, de modo a garantir em um futuro próximo a adoção generalizada da escalabilidade nos sistemas de codificação de vídeo mais populares.

REFERÊNCIAS

AL-SHAYKH, O. K.; MILOSLAVSKY, E.; NOMURA, T.; NEFF, R.; ZAKHOR, A. Video Compression Using Matching Pursuits. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.9, n.1, p.123–143, Feb. 1999.

AU, J.; LIN, B.; JOCH, A.; KOSSENTINI, F. Complexity Reduction and Analysis for Deblocking Filter. **JVT-C094, Reunião JVT (ISO/IEC MPEG ITU-T VCEG)**, Fairfax, May 2002. Disponível em: < http://ftp3.itu.ch/av-arch/jvt-site/2002_05_Fairfax/JVT-C019.doc >, Acesso em: Maio, 2006.

BENETIERE, M.; DUFOUR, C. Matching Pursuits Residual Coding for Video Fine Granular Scalability. **ISO/IEC JTC1/SC29/WG11 M4008 - Reunião MPEG**, Atlantic City, Oct. 1998, [S.l.:s.n].

BJONTEGAARD, G. Calculation of Average PSNR Differences Between RD-Curves. **ITU-T/SG16/VCEG-M33 - Reunião VCEG**, Austin, Apr. 2001. Disponível em: < http://ftp3.itu.ch/av-arch/video-site/0104_Aus/VCEG-M33.doc>, Acesso em: Maio, 2006.

BJONTEGAARD, G.; LILLEVOLD, K. Context-adaptative VLC (CVLC) Coding of Coefficients. **JVT-C028, Reunião JVT (ISO/IEC MPEG and ITU-T VCEG)**, Fairfax, May 2002. Disponível em: < http://ftp3.itu.ch/av-arch/jvt-site/2002_05_Fairfax/JVT-C028.doc >, Acesso em: Maio, 2006.

BRADY, N. MPEG-4 Standardized Methods for the Compression Arbitrarily Shaped Video Objects. **IEEE Transactions on Circuits and Systems for Video Technology - Special Issue on Object-Based Video Coding and Description**, [S.l.], v.9, n.8, p.1170–1190, Dec. 1999.

BRADY, N.; BOSSEN, F.; MURPHY, N. Context-based Arithmetic Encoding of 2D Shape Sequences. In: International Conference on Image Processing, **Proceedings**, [Sl.]: IEEE, 1997. v1, p. 29-32.

BURT, P. J.; ADELSON, E. H. The Laplacian Pyramid as a Compact Image Code. **IEEE Transactions on Communications**, [S.l.], v.COM-31, n.4, p.532–540, Apr. 1983.

CHADDHA, N.; WALL, G.; SHMIDT, B. An End to And Software Only Scalable Video Delivery System. In: 5th International Workshop on Network and Operating System Support for Digital Audio and Video, **Proceedings**, 1995. v1, p. 130-141 [Sl; sn].

CHANG, E.; ZAKHOR, A. Variable Bit Rate MPEG Video Storage on Parallel Disk Arrays. In: International Workshop on Community Networking Integrated Multimedia Services to the Home, **Proceedings**, 1994, [Sl: sn].

CHEN, Y.; DUFOUR, C.; RADHA, H.; COHEN, R. A.; BUTEAU, M. Request for fine Granular Video Scalability for Media Streaming Applications. **ISO/IEC JTC1/SC29/WG11 M3792 - Reunião MPEG**, Dublin, June 1998, [S.l.:s.n].

CHEUNG, S. C. S.; ZAKHOR, A. Matching Pursuits Coding for Fine Granular Scalability. **ISO/IEC JTC1/SC29/WG11 M3991 - Reunião MPEG**, Atlantic City, Oct. 1998, [S.l.:s.n].

CHIARIGLIONE, L. Terms of Reference for a Joint Project between ITU-T Q.6/SG16 and ISO/IEC JTC1/SC29/WG11 for the Development of new Video Coding Recommendation and International Standard. **ISO/IEC JTC1/SC29/WG11 N4400**, Pattaya, Dec. 2001. Disponível em: < http://www.itscj.ipsj.or.jp/sc29/29w12911jvt.pdf >. Acesso em: Maio, 2006.

CHIARIGLIONE, L. Resolutions of the 63th Meeting. **ISO/IEC JTC1/SC29/WG11 N5316**, Awaji, Dec. 2002. Disponível em: <http://www.itscj.ipsj.or.jp/sc29/open/29view/29n51851.doc>. Acesso em: Maio, 2006.

CLARK, R. **Transform Coding of Images**. London, UK: Academic Press, 1990.

CONKLIN, G. J.; HEMAMI, S. S. A Comparison of Temporal Scalability Techniques. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.9, n.6, p.909–919, Sept. 2000.

LSI Logic Corporation. Disponível em: <http://www.lsilogic.com.> Acesso em: nov 2005.

CROCHIERE, R.; WEBER, S.; FLANAGAN, J. Digital Coding of Speech in Subbands. **Bell System Technology Journal**, [S.l.], n.55, p.1069–1085, Oct. 1976.

DOMANSKI, M.; LUCZAK, A.; MACKOWIAK, S. Spatio-Temporal Scalability for MPEG Video Coding. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.10, n.7, p.1088–1093, Oct. 2000.

EBRAHIMI, T. MPEG-4 Video Verification Model: a video encoding/decoding algorithm based on content representation. **Signal Processing: Image Communication, Special Issue on MPEG-4**, [S.l.], v.9, n.4, p.367–384, May 1997.

EBRAHIMI, T.; HORNE, C. MPEG-4 Natural Video Coding - An Overview. **Signal Processing: Image Communication, Tutorial Issue on the MPEG-4 Standard**, [S.l.],

v.15, n.4-5, p.365–385, Jan. 2000.

ERIKSSON, H. MBONE: the multicast backbone. **Communications of the ACM**, [S.l.], v.37, n.8, p.54–61, Aug. 1994.

GALLANT, M.; KOSSENTINI, F. Rate-Distortion Optimized Layered Coding with Unequal Error Protection for Robust Internet Video. **IEEE Transactions on Circuits and Systems for Video Technology - Special Issue on Streaming Video**, [S.l.], v.11, n.3, p.357–372, Mar. 2001.

GIROD, B.; HORN, U.; BELZER, B. Scalable Video Coding with Multiscale Motion Compensation and Unequal Error Protection. In: Symposium on Multimedia Communications and Video Coding, 1995. **Proceedings**, USA. IEEE, 1995, [s.n].

HALBACH, T.; WIEN, M. Concepts and Performance of Next-Geration Video Compression Standardization. In: 5th Nordic Signal Processing Symposium, 2002. **Proceedings**, Trondheim, 2002.

HORN, U.; GIROD, B. Scalable Video Transmission for the Internet. **Computer Networks and ISDN Systems**, [S.l.], v.29, n.15, p.1833–1842, Nov. 1997.

