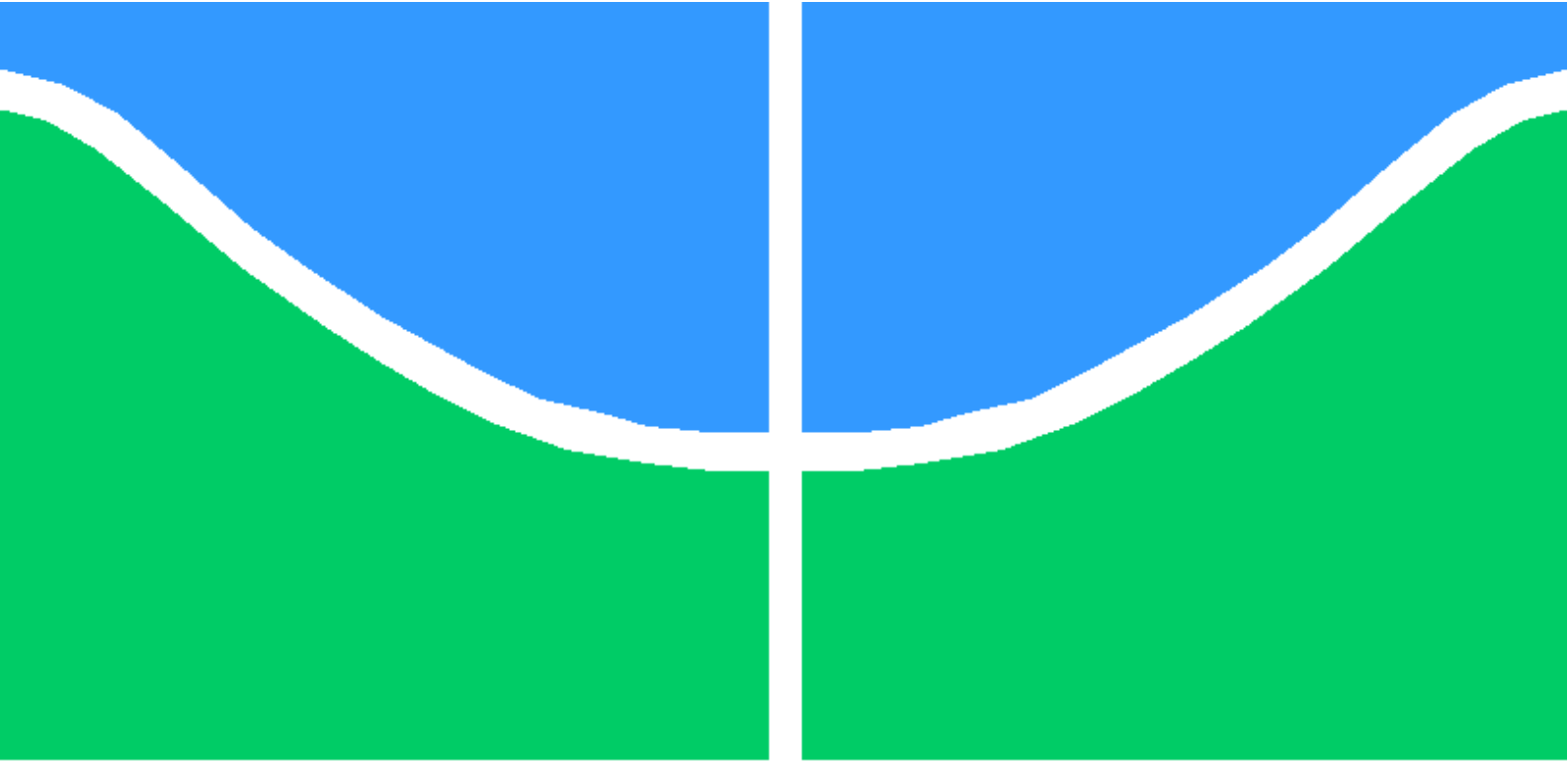


TESE DE DOUTORADO EM ENGENHARIA DE SISTEMAS ELETRÔNICOS E DE AUTOMAÇÃO

CODIFICAÇÃO DE VÍDEO ESCALONÁVEL EM COMPLEXIDADE E EM ENERGIA

Tiago Alves da Fonseca

Brasília, agosto de 2012

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

CODIFICAÇÃO DE VÍDEO ESCALONÁVEL EM COMPLEXIDADE E EM ENERGIA

Tiago Alves da Fonseca

ORIENTADOR: Ricardo Lopes de Queiroz

TESE DE DOUTORADO EM ENGENHARIA DE SISTEMAS ELETRÔNICOS E DE AUTOMAÇÃO

Publicação: PGEA.TD 057/2012
Brasília/DF: Agosto - 2012

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

TESE DE DOUTORADO EM ENGENHARIA DE SISTEMAS ELETRÔNICOS E DE AUTOMAÇÃO

CODIFICAÇÃO DE VÍDEO ESCALONÁVEL EM COMPLEXIDADE E EM ENERGIA

Tiago Alves da Fonseca

Tese de doutorado submetida ao Departamento de Engenharia Elétrica da Faculdade de Tecnologia da Universidade de Brasília, como parte dos requisitos necessários para a obtenção do grau de doutor.

Banca Examinadora

Prof. Ricardo Lopes de Queiroz, PhD. _____
UnB/ ENE (Orientador)

Prof. Eduardo Antônio Barros da Silva, PhD. _____
UFRJ/ COPPE (Examinador Externo)

Prof. Camilo Chang Dorea, PhD. _____
UnB/ CIC (Examinador Externo)

Prof. Francisco Assis de O. Nascimento, Dr. _____
UnB/ ENE (Examinador Interno)

Profa. Mylène Christine Q. de Farias, PhD. _____
UnB/ ENE (Examinadora Interna)

FICHA CATALOGRÁFICA

FONSECA, TIAGO ALVES
Codificação de Vídeo Escalonável em Complexidade
e em Energia. [Distrito Federal] 2012.
xx, 139pp., 210 mm x 297 mm (ENE/FT/UnB, Doutor, Engenharia de Sistemas Eletrônicos
e de Automação, 2012). Tese de Doutorado.
Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Elétrica.

1. Compressão de vídeo	2. Controle de complexidade
3. Vídeo de alta resolução	4. DPCM
5. H.264	6. Sistemas Verdes
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

FONSECA, T. A. da (2012). Codificação de Vídeo Escalonável em Complexidade e em Energia. Tese de Doutorado em Engenharia de Sistemas Eletrônicos e de Automação, Publicação PGEA.TD - 057/2012, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 139pp.

CESSÃO DE DIREITOS

NOME DO AUTOR: Tiago Alves da Fonseca.

TÍTULO DA TESE DE DOUTORADO: Codificação de Vídeo Escalonável em Complexidade e em Energia.

GRAU / ANO: Doutor / 2012

Tiago Alves da Fonseca
SQN 114, Bloco G, Ap. 610
70.764-070 Brasília - DF - Brasil.

Dedicatória

A meus pais, que cedo mostraram-me a importância da formação e sempre lutaram para me proporcionar uma educação de qualidade. À Eliete e à Vanessa, que têm de "aturar" meu humor e "conviver" com minha "ausência" todos os dias. À Sarah, pelos momentos agradáveis de companheirismo e motivação. Aos meus irmãos, à minha família e aos amigos, pelos momentos de diversão e pela motivação.

Tiago Alves da Fonseca

"Consulte não a seus medos, mas a suas esperanças e sonhos. Pense não sobre suas frustações, mas sobre o seu potencial não usado. Preocupe-se não com o que você tentou e falhou, mas com aquilo que ainda é possível a você fazer."

Karol Jozéf Wojtyła

Agradecimentos

*Agradeço à minha família, por me incutir a importância da formação desde cedo, especialmente a meus pais (Eliete, Valdir, Afonso Celso e Maria da Conceição (**in memoriam**)). Em nossa convivência, pude aprender com suas experiências de vida.*
Aos irmãos (Mariana, Felipe, Joaquim e Vanessa) pela paciência e compreensão.
À família, por sua curiosidade em saber do andamento do trabalho que, por sua vez, servia de ânimo para a continuação da longa jornada desta tese.
Aos amigos Britto e Carrijo, por garantirem que a chama do doutorado não fosse apagada por turbilhões de adversidades enfrentados.
À turma do GPDS (Mintsu, Zaghetto, Bruno, Diogo, Renan, Thacio, Jorge, Chaffim e Karen) pelo companheirismo, apoio, motivação e por servirem de exemplos para minha vida acadêmica.
À turma do SG11, por manter a "máquina" funcionando e, em especial, a Walter (Gaúcho), Alvino e Cícero. Ao amigo Filomeno, pelas manhãs de sábado regadas a café e histórias de vida.
Ao Dr. Ricardo, por sua incansável disposição em revisar os textos técnicos derivados deste trabalho.
Aos amigos, pelos momentos descontraídos e pelo companheirismo.
Ao colega Carlos, do GSEP, por sua disposição incondicional em acompanhar os ensaios de oscilografia necessários a este trabalho.
Aos colegas do CEPESC que sabem do valor de uma formação científica sólida e das renúncias necessárias na busca pela excelência.
*Por fim, ao meu professor **orientador** Ricardo Lopes de Queiroz, pela oportunidade de trabalho, pelo compartilhamento de seu conhecimento e motivação à pesquisa.*

Tiago Alves da Fonseca

RESUMO

Um dos tipos de sinais que mais se beneficiou dos avanços tecnológicos e industriais recentes foi o vídeo digital. O barateamento de sistemas de aquisição e a evolução das técnicas de processamento de sinais difundiu o emprego de sistemas de vídeo digital nas mais diversas aplicações. Uma das peças fundamentais dessa popularização foi a evolução dos codificadores de vídeo digital, culminando com o padrão H.264/AVC, considerado estado da arte em compressão de vídeo. Sua ampla gama de ferramentas de codificação tornou o conjunto complexo em termos computacionais, deixando como desafio a projetistas de sistemas de *hardware* e de *software* a otimização das metodologias do padrão para a devida realização do H.264/AVC em produtos comercialmente viáveis. Esta tese abordará a análise do codificador H.264/AVC sob a ótica do esforço computacional envolvido em sua operação a partir de implementações em *software* executadas em computadores pessoais.

A primeira contribuição trata de uma metodologia de otimização *on-line* do módulo de predições de forma a restringir a complexidade computacional da codificação a uma determinada provisão.

A segunda contribuição apresentada estende o conceito de otimização RD com a inserção de mais um eixo de análise, o eixo da complexidade C. Duas implementações de alto desempenho computacional foram estudadas e otimizadas em termos de RDC. Derivou-se, a partir de treinamento *off-line*, dois arranjos de codificadores capazes de comprimir vídeo digital a velocidades controladas em faixas de valores de interesse prático.

Por fim, uma última contribuição altera o esquema de otimização RDC e adiciona o eixo da energia demandada E ao problema de otimização RD, resultando num sistema em tempo real otimizado em termos de RDE. O codificador proposto otimizado por demanda energética é capaz de escalonar o consumo de energia em valores significativos às custas de impacto mínimo em termos de desempenho RD. Essa contribuição resume-se em um exemplo real de computação verde, em que uma atividade computacional é realizada por um mesmo equipamento, gastando menos energia e exposto a pequenas penalidades em termos de desempenho.

Com isso, esperamos estar contribuindo para um sistema mais “verde”, reduzindo as emissões de carbono de servidores de computação intensiva.

ABSTRACT

Digital video communications were largely benefited from advances in technology and in industrial processes. The falling prices of acquisition devices and the evolution of signal processing made digital video an ubiquitous technology. Digital video encoders are the cornerstone for the popularity of video technologies and its *state-of-the-art* is represented by the H.264/AVC standard. The myriad of coding tools made the H.264/AVC a massively complex application, imposing challenges to hardware and software designers when realizing commercial appliances. This thesis analyses the H.264/AVC complexity when implemented in software and executed on personal computers.

The first contribution leads to a on-line optimization method for the prediction stage in order to constrain the complexity to a certain level. The approach uses mode ranking and yields substantive complexity reduction.

The second contribution extends the RD optimization framework adding a third analysis axis, the complexity C axis. Two high performance implementations were studied and RDC optimized. We derived a framework that allow for practical values of encoding speed with minor performance penalties.

The RDC optimization framework was also modified by adding another axis to the optimization: the energy E axis. We provide a real-time RDE optimized scheme which is capable of scaling the energy demands in a significant range, slightly impacting the RD performance. This third contribution is a true example of green computing where the same task is accomplished in the same hardware system with much less energy consumption, incurring only is small performance penalties.

Since we can provide settings to meet the rate and distortion targets, as well as the maximum encoding speed, using less energy, we hope to contribute towards a “greener” system, reducing the carbon footprint of video compression servers.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS, ABREVIAÇÕES E ACRÔNIMOS

Abreviações, Acrônimos e Siglas

1080p	quadros de 1920×1080 *pixels*, captura progressiva
$4:2:0$	método de amostragem em que as componentes de crominância possuem metade da resolução na direção vertical e na direção horizontal da componente de luminância
720p	quadros de 1280×720 *pixels*, captura progressiva
AVC	*Advanced Video Coding*
CABAC	*Context-Based Adaptive Binary Arithmetic Coding*
CAVLC	*Context-Adaptive Variable Length Coding*
CD	*Compact Disk*
CIF	*Common Intermediate Format*, quadros de dimensões 352×288 *pixels*
codec	Codificador e Decodificador
DVD	*Digital Versatile Disk*
FRExt	*Fidelity Range Extensions*
GB	Giga bytes (1073741824 bytes)
HD	*High Definition*
HEVC	*High Efficiency Video Coding*
HVS	*Human Visual System*
IEC	*International Electrotechnical Commission*
ISO	*International Standards Organization*
ITU	*International Telecommunication Union*
JPEG	*Joint Photographic Experts Group*
JVT	*Joint Video Team*
LCH	*Lower Convex Hull*
MPEG	*Motion Picture Experts Group*
MSE	*Mean Square Error*
PC	*Personal Computer*, computador pessoal
PSNR	*Peak Signal to Noise Ratio*
QP	*Quantization Parameter*
RD	*Rate vs. Distortion*
RDC	*Rate vs. Distortion vs. Complexity*
RDE	*Rate vs. Distortion vs. Energy*
RDO	*Rate-Distortion Optimization*
RGB	espaço de cores *Red/Green/Blue* (vermelho, verde e azul)
SD	*Standard Definition*
SSIM	*Structural SIMilarity Index*
UMHS	*Hybrid Unsymmetrical-cross Multi-hexagon Grid Search*
VCEG	*Video Coding Experts Group*
VHS	*Video Home System*
VLC	*Variable Length Code*
YUV	Espaço de cores

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

O avanço e a popularização da tecnologia digital têm sido notáveis nas últimas duas décadas. Canais de comunicação tiveram, e ainda têm, suas capacidades de transmissão aumentadas constantemente. A disponibilidade de canais de banda larga em domicílios já é uma realidade. Os meios de armazenamento de informações digitais em massa têm sua capacidade de armazenamento superada a cada mês. Um olhar superficial sobre esses avanços poderia levar a acreditar que não se justifica o uso de técnicas de compressão de sinais [1]. Uma análise mais detalhada, contudo, nos revela que até mesmo o DVD (*Digital Versatile Disc*), um dos meios mais difundidos com grande capacidade de armazenamento de dados digitais, seria inútil sem o auxílio de compressão. Tome-se como exemplo o armazenamento de um vídeo de definição padrão (SD, *standard definition*) com quadro de dimensões de 720 *pixels* (pontos elementares de imagem) de largura por 480 *pixels* de altura e capturado a uma taxa de 30 quadros por segundo. Para armazenar um quadro de vídeo em cores sem compressão, gastariam-se 518.400 bytes. Sendo a taxa de aquisição de 30 quadros por segundo, para armazenar um segundo gastariam-se $15.552.000$ bytes. Dada a capacidade de armazenamento de um DVD, $4,7$ GB ($4,7 \times 2^{30}$ bytes), conclui-se que seria possível armazenar apenas $5,4$ minutos de vídeo sem compressão. Sabe-se que um DVD é capaz de armazenar não só vídeos de mais de duas horas, mas também legendas, áudios e outros conteúdos extras usados na promoção de filmes comerciais. É necessária, portanto, a compressão de sinais para o armazenamento dessas informações digitais.

Estimativa recente revelou que 24 horas de vídeo são carregadas no Youtube[1] a cada minuto [2]. Considerando-se que os vídeos sejam carregados a taxas proporcionadas por canais comuns de banda larga entre 2 e 5 Mbps, seriam demandados entre 45 e 75 terabytes de espaço em disco por dia para armazenamento de dados previamente codificados. Se esses sinais fossem tratados em formato bruto, essas cifras anteriores ganhariam algumas ordens de grandeza.

Além dos empecilhos que têm sido tratados para as plataformas de comunicação por difusão (*broadcast*), a transmissão de vídeo a partir de dispositivos móveis é um campo que traz uma série de desafios para a indústria de telecomunicações. As restrições que envolvem esse tipo de dispositivo

[1] http://www.youtube.com

vão do limitado espectro de frequências de transmissão (esse é um recurso escasso e que não pode ser aumentado) ao poder de processamento restrito e aos recursos finitos de energia. Somente a agregação das técnicas mais modernas de codificação de sinais, dentre elas a compressão, garantirão que tais dispositivos consigam capturar e transmitir vídeo em tempo real, com a maior qualidade possível, usando a menor largura de banda e pela maior quantidade de tempo. Mesmo com avanços constantes nas capacidades de armazenamento e transmissão de dados digitais, a compressão ainda será um componente essencial para serviços de comunicação multimeios por muitos anos.

A compressão de um **sinal** (denominação dada a um ente que carrega **informação** [3]) pode ser encarada como um processo que resulta numa representação mais eficiente do mesmo. Sob a óptica de um sistema digital, compressão resume-se em um processo que gera uma representação que necessita de menos bits (unidade básica de informação digital) que a representação dos dados brutos (*raw data*, no inglês) [4]. São duas as classes principais de técnicas que podem ser empregadas nesse processo: as técnicas de compressão sem perdas e as de compressão com perdas.

Em um sistema de compressão sem perdas, a redundância estatística é removida de forma que o sinal original possa ser reconstruído perfeitamente no receptor. O princípio básico da compressão sem perdas é gastar a menor quantidade de bits possível para codificar um sinal sem perder informação. Esse tipo de sistema é extremamente importante quando um erro na versão comprimida do sinal pode comprometer a integridade da informação, como é o caso de documentos eletrônicos, registros bancários etc. Contudo, apresenta desempenho modesto quando aplicado diretamente a imagens ou vídeos digitais [4].

Por outro lado, há certos tipos de informações em que falhas na reconstrução exata não constituem um problema. Na codificação de sinais de áudio, o valor exato da amostra digitalizada não é necessário para garantir a inteligibilidade e uma certa quantidade de informação pode ser descartada a depender do contexto. Por exemplo, um sinal de voz pode ser transmitido por meio de canais bastante degradados e, mesmo com perda significativa da informação de sinal, é possível a compreensão da informação a ser transmitida [5]

As técnicas de codificação baseadas no paradigma da compressão com perdas implicam grande compressão atingida às custas de perda de qualidade do sinal tratado. Dessa forma, a versão decodificada passa a ser diferente da versão original do sinal; à diferença resultante do processo de codificação do sinal dá-se o nome de **distorção**. A comunicação de sinais de vídeo, por demandar a transmissão de grandes quantidades de informação, baseia-se notadamente em técnicas de compressão com perdas.

1.2 DEFINIÇÃO DO PROBLEMA

1.2.1 Codificação de vídeo

A codificação[2] de vídeo é caracterizada pelo processo de compressão e descompressão de um sinal de vídeo digital. A compressão pode ser entendida como o processo que procura a representação mais eficiente do sinal em termos de uma determinada função de custo, a qual pode relacionar fidelidade, quantidade de bits no armazenamento e outros atributos de relevância do sinal. A observação superficial de um sinal de vídeo mostra que há similaridade entre quadros consecutivos de uma sequência de vídeo (veja a Figura 1.1).

(a) (b)

Figura 1.1: Quadros consecutivos da sequência “Foreman”.

Note que boa parte do conteúdo visual da cena permanece constante entre um quadro e outro. Uma estratégia inicial para compressão poderia ser mandar somente as diferenças existentes entre os quadros. Convencionou-se chamar de codificação **Inter-quadros** [4, 6] as técnicas existentes na literatura que abordam a redundância temporal do sinal de vídeo. Como, então, é enviado o primeiro dos quadros de uma sequência de vídeo digital? Usar um algoritmo de compressão de imagens é uma solução possível. As estratégias mais difundidas para extração de redundâncias espaciais em uma imagem digital baseiam-se na aplicação de uma operação matemática chamada de **transformada**. Seu princípio de funcionamento é representar o sinal de uma maneira mais conveniente a uma dada situação. Em compressão, os sinais são transformados de forma a “compactar” a **energia de sinal**. O emprego exclusivo de técnicas de codificação de imagens em que não são feitas referências a outros quadros na compressão de um quadro denomina-se de codificação **Intra-quadro** [4, 6].

[2]Esta tese tratará de codificação de fonte. Doravante, as alusões a codificação se restringirão ao contexto de codificação de fonte.

Um codificador de vídeo é usualmente composto pela agregação de técnicas de codificação Intra-quadro e Inter-quadros, de onde se deriva a nomenclatura codificador de vídeo híbrido. À primeira vista, o projeto de um codificador de vídeo pode parecer preocupado somente com a redução das demandas de armazenamento ou de largura de banda; todavia, envolvem-se outros aspectos relativos ao desempenho do sistema de codificação. Parâmetros como critérios de taxa $\times$ distorção, complexidade do algoritmo utilizado, características do canal de transmissão, estatísticas da fonte de vídeo, entre outros, devem ser levados em consideração na seleção das técnicas de codificação mais convenientes. Como são inúmeras as possibilidades para a construção de um codificador, há a necessidade de se estabelecerem padrões de compressão de vídeo para promover a interoperabilidade entre produtos semelhantes de diferentes fabricantes.

Pela cronologia, o H.261 [6], desenvolvido pela ITU-T (*International Telecommunication Union, Telecommunication Standardization Sector*), foi o primeiro padrão de codificação de vídeo difundido com sucesso, capaz de comprimir vídeo para as taxas de transmissão mais comuns na época (entre 80 e 320 kbit/s para comunicações em vídeo digital). Sua estrutura ainda é herdada por muitos codificadores modernos. Em seguida, surge o MPEG-1 [7], desenvolvido conjuntamente pela ISO (*International Organization for Standardization*) e a IEC (*International Electrotechnical Commission*), codificador bem difundido, com faixa de operação maior que o padrão anterior (1 a 2 Mbit/s) e capaz de proporcionar qualidades superiores às de fitas VHS (*Video Home System*) consumindo 1,5 Mbit/s de taxa de transmissão. O MPEG-2 [8], seu sucessor, foi desenvolvido de forma a superar o desempenho de padrões anteriores. Ele se destaca por promover um avanço maior na qualidade de imagem e por sua popularidade: é o responsável por impulsionar a difusão de sinais de televisão digital em resolução padrão e alta definição [9].

Com foco em transmissão de vídeo em baixas taxas, o H.263 [10] foi considerado o estado da arte da sua época. Como sucessor do H.261, sua faixa de operação era inicialmente em torno de 10 a 30 kbit/s, mas foi estendida para 10 a 2048 kbit/s. Retornando à família de codificadores MPEG, surge o MPEG-4 [11], criado com objetivo de padronizar os métodos de codificação mais eficientes e mais genéricos disponíveis até o momento de sua proposição, capazes de manipular vários tipos de dados audiovisuais. Seu ponto de partida é o codificador H.263 no perfil *Baseline*. Isso implica que decodificadores compatíveis com MPEG-4 devem ser capazes de decodificar vídeos comprimidos com H.263 *Baseline*. [12]

O H.264/AVC (*Advanced Video Coding*) [13] é um dos mais novos padrões de compressão de vídeo e resulta da colaboração entre as equipes *ISO/IEC Moving Picture Experts Group* (MPEG) e *ITU-T Video Coding Experts Group* (VCEG) sobre a agregação denominada JVT (*Joint Video Team*). Além

de promover a desejada interoperabilidade, essa padronização oferece elevada eficiência de compressão para a classe específica de sinais de vídeo resultantes da captura de cenas reais (e não sintéticas), além de proporcionar representação de vídeo conveniente à transmissão tanto para aplicações interativas quanto para não-interativas.

Os ganhos de compressão trazidos por esse novo padrão vieram, de certa maneira, às custas de elevada complexidade computacional. A título de ilustração, a complexidade computacional de seu decodificador é quatro vezes maior que a do MPEG-2 e duas vezes maior que a do MPEG-4 *Visual* [14].