HORN, U.; GIROD, B.; BELZER, B. Scalable Video Coding in Multimedia Applications and Robust Transmission overWireless Channels. **7th International Workshop on Packet Video**, Brisbanne, Australia, Mar. 1996.

ILLNGNER, K.; MULLER, F. Spatially Scalable Video Compression Employing Resolution Pyramids. **IEEE Journal on Selected Areas in Communications, Special Issue on Very Low Bit Rate Coding**, [S.l.], v.15, n.9, p.1688–1704, Dec. 1997.

INC, V. H,264/AVC Real-Time SD Encoder Demo. **JVT-D023 JVT (ISO/IEC MPEG ITU-T VCEG)**, Klagenfurt, July 2002. Disponível em: <http://ftp3.itu.ch/av-arch/jvt-site/2002_07_Klagenfurt/JVT-D023.pdf>. Acesso em: dec 2005.

ISO/IEC 14496-1:2004 Information Technology - Coding of audio-visual objects - Part 1: Systems. Disponível em: < http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38559 &ICS1=35&ICS2=40&ICS3=> Acesso em: dec 2005.

ISO/IEC 14496-2:2004 Information Technology - Coding of audio-visual objects - Part 2: Visual. Disponível em: < http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=39259 &ICS1=35&ICS2=40&ICS3=> Acesso em: dec 2005.

ISO/IEC 14496-3:2005 Information Technology - Coding of audio-visual objects - Part 3: Audio. Disponível em: < http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=42739> Acesso em: fev 2006.

ISO/IEC 14496-4:2004 Information Technology - Coding Audiovisual Objects - Part 4: conformance testing. 2000. Disponível em: <http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=36084 > Acesso em: dec 2005.

ISO/IEC 14496-5:2001 Information Technology - Coding of audio-visual objects - Part 5: Reference software. Disponível em: < http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=36086> Acesso em: fev 2006.

ISO/IEC 14496-6:2000 Information Technology - Coding of audio-visual objects - Part 6: Delivery Multimedia Integration Framework (DMIF). Disponível em: < http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=34418 &ICS1=35&ICS2=40&ICS3=> Acesso em: fev 2006.

ISO/IEC 14496-9:2004 Information Technology - Coding of audio-visual objects - Part 9: Reference hardware description. Disponível em: < http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38147 &ICS1=35&ICS2=40&ICS3=> Acesso em: dec 2005.

ITU Telecommunications Standardization Sector. ITU-T Recommendation H.263. 2001. Disponível em: <http://www.itu.int/itudoc/itu-t/com16/implgd/old/h263ig01.pdf> Acesso em: nov 2005.

JAIN, A. K. **Fundamentals of Digital Image Processing**. Englewood Cliffs: Prentice Hall, 1989.

JEON, B. Entropy Coding Efficiency of H.26L. **ITU-T/SG16/Q15-J57 Reunião VCEG**, Osaka, May 2000. [S.l.:s.n].

JIANG, H. Experiment on Post-Chip FGS Enhancement. **ISO/IEC JTC1/SC29/WG11 M5669 - Reunião MPEG**, Noordwijkerhout, Mar. 2000. [S.l.:s.n].

JIANG, H.; THAYER, G. M. Using Frequency Weighting in FGS Bitplane Coding for Natural Video. **ISO/IEC JTC1/SC29/WG11 M5489 - Reunião MPEG**, Maui, Dec. 1999. [S.l.:s.n].

JIANG, H.; THAYER, G. M. Using FrequencyWeighting in Fine-Granularity-Scalability Bit-plane Coding for Natural Video. **ISO/IEC JTC1/SC29/WG11 M5489 – Reunião MPEG**, Maui, Dec. 2003. [S.l.:s.n].

JOCH, A.; IN, J.; KOSSENTINI, F. Demonstration of FCD-conformant Baseline Real-Time Codec. **JVT-E136 - Reunião JVT(ISO/IEC MPEGITU-T VCEG)**, Genebra, Oct. 2002. Disponível em: < http://ftp3.itu.ch/av-arch/jvt-site/2002_10_Geneva/JVT-E136.doc>. Acesso em: jan 2006.

KARLSSON, G.; VETTERLI, M. Packet Video and its Integration into the network Architecture. **IEEE Journal on Selected Areas in Communications**, [S.l.], v.10, n.8, p.739–751, June 1989.

KATATA, H.; ITO, N.; AONO, T.; KUSAO, H. Object Wavelet Transform for Coding of Arbitrarily-Shaped Image Segments. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.7, n.1, p.234–237, Feb. 1997.

KAUFF, P.; SCHUUR, K. An Extension of Shape-Adaptative DCT Towards DC Separation and Delta DC Correction. In Picture Coding Symposium, 1997. **Proceedings**, Berlim, 1997.

KEROSFSKY, L. Entropy Coding of Transform Coefficients. **ITU-T/SG16/Q15-K45 Reunião VCEG**, Portland, Aug. 2000. [S.l.:s.n].

KIM, B. J.; XIONG, Z.; PEARLMAN, W. A. Low Bit-Rate Scalable Video Coding with 3D Set Partitioning in Hierarchical Trees (3D SPIHT). **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.10, n.8, p.1374–1386, Dec. 2000.

KOENEN, R. Profiles and Levels in MPEG-4: approach and overview. **Signal Processing: Image Communication, Tutorial Issue on the MPEG-4 Standard**, [S.l.], v.15, n.4-5, p.463–478, Jan. 2000.

KURCEREN, R.; KARCZEWICZ, M. Synchronization-Predictive Coding for Video Compression: the sp frames design for jvt/h.26l. In: International Conference on Image Processing, 2002. **Proceedings**, New York: IEEE, 2002. v.2, p. 497-500.

LI, S.; LI, W. Shape-Adaptative DiscreteWavelet Transforms for Arbitrary Shaped Visual Object Coding. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.10, n.5, p.725–743, Aug. 2000.

LI, S.; SODAGAR, I. Generic, Scalable and Efficient Shape Coding for Visual Texture Objects in MPEG-4. **IEEE International Symposium on Circuits and Systems**, Switzerland, May 2000.

LI, W. Overview of Fine Granularity in MPEG-4 Video Standard. **IEEE Transactions on Circuits and Systems for Video Technology - Special Issue on Streaming Video**, [S.l.], v.11, n.3, p.301–317, Mar. 2001.

LIANG, J. High Scalabel Image Coding for Multimedia Applications. **Proceedings of the ACM Multimedia Communication Conference**, Washington, USA, 1997.

LING, F.; LI, W.; SUN, H. Bitplane Coding of DCT Coefficients for Image and Video Compression. **SPIE Visual Communications and Image Processing**, California, USA, Jan. 1999.

LU, Y.; GAO,W.; WU, F. Fast and Robust Sprite Generation for MPEG-4 Video Coding. In: PACIFIC-RIM CONFERENCE ON MULTIMEDIA, 2001. Pequim, China. **Proceedings,** [Sl]: IEEE, 2001.