O estado da arte em compressão de vídeo atingido pelo H.264/AVC foi em grande parte resultado do refinamento de técnicas de codificação aplicadas a outros padrões. Como destaque, podemos elencar:

- estágio preditivo variado, composto por compensação de movimentos com variadas partições, flexibilidade no emprego de quadros de referência e vetores de movimento com precisão refinada de até 1/4 de *pixel* e predição Intra-quadro com variadas partições;
- módulo de transformada com suportes de diferentes tamanhos, com empregar transformadas inteiras reversíveis de tamanhos 4×4 e 8×8 para sinais de luminância e de tamanho de 2×2 para sinais de crominância; e
- codificador de entropia mais eficiente e que emprega contextos adaptativos na codificação dos elementos sintáticos.

A aplicação dos refinamentos listados acima resultam aumento de complexidade tanto no codificador, como no decodificador [14, 15].

Este último padrão sofreu um adendo em 2004 por meio da inclusão de novas técnicas que lidam com questões de fidelidade do sinal, comumente referenciadas por FREext (*Fidelity Range Extensions*) e representadas pelo perfil de codificação *High Profile* [16]. A adição desse conjunto de ferramentas tornou o H.264/AVC mais atrativo de forma a estabelecê-lo como item desejável na maioria das aplicações de vídeo digital. Embora importantes extensões tenham sido adicionadas ao padrão H.264/AVC desde então — entre elas, *Professional Profiles* [17], *Scalable Video Coding* (SVC) [18] e codificação em múltiplas vistas (*3D Stereo/Multiview Video Coding*, MVC) [19, 20] —, os mesmos abordaram outras questões em vez de pura e simples eficiência de codificação.

Dada a elevada demanda por qualidade na transmissão de vídeo digital, as organizações de padronização, especificamente ITU-T *Video Coding Experts Group* (VCEG) e ISO/IEC *Moving Picture Experts*

Group (MPEG), decidiram iniciar um trabalho conjunto de padronização de mais um novo padrão para a compressão de vídeo digital por meio da agregação dos avanços tecnológicos recentes. Foi estabelecido um grupo de trabalho denominado *Joint Collaborative Team on Video Coding* (JCT-VC) e expedida uma chamada de trabalhos conjunta. O resultado da apresentação de propostas do primeiro encontro do JCT-VC [21] lista tecnologias capazes de superar substancialmente o desempenho do H.264/AVC e que, em breve, serão agregadas pela definição do mais novo padrão de compressão de vídeo denominado HEVC, *High Efficiency Video Coding* [22].

1.2.2 Escalabilidade

Em eletrônica, o termo escalabilidade refere-se à capacidade de um sistema gerenciar o crescimento da carga de trabalho de maneira efetiva, isto é, a capacidade de se adaptar para acomodar esse crescimento [23]. Como propriedade de um sistema, o conceito de escalabilidade é geralmente atrelado à definição dos requisitos específicos nas variáveis/dimensões em relação às quais a escala é relevante. Um sistema cujo desempenho melhora depois do incremento de recursos, proporcionalmente à capacidade instalada, é dito um sistema escalonável.

1.2.3 Complexidade e Energia

Complexidade é o nome dado à medida de intensidade de esforço computacional necessário para a execução de uma tarefa com auxílio de computador. O trabalho seminal de Hartmanis e Stearns [24], contemporâneo à aplicação de computadores digitais na execução de tarefas, preocupava-se com a classificação de problemas de forma a discernir aqueles que são possíveis de serem tratados por um computador digital dadas restrições de tempo e memória e apresentava o termo complexidade. A restrição de tempo de execução de uma tarefa em um computador foi investigada no trabalho de Yamada [25], onde aparece o termo computação em tempo-real (do inglês *real-time*).

Na medida em que a pesquisa avançou, o termo complexidade foi generalizado e, como medida de esforço, passou a ser analisado por meio de diferentes métricas que dependem, basicamente, do tipo de tarefa computacional. É possível considerar o tempo gasto ou o número de operações básicas necessárias para executar uma tarefa, ou a quantidade de memória para executar uma atividade computacional, ou a quantidade de portas lógicas necessárias para que um determinado dispositivo eletrônico seja capaz de executar determinada aplicação [26].

Quando se trata de aplicações multimeios, o poder de computação deve ser considerado na escolha de um dispositivo ou plataforma de maneira a garantir o processamento em tempo-real, ou seja, em velocidade pelo menos igual à com que as informações a serem processadas são disponibilizadas. Ao se codificar vídeo digital, o menor poder de computação deve ser capaz de garantir processamento do sinal numa velocidade pelo menos igual à velocidade com que os quadros de vídeo chegam no codificador. A complexidade de um *codec* em termos do número de operações por segundo é uma medida que determina os requisitos mínimos de um equipamento escolhido como plataforma de execução capaz de comprimir o vídeo em tempo-real.

Uma vez definida a plataforma que executará a compressão, é possível tomar a quantidade de energia necessária no processo como medida de esforço e, portanto, como medida de complexidade. Uma das grandes vantagens de se utilizar medidas de energia é a possibilidade de vinculação direta a custos de operação, por exemplo, às contas de energia elétrica.

1.2.4 Codificação de Vídeo Escalonável em Complexidade e Energia

Os avanços em ganhos de compressão se fizeram às custas de elevação do esforço computacional envolvido na codificação dos sinais de vídeo. As tarefas computacionais, por sua vez, são realizadas mediante a aplicação de energia às plataformas eletrônicas de processamento cada vez mais potentes e rápidas. Entretanto, a conservação energética passou a ser uma preocupação fundamental atualmente. Programas governamentais têm fornecido incentivos para o desenvolvimento e para a aplicação de tecnologias de baixo impacto ambiental e, quando possível, sustentáveis. Por outro lado, as pessoas e as empresas passam a procurar produtos eficientes energeticamente de maneira a reduzir o custo de operação de suas casas e instalações, haja vista o elevado custo da energia elétrica necessária para operar os diversos aparelhos.

No caso de sistemas de computação intensiva, o consumo energético é fator crítico tanto em termos de custos operacionais como em termos de disponibilidade. Mesmo modernos *datacenters* são significativamente onerados pelas despesas em energia elétrica. Há estimativas que apontam que custos de propriedade dos *datacenters* poderão ser superados pelos custos em energia elétrica por uma ampla margem [27]. A dissipação térmica, derivada da aplicação da energia, é outro problema correlato quando se trata de dispositivos móveis alimentados por bateria. Em função disso, observa-se que a manipulação da energia tornou-se restrição fundamental no projeto de sistemas computacionais. É possível abordar o consumo energético por meio de soluções em *hardware* e *software*. Projetistas de *hardware* têm explorado

diversas alternativas na redução da demanda por energia de seus produtos. Por outro lado, emerge um interesse considerável na criação de técnicas algorítmicas para economia de energia [28].

1.3 OBJETIVOS

O presente trabalho apresenta novas metodologias para implementações em *software* de codificadores, de forma a torná-las capazes de gerenciar a complexidade computacional na codificação de sinais de vídeo. A abordagem de otimização por taxa-distorção-complexidade (RDC) é apresentada de forma a restringir a complexidade do compressor num cenário de codificação em tempo-real. A partir dos parâmetros otimizados no sentido RDC, um arranjo de codificação em tempo-real com complexidade controlada é montado e avaliado. De forma a prover um codificador de vídeo em tempo-real de complexidade escalonável e com controle de taxa de transmissão, esta pesquisa apresenta modificações a aspectos não normativos do módulo de predições de implementações do H.264/AVC.

O cenário de escalabilidade tratado nesta tese consiste em um sistema cuja entrada é um fluxo de vídeo digital em que quadros, geralmente de alta-resolução, são apresentados ao codificador. A saída do sistema, por sua vez, consiste num fluxo de bits que representa o vídeo comprimido de acordo com as demandas de taxa de bits (R) e qualidade (distorção, D) de um usuário. A Figura 1.2 ilustra os componentes principais de um arranjo típico de testes desta tese. A plataforma que executa o codificador deve garantir que a velocidade com que os quadros de vídeo serão comprimidos (C) pelo codificador não seja inferior à taxa com que os quadros de vídeo chegam ao mesmo. Durante a compactação dos quadros de vídeo digital, uma taxa de bits por segundo fica estabelecida como largura do canal de transmissão. Para atender às restrições do canal, serão tolerados apenas ajustes na qualidade do vídeo; portanto, não será permitida qualquer alteração das resoluções espaciais (dimensões dos quadros em *pixels*) ou temporais (descarte de quadros de vídeo). As demandas energéticas (E) também serão levadas em consideração durante o processo de compressão.

Os métodos propostos podem ser prontamente utilizados em sistemas de comunicações móveis cujas capacidades de computação crescem na direção de nível de computadores pessoais, contudo ainda restritos a critérios estritos de eficiência energética [29]. Nesta tese, a grandeza complexidade tem seu significado abstraído no contexto de esforço de computação. Para medir complexidade durante o processo de

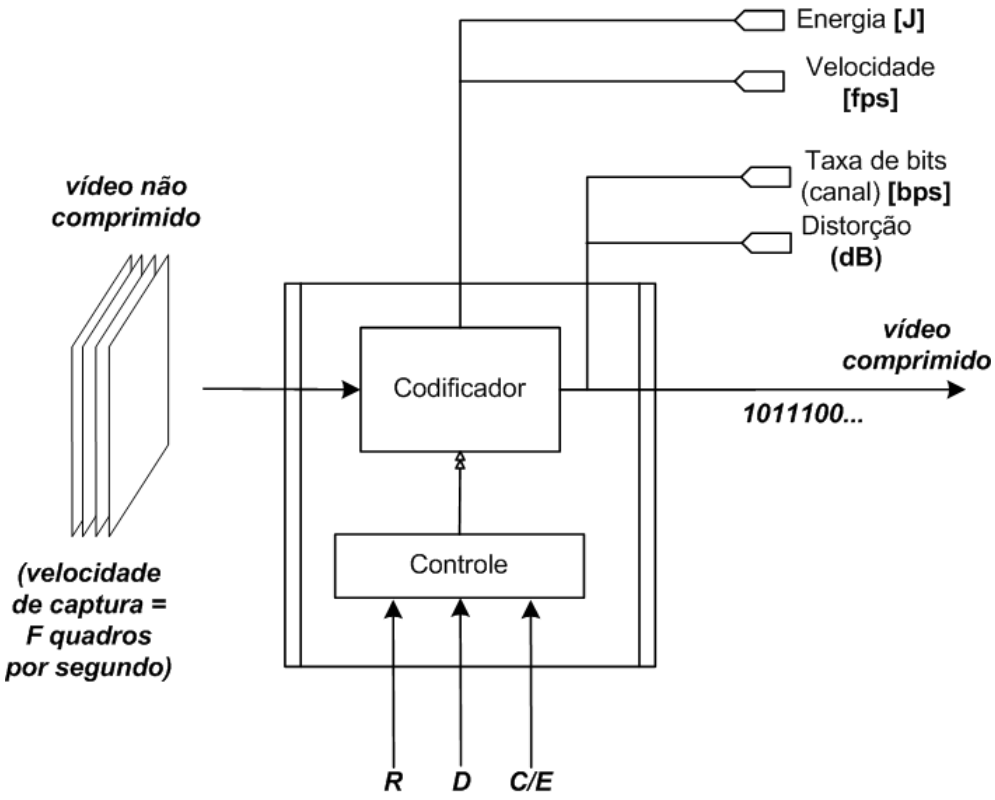


Figura 1.2: Cenário básico mostrando os componentes e as grandezas envolvidas no processo de otimização de implementação de um codificador de vídeo. Os alvos de taxa de bits (R), qualidade do sinal comprimido (D), complexidade (C, medida pelo gasto na compressão) e energia (E) são apresentados ao sistema de codificação como referências a serem respeitadas. Pequenos pentágonos irregulares da metade superior do esquema são usados para representação gráfica de sondas de medição das grandezas descritas em seu lado direito.

compressão, usamos tempo de codificação (quantos segundos o codificador levou na compressão de determinado quadro).

1.4 ORGANIZAÇÃO DA TESE

No Capítulo 2, é feita uma breve introdução acerca de processamento digital de sinais, com ênfase no processamento de vídeo. Serão apresentadas as técnicas envolvidas desde a captura do vídeo a partir de uma cena até o tratamento das amostras do vídeo de forma a proporcionar representação eficiente do sinal em termos de taxa e distorção. Ainda nesse capítulo, apresenta-se o codificador H.264/AVC, correlacionando as técnicas de compressão introduzidas com aquilo que é explorado e realizado em um codificador de vídeo do *estado da arte*.

A complexidade computacional na codificação de vídeo é abordada no Capítulo 3, em que é feita uma revisão bibliográfica das iniciativas de manipulação de complexidade na codificação de vídeo. É estabelecida a abordagem geral de controle do esforço de computação requerido pelo codificador H.264/AVC após a discussão do perfil de complexidade dos seus módulos constituintes.

O Capítulo 4 discute a primeira contribuição para o controle de complexidade de um codificador de vídeo. Por meio de uma metodologia de análise da recorrência de uso de ferramentas de predição de um *codec*, apresenta-se um processo de treinamento *on-line* por meio do qual a complexidade computacional de um codificador é estimada e controlada enquanto se comprimem quadros de vídeo. A supressão seletiva de ferramentas usadas com menor frequência durante a codificação de um quadro é explorada para escalonar a complexidade total de codificação. A metodologia proposta foi implementada no *software* de referência do H.264/AVC e a viabilidade da técnica é discutida a partir dos resultados experimentais obtidos.

Um esquema de otimização RDC, $Rate \times Distortion \times Complexity$ (Taxa *vs.* Distorção *vs.* Complexidade), é introduzido no Capítulo 5. Essa segunda contribuição utilizará de treinamento *off-line* para determinação do LCH (*Lower Convex Hull*, do inglês Casco Convexo Inferior) de um codificador de vídeo H.264/AVC a partir de uma busca extensiva de suas configurações de ferramentas de predição. As configurações ótimas desejadas são compiladas numa tabela de consultas que lista aquelas capazes de codificar vídeo com a menor demanda de taxa de bits, a menor distorção de sinal e a menor complexidade computacional, medida como tempo de computação. Determinadas as tabelas de configurações ótimas,

insere-se essa tabela no codificador de forma que o usuário possa escolher uma determinada complexidade computacional e o sistema se autoconfigurar para prover as configurações ótimas de compressão para aquela complexidade. O desempenho dessa contribuição é avaliado pela sua inserção em implementações do H.264/AVC de alto desempenho em termos de velocidade de compressão: o codificador de exemplo da biblioteca Intel® IPP (Intel *Performance Primitives*), doravante chamado de codificador IPP, e o codificador em código fonte aberto x264.

O Capítulo 6 trata da codificação de vídeo digital restrita pelas demandas de energia elétrica. Comunicações de vídeo têm demandado poder de computação cada vez maior e, portanto, maior consumo de energia. Para permitir soluções *verdes*[3] de codificação de vídeo baseadas em *software*, propõe-se um método para otimizar o estágio de predição do codificador H.264/AVC em termos de energia (E). Além de buscar as melhores opções de codificação para o melhor desempenho em termos de RD (*Rate-distortion*), o processo é restrito para adequar-se a uma determinada provisão de energia por meio de otimização no espaço RDE. Mede-se E pela energia elétrica demandada por um computador pessoal que executa o sistema de codificação. Os resultados apresentados mostram um *codec* otimizado por RDE que permite compressão de vídeo por *software* em tempo real restrita a um alvo de consumo de energia. Como implementação base para modificações, escolheu-se o x264.

Por fim, o Capítulo 7 é destinado à apresentação das conclusões, bem com das propostas de trabalhos futuros.

[3]Por *sistema verde* entenda-se sistema a par dos impactos ambientais envolvidos na sua construção e operação.

2 COMPRESSÃO DE IMAGENS E VÍDEO

Este capítulo trata de conceitos básicos de processamento de vídeo digital e dos componentes fundamentais de codificadores de vídeo cuja integração tem contribuído para a popularização de sistemas de comunicações multimeios.

2.1 CONSIDERAÇÕES INICIAIS

A compressão de um sinal pode ser tratada como uma representação mais eficaz que trata melhor as informações redundantes que o mesmo carrega. A fim de se obter tal representação, opta-se por uma de duas classes de técnicas: técnicas de compressão sem perdas e de compressão com perdas. Em um sistema de compressão sem perdas, a redundância estatística é removida de forma que o sinal original possa ser reconstruído perfeitamente no receptor. Contudo, esses métodos apresentam desempenhos modestos quando aplicados diretamente a sinais de imagens ou vídeo. As técnicas de compressão de vídeo amplamente difundidas são baseadas no paradigma da compressão com perdas, em que grande compressão é atingida às custas de perda de qualidade do sinal tratado, cuja versão decodificada agora passa a ser diferente, mas de certa maneira fiel à versão original. Nesse conjunto de técnicas, quanto maior a degradação inserida no sinal, menor será a versão comprimida. O desafio de um algoritmo de compressão de vídeo é comprimir eficientemente minimizando a distorção inerente ao processo.

Este trabalho, por tratar de codificação de vídeo digital de propósito geral, está inserido na classe das técnicas de compressão com perdas, apesar de lançar mão de técnicas de codificação entrópica, conforme será mostrado adiante.

Este capítulo trata de conceitos básicos de processamento de vídeo digital e dos componentes fundamentais de codificadores de vídeo cuja integração tem contribuído para a popularização de sistemas de comunicações multimeios.

2.2 CONCEITOS BÁSICOS DE PROCESSAMENTO DE VÍDEO DIGITAL

Antes de iniciar a apresentação do processo de compressão de um sinal de vídeo digital, é conveniente definir alguns conceitos.

2.2.1 Aquisição de sinais de vídeo digital

Um sinal de vídeo digital é composto por uma sequência de imagens digitais (quadros) capturadas de uma cena real ou geradas sinteticamente. Uma imagem digital de uma cena é resultado de transformação de um sinal multidimensional de parâmetros contínuos do mundo real para uma versão minimamente bidimensional de parâmetros discretos e de amplitudes quantizadas, manipuláveis por sistemas digitais [30, 31].

O processo de transformar parâmetros contínuos em parâmetros discretos é definido como amostragem. Mediante amostragem espacial, a vista (uma primeira projeção do sinal multidimensional em um sistema bidimensional) composta por infinitos pontos passa a ser representada por uma matriz retangular com um número finito de pontos. O processo que limita o domínio dos valores de intensidade para cada um dos pontos amostrados é chamado de quantização. [32]

A fim de adquirir adequadamente a sequência de vídeo, ainda é necessária a realização de amostragem no domínio do tempo que proporcione a mesma sensação de continuidade temporal existente no mundo real quando da posterior exibição consecutiva dos quadros amostrados e quantizados. Assim, o vídeo digital fica representado por amostras espaço-temporais cuja intensidade é composta por um conjunto de números que representam o brilho (luminância) e a cor (crominância) da amostra.

2.2.2 Codificação de cores

A representação de cenas reais por imagens digitais leva em consideração o conteúdo de cores presente na cena, que será posteriormente mostrada em aparato de apresentação. Enquanto imagens monocromáticas necessitam apenas de um valor por amostra espacial para representar o conteúdo do sinal (no caso composto somente por brilho ou luminância), imagens coloridas requerem mais dois valores para representar as informações de crominância.

A maneira mais tradicional de representar uma imagem colorida é por meio do uso do espaço de cores RGB, no qual as informações de crominância e luminância de uma amostra do sinal são codificadas

levando em consideração que cores visíveis podem ser representadas aproximadamente por ponderações de três componentes primárias de cor: vermelho (*Red*), verde (*Green*) e azul (*Blue*) [33]. Logo para cada elemento/amostra da imagem (*pixel*), são necessários três valores para armazenar adequadamente as informações de cores. A simplicidade na captura das cores e na geração das mesmas por dispositivos eletrônicos justifica a popularidade deste tipo de codificação.

O fato de o sistema RGB não levar em conta detalhes do comportamento do sistema visual humano, deixa-o, contudo, em desvantagem em relação a outros espaços de cores. Sabe-se que o olho humano possui menor resolução para informações de crominância do que de luminância. [34] O espaço de cores YCbCr usa essa característica como vantagem, tornando-se mais eficiente ao representar imagens coloridas. Este codifica informações de cores separadamente das informações de luminância, que necessitam de uma maior resolução. Para a determinação das suas componentes, usam-se as seguintes expressões:

$$\begin{aligned} Y &= k_r R + (1 - k_b - k_r)G + k_b B \\ C_b &= \frac{0.5}{1 - k_b}(B - Y) \\ C_r &= \frac{0.5}{1 - k_r}(R - Y) \end{aligned} \tag{2.1}$$

onde k são fatores de ponderação [35] e os valores de R, G, B e Y pertencem ao intervalo $[0,\ 1]$, enquanto C_b e C_r pertencem a $[0,\ 0,5]$. Um dos melhores atributos desse espaço de cores é que as componentes C_r e C_b podem ser representadas em resolução espacial menor que Y devido à supracitada característica do sistema visual humano (HVS). Um dos tipos mais comuns de representação do sinal de vídeo digital denomina-se espaço de cores YUV420 ou, também, YUV12, para o qual a representação de cada uma das componentes de crominância usa metade da resolução espacial em cada dimensão. Dessa forma, enquanto que para cada 4 *pixels* do sistema RGB usam-se 12 amostras de sinal, para o sistema YUV420 usam-se 4 amostras de luminância, 2 duas de crominância do canal azul (C_b ou U) e 2 duas de crominância do canal vermelho (C_r ou V). Isso implica redução da quantidade de dados necessária para codificar cores sem perdas de qualidade visual, por si só uma etapa de compressão do sinal.

2.2.3 Métricas de avaliação da qualidade de sinais codificados

Quantificar a qualidade visual é um problema difícil e impreciso dada a gama de fatores que podem interferir nos resultados. Numa tentativa de dispor de um método objetivo e de baixo custo computacional foram desenvolvidas algumas métricas em cujo cálculo não é levada em consideração a interação com observadores [1].

A métrica mais popular para avaliação da qualidade de vídeos codificados é a PSNR (*Peak Signal to Noise Ratio*), medida em escala logarítmica e calculada a partir da razão entre o quadrado da maior intensidade que o sinal pode assumir (usualmente $(2^n - 1)^2$, onde n é o número de bits usados para codificar uma amostra do sinal) e o erro quadrático médio (*Mean Squared Error* ou MSE) do sinal original para o sinal codificado.

$$PSNR_{dB} = 10 \log_{10} \frac{(2^n - 1)^2}{MSE} \tag{2.2}$$

Sua simplicidade algorítmica é um dos fatores que a tornaram extremamente atrativa, todavia deve-se levar em conta suas desvantagens. Transformações simples aplicadas a uma imagem, como deslocamentos por uma amostra, resultam em valores pobres de qualidade objetiva, apesar de os sinais permanecerem visualmente idênticos.

Ainda na classe de métricas objetivas, estudos recentes sugerem novas metodologias para aferição de qualidade não mais fundamentadas nas técnicas comuns que tipicamente calculam a diferença de intensidade entre o sinal distorcido e o sinal de referência. Essas novas metodologias tentam quantificar a diferença de forma perceptual pela incorporação de propriedades conhecidas do HVS [36]. Novas metodologias de indicação de qualidade têm proposto o emprego da similaridade estrutural entre sinais, onde a estrutura provém da grande dependência que as amostras dos sinais exibem entre si.

Partindo da hipótese de que o HVS é altamente adaptado para extrair essas informações estruturais do campo de visão, Wang et al. propuseram uma medida de similaridade estrutural usando uma abordagem que separa a medida de similaridade na composição das comparações de três grandezas: a luminância, o contraste e a estrutura [37]. Sugere-se a função SSIM (do inglês, *Structural SIMilarity Index*) para a comparação de dois sinais $\mathbf{x}$ e $\mathbf{y}$, definida da seguinte maneira:

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{2.3}$$

em que μ_x é o valor esperado do sinal $\mathbf{x}$ (medida de luminância), σ_x, seu desvio padrão (estimativa de contraste), σ_{xy} a covariância entre os sinais $\mathbf{x}$ e $\mathbf{y}$ e C_1 e C_2 são constantes para estabilizar uma eventual divisão com denominador próximo a zero. Da forma proposta, a Eq. 2.3 satisfaz três condições:

1. Simetria: $\text{SSIM}(\mathbf{x}, \mathbf{y}) = \text{SSIM}(\mathbf{y}, \mathbf{x})$. Como a intenção é quantificar a similaridade entre dois sinais, a troca da ordem dos sinais de entrada não deve afetar a medida de similaridade.

2. Limite de abrangência: $\text{SSIM}(\mathbf{x}, \mathbf{y}) \leq 1$. O fato de a função ter um limite superior é uma propriedade

útil para uma métrica de similaridade pois um supremo pode servir de indicador do quanto os dois sinais estão próximos de serem perfeitamente idênticos.

3. Máximo único: $\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = 1$ se, e somente se, $\mathbf{y} = \mathbf{x}$. Em outras palavras, a medida de similaridade deve quantificar quaisquer variações que possam existir entre os sinais de entrada; o valor máximo será atingido apenas quando os sinais usados na comparação forem idênticos.

O algoritmo de avaliação SSIM surge como opção viável não apenas por sua formulação simples e baixa complexidade computacional de implementação, mas, fundamentalmente, por sua boa precisão na predição da qualidade subjetiva dos sinais [37].

2.3 TÉCNICAS DE COMPRESSÃO DE VÍDEO

Uma vez disponíveis sinais digitais de vídeo, é possível fazer o uso de diversas técnicas de compressão a fim de representá-los convenientemente. A primeira ideia ao se pensar em comprimir sinais de vídeo é comprimir cada quadro da sequência separadamente usando técnicas de compressão de imagens.

Uma grande evolução dos métodos de compressão de imagens foi observada no início da década de 70, com a publicação de trabalhos sobre a aplicação de transformadas de blocos [38, 39, 40] às imagens e o posterior tratamento e codificação dos coeficientes transformados, lançando mão de características do HVS [4].

Caso de sucesso em padronização, o codificador JPEG [41] lança mão extensivamente das abordagens apresentadas nesses trabalhos seminais e é usado aqui como paradigma para a apresentação dessa classe de técnica de tratamento de figuras. Seu fluxo de operação inicia-se com a fragmentação da imagem em pequenos blocos de tamanhos iguais; para cada bloco, aplica-se a transformada discreta de cosseno (*Discrete Cosine Transform* ou DCT).

2.3.1 A Transformada Discreta de Cosseno (DCT): viabilização da compressão

A DCT [40, 42] é uma transformada muito usada em padrões de compressão de imagem e vídeo devido à sua eficiência na compactação de energia. Por capacidade de descorrelação entende-se a habilidade de uma transformada converter um conjunto de dados altamente correlacionados em outro conjunto de dados relativamente independentes (reduzindo a redundância estatística) enquanto eficiência em compactação de

energia refere-se à habilidade de uma transformada compactar o conteúdo energético de um sinal na menor quantidade possível de coeficientes. O fato de a DCT ser uma transformada independente de dados e a disponibilidade de implementações rápidas são argumentos que justificam sua popularidade [43].

Para um sinal de $N \times N$ amostras, tomado aqui como uma fração (bloco) de um quadro a ser comprimido, os coeficientes de sua matriz de transformação **C** são obtidos por funções de cossenos [4], conforme verificado na Equação 2.4:

$$[\mathbf{C}]_{ij} = \begin{cases} \sqrt{\frac{1}{N}} \cos \frac{(2j+1)i\pi}{2N} & i = 0;\ j = 0, 1, .., N-1 \\ \sqrt{\frac{2}{N}} \cos \frac{(2j+1)i\pi}{2N} & i = 1, 2, ..., N-1;\ j = 0, 1, .., N-1. \end{cases} \tag{2.4}$$

O resultado da aplicação da DCT em um bloco de $N \times N$ amostras é um conjunto de $N \times N$ coeficientes representando o bloco no domínio transformado, coeficientes que podem ser considerados como ponderações para um conjunto de matrizes de base, ilustrado na Figura 2.1[1] para o caso em que $N = 8$. Dessa forma, a representação no domínio transformado pode ser encarada como a representação do sinal pela combinação de todas as $N \times N$ matrizes de base, cada qual multiplicada por seu fator de ponderação apropriado [1].

2.3.2 Quantização e Codificação de Entropia: realização da compressão

A grosso modo, a finalização da compressão de um bloco da imagem usando JPEG consiste na aplicação de processo de quantização aos coeficientes transformados pela DCT. Essa etapa é responsável pela perda seletiva de informações no processo de compressão pela exploração de características do HVS como diferentes sensibilidades na percepção das frequências espaciais [4]. No término do processo, os coeficientes quantizados são devidamente organizados (varredura em "zig-zag") [39] e passam por processo de codificação de entropia.

Apesar de bastante simples quando comparado ao conjunto de etapas que contemplam o fluxo de execução de um codificador moderno, o método descrito anteriormente ainda é encontrado em codificadores de vídeo considerados estado da arte em tecnologia e é comumente denominado de processo de codificação **Intra-quadro**, uma vez que as informações usadas na compressão do sinal não dependem de quadros anteriores. O próprio codificador o JPEG2000 [45], sucessor do JPEG e considerado estado da arte na compressão de imagens estáticas, é tomado como alternativa na codificação de vídeo para aplicação

[1] Adaptado de [44].

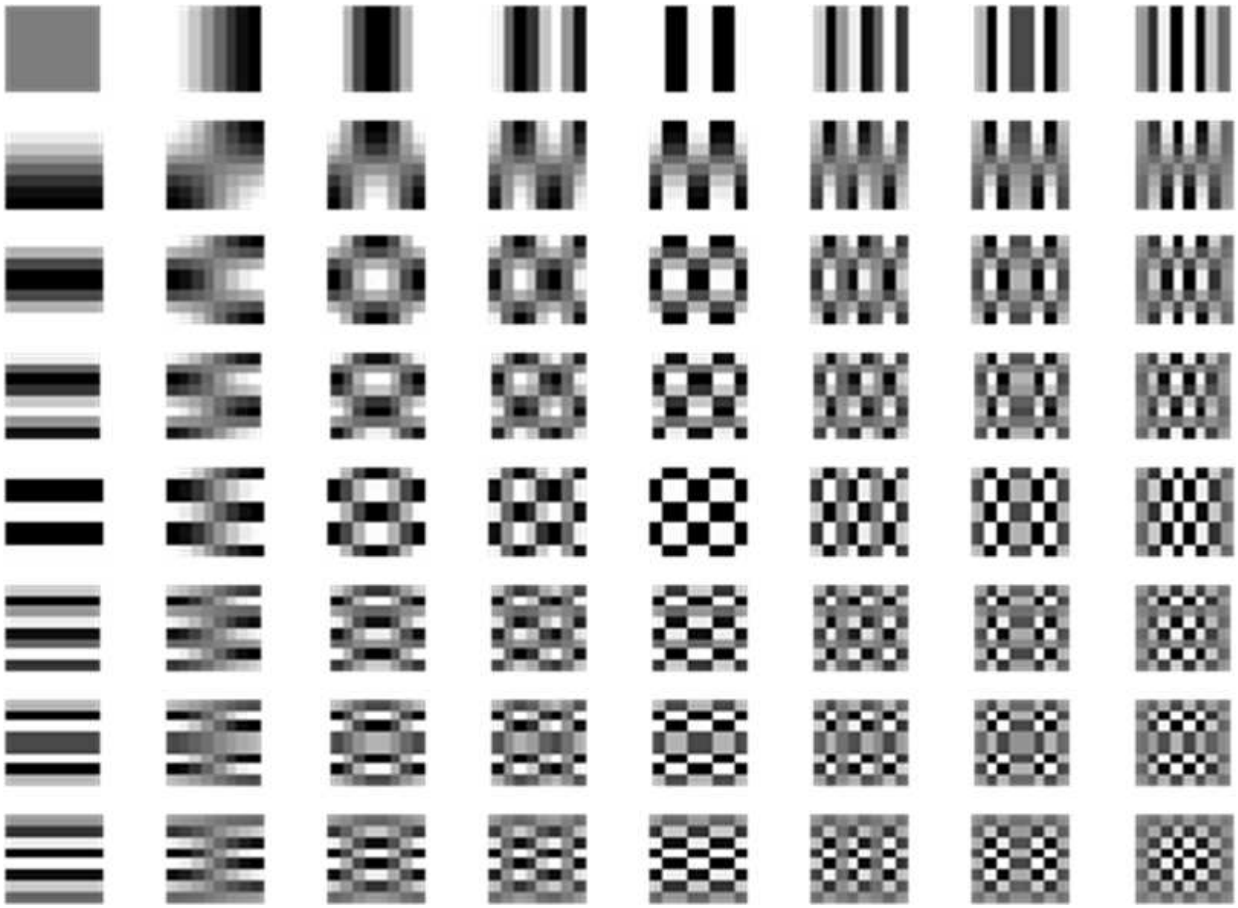
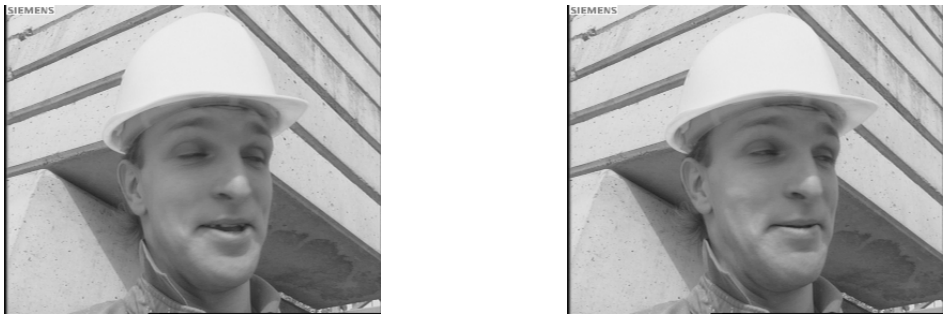


Figura 2.1: Bases da DCT para blocos de 8×8 *pixels*.

de cinema digital [46]. Todavia, um grande desempenho em compressão ainda pode ser obtido se levarmos em consideração a redundância temporal em um sinal de vídeo.

2.3.3 Explorando a dimensão temporal na codificação de quadros de vídeo



(a) (b)

Figura 2.2: Dois quadros consecutivos da sequência “Foreman”.

Uma rápida observação da Figura 1.1, repetida aqui na Figura 2.2, permite-nos perceber que muito do conteúdo visual da cena é repetido entre quadros consecutivos sem mudanças significativas. Uma maneira mais eficiente de representar o sinal de vídeo seria pela codificação somente das mudanças no conteúdo de vídeo em vez de codificar o quadro completamente. O conjunto de técnicas que exploram redundâncias temporais, usualmente denominado codificação **Inter-quadros**, é responsável pelo grande ganho de codificação que distingue um codificador de vídeo de um codificador de imagens aplicado sequencialmente. Ainda no começo da década de 70, o trabalho de F. W. Mounts [47] já analisava o uso das dependências temporais entre os quadros para a redução na largura de banda usada para transmitir os sinais de vídeo.

No processo de codificação Inter-quadros, usualmente se fragmenta o quadro em pequenos blocos de igual tamanho e verifica-se o sinal de diferença entre o bloco do quadro atual e o bloco que ocupa a mesma posição no quadro anterior. O sinal de diferença, denominado resíduo, deve ser encaminhado ao decodificador de forma que esse seja capaz de reconstruir o quadro codificado. Tal método pode ser refinado se uma análise mais detalhada da correlação existente entre os quadros for realizada por meio de uma técnica de predição chamada **compensação de movimentos**.[2]

[2] As pesquisas seminais [48, 49, 50] já apresentavam as vantagens da compensação de movimentos em blocos na remoção de redundâncias temporais.

A maioria das mudanças existentes entre os quadros de vídeo é devida ao movimento de objetos da cena em relação a um fundo estático. Pequenas quantidades de movimento podem resultar em grandes diferenças entre blocos co-localizados em quadros consecutivos. Tipicamente, a busca de um bloco feita numa janela ao redor da posição do bloco no quadro anterior, chamado de quadro de referência, pode reduzir significativamente a quantidade de informação necessária para codificar adequadamente o quadro. O uso de deslocamentos espaciais como forma de aproximação do sinal é denominado **compensação de movimentos** e ao processo de busca pelo melhor casamento entre blocos do quadro atual e dos anteriores dá-se o nome de **estimação de movimentos**. Nesse caso, além das diferenças, também chamadas de resíduo de predição, é necessário o envio de informações que indiquem qual foi o bloco do quadro anterior usado na aproximação. A essa informação lateral de deslocamento dá-se o nome de **vetor de movimento**.[3]

Na Figura 2.3 é apresentado um campo vetorial cujos vetores indicam de qual bloco (aqui de tamanho de 8×8 *pixels* ou pontos elementares de imagem) do quadro anterior provém o melhor casamento com o bloco do quadro atual. Na Figura 2.4 são comparadas a versão original do quadro atual (2.4(b)) e sua versão predita a partir de compensação de movimentos (2.4(c)) em relação ao quadro anterior (2.4(a)). Observa-se ao comparar as Figuras 2.4(d) e 2.4(e) como a energia do resíduo entre o quadro e sua versão predita por compensação de movimentos é bem menor do que o resíduo entre o mesmo quadro e o subsequente. É esse resíduo, juntamente com os vetores de movimento, que precisa ser enviado ao decodificador para garantir a reconstrução do sinal de vídeo do outro lado do canal. A fim de tirar o máximo proveito das técnicas de compressão, faz-se conveniente a aplicação de técnicas de codificação de imagens quando do tratamento do resíduo de compensação de movimentos.

2.3.4 Codificação Entrópica Revisitada

Em um codificador de vídeo, os parâmetros extraídos pelo processo de compressão (coeficientes quantizados, vetores de movimento etc.) precisam ser representados eficientemente e sem erros de forma que o decodificador possa recompô-los e decodificar o sinal tal qual foi reconstruído no codificador. Para comprimir essas informações sem perdas, usa-se codificação de entropia. As técnicas de codificação entrópica apresentam bons resultados para fontes de informação sem memória (fontes cujo valor de cada amostra não apresentam dependência entre si) minimizando a taxa de bits necessária para a codificação pela associação de códigos de comprimento variável para amostras de entrada de acordo com a função de

[3]No começo da década de 80, um codificador de vídeo usando predições por compensação de movimentos é proposto por Koga et al. [51].

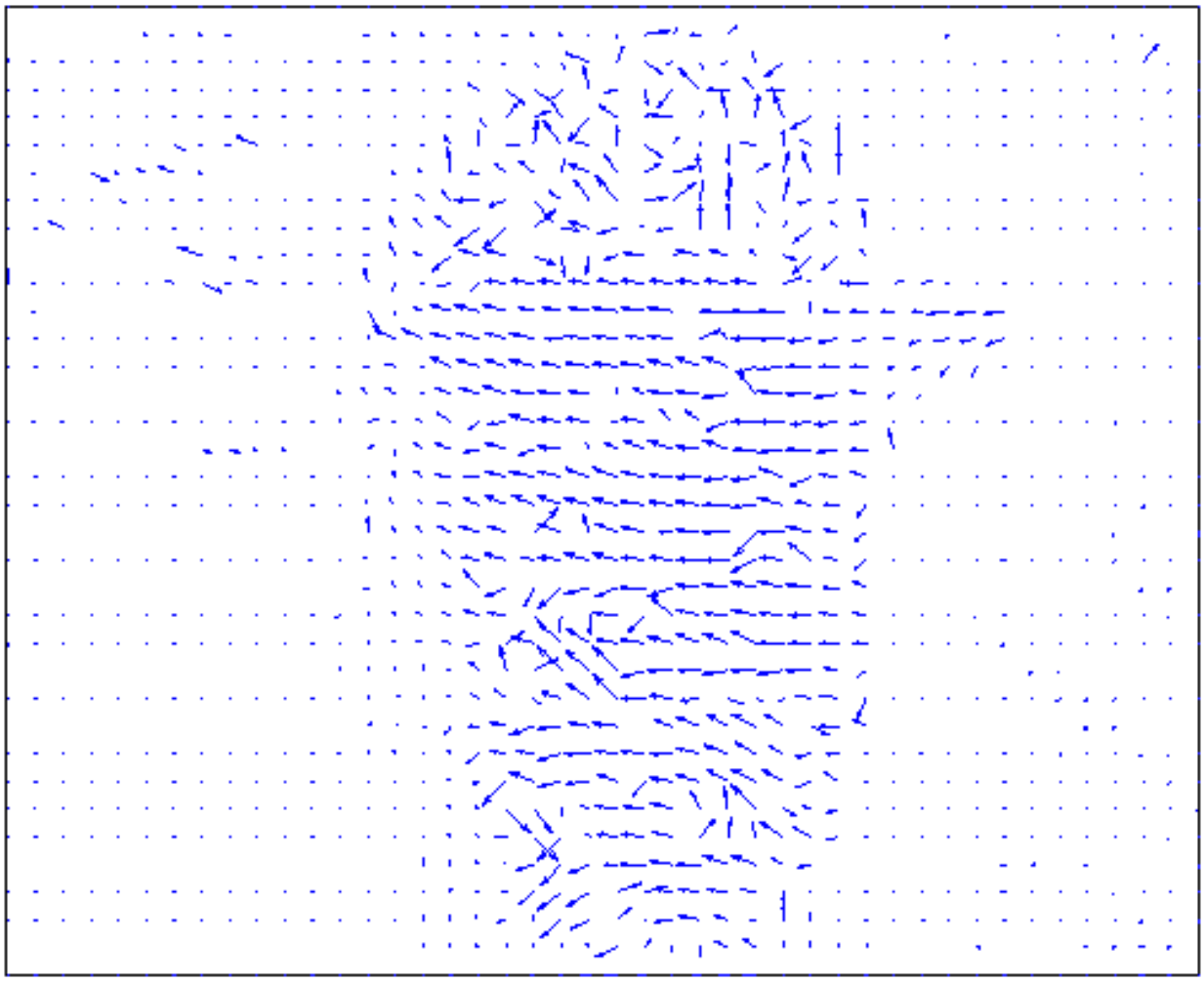


Figura 2.3: Fluxo óptico composto por vetores de movimento para cada bloco de 8×8 *pixels* da imagem da Figura 2.2(b) com relação à Figura 2.2(a).

(a)

(b)

(c)

(d)

(e)

Figura 2.4: Quadro (a) da Figura 2.2(a), seu sucessor (b)(Figura 2.2(b)) e versão compensada (c) do quadro (b) com relação ao quadro (a). O quadro de diferenças entre (b) e (a) é apresentado em (d), onde valores muito claros ou muito escuros indicam grandes diferenças entre as intensidades dos *pixels*. O quadro de diferenças (e), também chamado de resíduo de compensação de movimentos, mostra o efeito da compensação de movimentos. Após a aplicação do fluxo óptico da Figura 2.3, os valores extremos de intensidades são menos frequentes.

densidade de probabilidade (fdp) das amostras[4]. O princípio básico de operação é simples: para as amostras mais recorrentes usam-se símbolos mais curtos enquanto às amostras menos frequentes são associados símbolos mais longos em bits [4].

Assumir que fontes de informação não possuem memória é uma abordagem válida, porém simplista. A fim de extrair o máximo da redundância existente em dados a serem comprimidos, é possível conjugar a codificação entrópica a codificadores preditivos, que tratam mais adequadamente fontes com memória, ou seja, fontes em que cada amostra apresenta dependência estatística em relação às amostras adjacentes. Essa classe de codificadores reduz a entropia e é aplicada em larga escala em modernos compressores de sinais com perdas [52, 1].

2.3.5 Codificador de Vídeo DPCM Híbrido

Ao agregado de técnicas de estimação/compensação de movimentos, de codificação de imagens e de entropia empregado em um codificador de vídeo dá-se o nome de codificação de vídeo híbrida [51]. As Figuras 2.5(a) e 2.5(b)[5] apresentam, respectivamente, diagramas de um codificador e de um decodificador DPCM [4] híbrido baseado em DCT.

O codificador do tipo DPCM é uma estrutura muito usada no tratamento de fontes que apresentam elevado grau de correlação entre as amostras. Ele procura explorar a correlação para prever novas amostras a partir daquelas já comprimidas e codificar e transmitir apenas as diferenças entre a predição e o novo valor. O decodificador DPCM deve "imitar" (Seção 2.3.5.2) o codificador para gerar predições idênticas e adicioná-las às diferenças enviadas pelo codificador, resultando nas amostras decodificadas.

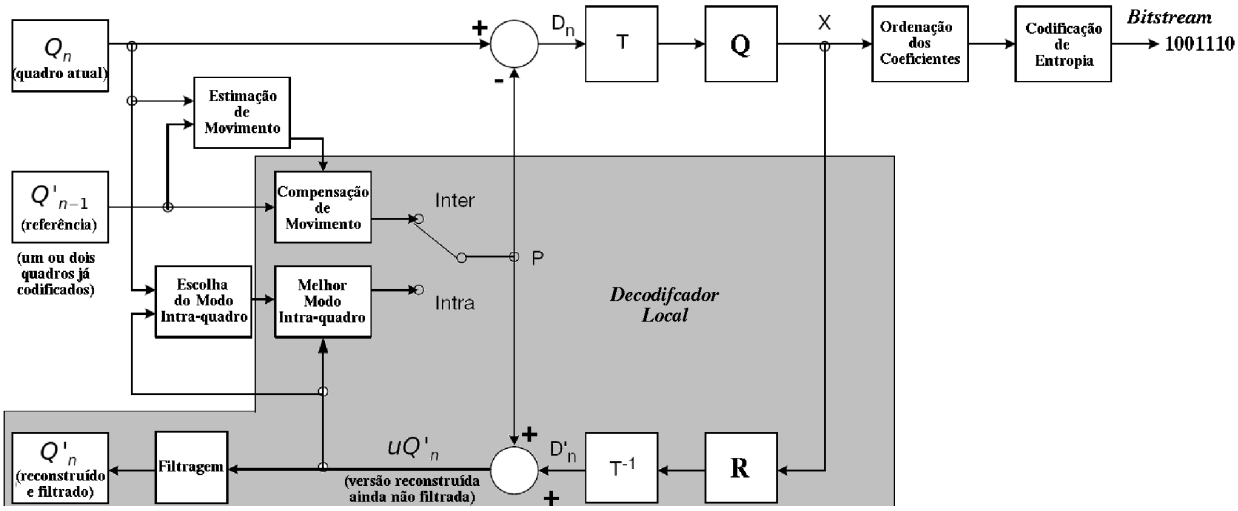
No codificador da Figura 2.5(a), um quadro de vídeo (Q_n) é processado para produzir a sequência de bits (*bitstream*) comprimido e, no decodificador (Figura 2.5(b)), o *bitstream* comprimido é decodificado para produzir o quadro reconstruído (Q'_n) ligeiramente diferente do quadro original. Nas figuras estão realçados elementos comuns ao codificador e decodificador.

2.3.5.1 Fluxo de dados do codificador

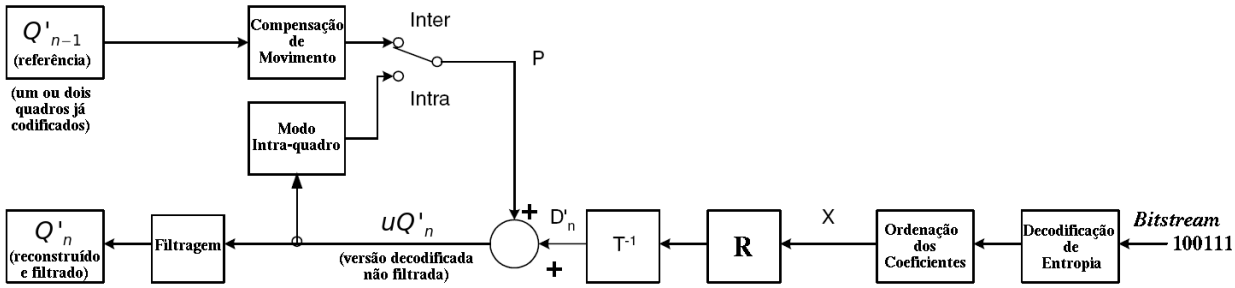
Há dois fluxos principais de dados no codificador: o caminho de codificação e o caminho de decodificação. Os passos de codificação são os seguintes:

[4] Assume-se aqui codificação de entropia de ordem zero.

[5] Adaptado de [1].



(a)



(b)

Figura 2.5: Codificador (a) de vídeo DPCM híbrido e seu respectivo decodificador (b).

1. Um quadro de entrada Q_n entra no codificador e é processado em pequenas frações, chamadas macroblocos, tomadas aqui como blocos com 16×16 *pixels*.

2. Q_n é comparado com um quadro de referência que pode ser o quadro codificado anteriormente (Q'_{n-1}). Uma função de estimação de movimentos encontra uma região de 16×16 amostras em Q'_{n-1} que casa com o macrobloco de Q_n que está sendo codificado no momento. O deslocamento entre as posições do macrobloco atual e o de melhor casamento do quadro de referência é o vetor de movimento VM.

3. A partir do vetor de movimento, uma predição P por compensação de movimentos é gerada: predição Inter-quadros. É possível gerar o sinal P a partir de extrapolação espacial dos seus vizinhos contidos numa janela causal: predição Intra-quadro.