LUTHRA, A.; GHANDI, R.; PANUSOPONE, K.; MCKOEN, K.; BAYLON, D. Performance of MPEG-4 Profiles Used for Streaming Video. **Proceedings of Workshop and Exihbition on MPEG-4**, California, USA, June 2001.

MA, S.; GAO, W.; LU, Y.; LU, H. Proposed Draft Description of Rate Control on JVT Standard. **ISO/IEC JTC1/SC29/WG11 N4920 - Reunião MPEG**, Awaji, Dec. 2002.

MALVAR, H.; et al. Low-Complexity Transform and Quantization with 16-bit Arithmetic for H.26L. In: International Conference on Image Processing, 2002. New York, USA. **Proceedings,** IEEE, 2002.

MARPE, D.; et al. Context-base Adaptative Binary Arithmetic Coding in JVT/H.26L. In: International Conference on Image Processing, 2001. Salonica, Greece. **Proceedings**, [Sl], IEEE, 2001.

MARTUCCI, S. A.; et al. A ZerotreeWavelet Video Coder. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.7, n.1, p.109–118, Feb. 1997.

MOCCAGATTA, I.; RATAKONDA, K. A Performance Comparison of CABAC and VCL-based Entropy Coders for SD and Hd Sequences. **JVT-E079 JVT (ISO/IEC MPEGITU-T VCEG)**, Genebra, Oct. 2002. Disponível em: < http://ftp3.itu.ch/av-arch/jvt-site/2002_10_Geneva/JVT-E079.doc>. Acesso em: mar 2006.

MPEG-4 description Disponível em: < http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm.> Acesso em: nov 2005.

NEFF, R.; ZAKHOR, A. Very Low Bit-Rate Video Coding Based on Matching Pursuits. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.7, n.1, p.158–171, Feb. 1997.

PEREIRA, F. MPEG-4: why, what, how and when ? **Signal Processing: Image Communication, Tutorial Issue on the MPEG-4 Standard**, [S.l.], v.15, n.4-5, p.271–279, Jan. 2000.

PERKIS, A. On the Importance of Error Resilience in Visual Communications over Noisy Channels. **Birkhauser Boston Transactions on Circuits, Systems and Signal Processing, Special issue on Multimedia Communications**, [S.l.], v.20, n.3, p.415–445, 2001.

PROJECT, D. V. B. **DVB - Digital Video Broadcasting**. Disponível em: <http://www.dvb.org>. Acesso em: jan 2006.

RADHA, H.; et al. Scalable Internet Video Using MPEG-4. **Signal Processing: Image Communication, Special Issue on Realtime Video over the Internet**, [S.l.], v.15, n.1-2, p.95–126, Sept. 1999.

RADHA, H. M.; SCHAAR, M.; CHEN, Y. The MPEG-4 Fine-Grained Scalable Video Coding Methodo for Multimedia Streaming Over IP. **IEEE Transactions on Multimedia**, [S.l.], v.3, n.1, p.53–68, Mar. 2001.

RAO, K. R.; YIP, P. **Discrete Cosine Transform**: algorithms, advantages, applications. Boston, USA: Academic Press, 1990.

RHEE, I. Error Control Techniques for Interative Low bit Rate Video Transmission over the Internet. **Computer Communication Review**, New York, v28, n4, Oct 1998. Trabalho apresentado na ACM SIGCOMM Conference, 1998, Vancouver.

SAENZ, M.; et al. And evaluation of Color Embedded Wavelet Image Compression Techniques. In: SPIE Conference on Visual Communications and Image Processing, 1999. California, USA, **Proceedings**, 1999.

SAID, A.; PEARLMAN, W. A New, Fast and Efficient Image Codec Based on Set Partitionig in Hierachical Trees. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.6, n.3, p.243–250, June 1996.

SAPONARA, S.; BLANCH, C.; DENOLF, K.; BORMANS, J. Data Transfer and Storage Complexity Analysis of the H.264/AVC Codec on a Tool-by-Tool Basis. **ISO/IEC JTC1/SC29/WG11 M8547 - Reunião MPEG**, Klagenfurt, July 2002. [Sl; sn]

SCHAAR, M.; CHEN, Y.; RADHA, H. Adaptative Quantization Modes for Fine-Granular Scalability. **ISO/IEC JTC1/SC29/WG11 M5589 - Reunião MPEG**, Vancouver, Canada, July 1999.

SCHAAR, M.; CHEN, Y.; RADHA, H. Embedded DCT and Wavelet Methods for Fine Granular Scalable Video Coding: analysis and comparison. **SPIE Image and Video Communications and Processing**, California, USA, Mar. 2000.

SCHAAR, M.; LIN, Y. T. Content-Based Selective Enhancement for Streaming Video. **IEEE International Conference on Image Processing**, Salonica, Greece, Oct. 2001.

SCHAAR,M.; RADHA, H. A Hybrid Temporal-SNR Fine-Granular Scalability for Internet Video. **IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Streaming Video**, [S.l.], v.11, n.3, p.318–331, Mar. 2001.

SCHAFER, R.; WIEGAND, T.; SCHWARZ, H. **The Emerging H.264/AVC Standard**. [S.l.]: EBU Technical Review, 2003. (293).

SCHUSTER, B. Fine Granular Scalability with Wavelets Coding. **ISO/IEC JTC1/SC29/WG11 M4021 - Reunião MPEG**, Atlantic City, Oct. 1998.

SCHUSTER, B.; B, T.; LI, W.; CHEN, Y.; FRANCOIS, E. Fine Granular SNR Scalability: target applications. **ISO/IEC JTC1/SC29/WG11 M4426 - Reunião MPEG**, Seul, Mar. 1999.

SCHWARZ, H.; MARPE, D.; BLATTERMANN, G.; WIEGAND, T. Improved CABAC. **JVT-C060 JVT (ISO/IEC MPEGITU-T VCEG)**, Fairfax, May 2002.

SHAPIRO, J. Embedded Image Coding Using Zerotrees of Wavelet Coefficients. **IEEE Transactions on Signal Processing**, [S.l.], v.41, n.12, p.3445–3462, Dec. 1993.

SHEN, G.; ZENG, B.; LIOU, M. L. A New Padding Technique for Coding Arbitrarily Shaped Image/Video Segments. **IEEE International Conference on Image Processing**, Kobe, Japan, Oct. 1999.

SCIENTIFIC ATLANTA. Disponível em: <http://www.scientificatlanta.com.> Acesso em fevereiro de 2006.

SIKORA, T. Low Complexity Shape Adaptative DCT for Coding Arbitrarily Shaped Image Segments. **Signal Processing: Image Communication, Special Issue on coding Techniques for Very Low Bit Rate Video**, [S.l.], v.7, n.4-6, p.381–396, Nov. 1995.

SODOGAR, I.; LEE, H. J.; HATRACK, P.; ZHANG, Y. Q. Scalable Wavelet Coding for Synthetic/Natural Hybrid Images. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.9, n.2, p.244–254, Mar. 1999.

SON, S. H.; JANG, E. S.; LEE, S. H.; CHO, D. S.; SHIN, J. S.; SEO, Y. S. Scan Interleaving Based Scalable Binary Shape Coding. **Signal Processing: Image Communication, Special Issue on Shape Coding Emerging Multimedia Applications**, [S.l.], v.15, n.7-8, p.619–629, May 2000.

SUEHRING, K. **H.264/AVC Reference Software**. Disponível em: <http://iphome.hhi.de/suehring/tml>. Acesso em: maio 2006.

SULLIVAN, G.; WIEGAND, T. Rate-Distortion Optimization for Video Compression. **IEEE Signal Processing Magazine**, [S.l.], v.15, n.6, p.74–90, 1998.

SUN, M. T.; REIBMAN, A. Compressed Video over Networks. **Signal Processing and Communications Series**, New York, USA, 2001.

TALLURI, R. Error Resilient Video Coding in ISO MPEG-4 Standard. **IEEE Communication Magazine**, [S.l.], v.6, n.6, p.112–119, June 1998.

TAN, K. H.; GHANBARI, M. Layered Image Coding Using the DCT Pyramid. **IEEE Transactions on Image Processing**, [S.l.], v.4, n.4, p.512–516, Apr. 1995.

TAUBMAN, D. High Performance Scalable Image Compression with EBCOT. **IEEE Transactions on Signal Processing**, [S.l.], v.9, n.7, p.1158–1170, June 2000.

TAUBMAN, D.; ZAKHOR, A. Multirate 3D Subband Coding of Video. **IEEE Transactions on Image Processing**, [S.l.], v.3, n.5, p.572–588, Sept. 1994.

THAM, J. Y.; RANGANATH, S.; KASSIM, A. A. High Scalable Wavelet-Based Video

Codec for Very Low Bit-Rate Environment. **IEEE Journal on Selected Areas in Communications**, [S.l.], v.16, n.1, p.12–27, Jan. 1999.

TREES, H. V. **Detection Estimation and Modulation Theory**. New York, USA: Wiley, 1968.

UGUR, K.; NASIOPOULUS, P. Design Issues and Proposal for H.264 Based FGS. **ISO/IEC JTC1/SC29/WG11 M9505 - Reunião MPEG**, Pattaya, Mar. 2003. [Sl; sn]

USEVITCH, B. E. A Tutorial on Modern Wavelet Image Compression: foundations of jpeg2000. **IEEE Signal Processing Magazine**, [S.l.], v.18, n.5, p.22–36, Sept. 2001.

VETTERLI, M. Multidimensional Subband Coding: some theory and algorithms. **Signal Processing**, [S.l.], v.6, n.2, p.97–112, Apr. 1984.

VIDEO; GROUP, T. Draft 0.1 AVC Video Verification Test Plan. **ISO/IEC JTC1/SC29/WG11 N5124 - Reunião MPEG**, Shangai, Oct. 2002. Disponível em: < http://www.gammassl.co.uk/ist33/N833.pdf>. Acesso em: mar 2006.

WANG, Y.; ZHU, Q. F. Error Control and Concealment for Video Communications: A review. **Proceedings of the IEEE**, [S.l.], v.85, n.5, p.974–977, May 1998.

WEBCAST. **WebCastTechnologies**. Disponível em: <http://www.webcasttechnologies.com>. Acesso em: nov 2005.

WIEGAND, T.; SULLIVAN, G.; LUTHRA, A. Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC). **JVT-G050r1 JVT(ISO/IEC MPEGITU-T VCEG)**, Genebra, May 2002. Disponível em: < http://www.dspr.com/www/technology/h264.pdf>. Acesso em: mar 2006.

WU, F.; LI, S.; ZHANG, Y. Q.; SCHAAR, M.; LI, W. The Requeriments on Advanced FGS (AFGS). **ISO/IEC JTC1/SC29/WG11 M8734 - Reunião MPEG**, Klagenfurt, July 2003.[Sl; sn]

XIONG, Z.; RAMCHANDRAN, K.; ORCHAD, M. Space-frequency Quantization for Wavelet Image Coding. **IEEE Transactions on Image Processing**, [S.l.], v.6, n.5, p.677–693, May 1997.

YAN, R.; WU, F.; LI, S.; ZHANG, Y. Q. Error Resilience Methods in FGS Video Enhancement Bitstream. **ISO/IEC JTC1/SC29/WG11 M6207 - Reunião MPEG**, Pequim, July 2000. [Sl; sn]

ANEXO A

Seqüências de Teste

Neste anexo é apresentado uma breve caracterização das seqüências de teste utilizadas ao longo deste trabalho, evitando-se assim a descrição de cada seqüência de teste, cada vez que esta for utilizada. O leitor pode utilizar este anexo como referência cada vez que necessite de detalhes sobre as características de cada seqüência, sem ter que procurar o local exato da descrição ao longo do trabalho.

Na tabela A.1, apresenta-se uma breve descrição das principais características de cada seqüência.

Tabela A.1: Características das seqüências de teste.

Nome da seqüência	Boat	Canoa	Carphone	Coastguard	Rugby
Resoluções espaciais disponíveis	CIF, QCIF	CIF, QCIF	CIF, QCIF	CIF, QCIF	CIF, QCIF
No. de quadros	260	220	382	300	260
Frequência do quadro	30	25	25	25	30
Origem	VQEG	VQEG	MPEG	MPEG	VQEG

Nome da seqüência	Foreman	Stefan	Table Tennis	Tempete	Waterfall
Resoluções espaciais disponíveis	CIF, QCIF	CIF, QCIF	CIF, QCIF	CIF, QCIF	CIF, QCIF
No. de quadros	300	300	300	260	260
Frequência do quadro	25	25	25	30	30
Origem	MPEG	MPEG	MPEG	VQEG	VQEG

As seqüências de teste foram obtidas de duas fontes distintas: o grupo MPEG do ISO/IEC e o grupo VQEG (*Vídeo Quality Experts Group*) da ITU-T. De todas as seqüências disponibilizadas no contexto de ambos os grupos, foram escolhidas cinco do grupo MPEG e cinco do grupo VQEG. As seqüências de teste foram escolhidas para representar uma grande variedade de cenários, esta variedade de conteúdos corresponde também a grande variedade de dificuldades e particularidades de codificação como

convém para que se obtenham resultados representativos para casos extremos. Apresenta-se a seguir uma descrição mais detalhadas das seqüências de teste.

A.1 Seqüência *Boat*

Entre todas as seqüências de teste, esta é a que apresenta a menor quantidade de movimento ao longo do tempo. Consiste em uma cena de um barco atracado em um porto, filmada com câmera que se mantem fixa durante toda a seqüência. As duas bandeiras do barco apresentam algum movimento, devido ao vento, e as nuvens deslocam-se lentamente provocando uma pequena variação de iluminação na popa do barco. Algumas imagens desta seqüência são apresentadas na figura A.1, com um espaçamento temporal de 52 imagens.