4. A melhor predição encontrada pelos processos Intra ou Inter-quadros é tomada (P) e subtraída do macrobloco atual para produzir o macrobloco residual D_n.

5. D_n é partido em blocos menores e passa por um processo de transformação, por exemplo transformado pela DCT.

6. A versão transformada de D_n é quantizada (X), ou seja, os coeficientes transformados passam por um mapeamento em que o intervalo de definição de seus valores é reduzido para um conjunto finito menor e que requer menos bits para ser representado. É nesta etapa que ocorrem as perdas irreversíveis do processo de codificação.

7. Os coeficientes de cada bloco são rearranjados e submetidos à codificação de entropia juntamente com informações a respeito do tipo de predição empregado, seus parâmetros (vetores de movimento ou modo de predição espacial) e informações de cabeçalho, produzindo o *bitstream* comprimido.

Os passos de reconstrução são:

1. Cada macrobloco quantizado X é re-escalonado (R), num processo em que os valores de seus coeficientes transformados são mapeados de volta para o intervalo de definição original. Em seguida, é submetido à transformada inversa da DCT para obtenção do resíduo decodificado D'_n. Note que o processo de quantização não é reversível, o que significa que D'_n não será idêntico a D_n. A diferença entre essas duas versões é a causa da **distorção** no processo.

2. Independentemente do tipo de predição empregado, Intra-quadro ou Inter-quadros, o sinal predito P é adicionado ao resíduo D'_n para produzir um macrobloco reconstruído e os macroblocos reconstruídos são salvos para a montagem do quadro reconstruído Q'_n.

3. Antes de ser exibido e guardado na memória de quadros de referência, o sinal ainda passa por um processo de filtragem de tratamento de efeitos de quantização e de análise em blocos, o que leva à obtenção do quadro filtrado Q'_n.

Uma vez terminada sua codificação completa, o quadro Q'_n pode ser usado como quadro de referência para o próximo quadro da sequência de vídeo, Q_{n+1}. Note que esta estrutura é muito semelhante à do decodificador, o que justifica sua comum denominação de decodificador local.

2.3.5.2 Fluxo de dados do decodificador

Por sua vez, no decodificador há apenas um fluxo de dados, descrito a seguir:

1. O *bitstream* comprimido é decodificado entropicamente para a extração dos coeficientes, parâmetros de predição (vetores de movimento ou direção de predição espacial) e cabeçalhos de cada macrobloco. Os coeficientes são rearranjados para produzir a versão quantizada do macrobloco, X.

2. X é re-escalonado e a ele é aplicada a transformada inversa para gerar o resíduo decodificado D'_n.

3. O vetor de movimento decodificado é usado para localizar a região de 16×16 amostras na cópia do quadro de referência Q'_{n-1} presente no decodificador. Essa região é a predição por compensação de movimentos P para o caso Inter-quadros. Caso a predição sinalizada seja Intra-quadro, é realizada a composição do sinal P pela extrapolação espacial a partir de vizinhos em uma janela causal de codificação.

4. O sinal predito P é adicionado ao resíduo D' para produzir o macrobloco reconstruído e os macroblocos reconstruídos são armazenados para a montagem do quadro reconstruído uQ'_n.

5. O quadro reconstruído uQ'_n passa por processo de filtragem para tratamento de efeitos da análise do sinal em blocos e demais efeitos de quantização e, em seguida, é inserido no *buffer* de quadros reconstruídos, usados na exibição e como referência para a decodificação de outros quadros.

Depois que o quadro for completamente decodificado, Q'_n estará pronto para exibição e poderá ser armazenado como quadro de referência para o próximo quadro a ser decodificado, Q'_{n+1}.

2.4 O CODIFICADOR H.264/AVC

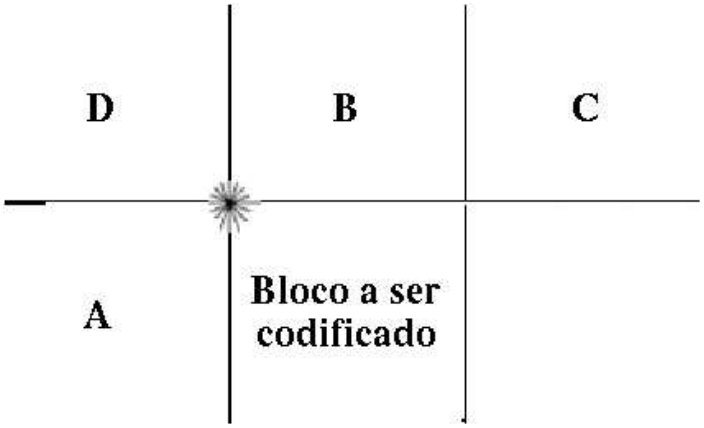
H.264/AVC é considerado como um dos codificadores estado da arte em codificação de vídeo. Ele foi desenvolvido em trabalho conjunto entre o *Video Coding Experts Group* (VCEG, ITU-T) e o *Moving Picture Experts Group* (MPEG, ISO/IEC) [13]. Esse padrão apresenta uma série de melhoramentos [53] em relação a padrões anteriores que resultam em ganhos de codificação para uma vasta gama de aplicações, entre elas a vídeo conferência, TV digital, cinema digital e transmissão de vídeo. A seguir serão apresentados os refinamentos em relação à estrutura de codificador híbrido agregados ao H.264/AVC.

2.4.1 Predição Intra-quadro

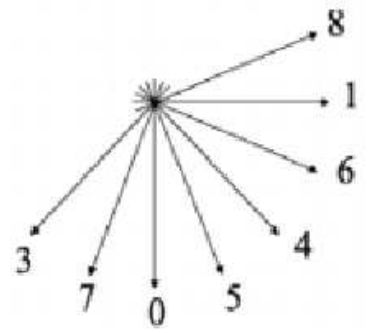
A predição Intra-quadro é a técnica pela qual amostras do sinal de vídeo agrupadas em macroblocos de 16×16 *pixels*, tomados no H.264/AVC como unidade básica de codificação, são preditos usando informações de macroblocos já codificados e pertencentes ao mesmo quadro. Nos perfis mais básicos do H.264/AVC (vide seção 2.4.6) existem inicialmente dois tipos de predição Intra-quadro para as componentes de Y luminância: INTRA_4×4 e INTRA_16×16. Com o advento do adendo FRExt (*Fidelity Range Extensions*) [16], foi acrescentado mais um tipo de predição, INTRA_8×8, para o perfil *High*.

Os tipos Intra_4×4 e Intra_8×8 particionam o macrobloco em blocos menores de tamanhos 4×4 ou 8×8 *pixels*, respectivamente. Para cada um desses blocos, são disponibilizados nove modos de predição em que técnicas espaciais são empregadas. Um modo de predição é o DC, no qual todas as amostras de cada sub-bloco de 4×4 são preditas pela média das amostras vizinhas à esquerda (A) ou acima (B, C e D) do bloco atual (veja Figura 2.6(a)) e que já tenham sido reconstruídas no codificador e no decodificador. Os outros oito modos exploram direções para predição planar, mostradas na Figura 2.6(b), também a partir da referida janela causal.

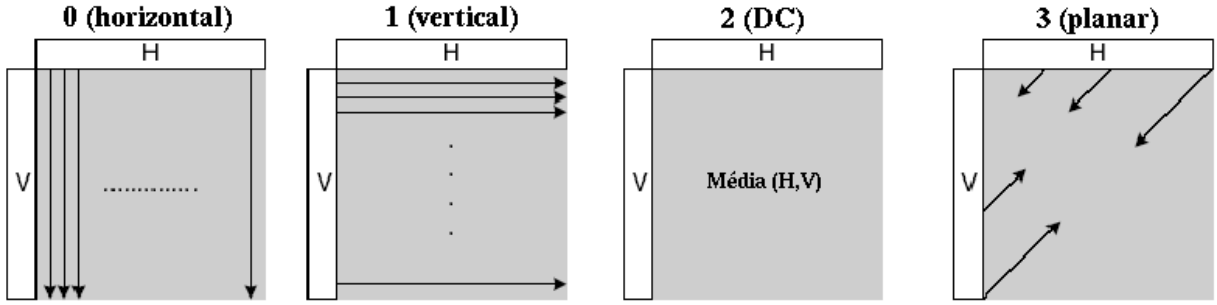
No tipo INTRA_16×16, o macrobloco completo é predito por apenas um modo. Na Figura 2.6(c) são apresentados os quatro modos suportados: predição vertical, predição horizontal, predição DC e predição planar. A predição Intra-quadro por macrobloco completo é muito eficiente quando o sinal a ser codificado apresenta variações suaves.



(a)



(b)



(c)

Figura 2.6: Janela causal (a) para codificação e modos de predição para codificação Intra-quadro com suas direções de interpolação planar para blocos 4×4 e 8×8 (b) e 16×16 (c). O modo 2, não apresentado em (b), é o DC.

Quando um quadro é inteiramente codificado por meio do uso de técnicas de predição Intra-quadro, ele é chamado de quadro I.

2.4.2 Predição Inter-quadros

Quando é mais interessante usar informações de quadros anteriores na geração da predição de um macrobloco, a compensação de movimentos é empregada como técnica para prever o sinal a partir de quadros já transmitidos, denominados quadros de referência.

Um dos grandes avanços do H.264/AVC em relação a outros padrões vem do fato de que cada macrobloco pode ser dividido em blocos menores, chamados de partições. Os tamanhos possíveis de partição são: 16×16, 16×8, 8×16 e 8×8 *pixels*; a partição de 8×8 pode ainda ser subdividida em partições menores de 8×4, 4×8 e 4×4 *pixels*. As partições são ilustradas na Figura 2.7.

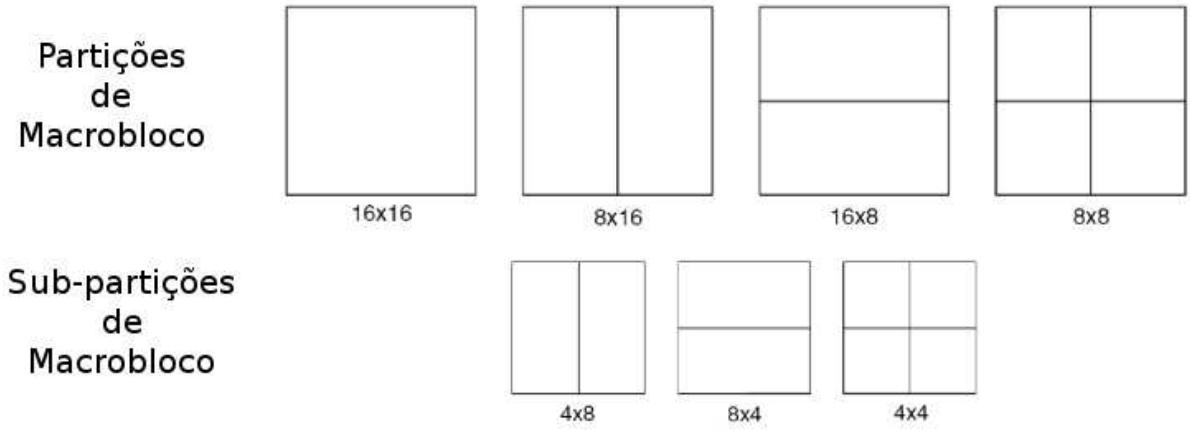


Figura 2.7: Partições de macroblocos.

2.4.2.1 Estimação de Movimentos

A maneira como foram propostas as partições de macrobloco da Figura 2.7 permite estimação de movimento refinada, pois movimentos de objetos complexos ficam melhor mapeados pela maior variedade de tamanhos e formatos dos blocos. Além disso, os tamanhos utilizados podem ser convenientemente representados em uma estrutura hierárquica denominada *quadtree*, ilustrada na Figura 2.8[6].

A precisão no cálculo dos vetores de movimento é outro aspecto bem explorado no H.264/AVC, em

[6] Adaptado de [44].

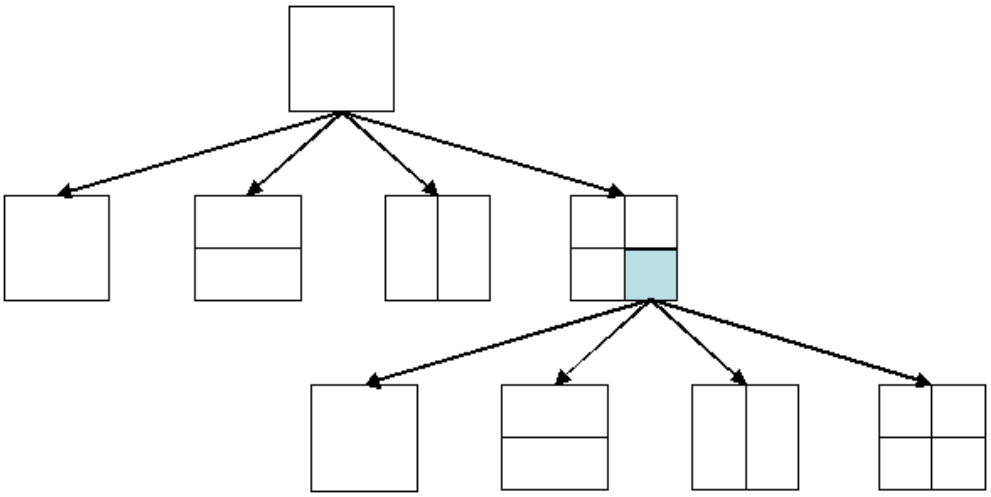


Figura 2.8: Organização das partições em uma estrutura hierárquica.

que os vetores de movimento são calculados com precisão de até um quarto de *pixel* (1/4-*pel*). Vetores de movimento com resolução fracionária referem-se a posições dos quadros de referência que estão situadas entre as amostras do quadro digital. Para gerar vetores de movimento fracionários empregados na compensação, o sinal do quadro de referência precisa ter preenchidas as posições sub-*pixel* por meio de interpolação.

Quanto às técnicas de busca pelo melhor casamento durante a estimação de movimentos, cabe ressaltar duas técnicas que serão referenciadas ao longo do texto: FastFullSearch e UMHS ou UMHexagonS [54].

A primeira técnica, FastFullSearch, usa do fato de que a menor partição (de um plano de luminância de um quadro) utilizada na estimação de movimentos, de acordo com a recomendação H.264/AVC, possui 4×4 *pixels* de tamanho. O artifício proposto sugere que a estimação de movimentos completa (restrita a um tamanho de janela) seja realizada em um quadro apenas em blocos de 4×4 *pixels* e que os resultados de estimações de movimentos para partições maiores sejam derivados da composição de unidades elementares de 4×4 *pixels*. Por exemplo, com os vetores de estimação de movimentos para blocos de 4×4 *pixels*, é possível derivar as estimações de movimentos para blocos de 8×8 *pixels* por meio do agrupamento dos resíduos, para determinado vetor/direção, dos 4 blocos de 4×4 *pixels* constituintes do bloco de 8×8 *pixels*. Isso evita, portanto, a repetição do processo custoso de busca para todas as partições que podem ser montadas por meio de blocos de 4×4 *pixels*.

A UMHS ou UMHexagonS (*Hybrid Unsymmetrical-cross Multi-hexagon Grid Search*) foi adotada pelo JVT devido a seu excelente desempenho em termos de RD quando comparado com a FastFullSearch

enquanto, ao mesmo tempo, reduz consideravelmente o tempo gasto em estimação de movimentos [54]. O diagrama de blocos para o algoritmo da UMHS é apresentado na Figura 2.9.

Há, basicamente, três passos na UMHS. Segue, abaixo, a explicação de cada uma das etapas intermediárias deste algoritmo, com suas respectivas associações no fluxograma da Fig. 2.9 destacadas em itálico.

Passo 1) **Determinação do vetor de movimento de partida:** O vetor de movimento de partida é calculado pela melhor predição em termos de RD, determinada pela avaliação de quatro candidatos: predição por mediana espacial (mediana dos vetores de movimento dos blocos vizinhos espacialmente: *Checar a posição predita por Mediana*); predição pela camada superior (valores de vetores de movimentos para partições maiores, no mesmo macrobloco); predição por quadros de referência vizinhos (no caso de múltiplas referências, usa-se o vetor de movimento de uma referência escalonado para prever outros vetores de movimentos em diferentes quadros de referências) e da posição do bloco co-localizado no quadro anterior — vetor de movimento $(0, 0)$ — (*Checar outras posições preditas*) [13]. Um bom ponto de partida é fundamental para o sucesso do processo de busca, e essas predições tentam derivar uma localização perto do ponto ótimo (o que levará ao menor custo em termos de RD de codificação do bloco). Se isso não acontecer, é provável que visitas desnecessárias sejam realizadas na busca.

Passo 2) **Busca segundo padrões de varredura:** Aplica-se o padrão de busca *Unsymmetrical-cross search* (o qual inclui, além da *Busca em cruz*, uma *Busca em vizinhança quadrada de 5×5 pixels*) e o padrão *Uneven multi-hexagon-grid search* (o qual consiste de varredura em múltiplas malhas de 16 pontos com formato hexagonal: *Busca Hexagonal com 16 pontos em laço*) a partir do ponto de início derivado da aplicação do melhor vetor de movimento encontrado no passo anterior. Os padrões são ilustrados na Figura 2.10. Cada um desses padrões de busca possui um conjunto independente de pontos de visita, o que significa que cada grupo de pontos pode ser calculado em paralelo assim que o ponto de partida central tenha sido definido.

Passo 3) **Etapa de *Refinamento*:** Esta etapa consiste em dois sub-passos de varredura, compostos por um pequeno padrão de busca hexagonal e um pequeno padrão de busca em forma de diamante. Esses padrões são aplicados ao redor do ponto ótimo em termos de RD encontrado no passo anterior. O ponto ótimo atual é posto como ponto de partida depois de se processar cada padrão de varredura

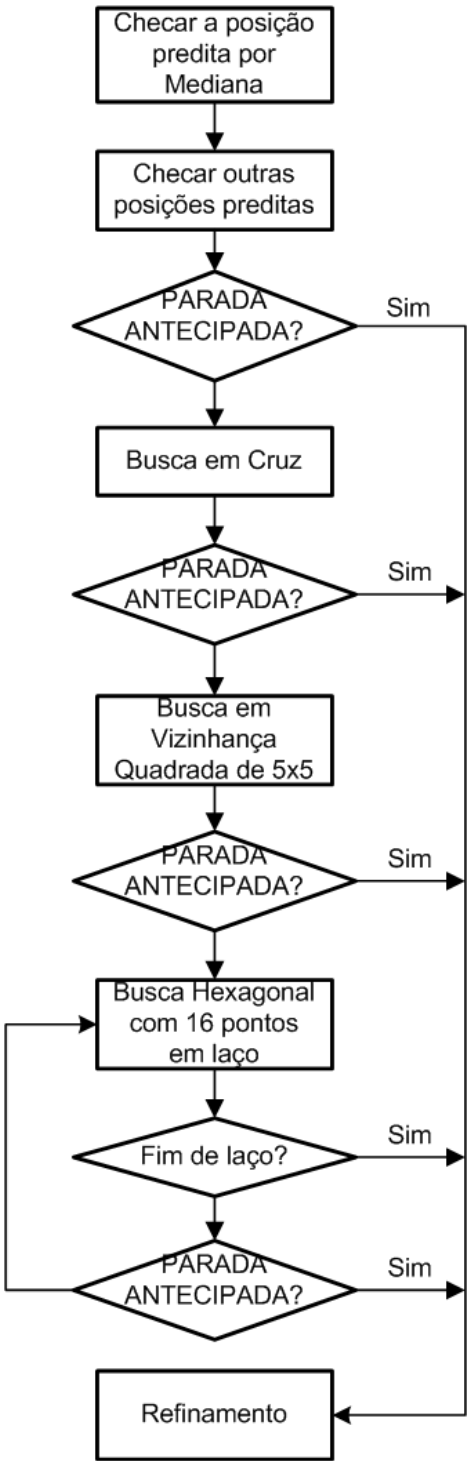


Figura 2.9: Fluxograma do algoritmo da estimação de movimentos UMHS.

e o processo de busca continuará, recursivamente, até que o ponto de partida (ponto central) seja também considerado o ponto ótimo de toda a janela de varredura (convergência).

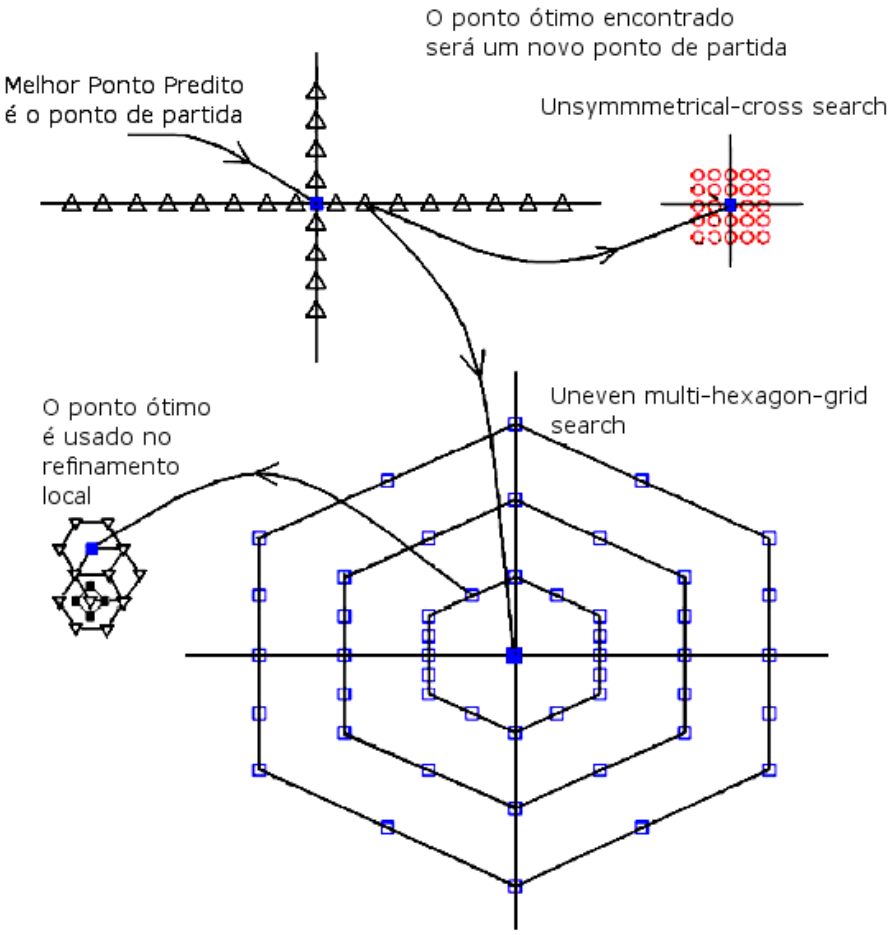
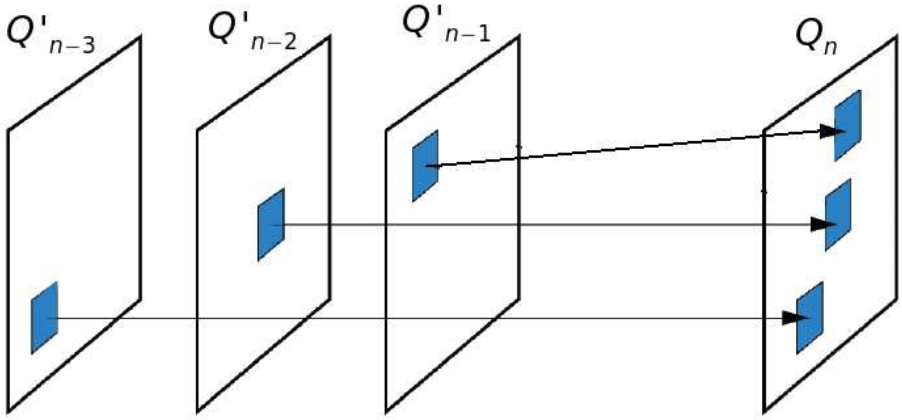


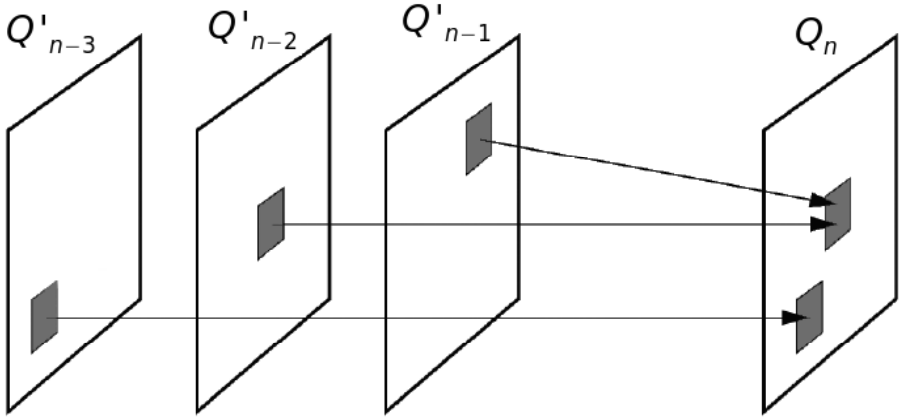
Figura 2.10: Esquema de varredura do algoritmo da estimação de movimentos UMHS, com a ilustração da sucessão de etapas em seus dois padrões de busca (*Unsymmetrical-cross search* e *Uneven multi-hexagon-grid search*) e no estágio de refinamento.