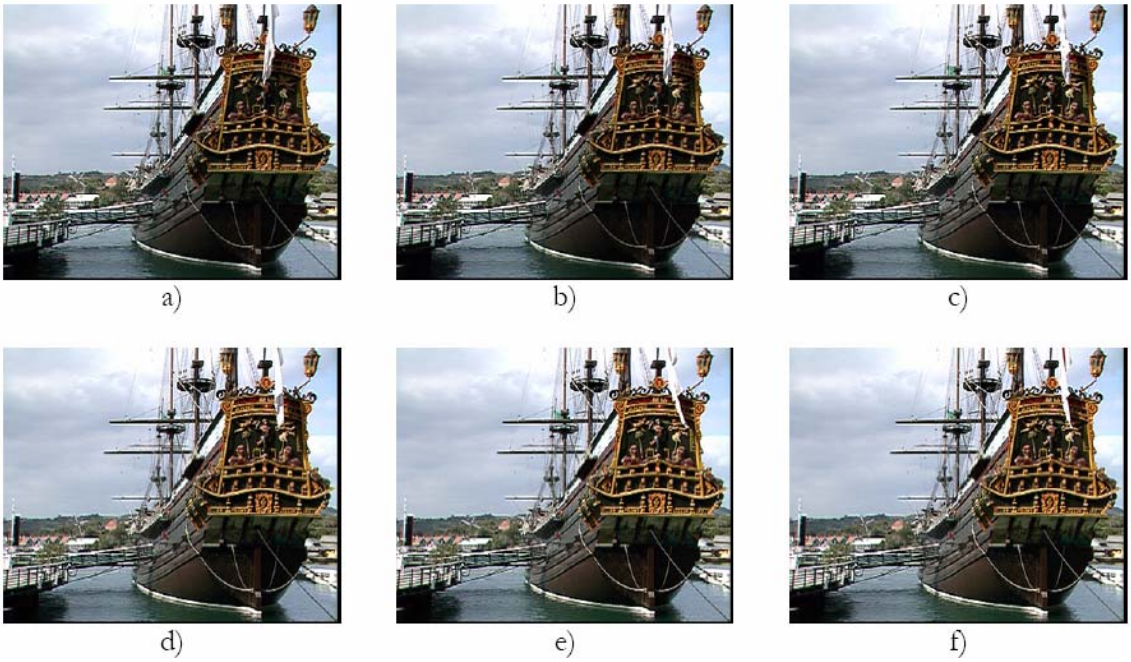


Figura A.1: Seqüência *Boat*: a) quadro 0; b) quadro 52; c) quadro 104; d) quadro 156; e) quadro 208; f) quadro 256.

A.2 Seqüência *Canoa*

A seqüência *Canoa* acompanha o movimento de um praticante de canoagem em um rio. Esta seqüência apresenta movimentos de câmera rápidos e com contraste elevado entre objetos presentes na cena (praticantes de canoagem) e o fundo. O praticante de canoagem, em primeiro plano, apresenta um movimento rápido, variado e difícil de caracterizar. Algumas imagens desta seqüência são apresentadas na figura A.2, com espaçamento temporal de 44 imagens.

Figura A.2: Seqüência *Canoa*; a) quadro 0; b) quadro 44; c) quadro 88; d) quadro 132; e) quadro 176; f) quadro 219.

A.3 Seqüência *Carphone*

A seqüência *Carphone* corresponde a uma conversa vídeo-telefonica em um automóvel em movimento. Possui zonas do fundo em que o movimento é quase inexistente (dentro do carro) e outras com uma quantidade de movimento mais significativa (na janela). O ator na conversa é bastante expressivo, com diversas reações faciais e largos movimentos. A câmera se mantém quase estática durante toda a seqüência mas existe a vibração do carro. Algumas imagens desta seqüência são apresentadas na figura A.3, com espaçamento temporal de 76 imagens.

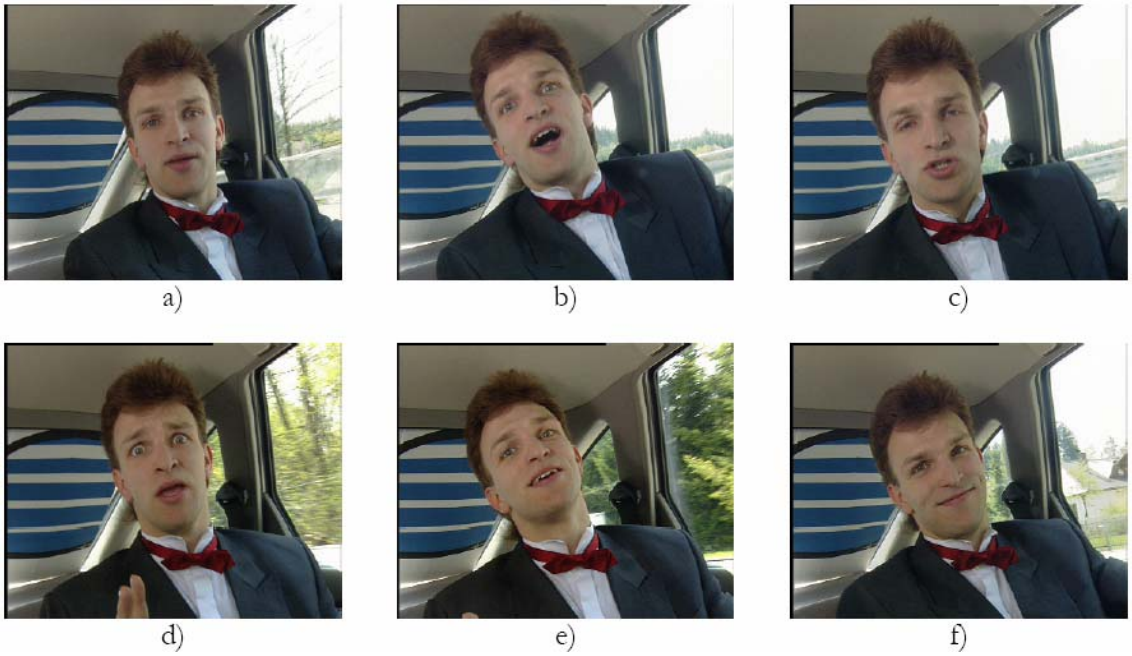


Figura A.3: Seqüência *Carphone*; a) quadro 0; b) quadro 76; c) quadro 152; d) quadro 228; e) quadro 304; f) quadro 380.

A.4 Seqüência *Coastguard*

Na seqüência *Coastguard*, a câmera segue um barco pequeno qu se desloca para esquerda até que um barco maior aparece do lado direito. Neste ponto, a câmera move-se rapidamente para cima e começa a seguir o barco maior para a direita. Os objetos presentes nesta cena apresentam um movimento bem definido e constante. Algumas imagens desta seqüência são apresentadas na figura A.4, com um espaçamento temporal de 60 imagens.

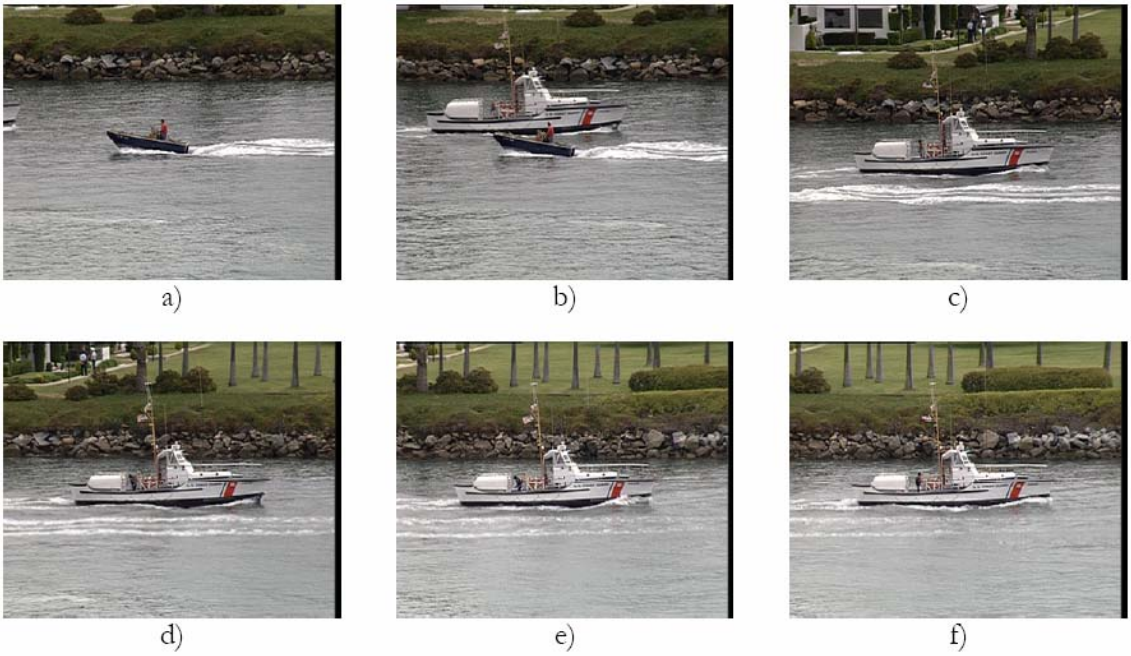


Figura A.4: Seqüência *Coastguard*; a) quadro 0; b) quadro 60; c) quadro 120; d) quadro 180; e) quadro 240; f) quadro 299.

A.5 Seqüência *Rugby*

A seqüência *Rugby* é a seqüência mais rápida do conjunto de seqüências e consiste em uma cena de futebol americano. A câmera tenta seguir a bola, através de movimentos de câmera rápidos, interromidos por paradas. Além disso, todos os jogadores se movimento de uma forma muito rápida. Algumas imagens desta seqüência são apresentadas na figura A.5, com um espaçamento temporal de 52 imagens.

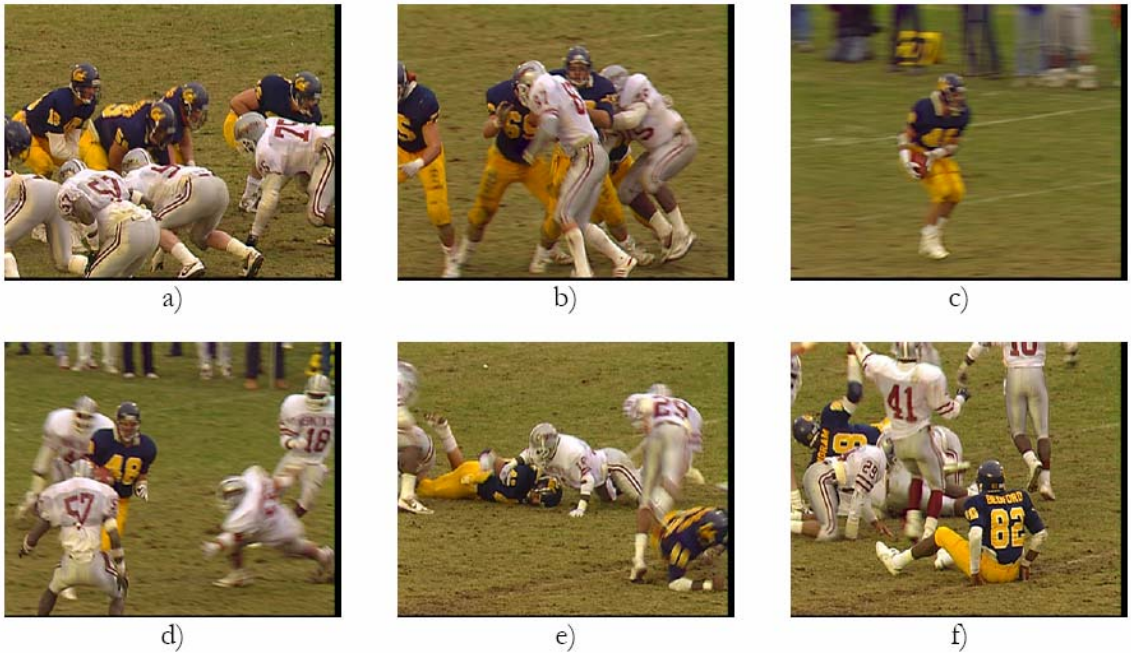


Figura A.5: Seqüência *Rugby*; a) quadro 0; b) quadro 52 c) quadro 104; d) quadro 156; e) quadro 208; f) quadro 259.

A.6 Seqüência *Foreman*

Esta seqüência pode ser claramente dividida em duas partes: uma cena de vídeo telefonia onde o telefone está na mão de quem fala, em seguida uma mudança rápida para uma cena de um prédio em construção. Durante a primeira cena, o movimento da câmera é reduzido; no entanto, o ator treme um pouco e movimenta-se, aproximando e afastando-se da câmera. Na segunda parte, o movimento da câmera é significativo, com a ocorrência de um *pan-left*. Nesta parte, os objetos não apresentam qualquer movimento. Algumas imagens desta seqüência são apresentadas na figura A.6; com um espaçamento temporal de 60 imagens.