2.4.2.2 Compensação com Múltiplos Quadros de Referências

No que se refere a partições de um quadro, padrões anteriores permitiam apenas um número limitado de opções, por exemplo, 16×16 e 8×8 *pixels* no caso do H.263[10] ou MPEG-4[11]. Vetores deslocamento eram estimados e transmitidos para cada um desses blocos e referiam-se à posição da região de melhor casamento com o bloco atual em um quadro de referência, usualmente o último quadro reconstruído. No H.264/AVC, é possível fazer referência a várias imagens precedentes simultaneamente, sendo necessário



(a)



(b)

Figura 2.11: Compensação de movimento com múltiplos quadros de referência em quadros P (a) e em quadros B (b).

o envio de informações dos quadros de referência empregados juntamente com os vetores de movimento ao decodificador. A esta técnica dá-se o nome de predição por compensação de movimento com múltiplos quadros de referência, a qual é ilustrada nas Figuras 2.11(a) e 2.11(b) para quadros P e B respectivamente.

Chamamos quadros P os quadros codificados usando apenas uma referência para predição de cada macrobloco (Figura 2.11(a)); por sua vez, quadros B usam múltiplas referências simultaneamente na compensação de movimentos para cada macroblocos (Figura 2.11(b)) e também podem ser referidos por quadros bi-preditos. Comparado a padrões anteriores de compressão de vídeo, o conceito clássico de quadros bi-preditos é generalizado no H.264/AVC ao se permitir uma combinação linear de duas predições por meio da aplicação de pesos arbitrários e deixar em aberto a direção das predições temporais. Com isso, é possível explorar melhor as correlações temporais entre os quadros, rearranjando-os de maneira conveniente e possibilitando que a ordem de codificação dos quadros seja diferente da ordem de exibição.

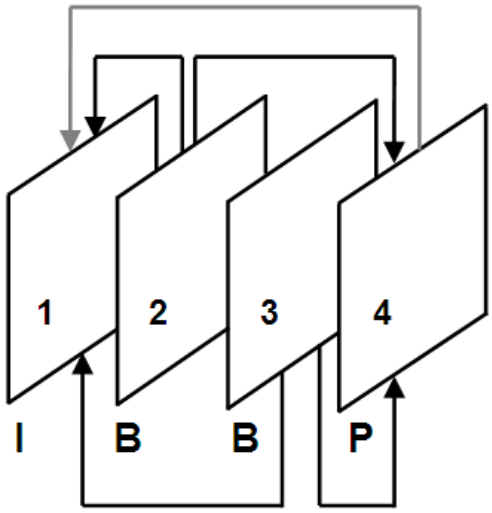
Um exemplo da diferença entre a sequência de codificação e a de exibição é apresentado na Figura 2.12(a)[7], cujo quadro 4 é do tipo P por empregar compensação de movimentos em sua codificação e usar como quadro de referência o quadro 1, quadro I previamente codificado. Enquanto isso, os quadros 2 e 3, do tipo B, usam os quadros 1 e 4 simultaneamente para geração de suas predições. Note que os quadros 2 e 3, por dependerem de versões codificadas dos quadros 1 e 4, são codificados posteriormente ao quadro 4, ilustrando o fato de a ordem de codificação não ser necessariamente ditada pela ordem de exibição. A ordem de codificação para o caso da Figura 2.12(a) é apresentada na Figura 2.12(b).

2.4.3 Codificação por transformadas

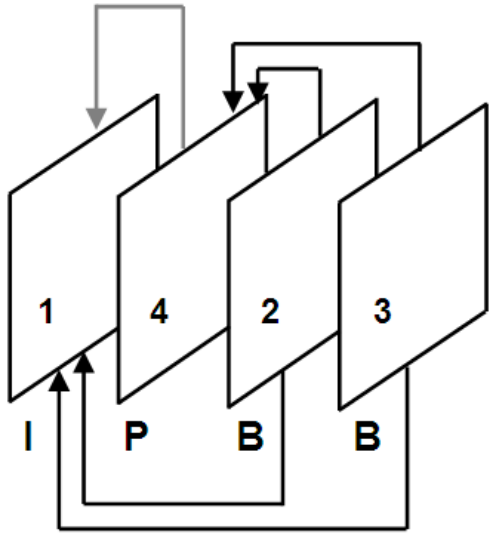
Seguindo a tendência dos padrões de compressão anteriores, a codificação por transformadas continua sendo empregada para viabilizar a codificação do sinal residual por meio da redução da redundância espacial existente no erro de predição. Em vez de aplicar a DCT bidimensional em blocos de 8×8 *pixels*, como feito em padrões como MPEG-1[7] e MPEG-2[8], no H.264/AVC são usadas transformadas inteiras reversíveis com tamanhos variados de 8×8, 4×4 a 2×2[8] amostras. A possibilidade do emprego de transformadas menores que a DCT de 8×8 permite ao compressor codificar melhor o resíduo de predição no caso em que são selecionadas partições de tamanhos menores que 8×8, o que ocorre frequentemente na estimação de movimento de bordas de objetos, além de conseguir compactar melhor o sinal devido à menor variância esperada em pequenas regiões de um quadro digital de uma cena real.

[7] Adaptado de [44].

[8] Transformadas de suporte 2×2 são usadas no H.264/AVC para a compressão de sinais de crominância.



(a)



(b)

Figura 2.12: Quadros de compressão de vídeo ordenado de acordo com a sequência de exibição (a) e de acordo com a sequência de compressão (b).

Levando em consideração apenas amostras de luminância, três tipos de transformadas são usados. O primeiro tipo é aplicado a todas as amostras do resíduo de predição, não importa se resultantes de predição por compensação de movimentos ou de predição Intra-quadro, e tem suporte 4×4. Se o macrobloco for predito usando INTRA_16×16, é necessária a aplicação da segunda transformada de suporte 4×4, do tipo Hadamard [55], aos 16 coeficientes DC dos blocos de luminância já codificados [13]. Com o adendo FRExt[16], foi adicionada uma transformada inteira de 8×8 para a predição Intra-quadro.

Os coeficientes transformados são quantizados escalarmente com passo de quantização determinado pelo parâmetro de quantização, QP, que pode assumir valores inteiros entre 0 e 51. Devido ao emprego de escala logarítmica, o passo de quantização duplica a cada incremento de 6 no QP, e um incremento unitário no QP resulta na redução de aproximadamente 12,5% na taxa de transmissão [14].

2.4.4 Codificação de Entropia

Uma das desvantagens das técnicas tradicionais de codificação de entropia vem da hipótese de que a estatística dos sinais seria estacionária e que seria possível levantar, de antemão, códigos de comprimento variável de tamanho ótimo para os elementos sintáticos normatizados. Essa hipótese raramente era verificada e nem sempre esse tipo de compressão conseguia remover com eficiência a redundância do sinal. A incorporação de um modelamento de contextos no sistema de codificação de entropia é que proporciona um alto grau de adaptação do H.264/AVC às fontes que serão comprimidas e contribui para seu desempenho superior.

São dois os métodos de codificação de entropia disponíveis no H.264/AVC: uma técnica de baixa complexidade, baseada no uso de conjuntos de códigos de comprimento variável e de contexto adaptativo, ou CAVLC (*Context Adaptative Variable Length Codes*), e um algoritmo mais complexo que emprega codificação aritmética binária e de contexto adaptativo, ou CABAC (*Context Adaptative Binary Arithmetic Coding*)[52].

2.4.5 Filtro de *Deblocking* Adaptativo

A estrutura baseada no fracionamento de quadros em macroblocos, que, por sua vez, podem ser divididos em partições menores para compensação de movimentos e para aplicação de transformadas de blocos, sujeita o sinal de vídeo codificado a artefatos de bloco. A aplicação de filtros nas bordas dos blocos é tida como uma poderosa ferramenta para reduzir a percepção desses artefatos.

A filtragem para redução de efeitos de blocos, também chamada de filtragem de *deblocking*, era tomada como uma etapa posterior ao processo de codificação no padrão H.263, alterando o sinal apenas antes da apresentação do mesmo. Entretanto, uma qualidade visual superior pode ser alcançada por meio da aplicação da etapa de filtragem no laço de codificação o que implica que todos os quadros de referência sejam versões filtradas dos quadros reconstruídos, abordagem introduzida pelo H.261[6] e seguida pelo H.264/AVC[56] e pelo anexo J do H.263+ [57].

O filtro de *deblocking* em laço do AVC é um filtro adaptativo ao conteúdo e à quantização que ajusta a sua intensidade de acordo com:

- o modo de predição do macrobloco (Intra ou Inter);
- o parâmetro de quantização;
- o vetor de movimento;
- a decisão por quadro ou campo (no caso em que se tratam sequências entrelaçadas); e
- o valor de intensidade dos *pixels*.

O objetivo do filtro é garantir que as bordas reais da cena não sejam suavizadas enquanto os artefatos de blocagem são reduzidos.

2.4.6 Perfis do H.264/AVC

O H.264/AVC foi desenvolvido para ser o mais genérico possível e capaz de atender uma vasta gama de aplicações, taxas, resoluções, qualidades e serviços. Todavia, diferentes aplicações apresentam requisitos diversos. De forma a maximizar a inter-operabilidade e proporcionar grande aceitação do padrão, a especificação do H.264/AVC define perfis e níveis conforme ilustrado na Figura 2.13.

Um perfil é definido como um subconjunto da sintaxe completa do *bitstream* ou, em outras palavras, um subconjunto de ferramentas de codificação. De forma a operar em um subconjunto da sintaxe, elementos sintáticos específicos para sinalização indicam a presença ou a ausência de elementos sintáticos que venham a ser usados posteriormente no *bitstream*. Todos os decodificadores compatíveis com determinado perfil devem suportar as ferramentas de codificação correspondentes àquele.

Na primeira versão do H.264, três perfis foram definidos: *Baseline*, *Extended* e *Main*. O perfil Baseline foi projetado com o foco em aplicações para dispositivos móveis, celulares e equipamentos de

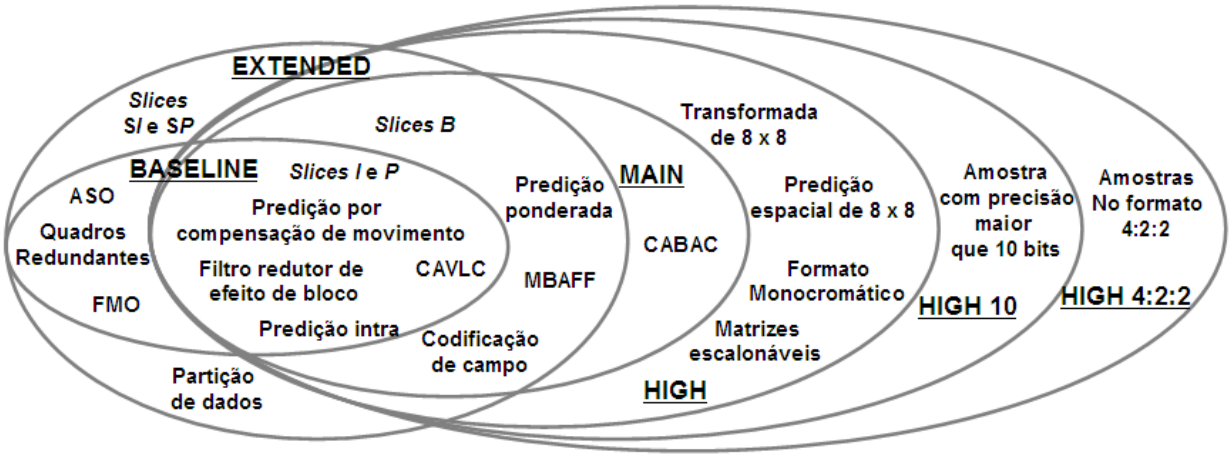


Figura 2.13: Ferramentas de compressão do H.264 agrupadas nos respectivos perfis do recomendados pelo padrão: BASELINE, MAIN, EXTENDED e HIGH.

videoconferência. Nessas aplicações, o consumo de potência é uma restrição importante e, portanto, as complexidades dos codificadores e decodificadores devem ser baixas. Além disso, espera-se que as resoluções tratadas não sejam maiores que CIF ou QVGA. Esse perfil suporta todas as características do H.264/MPEG4-AVC, versão 1 (2003), exceto pelos conjuntos de características a seguir:

1. *slices B*, codificação de campo, chaveamento adaptativo entre quadro e campo (MBAFF - *Macroblock Adaptive Switching Between Frame and Field*) e predição ponderada.

2. CABAC; e

3. *slices SI (Switching I)* e *SP (Switching P)* e partição de dados com *slice*.

Os dois primeiros itens contêm um conjunto de características que são suportados pelo perfil *Main*, em adição às características suportadas pelo *Baseline*. A intenção é proporcionar alta eficiência de codificação com o alvo em aplicações de TV digital. Note a presença dos dois codificadores de entropia normatizados pela recomendação bem como a remoção de algumas ferramentas de resiliência devido à baixa taxa de erros dos canais que transportam os sinais de TV [58].

O perfil *Extended* suporta todas as características do perfil *Baseline* adicionado aos itens um e três da enumeração acima. A grosso modo, o perfil *Extended* foi desenvolvido para promover um compromisso entre os perfis *Baseline* e *Main* com um foco para necessidades específicas de aplicações com *streaming* de vídeo adicionado à robustez a erros e perda de pacotes.

O padrão H.264/AVC sofreu um adendo, denominado FREXt [16], em que se adicionaram outros três perfis com base no perfil *Main*: *High, High 10* e *High 4:2:2*. Esses perfis foram desenvolvidos de forma a munir o codificador H.264/AVC de ferramentas demandadas por ambientes profissionais de edição de vídeo e com foco em resoluções mais elevadas. O perfil *High* possibilita o uso de sinais de vídeo de altas resoluções porém ainda não possibilita o emprego de formatos de crominância mais precisos ou de amostras com maior acurácia; essa passa a ser disponibilizada pelo perfil *High 10* onde 10 bits são gastos por amostra do sinal. Os perfis *High 4:2:2* e *High 4:4:4* (previsto, mas removido com a emenda [59], por isso não ilustrado na Figura 2.13[9]) representam cores com maior precisão permitindo amostragem de crominância do tipo 4:2:2 e 4:4:4, respectivamente. Os grandes avanços proporcionados pelo perfil High devem-se à adição de transformada inteira de suporte 8×8 *pixels*, predição intra com blocos de 8×8 *pixels* e tabelas de quantização/ponderação ajustáveis. Em média, esse perfil é capaz de prover 10% de eficiência de codificação quando seu *bitstream* é comparado com o gerado pelo perfil Main para uma mesma sequência de vídeo no formato 720p (quadros nas dimensões de 1280×720 *pixels*, não entrelaçados) [60].

2.5 AVALIAÇÃO DE DESEMPENHO COMPARATIVO DE CODIFICADORES DE VÍDEO

Durante o desenvolvimento dos padrões de compressão de vídeo, a necessidade da comparação de desempenho entre diferentes propostas levou o VCEG (*Video Coding Experts Group*) a definir uma metodologia padrão para comparação de curvas RD[10]. Ficou estabelecida a técnica de Bjontegaard [61] como método para o cálculo de diferenças médias entre curvas RD.

Basicamente, segue-se a seguinte rotina:

1. Escolha duas curvas RD para comparação. Para cada uma, determine de quatro pontos de simulação para análise de desempenho (Figura 2.14(a)).

2. Execute ajuste de curvas para cada curva (Figura 2.14(b)). O ajuste deve ser feito de forma que a função interpolada passe por cada um dos quatro pontos de análise escolhidos no passo anterior.

[9] Adaptado de [44].

[10] Curvas RD são linhas de desempenho derivadas da observação das variáveis operacionais taxa (R) e distorção (D) quando se variam parâmetros de codificação de determinado *codec* de vídeo.

3. A partir da funções ajustadas, determine a área A entre as mesmas sobre o intervalo de interesse para análise $[R_i, R_f]$ (Figura 2.14(c)).

4. Defina a diferença média de desempenho entre duas curvas pela diferença entre as integrais das curvas, dividida pelo comprimento do intervalo de integração: $\frac{A}{R_f - R_i}$.

O VCEG sugere a interpolação de curva RD da seguinte forma,

$$SNR = a + b * t + c * t^2 + d * t^3, \tag{2.5}$$

em que t é o valor da taxa de bits, tomado em escala logarítmica, e SNR é o valor da PSNR. Os parâmetros a, b, c e d são determinados de forma que a curva passe por todos os quatro pontos RD de uma determinada configuração do codificador. O uso da escala logarítmica para as taxas impede que o valor da integral da região de altas taxas enviese o processo de comparação.

É possível, também, definir a taxa de bits em função da PSNR, ou seja:

$$t = a + b * SNR + c * SNR^2 + d * SNR^3. \tag{2.6}$$

Desta forma, é possível encontrar tanto a diferença média, em dB, para toda uma faixa de taxa de bits, como encontrar a diferença média percentual de taxas de bits para toda uma faixa de PSNR.

A grande vantagem dessa metodologia é que a diferença entre as curvas de desempenho RD é computada por uma métrica objetiva, facilitando a análise quando os números de desempenho são muito próximos.

Uma vez apresentada a estrutura funcional básica do codificador AVC e uma rotina para comparação de desempenho entre curvas RD geradas por diferentes configurações deste codificador, exploraremos aspectos de sua implementação de modo a propor um arranjo que permita codificação de vídeo restrita por velocidade de compressão e por demanda energética. As vantagens e desvantagens de cada proposta serão discutidas por meio dos números de diferença de desempenho observados.

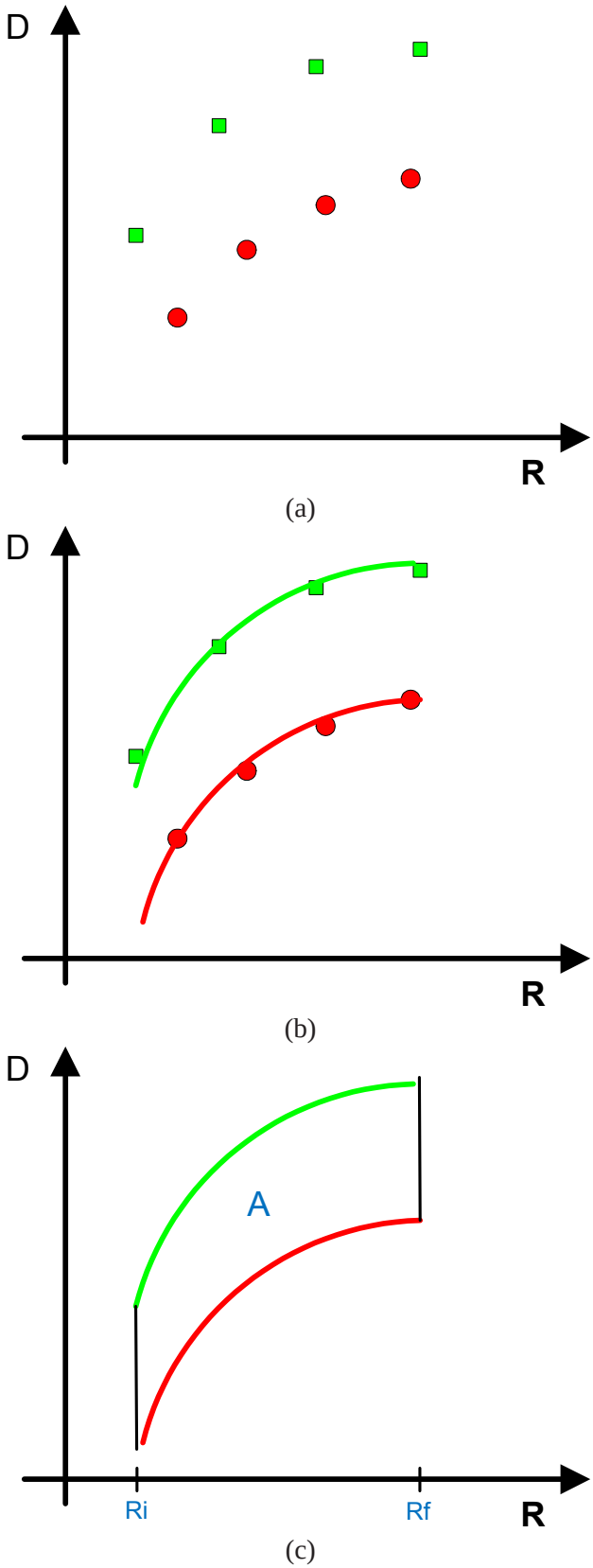


Figura 2.14: Procedimento para o cálculo de diferença de desempenho entre curvas RD de configurações de codificadores de vídeo. Partindo de (a) quatro pontos RD de simulação de cada codificador sob análise, interpolam-se curvas RD (b) e calcula-se a área A entre as curvas. A métrica de diferença consiste na razão $\frac{A}{R_f - R_i}$, em que R_i representa o valor inicial do intervalo enquanto R_f, o valor final.

3 AVALIAÇÃO E CONTROLE DE COMPLEXIDADE DO CODIFICADOR H.264/AVC

Este capítulo caracteriza o codificador H.264/AVC quanto à complexidade computacional e propõe plataformas simplificadas para seu controle.

3.1 CONSIDERAÇÕES INICIAIS

A maneira como foi proposto o estágio preditivo do H.264/AVC levou-o a se tornar a etapa do processo de compressão mais onerosa computacionalmente. Isso é devido aos inúmeros testes aplicados na escolha do melhor dos **modos de predição** disponíveis em termos de taxa $\times$ distorção (RD, do inglês *Rate* $\times$ *Distortion*). É natural, portanto, que uma redução do tempo de codificação de sequências de vídeo digital deva passar pela modificação do estágio preditivo, especialmente pela etapa de estimação de movimentos.

Neste capítulo, será feita revisão bibliográfica sobre o assunto e será discutida a distribuição e o tratamento de complexidade nesta tese.

3.2 REVISÃO BIBLIOGRÁFICA

Na literatura, encontram-se vários trabalhos com enfoque no estágio de predição como estratégia de redução da complexidade computacional do codificador. Alguns trabalhos [62, 63, 64] exploraram técnicas de estimação de movimento subótimas que, posteriormente, foram incorporadas ao código de referência do H264/AVC. A ideia básica foi empregar heurísticas para reduzir a quantidade de visitas necessárias durante a varredura da janela de busca na estimação de movimentos, propiciando ganhos consideráveis em tempo de codificação com pequenas perdas de qualidade do sinal em relação à técnica de busca completa.

Há propostas para redução de complexidade em que a estimação de movimentos extensiva é evitada e aplicada somente na partição de macrobloco mais provável, partição esta determinada por meio de heurísticas que exploram informações dos macroblocos previamente codificados. Kuo & Chan aplicaram a verossimilhança e a correlação dos fluxos ópticos (campos de movimentos) na seleção da melhor partição

do macrobloco [65]. Já Kim et al. empregaram informações sobre as partições de macroblocos de quadros anteriores para a formação de contexto usado na indicação das melhores partições, somente sobre as quais será aplicada estimação de movimentos [66].

Apesar de não representar parcela significativa da complexidade do codificador, os testes empregados para predição Intra-quadro podem ser reduzidos por meio da seleção prévia do melhor modo de predição de acordo com heurísticas [67, 68].

Ao se incorporar a estimação de movimento rápida ao processo de predição, linhas de trabalho mais recentes encaram o processo global de decisão pela melhor predição de determinado macrobloco. A complexidade computacional do codificador passa a ser tratada como um recurso escasso e o conceito de otimização de taxa *vs.* distorção é estendido com a adição de um terceiro eixo na análise: o eixo da complexidade. Uma aplicação para esse tipo de abordagem é a comunicação de vídeo em sistemas sem fio com restrições de energia [69], em que recursos específicos da plataforma que executa o sistema de codificação são manipulados em conjunto com parâmetros de codificação do codificador H.263+. Ademais, as sequências testadas são de conteúdo de baixa resolução e o descarte de quadros (e, portanto, de informações) é utilizado para garantir a velocidade mínima de codificação. O trabalho de Akyol et al., por sua vez, enfatiza a transmissão em tempo real de sinais de vídeo desenvolvendo controle conjunto do preenchimento dos *buffers* de complexidade e de transmissão [70]. É frequente o uso de paradas antecipadas no processo de determinação da melhor predição de um macrobloco nessa linha de abordagem.

Moecke & Seara [71] sugerem metodologia capaz de promover uma redução considerável de complexidade por alteração da implementação do *software* de referência. Contudo, o intervalo disponibilizado para escala em complexidade não é percebido quando a proposta é realizada usando uma implementação de codificador que lance mão de algoritmos devidamente otimizados, de bibliotecas de alto desempenho e de alguns recursos disponibilizados pela própria plataforma em que se executa o codificador [72, 73]. Modelos para o tratamento da complexidade de um compressor também foram propostos [74, 75]. Todavia, os resultados são geralmente comparados com o codificador H.264/AVC em seu *software* de referência, que carece de cuidados básicos quanto à otimização de complexidade na sua implementação; além disso, apenas sequências de baixas e médias resoluções são ensaiadas.

Novas propostas de padrões de compressão de vídeo surgem com forte viés quanto ao tratamento de sequências de vídeo de alta resolução, alta qualidade de imagem, amplo espaço de cores, escala dinâmica aumentada etc. Isso é geralmente alcançado às custas de elevado esforço computacional. A mais nova

proposta de padrão, o High Efficiency Video Coding, HEVC, listou explicitamente complexidade entre as principais preocupações dos esforços de padronização [22].

3.3 PERFIL DE COMPLEXIDADE DO CODIFICADOR H.264/AVC *HIGH PROFILE*

Dada a sua estrutura DPCM híbrida, o codificador H.264/AVC possui um módulo de predição, um módulo de transformada, uma etapa diferencial e um laço de realimentação [1]. Uma discussão detalhada do funcionamento de cada item de uma estrutura DPCM híbrida é encontrada na Seção 2.3.5. O estágio de predição do codificador H.264 usa amostras previamente reconstruídas (e filtradas) como parâmetro de entrada do processo dos modelos de predição. Isso evita descasamentos entre os dados empregados pelo codificador e pelo decodificador e inibe a deriva[1], ou seja, garante perfeito sincronismo entre a codificação e a decodificação. O estágio de predição do H.264/AVC é relativamente complexo devido a seu rico conjunto de ferramentas, conforme discutido nas Seções 2.4.1 e 2.4.2.

Em síntese, a predição para determinado macrobloco de um quadro é criada a partir de amostras do quadro já codificadas, sejam pertencentes ao mesmo quadro (técnicas Intra-quadros), sejam pertencentes a quadros previamente codificados (técnicas Inter-quadros). A predição é subtraída do macrobloco original (etapa diferencial) e o resíduo é convertido para o domínio transformado (módulo de transformada), comprimido (por quantização e codificação entrópica) e transmitido em conjunto com as informações necessárias ao decodificador para a repetição do processo de predição (vetores de movimentos, modos de predição, etc.). O decodificador cria uma predição idêntica à do codificador e adiciona-a ao bloco de resíduo já decodificado.

Conforme exposto na Seção 2.4, o estágio de predição do codificador H.264/AVC disponibiliza um amplo conjunto de testes para a escolha do melhor modo de predição em termos de RD. É imediato relacionar que a redução de complexidade de codificação possa ser alcançada por meio da simplificação do estágio de predição e, em especial, da sua etapa de estimação de movimentos.

A fim de verificar o perfil de complexidade do codificador H.264/AVC *High Profile* e confirmar as estimativas de complexidade dos sub-módulos de codificação apresentadas em [15], fez-se o uso de ferramentas de programação chamadas de *profiler*, empregadas no levantamento de informações a respeito de um programa, tais como funções usadas em uma tarefa, tempo gasto para executá-las, quantidade

[1] Tradução do termo inglês *drifting*.

de vezes em que foram chamadas, suas interrelações e quantidade de memória usada. Para o sistema operacional *Linux*, há a ferramenta `gprof`, um *profiler* que se enquadra na categoria de *software livre* e é capaz de indicar o tempo gasto em cada função de um programa.

Inicialmente, analisou-se a implementação de referência do H.264/AVC, o *software* JM[2]. A sequência escolhida foi "Pedestrian Area", de dimensões 1920×1080 *pixels*, ou seja, de alta definição. As opções de codificação foram: otimização RD para codificação de macroblocos ativada; quatro quadros de referência, estimação de movimentos FastFullSearch e quadros B desativados. Os números são apresentados na Tabela 3.1 para a codificação de vídeo apenas com técnicas Intra-quadros e, na Tabela 3.2, para o caso geral.

Tabela 3.1: Contribuição relativa na complexidade computacional para a codificação da sequência HD "Pedestrian Area" usando somente técnicas de predição Intra-quadros na versão JM12.3 do *software* de referência H.264/AVC em um computador pessoal com o processador Intel® Pentium® D.

Etapa de Codificação	**Percentual [%]**
Pred. Intra em blocos de 4×4	25,2
Pred. Intra em blocos de 8×8	20,9
Pred. Intra em blocos de 16×16	6,1
Outras etapas	47,8
Total	100,0

A Tabela 3.1 mostra as contribuições relativas em complexidade dos modos de predição Intra-quadros do H.264/AVC. Verifica-se que a complexidade envolvida na predição de blocos de tamanho 4×4 e 8×8 *pixel* é maior que a envolvida para os blocos de 16×16 *pixels*. Isso é devido ao maior número de modos de predição disponíveis (nove) para tamanhos de blocos de 4×4 e 8×8 *pixels*.

As medidas de complexidade de Huang et al. [15] são confirmadas e estendidas na Tabela 3.2, na qual diferentes tamanhos de janelas de estimação de movimentos são utilizados na codificação da sequência HD. Verificou-se que o codificador gasta a maior parte do tempo de processamento no estágio de estimação de movimentos, pois é grande o número de testes exaustivos usados pelos algoritmos de varredura para busca do melhor casamento em termos de RD.

[2] JM disponível em: http://iphome.hhi.de/suehring/tml/

Tabela 3.2: Contribuição relativa na complexidade computacional para a codificação da sequência HD "Pedestrian Area" para vários tamanhos de janelas de estimação de movimentos na versão JM12.3 do *software* de referência H.264/AVC em um computador pessoal com o processador Intel® Pentium® D.

	Tamanho da janela (*pixels*)				
Etapa de Codificação	**8**	**16**	**32**	**64**	**128**
Estimação de Movimentos	65,8%	78,5%	90,8%	97,1%	99,1%
Pred. Intra em blocos de 4×4	6,1%	3,8%	1,6%	0,5%	0,1%
Pred. Intra em blocos de 8×8	4,9%	3,2%	1,3%	0,4%	0,1%
Pred. Intra em blocos de 16×16	1,4%	1,0%	0,4%	0,1%	0,0%
Outras etapas	21,8%	13,5%	5,9%	1,9%	0,7%
Total	100,0%				

A utilização de algoritmo subótimo de estimação de movimentos acarreta alterações significativas no tempo de execução de uma atividade de compressão com o codificador H.264/AVC na sua implementação de referência JM12.3 [54]. O impacto da ativação da técnica de estimação de movimentos UMHS no perfil de complexidade relativa do H.264/AVC é objeto de avaliação nas Tabelas 3.3 e 3.4 para a codificação de sequência CIF e 1080p (1920×1080 *pixels* por quadro) respectivamente. Nessas tabelas, também é possível avaliar a sensibilidade da variação do parâmetro de quantização QP na complexidade relativa .

Observa-se que, apesar de a distribuição de complexidade relativa dos módulos de predição perder parcelas de complexidade (a contribuição de outras etapas chega a alcançar a cifra de 25% na Tabela 3.4), a contribuição conjunta da etapa de predição continua a dominar o esforço de computação. A variação dos QPs sobre a distribuição de complexidade relativa mostrou que a participação relativa das outras etapas de codificação aumenta com a diminuição do parâmetro de quantização. Isso é devido ao fato de um número maior de coeficientes transformados resultarem da quantização mais precisa o que, por sua vez, aumenta o volume de informações tratadas pelo codificador de entropia.

Espera-se que a implementação eficiente do padrão H.264/AVC possa alterar esses números, embora a contribuição em complexidade relativa do estágio de predições tenda a dominar a parcela de outros estágios de codificação. Para avaliar o impacto de otimizações no perfil de complexidade do *codec*, utilizou-se o codificador x264, uma implementação do padrão H.264/AVC cujo alto desempenho em termos de tempo

Tabela 3.3: Contribuição relativa na complexidade computacional para a codificação da sequência CIF "Mobile" para dois tamanhos de janelas de estimação de movimentos na versão JM12.3 do *software* de referência H.264/AVC em um computador pessoal com o processador Intel® Pentium® D. O parâmetro de quantização também foi variado na tabela.

	Tamanho da janela (*pixels*)/QP			
Etapa de Codificação	**32**/23	**32**/36	**64**/23	**64**/36
Predição Inter-quadros	64.7%	73.2%	73.2%	70.3%
Pred. Intra em blocos de 4×4	8.0%	6.1%	6.6%	4.9%
Pred. Intra em blocos de 8×8	6.9%	5.1%	5.8%	4.1%
Pred. Intra em blocos de 16×16	0.8%	0.8%	0.7%	0.7%
Outras etapas	19.6%	14.8%	16.6%	12.1%
Total	100.0%			

Tabela 3.4: Contribuição relativa na complexidade computacional para a codificação da sequência HD "Pedestrian Area" para dois tamanhos de janelas de estimação de movimentos na versão JM12.3 do *software* de referência H.264/AVC em um computador pessoal com o processador Intel® Pentium® D. O parâmetro de quantização também foi variado na tabela.

	Tamanho da janela (*pixels*)/QP			
Etapa de Codificação	**64**/16	**64**/28	**64**/36	**144**/28
Predição Inter-quadros	56,7%	66,4%	69,1%	72,2%
Pred. Intra em blocos de 4×4	10,1%	7,0%	5,9%	5,7%
Pred. Intra em blocos de 8×8	7,2%	5,1%	4,5%	4,2%
Pred. Intra em blocos de 16×16	1,1%	1,2%	1,2%	1,1%
Outras etapas	24,9%	20,3%	19,3%	16,8%
Total	100,0%			

de computação é devido ao uso de rotinas otimizadas em linguagem de montagem (linguagem de máquina, *assembly*) para a execução das etapas computacionalmente mais custosas.

Além do uso de rotinas de alto desempenho na realização dos blocos mais básicos de processamento, x264 também lança mão de paradas antecipadas durante o processo de otimização das decisões de macroblocos ou quadros por critérios taxa$\times$distorção, oferecendo aumento de 50 vezes na velocidade de compressão quando comparado ao codificador de referência e sem afetar o desempenho global em termos de taxa$\times$distorção [76]. Seu perfil de complexidade é verificado com os dados da Tabela 3.5, em que o x264 foi analisado na compressão das sequências "Mobile" CIF e "Mobcal" 720p, usando quatro quadros de referência, janela quadrada de estimação de movimentos de 64 *pixels* de suporte, otimização RD ativa na decisão dos modos de predição, quadros B desativados, estimação de movimentos UMHS e QP=28. Nesse ensaio, o perfil de complexidade dividiu a atividade de codificação em três etapas (Predições, Codificação Entrópica e Outras Etapas), pois o extensivo uso de algoritmos de paradas antecipadas no processo de predição poderia inviabilizar a medição de um modo de predição menos recorrente.

Tabela 3.5: Complexidade computacional relativa do x264 para a codificação das sequências "Mobile" CIF (352$\times$288 *pixels*) and "Mobcal" 720p (1280$\times$720 *pixels*) em um computador pessoal com o processador Intel® Core® 2 Quad. Nesses ensaios, QP=28.

	Resolução	
Etapa da Compressão	**CIF**	**720p**
Predições	91.24%	90.42%
Codificação Entrópica	6.07%	6.13%
Outras etapas	2.69%	3.45%
Total	100.0%	

Verifica-se da Tabela 3.5 que, para o x264, a alteração das dimensões dos quadros da sequência não altera o perfil de complexidade relativa de forma significativa. Isso é, em grande parte, devido à utilização da estimação de movimentos UMHS, que limita de forma significativa a quantidade de visitas durante a varredura em busca pela melhor referência para predição Inter-quadro. Dos resultados apresentados, conclui-se que a complexidade computacional do estágio de predição domina o tempo de codificação mesmo após a utilização de técnicas modernas para a decisão do modo de predição e para a estimação

de movimentos, juntamente com a implementação otimizada de rotinas por meio do uso de linguagem *assembly*.

3.4 ABORDAGEM PARA ESCALONAMENTO E CONTROLE DE COMPLEXIDADE

De forma a manipular a complexidade de todo o processo de compressão com o codificador H.264/AVC, podem ser ajustados parâmetros de configuração que afetam diretamente o esforço computacional de predição e, por sua vez, o esfoço de todo o processo de compressão. Entre os parâmetros, os de maior relevância para este trabalho são:

- a precisão da estimação de movimento *subpixel*;
- a forma como o processo de decisão do modo de predição é realizado para determinado macrobloco;
- o emprego de optimização na quantização dos valores dos coeficientes transformados;
- o conjunto de modos de predição disponíveis; e
- a quantidade de *threads* empregadas na codificação.

As propostas aqui descritas consistem em adaptar estes e outros parâmetros fortemente correlacionados com a complexidade computacional demandada pelo codificador H.264/AVC e sugerir uma arquitetura capaz de codificar sinais de vídeo gastando o menor intervalo de tempo na compressão, demandando a menor taxa de bits possível e resultando na melhor qualidade de sinal comprimido.

As propostas apresentadas nessa tese podem ser divididas em três classes:

- técnica de controle de complexidade com treinamento *on-line*;
- técnica de controle de complexidade com treinamento *off-line*; e
- técnica de controle de complexidade por energia com treinamento *off-line* e em malha fechada.

A primeira metodologia parte da investigação aprofundada do codificador de referência do padrão H.264/AVC de modo a restringir a complexidade de codificação. Um módulo de análise é inserido no codificador H.264/AVC e, durante a compressão de sinais de vídeo (*on-line*), o conjunto é capaz de

avaliar as estatísticas de uso de ferramentas de predição e descartar seletivamente módulos funcionais de maneira a escalonar a complexidade global de compressão. A segunda técnica, usando de alguns resultados da primeira como ponto de partida, faz uso de bibliotecas e implementações de alto desempenho destinadas a processadores da família x86 de forma a proporcionar a realização de sistemas em *software* capazes de prover compressão de vídeo digital em alta resolução a velocidades de codificação de interesse prático. Nessa abordagem, uma intensa etapa de treinamento *off-line* é realizada para descartar todas as configurações que não são capazes de operar na região ótima em termos de RDC. A terceira estratégia apresenta-se mais genérica. O H.264/AVC é analisado partindo-se de uma implementação em código fonte aberto que se destaca por seus artifícios de computação de alto desempenho oriundos de otimização de módulos computacionalmente intensivos e de técnicas de parada antecipada na análise de predições [76]. Em vez de otimizar o codificador por meio do ajuste de parâmetros que afetem sua velocidade, a otimização generaliza-se ao incluir um novo eixo na análise: o eixo da energia demandada (E). Um esquema de controle de energia consumida é acoplado à implementação otimizada, resultando em um sistema em tempo real capaz de escalonar a energia demandada na compressão de vídeo em uma ampla faixa, sem degradar o desempenho de codificação em termos de RD.

3.5 EQUIPAMENTOS DE COMPUTAÇÃO E IMPLEMENTAÇÕES USADAS NAS AVALIAÇÕES

Durante a realização das atividades de pesquisa afetas a esta tese, foram utilizados diferentes equipamentos de computação para a avaliação das técnicas propostas, todos dispostos em plataforma PC, *Personal Computer*. As variações de equipamentos ocorreram em função da própria evolução tecnológica disponibilizada pela indústria na medida em que a pesquisa era desenvolvida, agregada à necessidade de se atualizar e repor instrumentos de trabalho.

A lista abaixo mostra a relação dos processadores usados na execução das investigações de cada contribuição, as quais se sucedem de acordo com ordem cronológica ascendente:

- Contribuição 1: Intel® Pentium® D;
- Contribuição 2(a): Intel® Centrino 2®;
- Contribuição 2(b): Intel® Core i7® e AMD® Opteron®;

- Contribuição 3: Intel® Core i7® e AMD® Phenom®.

Exceto pelo uso de bibliotecas com primitivas de desempenho fornecidas fabricante Intel® e empregadas na Contribuição 2(a), as contribuições apresentadas proporcionam as vantagens relatadas independentemente do fabricante do processador para plataforma PC.

Quanto às realizações do codificador H.264/AVC, as diferentes escolhas de implementações também são resultados de busca por alto desempenho computacional em termos de velocidade de compressão de sinais de vídeo digital. Partindo da implementação de referência, o codificador JM, paradigma quanto a desempenho em termos de compactação dos sinais mas ruim em termos de velocidade de execução, seguiu-se analisando o codificador disponibilizado pelo fabricante Intel® (uma aplicação de referência de suas bibliotecas de primitivas de alto desempenho IPP®) e, por fim, estudando a implementação de elevada velocidade de compactação feita em código-fonte aberto, o x264.

3.6 SEQUÊNCIAS DE VÍDEO USADAS NAS AVALIAÇÕES

Antes de prosseguir com a discussão das contribuições desta tese, apresentam-se quadros ilustrativos das sequências de vídeo digital usadas ao longo da pesquisa. A Figura 3.1 apresenta sequências de baixa e média resolução. Por sua vez, as Figuras 3.2 e 3.3 mostram as sequências HD, com o destaque de que o último conjunto é restrito a sequências de vídeo-conferência.

Faz-se conveniente, também, uma breve avaliação de conteúdo dinâmico das sequências sob testes. Partindo da métrica de análise de conteúdo espaço-temporal descrita por Khan et al. [77], dispuseram-se as sequências de vídeo usadas nesta tese em um diagrama espaço-temporal, como o da Figura 3.4. Dependendo da posição da sequência em relação aos limites do diagrama, é possível classificar os conteúdos dinâmicos em quatro categorias:

- Sequências com baixa atividade temporal e baixa atividade espacial;
- Sequências com baixa atividade temporal e elevada atividade espacial;
- Sequências com elevada atividade temporal e baixa atividade espacial;
- Sequências com elevada atividade temporal e elevada atividade espacial.

O diagrama para as sequências de vídeo usadas nesta tese é mostrado na Figura 3.5, evidenciando

o conteúdo temporal baixo a moderado de sequências típicas de vídeo-conferência e discriminando sequências de vídeo de elevada atividade em diferentes resoluções, com destaque para as sequências “Mobile”, “Soccer” e “Pedestrian Area”. Isso é um indício de um processo de compressão trabalhoso para esses sinais.

(a)

(b)

Figura 3.1: Sequências de baixa e média resolução. Em (a), mostram-se as sequências CIF (352×288 *pixels* por quadro) usadas: “Akiyo”, “Foreman”, “Mobile” e “Silent”. Em (b), mostram-se sequências 4CIF (704×576 *pixels* por quadro), doravante referidas por SD, usadas neste trabalho: “City”, “Crew”, “Harbour”, “Ice” e “Soccer”. A ordem de apresentação das sequências segue o padrão de varredura de leitura.

(a)

(b)

Figura 3.2: Sequências de alta resolução. Em (a), apresentam-se as sequências 720p (1280×720 *pixels* por quadro) usadas: “Mobcal”, “Parkrun”, “Shields” e “Stockholm”. Em (b), apresentam-se sequências 1080p (1920×1080 *pixels* por quadro), usadas neste trabalho: “Pedestrian”, “Riverbed”, “Rushhour”, “Sunflower” e “Tractor”. A ordem de apresentação das sequências segue o padrão de varredura de leitura.

Figura 3.3: Sequências de vídeo-conferência em alta resolução (1280×720 *pixels* por quadro) utilizadas: “Seq05”, “Seq06”, “Seq12”, “Seq15”, “Seq17” e “Seq21”. A ordem de apresentação das sequências segue o padrão de varredura de leitura.

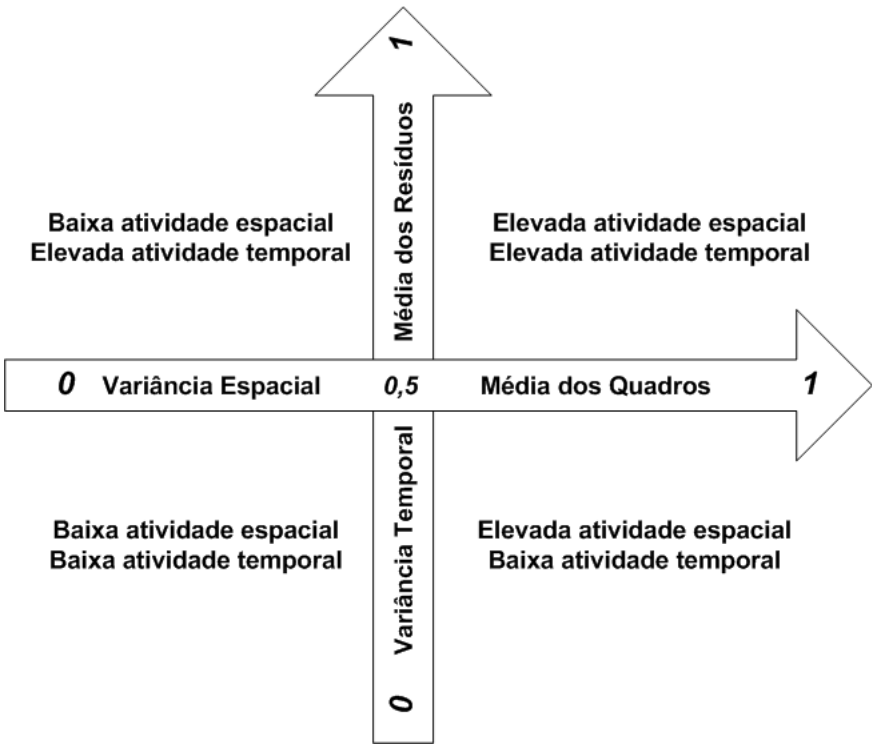


Figura 3.4: Diagrama de avaliação de conteúdo espaço-temporal. Quanto mais afastada da origem estiver situada uma sequência neste diagrama, maior o conteúdo espacial ou temporal.

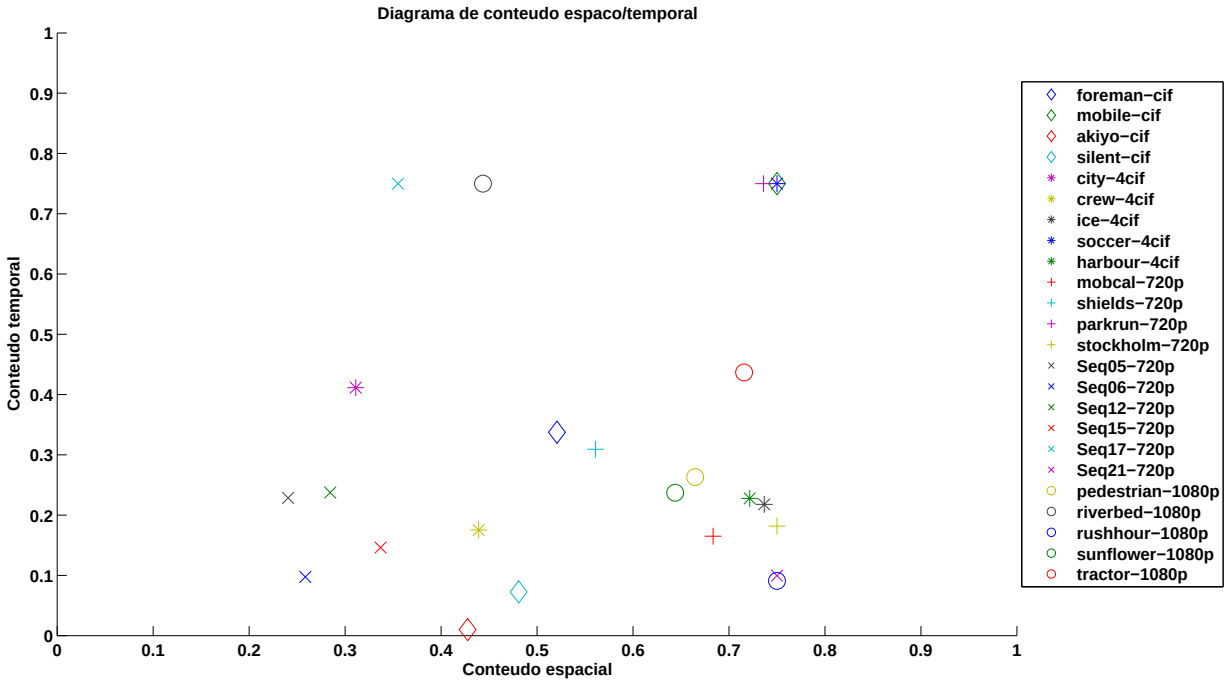


Figura 3.5: Diagrama de avaliação de conteúdo espaço-temporal para as sequências sob testes. As sequências “Mobile”, “Soccer” e “Pedestrian Area” destacam-se das demais pelo seu elevado conteúdo de atividade e espera-se que a compressão desses sinais seja bastante trabalhosa.