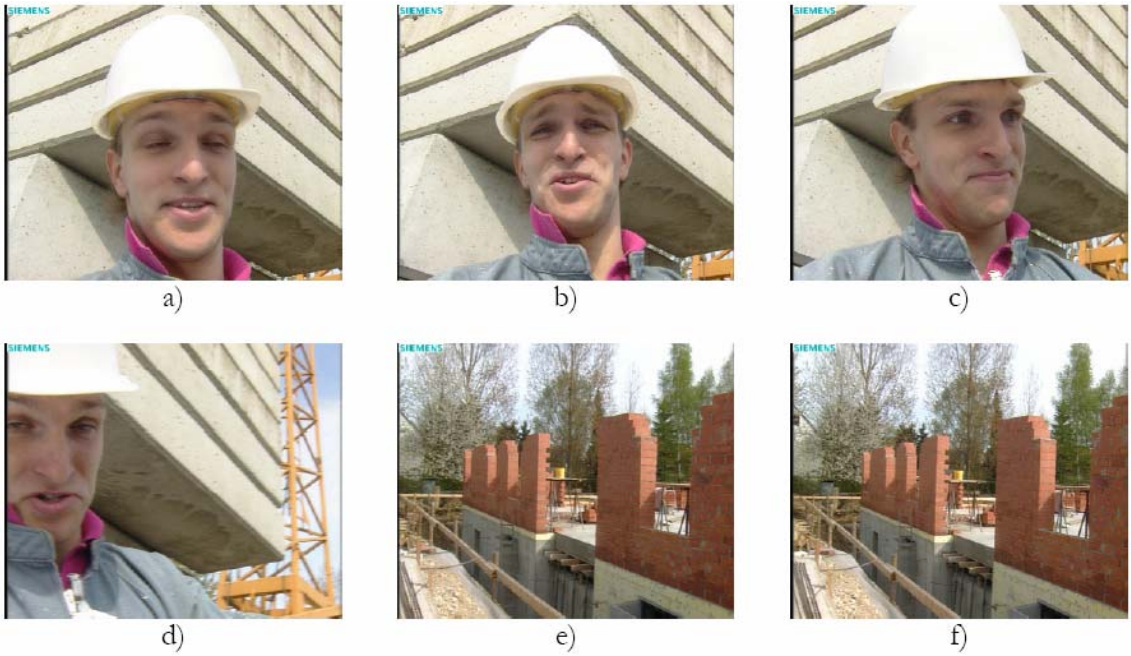


Figura A.6: Seqüência *Foreman*; a) quadro 0; b) quadro 60; c) quadro 120; d) quadro 180; e) quadro 240; f) quadro 299.

A.7 Seqüência *Stefan*

A seqüência *Stefan* é uma seqüência rápida que segue os movimentos de um jogador de tênis que se movimenta em todas as direções no campo. Por trás do jogado, encontra-se uma área com pouco movimento bastante texturada. Os movimentodas da câmera são praticamente horizontais e o movimento do jogador é bastante complexo ao longo do tempo. Algumas imagens desta seqüência são apresentadas na figura A.7, com espaçamento temporal de 60 imagens.

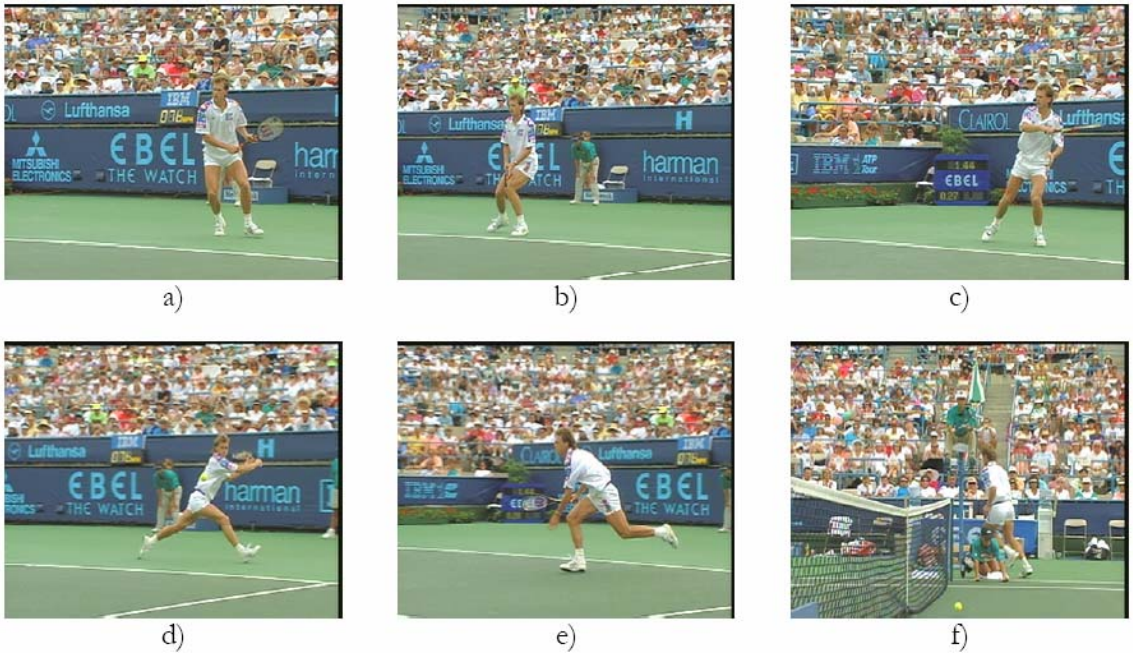


Figura A.7: Seqüência *Stefan*; a) quadro 0; b) quadro 60; c) quadro 120; d) quadro 180; e) quadro 240; f) quadro 299.

A.8 Seqüência *Table Tennis*

A seqüência *Table tennis* consiste em um jogo de tênis de mesa. Esta seqüência pode ser dividida em duas partes, separadas por um corte de cena. Na primeira metade, é filmando o início do jogo e ocorre um *zoom-out* sobre um dos jogadores; na segunda mentade, é filmado o segundo jogador e a câmera mantém-se fixa. Algumas imagens desta seqüência são apresentadas na figura A.8, com um espaçamento temporal de 60 imagens.

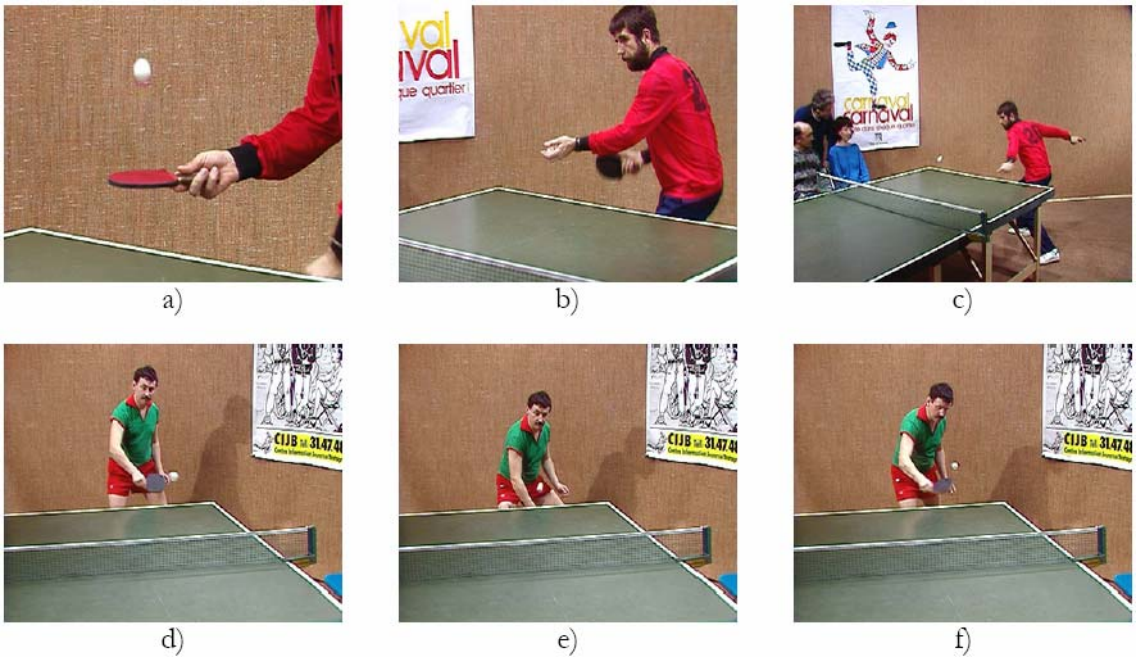


Figura A.8: Seqüência *Table*; a) quadro 0; b) quadro 60; c) quadro 120; d) quadro 180; e) quadro 240; f) quadro 299.

A.9 Seqüência *Tempete*

A seqüência *Tempete* é uma seqüência semi-sintética, uma vez que misura objetods do mundo rela com objetos gerados por computador. O único movimento de câmera que ocorre ao longo da seqüência é o *zoom-out*. Os únicos objetos que apresentam movimento são as folhas que caem em primeiro plano. Entre os objetos e o fundo (em tons de azul saturado), o contraste é elevado. Algumas imagens desta seqüência são apresentadas na figura A.9, com espaçamento temporal de 52 imagens.

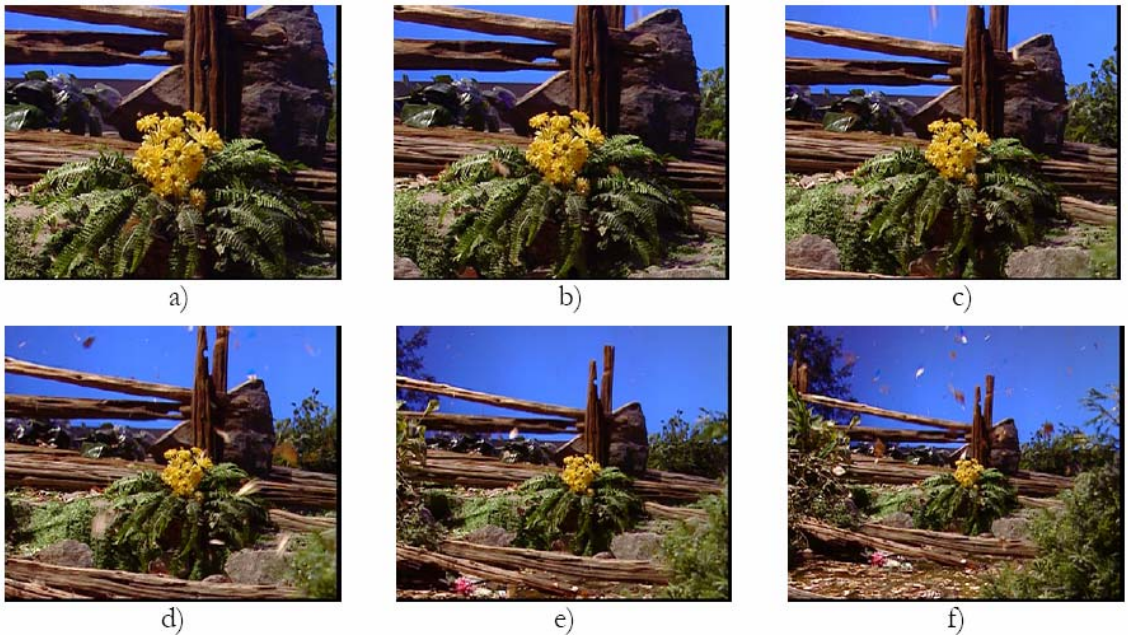
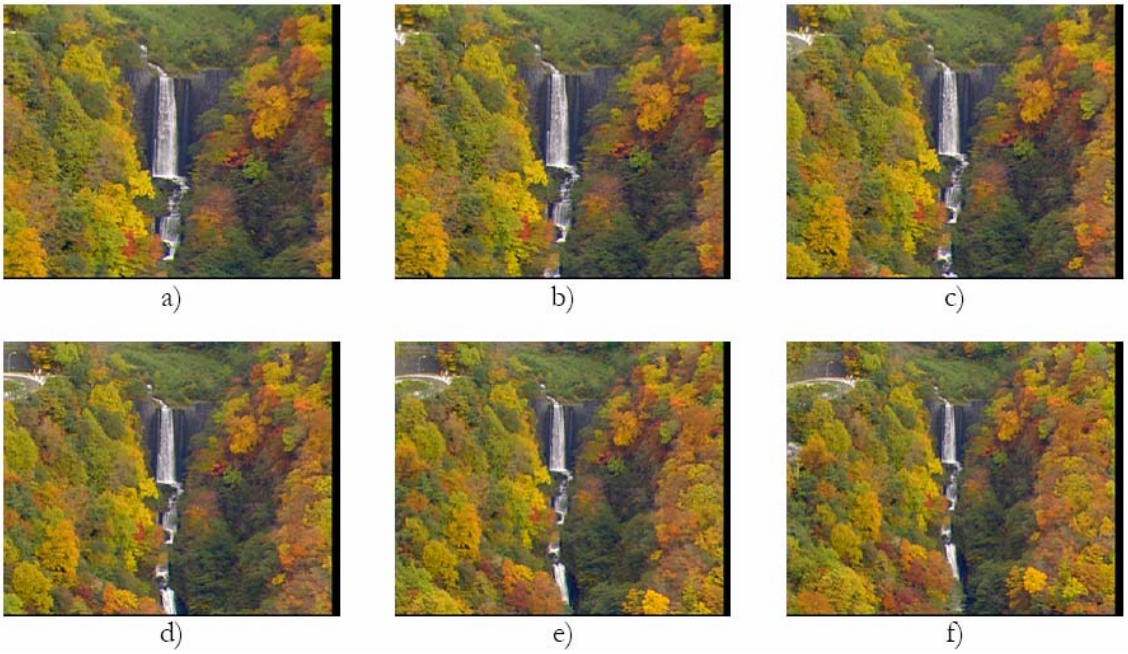


Figura A.9: Seqüência *Tempete*; a) quadro 0; b) quadro 52; c) quadro 104; d) quadro 156; e) quadro 208; f) quadro 259.

A.10 Seqüência *Waterfall*

A seqüência *Waterfall* mostra uma paisagem com uma queda de água distante. Apresenta uma textura rica, com muitas tonalidades de verdes e vermelhos. O único movimento de câmera que ocorre é o *zoom-out*. Algumas imagens desta seqüência são apresentadas na figura A.10, com um espaçamento temporal de 52 imagens.



A.10: Seqüência *Waterfall*; a) quadro 0; b) quadro 52; c) quadro 104; d) quadro 156; e) quadro 208; f) quadro 259.