4 CONTROLE DE COMPLEXIDADE DO H.264/AVC COM TREINAMENTO *ON-LINE* E SORTEIO COM ORDENAÇÃO DE MODOS DE PREDIÇÃO

Este capítulo apresenta uma metodologia para controle de complexidade do codificador H.264/AVC com treinamento on-line *e que utiliza o sorteio e a ordenação dos modos de predição no escalonamento da complexidade de compressão.*

4.1 CONSIDERAÇÕES INICIAIS

Neste capítulo apresenta-se uma metodologia para controle de complexidade do codificador H.264/AVC que altera o estágio de predição do compressor para proporcionar escala em complexidade. O codificador de referência JM foi modificado de forma a incorporar um sistema de treinamento *on-line* capaz de sortear macroblocos e acompanhar a distribuição estatística dos modos de predição, juntamente com seus custos, em termos de complexidade computacional.

Baseado no ordenamento dos modos de predição por meio da relevância, medida através da frequência de recorrência de um modo de predição, modos de predição menos frequentes são descartados seletivamente no processo de busca pelo melhor modo de predição em termos de RD. Desta forma, permanecerão os modos dominantes, cuja remoção acarretaria em grandes perdas de desempenho em termos de RD. O descarte dos modos não dominantes é feito de forma seletiva e é suspenso sempre que a complexidade computacional dos modos de predição remanescentes corresponda à uma provisão de complexidade pré-estabelecida.

4.2 ABORDAGEM PARA ESCALONAMENTO DE COMPLEXIDADE NO JM13.2

Conforme discutido na Seção 3.3, a complexidade computacional do codificador H.264/AVC pode ser reduzida mediante alterações no seu estágio de predições, responsável por parcela significativa de

esforço computacional na compressão de sequências e, notadamente, por meio da otimização do módulo de estimação de movimentos. Quando se trata da implementação de referência do padrão H.264/AVC, a atividade é facilitada pois o alto desempenho em termos de velocidade de computação não foi uma restrição crítica de projeto e há boa quantidade de opções de intervenções sobre o código fonte que são capazes de entregar bons números de desempenho.

Os inúmeros testes necessários para a escolha do melhor modo de predição para um macrobloco são os responsáveis pelo elevado esforço computacional dispendido durante o estágio de predições. Entretanto, verificou-se que os modos de predição aplicados para codificar os sinais de vídeo submetidos a compressão ficam agrupados em determinadas classes. A Figura 4.1 mostra o perfil de frequências dos modos de predição selecionados na compressão da sequência "Pedestrian Area" em diferentes tamanhos de quadros desde QCIF (176×144 *pixels* por quadro) até 1080p (1920×1080 *pixels* por quadro). Cada cor do gráfico de barras representa a frequência de determinado modo de predição e **P<=8x8** engloba todos os modos de predição que lançam mão de compensação de movimentos e para os quais os tamanhos das partições de macrobloco são inferiores ou iguais a 8×8 *pixels*. O que se nota dos perfis é que, para a sequência em análise, aumentos na resolução implicam em agrupamento dos modos de predição mais recorrentes nas partições de macrobloco maiores. Perfis construídos para outras sequências são apresentados nas Figuras 4.2 e 4.3 e o comportamento geral sugere que esforço computacional pode ser evitado por um processo de compressão que evite as partições de macroblocos menos recorrentes [78].

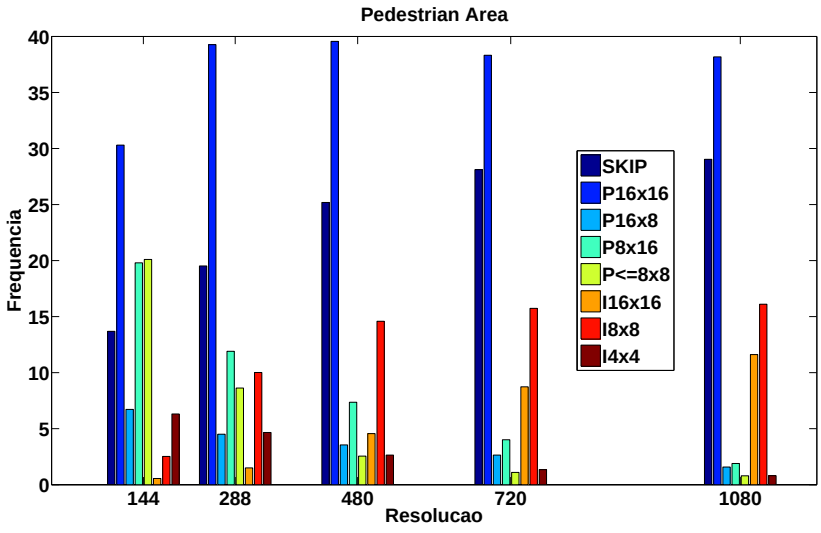


Figura 4.1: Frequência de ocorrência de modos de predição $\times$ dimensões de quadro para a sequência "Pedestrian Area".

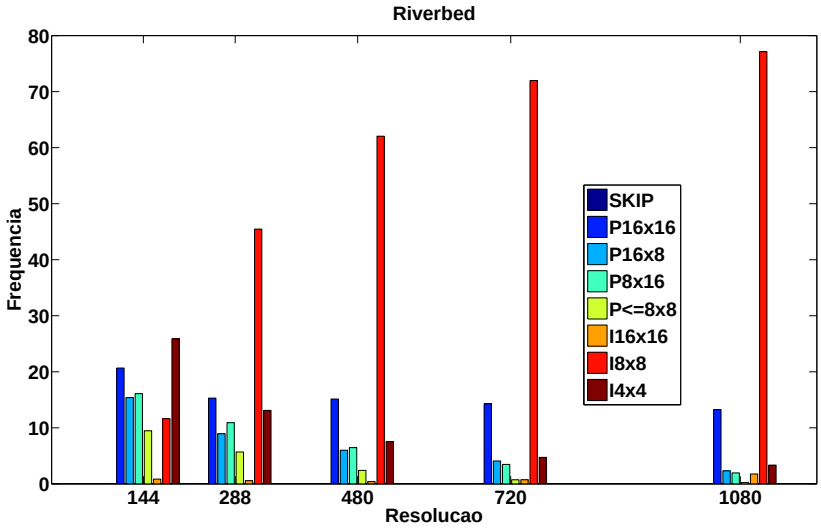


Figura 4.2: Frequência de ocorrência de modos de predição × dimensões de quadro para a sequência "Riverbed".

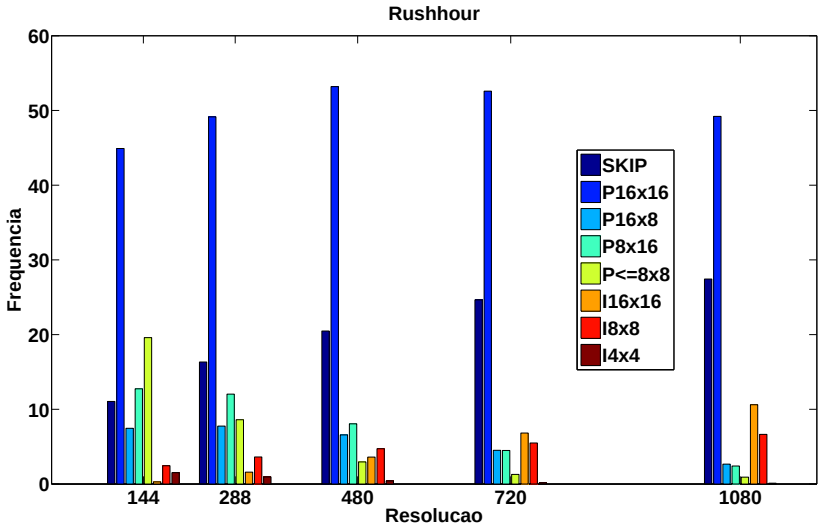


Figura 4.3: Frequência de ocorrência de modos de predição × dimensões de quadro para a sequência "Rushhour".

O escalonamento de complexidade proposto baseia-se na supressão de modos de predição menos frequentes por meio da aplicação do esquema ilustrado na Figura 4.4. Inicia-se pela seleção aleatória de um conjunto pequeno de macroblocos (população de referência) para prever a distribuição de frequências dos modos de predição do próximo quadro. Em seguida, selecionam-se os **modos dominantes**, aqueles modos de predição mais frequentes na população de referência. Os modos dominantes determinados serão usados no provimento da complexidade desejada C na codificação do próximo quadro de acordo com o seguinte algoritmo:

Cada quadro possui N macroblocos e um conjunto de modos dominantes ($\mathbf{D}$) que, inicialmente, é composto por todos os modos de predição disponíveis. Para o n-ésimo quadro P ou B:

1. Selecione, aleatoriamente, um conjunto ($\mathbf{S}$) de N_S macroblocos do n-ésimo quadro. Os $N - N_S$ macroblocos restantes formam o conjunto complementar $\mathbf{S}'$.
2. Teste todos os modos de predição para os macroblocos no conjunto $\mathbf{S}$ de forma a selecionar os melhores modos de predição em termos de RD.
3. Teste apenas os modos dominantes $\mathbf{D}$ para os macroblocos pertencentes a $\mathbf{S}'$.
4. Ordene por frequência de ocorrência os melhores modos de predição no conjunto $\mathbf{S}$. Enquanto a complexidade acumulada estiver dentro de determinada provisão de complexidade C, insira o próximo modo de predição mais frequente ao conjunto de modos dominantes $\mathbf{D}$.
5. Ajuste $n \leftarrow n + 1$ e repita o processo.

Embora a frequência de distribuição dos melhores modos de predição não seja estacionária, testes mostraram que estacionariedade em frequência pode ser tomada como uma boa aproximação para quadros adjacentes [78], o que constitui um sistema adaptativo com memória de um quadro. Erros na busca por modos dominantes refletem-se em pequenas degradações no desempenho do codificador em termos de RD. O tamanho da população de referência N_S foi determinado empiricamente. Quanto menor for N_S, maiores serão as reduções em complexidade, contudo, pior o desempenho. Arbitrou-se $N_S = N/10$ satisfatório na maioria dos casos e esse valor será empregado nas simulações analisadas nesta seção.

Na etapa de amostragem do algoritmo listado acima, a cada modo de predição associa-se uma frequência de ocorrência e uma complexidade (tempo estimado gasto nas predições mediante o modo de predição em questão). Determina-se, por meio de busca exaustiva, o conjunto de modos de predição tais que a soma das complexidades seja inferior ou igual à provisão de complexidade C, enquanto se maximiza a frequência acumulada do conjunto.

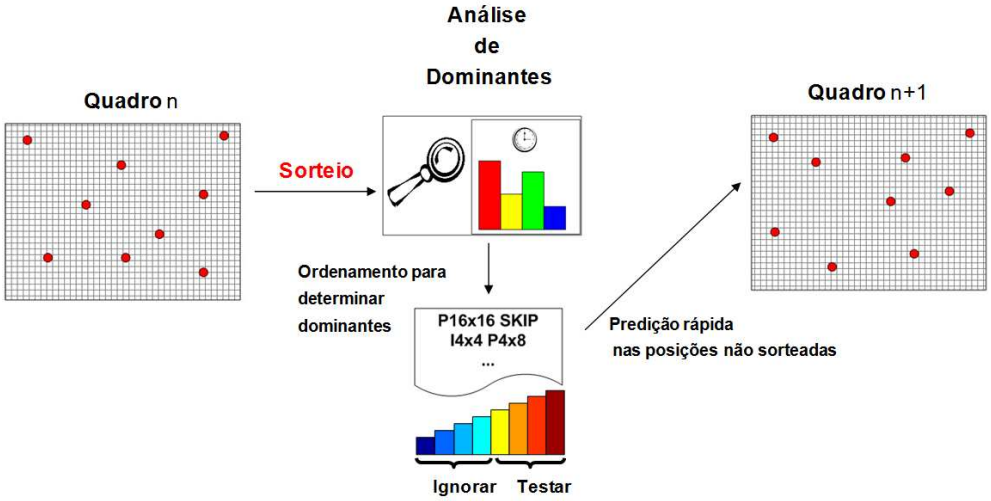


Figura 4.4: Esquema para escalonamento de complexidade. As posições sorteadas aparecem em vermelho; as posições não sorteadas consistem no restante da grade representante do quadro. Os modos dominantes do quadro anterior são usados na predição rápida dos macroblocos não sorteados do quadro seguinte.

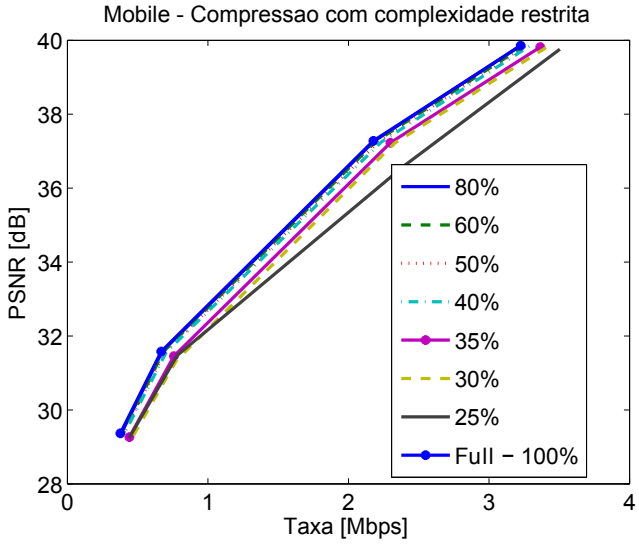
4.3 RESULTADOS EXPERIMENTAIS

O método proposto foi implementado no *software* de referência JM13.2 usando estimação de movimentos UMHS. A complexidade foi medida pelo tempo de execução total do trabalho de compressão[1]. A Figura 4.5 apresenta as curvas de desempenho RD para diferentes ajustes de complexidade: full (100% de complexidade na codificação), 90% de complexidade (total) de codificação, 80% de complexidade (total) de codificação e assim por diante. Observa-se que o desempenho do codificador modificado é muito próximo ao do codificador de referência, de forma que a avaliação das diferenças de desempenho em entre as curvas fica prejudicada pela justaposição das mesmas. As curvas de desempenho para a sequência 1080p (Figura 4.5 (b)) são mais justapostas do que as curvas para a sequência CIF (Figura 4.5 (a)), efeito de um agrupamento mais intenso dos modos de predição em poucas classes.

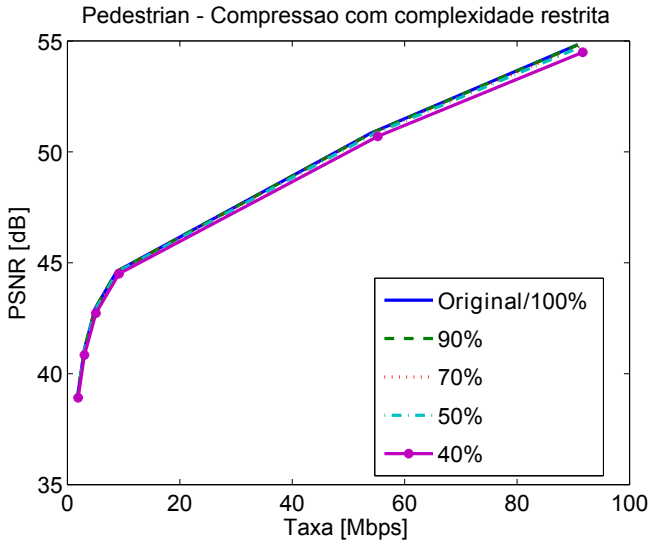
4.3.1 Resultados por diferenças médias de desempenho

Traçar as diferenças médias de desempenho entre as curvas RD para diferentes ajustes de complexidade de acordo com o recomendado pelo VCEG permite uma análise mais detalhada na comparação

[1] Apesar de alterações no estágio de predição serem usadas na escala de complexidade, as medidas de escala de complexidade tratadas nesta tese referem-se sempre ao esforço global de computação de sequências de vídeo digital, e não somente do esforço do estágio de predição.



(a)



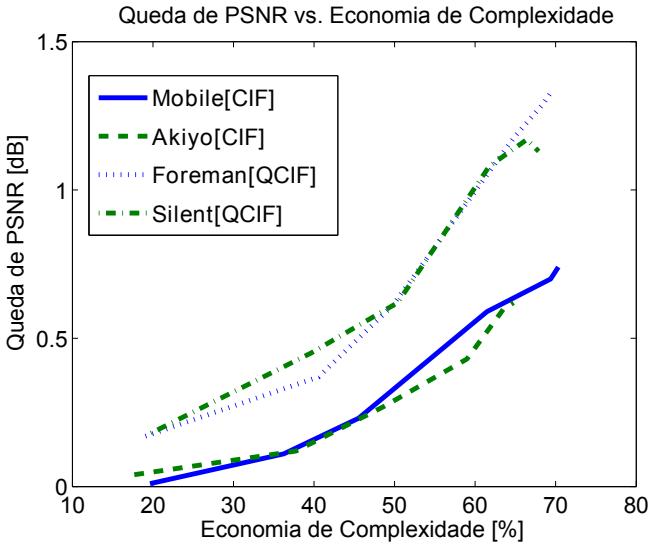
(b)

Figura 4.5: Desempenho em termos de RD para diversos ajustes de provisão de complexidade na compressão da sequência (a) “Mobile [CIF]” e (b)“Pedestrian [1080p]”.

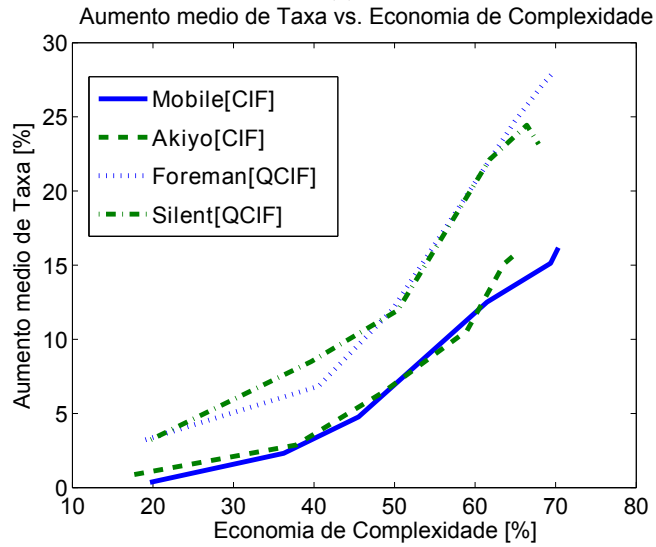
entre curvas. As Figuras 4.6 e 4.7 mostram as diferenças médias de PSNR e de taxa entre curvas foram calculadas de acordo com o discutido na Seção 2.5. A curva de referência de comparação foi a que representa a codificação com 100% de complexidade. Verificam-se pequenas perdas de qualidade quando se usam predições mediante modos dominantes; isso é devido a supressão de modos de predição que, apesar de pouco recorrentes, contribuíam de forma significativa ao desempenho em termos de RD. As perdas em taxas são inferiores a 5% (Figura 4.6 (b)) se as reduções de complexidade forem tomadas até 50% para sequências CIF. Para sequências QCIF, o aumento em taxa fica em torno de 10%. Nota-se que o desempenho em termos de RD é trocado por complexidade e as perdas podem ser ainda diminuídas se as resoluções espaciais das sequências forem aumentadas ou mesmo quando o enviesamento dos modos de predição for mais intenso ao redor de um subconjunto reduzido.

Quando se tratam de sequências HD, o algoritmo de restrição e controle de complexidade beneficia-se ainda mais do agrupamento dos modos de predição, conforme é notado pela a análise da Figura 4.7. Para reduções de complexidade abaixo de 50%, o aumento de taxa necessário é aproximadamente de 4% para o pior caso.

De maneira a avaliar o efeito da resolução espacial na metodologia de redução de complexidade proposta, sequências foram comprimidas em diferentes tamanhos de quadro, de QCIF a 1080p. Os resultados são apresentados na Figura 4.8, onde SD representa quadros de tamanho 720×480*pixels*. Maiores resoluções são beneficiadas por maiores reduções de complexidade para uma ampla faixa de redução de complexidade. A sequência "Riverbed" é reconhecida por sua dificuldade de compressão. Essa dificuldade advém do fato de que as suas cenas ricas em texturas não são otimamente codificadas pelos modelos de predição temporal Inter-quadros. Dessa, resulta um grande número de macroblocos são codificados com técnicas Intra-quadros, conforme o previsto pelo perfil de distribuição da Figura 4.2. Neste quesito, o algoritmo aqui proposto consegue detectar esse agrupamento diferenciado e economizar muito esforço computacional às custas de um acréscimo mínimo de taxa de codificação. É verificada, também, uma descontinuidade em formato de joelho para a Figura 4.2(b), a qual é devida à supressão de um modo de predição de relevância em termos de RD durante a redução seletiva do conjunto dos modos de predição disponíveis.

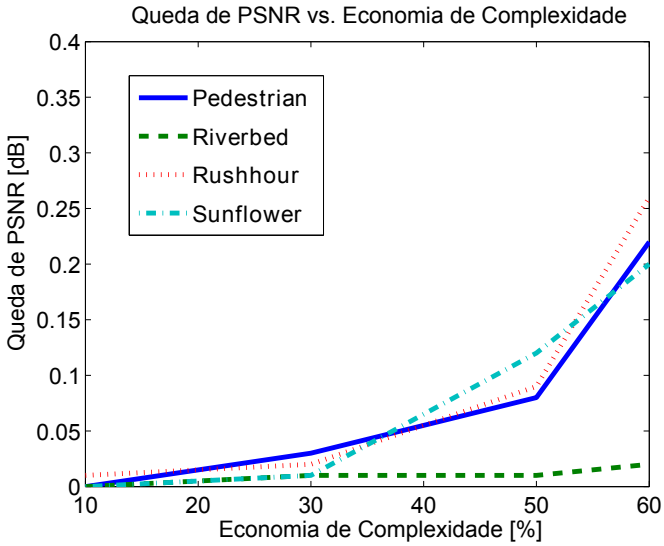


(a)

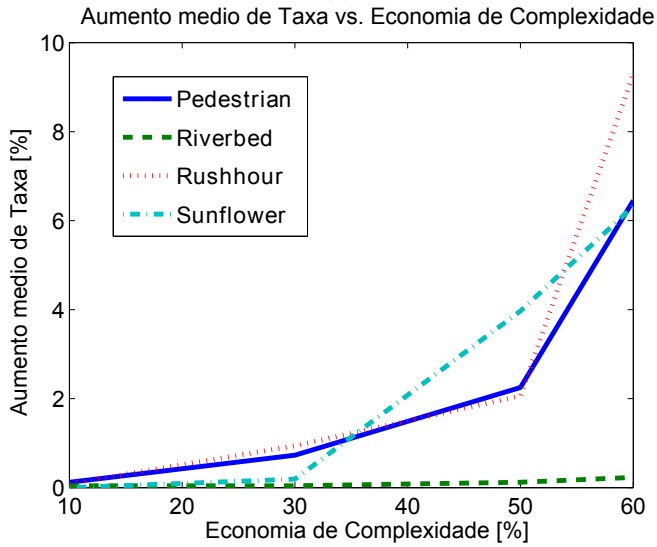


(b)

Figura 4.6: Avaliação de desempenho RD por diferenças médias para diferentes patamares de redução de complexidade de codificação. As sequências de testes utilizadas: “Mobile [CIF]”, “Akyio [CIF]”, “Foreman [QCIF]” e “Silent [QCIF]”. A diferença média de PSNR (a) e o aumento médio de taxa de bits (b) são plotados em relação à redução de complexidade.

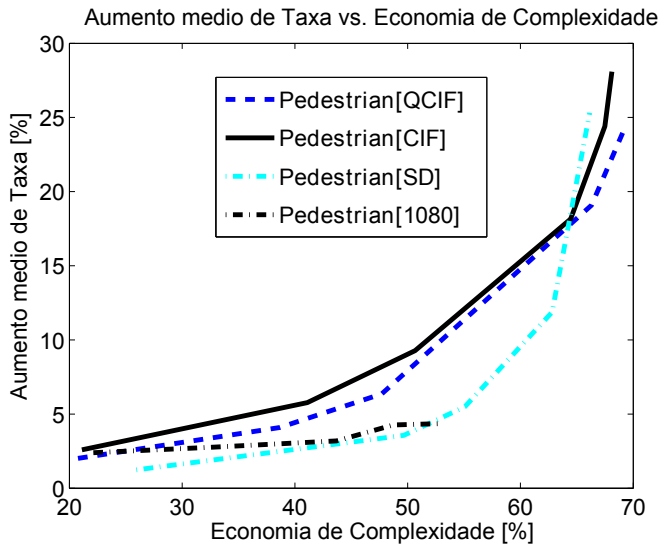


(a)

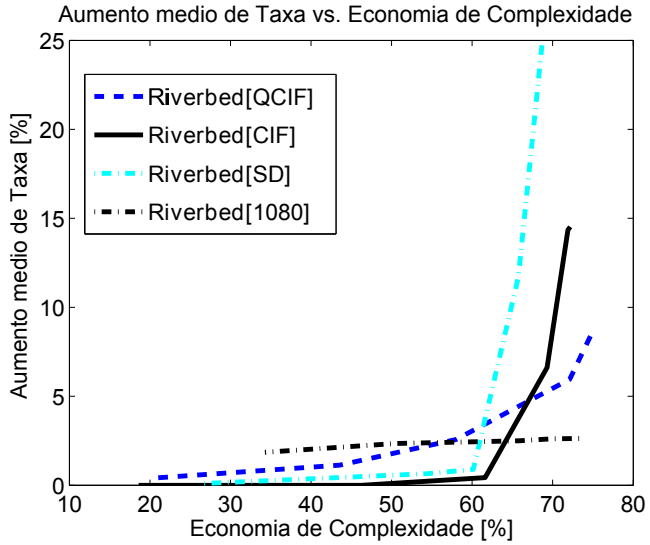


(b)

Figura 4.7: Avaliação de desempenho RD por diferenças médias para diferentes patamares de redução de complexidade de codificação. As sequências de testes utilizadas: “Pedestrian [1080p]”, “Riverbed [1080p]”, “Rushhour [1080p]” e “Sunflower [1080p]”. A diferença média de PSNR (a) e o aumento médio de taxa de bits (b) são plotados em relação à redução de complexidade.



(a)



(b)

Figura 4.8: Aumento médio de taxa *vs.* redução de complexidade para duas sequências de vídeo em diferentes resoluções espaciais: (a) “Pedestrian Area” e (b) “Riverbed”.

4.4 CONSIDERAÇÕES

Discutiu-se uma plataforma de escalonamento de complexidade usando a implementação de referência do codificador H.264/AVC que funciona para a maioria das aplicações. Em vez de testar todos os modos de predição disponíveis, analisaram-se apenas os modos dominantes (aqueles que são mais recorrentes) enquanto determinada provisão de complexidade é respeitada. A estimação é feita por meio da busca de todos os modos de predição disponíveis sobre uma população de referência; somente os modos dominantes daí derivados serão considerados na codificação dos macroblocos não pertencentes à população de referência do próximo quadro. Os testes apresentaram penalidades pequenas em desempenho RD devido ao descarte de busca de modos não-dominantes, enquanto reduções de complexidade expressivas são alcançadas.

A metodologia aqui exposta escalona a complexidade com restrições e pode ser usada como ponto de partida para o controle de complexidade em implementações otimizadas, oferecendo desempenhos melhores em termos de velocidade de codificação, conforme será visto nas próximas seções.

5 CONTROLE DE COMPLEXIDADE DO H.264/AVC COM TREINAMENTO *OFF-LINE* E IMPLEMENTAÇÕES DE ALTO DESEMPENHO

Este capítulo apresenta uma metodologia para controle de complexidade do codificador H.264/AVC com treinamento off-line *e que utiliza implementações de alto desempenho computacional.*

5.1 CONSIDERAÇÕES INICIAIS

A metodologia apresentada no Capítulo 4 foi avaliada a partir da implementação de referência do H.264/AVC, o codificador JM. Concomitante ao estabelecimento do padrão H.264/AVC, surgiram, no mercado, novas implementações em *software* do codificador. Essas realizações foram implementadas com restrições de projeto diferentes das tratadas pelo JM, o que resultou, por exemplo, num cuidado maior na otimização das rotinas mais intensas computacionalmente. Serão objeto de estudo neste capítulo duas implementações otimizadas do H.264/AVC:

1. O codificador disponibilizado pelas bibliotecas de primitivas de alto desempenho da empresa Intel® [73] (Intel® IPP, Intel *Performance Primitives*), aqui chamado de IPP.

2. O codificador de vídeo em codigo fonte aberto, x264 [76].

As próximas seções tratarão da análise das particularidades dessas duas realizações e, para cada uma delas, será apresentada uma metodologia de controle de complexidade com prévio treinamento *off-line*. Um diferencial das contribuições deixadas nesse capítulo será o projeto e a implementação de um módulo de controle de velocidade de codificação para o codificador H.264/AVC, em que a velocidade é ajustada em valores absolutos (quadros por segundo).

5.2 O CODIFICADOR IPP

O codificador H.264/AVC disponibilizado pela Intel® em seu produto Intel® IPP, doravante chamado de codificador IPP, foi implementado a partir de uma biblioteca de primitivas de alto desempenho. As primitivas aqui tratadas consistem em funções implementadas em *software* para atender demandas de aplicações de multimeios, de processamento de sinais e de comunicações. A intenção do produto é oferecer um conjunto otimizado de funções frequentemente utilizadas em algoritmos fundamentais desse tipo de aplicação.

A estratégia adotada pela Intel® é disponibilizar uma biblioteca capaz de ativar os múltiplos núcleos de processamento dos modernos processadores e utilizar de forma extensiva o conjunto de instruções e sub-sistemas de aumento de desempenho, como o *pipeline* de instruções. Por meio de otimizações de baixo-nível, frequentemente usando linguagem de montagem (*assembly*), a empresa afirma que sua biblioteca consegue ganhos de desempenho superiores aos que os modernos compiladores podem alcançar.

Entre o conjunto de artefatos utilizados para o tratamento com a biblioteca Intel® IPP, a empresa disponibilizou sua implementação em código fonte aberto do padrão H.264/AVC. Naturalmente, muitas funções primitivas (e de acesso a código fonte restrito) são empregadas na implementação; entretanto, blocos funcionais básicos podem ser identificados e alterados de maneira a intervir na complexidade computacional do codificador. Trabalhos datados de 2010 apontam o codificador IPP como capaz de comprimir até 480 vezes mais rápido um vídeo digital quando comparado com a versão mais recente do *software* de referência JM [79].

5.3 ESCALABILIDADE EM COMPLEXIDADE USANDO O CODIFICADOR IPP

5.3.1 Implementação H.264/AVC da Intel®

O codificador IPP é disponibilizado pela Intel® IPP por meio de códigos fonte exemplos de casos de uso de sua biblioteca de primitivas de alto desempenho. O aplicativo `umc_video_enc_con` apresenta a interface para um codificador H.264/AVC em perfil HIGH [16]. Além dos parâmetros de compressão presentes em outras implementações do codificador H.264/AVC, a implementação IPP oferece um conjunto extra de parâmetros para escalonar a complexidade computacional e, portanto, a distorção de uma sequência de vídeo comprimida.

O primeiro dos parâmetros adicionais, `Sub-block split`, é responsável pelo controle de como o codificador irá percorrer o esquema de particionamento de macroblocos em estrutura *quadtree* durante as predições Inter quadros. Há três opções:

(i) permitir compensação de movimentos usando apenas blocos de 16×16 *pixels*;

(ii) permitir compensação de movimentos usando blocos de tamanhos até 8×8 *pixels* e

(iii) permitir compensação de movimentos usando blocos de tamanhos até 4×4 *pixels*.

`Speed/Quality Grade` é o segundo parâmetro, responsável pela habilitação/desabilitação de ferramentas de otimização de forma que seja possível trocar velocidade por ganho de compressão. Sua função é gerenciar o processo de otimização RD, a precisão da estimação de movimentos (o nível de refinamento sub-*pixel*) e o tamanho do *buffer* de quadros de referência.

Durante a codificação, ele é representado por uma variável inteira. É permitido ao usuário ajustar a sua faixa de valores de acordo com a lista abaixo:

- `0`: As funções de custo são avaliadas a partir da SAD, *Sum of Absolute Differences*.
- `1`: Além da opção anterior, inclui partições Intra-quadro nas predições de quadros P.
- `2`: Além da opção anterior, inclui a estimação de movimentos independentes para os planos de crominância e os custos de um macrobloco em termos de taxa e distorção.
- `3`: Além da opção anterior, inclui quadros B e ativa a otimização das predições usando otimização por taxa *vs.* distorção.
- `4`: Além da opção anterior, permite que a estimação de movimentos seja realizada em precisões fracionárias de *pixels*.
- `5`: Além da opção anterior, permite que a utilização de todos os quadros de referência na estimação de movimentos.

O último parâmetro de relevância, `Optimal Quantization`, é basicamente auto-explicativo: habilita ou desabilita o processo de otimização na quantização dos coeficientes transformados.

Os parâmetros expostos acima manipulam o desempenho do estágio de predição e podem ser ajustados para prover escalabilidade em complexidade. A seguir, apresenta-se uma alteração aplicada à semântica do parâmetro `Sub-block split` para restringir de maneira suave a complexidade de codificação.

5.3.1.1 Refinamento do parâmetro *Sub-block split*

O parâmetro `Sub-block split` é limitado originalmente quanto ao arranjo possível de partições de predição a que um macrobloco pode ser submetido. O codificador IPP teve suas partições arranjadas em uma árvore do tipo *quadtree* que é cortada para reduzir a complexidade computacional. A ideia básica é que, quanto mais próximo da raiz for aplicado o corte (ou quanto maior for o tamanho das partições de macrobloco a ser testado), menores serão as opções de testes de predições disponíveis.

As alterações propostas visam a refinar a forma como se descartam os modos de predição. Em vez de ficar vinculado ao conceito de *quadtree*, a contribuição proposta por esta tese consiste em utilizar subconjuntos das partições disponíveis utilizando arranjos de partições agrupados em ordem crescente de complexidade.

Convém definir alguns símbolos empregados na modificação proposta do parâmetros. Sejam:

- I16MBCDONLY: macrobloco de 16×16 *pixels* predito por técnica Intra-quadro que usa no contexto apenas o nível DC dos macroblocos vizinhos;
- I16MB: macrobloco de 16×16 *pixels* predito por técnica Intra-quadros;
- I8MB: macrobloco de 8×8 *pixels* predito por técnica Intra-quadros;
- I4MB: macrobloco de 4×4 *pixels* predito por técnica Intra-quadros;
- ZMV: macrobloco de 16×16 *pixels* predito por técnica Inter-quadros e vetor de movimentos nulo (predição pelo macrobloco colocalizado no quadro de referência);
- P16x16: macrobloco de 16×16 *pixels* predito por técnica Inter-quadros;
- P16x8: macrobloco de 16×8 *pixels* predito por técnica Inter-quadros;
- P8x16: macrobloco de 8×16 *pixels* predito por técnica Inter-quadros;
- P8x8: macrobloco de 8×8 *pixels* predito por técnica Inter-quadros;
- P8x4: macrobloco de 8×4 *pixels* predito por técnica Inter-quadros;
- P4x8: macrobloco de 4×8 *pixels* predito por técnica Inter-quadros;
- P4x4: macrobloco de 4×4 *pixels* predito por técnica Inter-quadros.

Assume-se que partições de macroblocos maiores são menos complexas computacionalmente, já que é menor o número de subunidades necessárias para montar o macrobloco. Além disso, são demandados menos parâmetros na sua codificação. Em contrapartida, uma partição menor é capaz de ajustar-se melhor às particularidades do sinal de vídeo, mas exigirá um número maior de subunidades na montagem de um macrobloco e, para cada uma dessas subunidades, haverá um número maior de parâmetros de codificação.

Exceto por I16MBCDONLY e ZMV, todos os outros elementos possuem um representante sintático no padrão H.264/AVC. I16MBCDONLY foi escolhido como sendo a partição mais simples para a codificação de um macrobloco usando prediçao Intra-quadros; ZMV, por sua vez, é a versão mais simples para codificação Inter-quadros.

Para aumentar a granularidade das escalas de complexidade originalmente fornecidas pelo IPP, em vez de usar o corte em árvore *quadtree* em nós quadrados, como originalmente implementado no codificador IPP, o parâmetro `Sub-block split` foi estendido para adotar os valores da Tabela 5.1.

Tabela 5.1: Tabela de valores estendidos do parâmetro *Sub-block split*.

Valor	**Descrição**
`-3`	predição sem estimação de movimentos, usando apenas I16MBCDONLY
`-2`	predição sem estimação de movimentos, usando I16MB
`-1`	predição sem estimação de movimentos, usando I16MB, I8MB e I4MB
`0`	predição com estimação de movimentos, usando apenas ZMV
`1`	predição com estimação de movimentos, usando ZMV e I16MB, I8MB e I4MB
`2`	predição usando P16x16, I16MB, I8MB e I4MB
`3`	predição usando P16x16, P8x8, I16MB, I8MB e I4MB
`4`	predição usando P16x16, P16x8, P8x8, I16MB, I8MB e I4MB
`5`	predição usando P16x16, P16x8, P8x16, P8x8, I16MB, I8MB e I4MB
`6`	predição usando P16x16, P16x8, P8x16, P8x8, P4x4, I16MB, I8MB e I4MB
`7`	predição usando P16x16, P16x8, P8x16, P8x8, P8x4, P4x4, I16MB, I8MB e I4MB
`8`	predição usando todas as partições disponíveis

5.3.2 Abordagem de restrição de complexidade

A metodologia para compressão de vídeo HD com complexidade restrita consiste no controle do esforço computacional gasto na codificação de uma sequência pelo ajuste de parâmetros de codificação de forma que as quedas em desempenho RD sejam as menores possíveis em relação à compressão feita com o codificador ajustado com todas as ferramentas de desempenho RD habilitadas. A complexidade do codificador é medida pelo tempo de gasto na codificação de uma sequência de vídeo por um computador pessoal que execute o compressor IPP.

Inicialmente, modificou-se o parâmetro **Sub-block Split**, que é responsável pela forma como o codificador percorre o esquema baseado em *quadtree* para a determinação da partição de macrobloco na predição Inter. A alteração inseriu mais opções de valores de forma a ter escala mais fina de impacto em complexidade ao se manipular esse parâmetro.

A abordagem empregada estende a otimização RD [80] por meio de sua generalização com a adição de outro eixo (eixo C, de complexidade) ao bem conhecido problema bidimensional de encontrar a melhor representação de sinal de vídeo que utilize a menor taxa de canal (R) sobre determinado nível de distorção (D).

Restringiu-se o escopo para a avaliação do codificador IPP com a varredura dos seguintes parâmetros afetos à codificação:

- tamanho da janela de estimação de movimentos, configurada nos valores de `1`, `4`, `8`, `16`, `32`, `64`, `128`;
- valor do parâmetro de quantização QP, configurado nos valores de `23, 28, 33, 38`;
- `Sub-block split`, varrido entre os valores de `-3` a `8`;
- `Speed/Quality Grade`, varrido entre os valores de `0` a `5`;
- `Optimal Quantization`, habilitado ou não.

Para cada seleção k das combinações dos parâmetros listados, calcula-se a taxa total (R), a PSNR (D) média e a razão $C = \frac{T_k}{T_{\text{melhor } RD}}$ entre o tempo T_k para codificar a sequência de treinamento e o tempo $T_{\text{melhor } RD}$ gasto quando todos os parâmetros de ajustes do compressor são habilitados para obtenção de melhor desempenho RD[1]. Os pontos RDC são usados para montar um espaço amostral do qual

[1] O valor de $C = 1$ ou $C = 100\%$ representa o maior valor de complexidade computacional, que é demandado pela

extrairemos as configurações que gerem os pontos que compõem o casco convexo inferior (LCH, do inglês *lower convex hull*), conforme ilustrado na Figura 5.1 para taxas constantes. Depois de encontradas as combinações de parâmetros que pertencem à frente de Pareto [81] (aqui composta pelos pontos que, além de proverem o melhor desempenho RD, são os que demandam os menores tempos para a compressão do sinal), deriva-se uma tabela de consulta (do inglês, *look-up table*) que lista valores ótimos em termos de RDC. Os pontos intermediários não encontrados na tabela são alcançados mediante interpolação dos valores vizinhos.

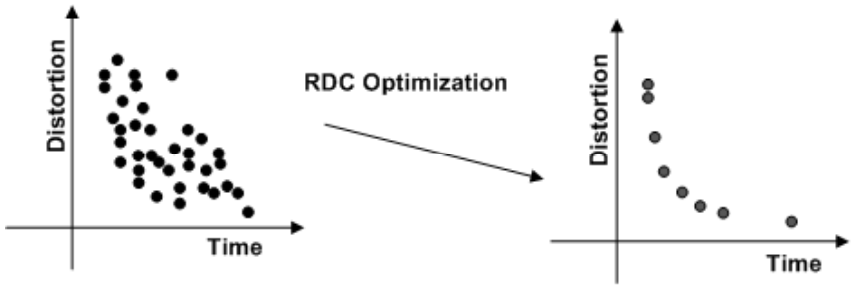


Figura 5.1: Busca pelo conjunto de pontos que compõem a frente de Pareto. Para essa figura, a taxa foi mantida constante.

Como resultado do processo de otimização, insere-se um parâmetro ao codificador H.264/AVC do IPP, o parâmetro **QPC**, *Quantization Parameter for Complexity*, uma analogia à variável **QP** encontrada em codificação de vídeo digital. **QPC** assume valores entre 20 e 100, onde o valor 100 representa compressão sem economia de esforço computacional (100% de complexidade).

5.3.3 Resultados de codificação restrita por complexidade

Aplicou-se o método proposto no codificador IPP, configurado com estimação de movimento UMHS e um quadro de referência. A complexidade C foi tomada pela razão entre o tempo total da codificação para a configuração de complexidade reduzida e o tempo demandado pelo codificador com todos os parâmetros de configuração ajustados para o melhor desempenho em termos de RD[2]. Em princípio, avaliou-se o desempenho do codificador na compressão de conjuntos de sequências de treinamento na resolução 720p (1280×720 *pixels*); dentre as sequências, usou-se “Shields”, “Parkrun” e “Mobile and Calendar”. Testaram-se diferentes parâmetros para ajuste da codificação: QP, tamanho da janela de estimação de

configuração que melhor comprime o vídeo em termos de RD: menor taxa de bits e maior qualidade possível. Espera-se que outras configurações do codificador resultarão em desempenho RD inferior a $C = 100\%$.

[2]O melhor desempenho RD representa $C = 100\%$.

movimentos, precisão da análise na otimização RD, quantização otimizada de coeficientes transformados e diversos subconjuntos de particionamento de macroblocos.

Para cada combinação dos parâmetros de compressão, comparou-se o desempenho do codificador alterado em relação ao codificador original com todas as ferramentas habilitadas ($C = 100\%$, que corresponde a $2,7$ fps, *frames per second*, para a plataforma de testes) por meio da avaliação das diferenças médias de taxa e de PSNR de acordo com a metodologia discutida na Seção 2.5 [61]. Isso fez-se necessário pois as curvas RD são muito próximas uma das outras.

Do espaço amostral dos valores de desempenho, delimita-se a frente de Pareto, ilustrada na Figura 5.2. O comportamento geral sugere que, na medida em que se reduz a complexidade para codificar sinais de vídeo, as quedas em desempenho tendem a aumentar. Há uma região em que o codificador modificado consegue ser capaz de ultrapassar o desempenho de 100% de esforço computacional. Isso é devido ao fato de a restrição da complexidade computacional permitir ao codificador de entropia explorar melhor os custos RD dos poucos elementos sintáticos utilizados, além de proporcionar redução significativa no tempo usado para comprimir determinada sequência.

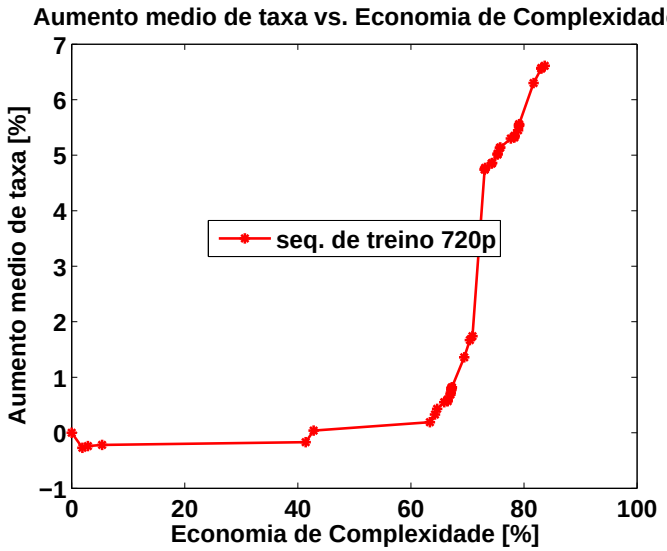


Figura 5.2: Média de degradação de desempenho *vs.* economia de complexidade para sequências de treinamento. Dada uma complexidade $C \leq 100\%$, a economia de complexidade é calculada por $100\% - C$. O valor $C = 100\%$ corresponde a $2,7$ fps em um computador portátil com processador Centrino 2® (frequência de relógio de $2,4$ GHz) e 4GB de memória RAM.

A Figura 5.3 apresenta a curva de queda de desempenho para as sequências de testes em resolução 720p para a faixa de redução de complexidade a partir de 40%. Escolheu-se este ponto de partida pois o

processo de otimização RDC identificou o ponto de melhor desempenho RD nesta região. Em termos gerais, observam-se pequenas perdas em desempenho RD (aumento médio de taxa de canal ao redor de 6%) ao se comprimir sequências em esforço computacional reduzido. Nota-se que o comportamento da curva é similar ao obtido para o conjunto de treinamento; a exceção fica por conta da sequência 720p "Rushhour", cujos valores são ligeiramente inferiores aos observados no processo de treinamento. Para as sequências de testes, o patamar de 100% de complexidade equivale a $2,3$ fps ("Stockholm"), $4,0$ fps ("Sunflower") e $3,0$ fps ("Rushhour") em nossa plataforma, um computador portátil com processador Centrino 2® (frequência de relógio de $2,4$ GHz) e 4 GB de memória RAM.

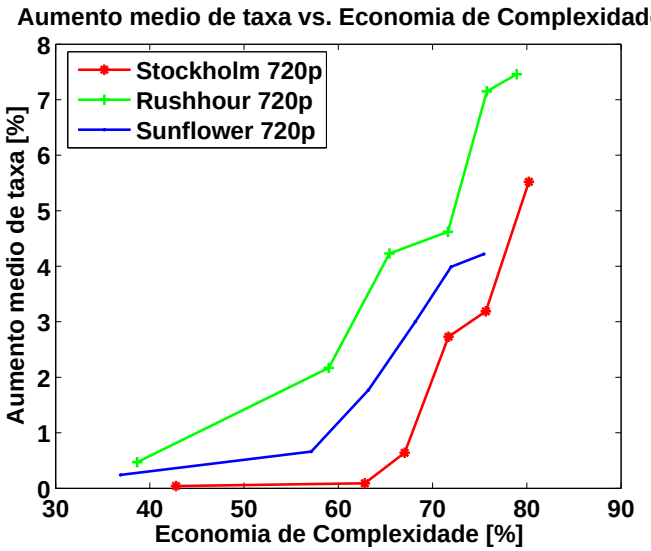


Figura 5.3: Aumento médio em taxa *vs.* economia de complexidade para sequências de vídeo 720p. Dada uma complexidade $C \leq 100\%$, a economia de complexidade é calculada por $100\% - C$.

O resultado principal dessa metodologia é um codificador H.264/AVC com velocidades de interesse prático. Pelo emprego dos resultados RDC, é possível extrair combinação de parâmetros de configuração que permita a maior velocidade de codificação atingida pelo implementação IPP do H.264/AVC para cada QP. Desta maneira, os resultados aparecem ligeiramente diferentes daqueles mostrados na Figura 5.2.

As curvas RD para o codificador com velocidade de aplicação prática (denominadas H.264-IPP RT) são apresentadas e comparadas ao melhor desempenho alcançado (H.264-IPP Opt, na Figura 5.4). Os aumentos médios de taxa de canal ficam em volta de 5% para "Stockholm" e "Sunflower", com valores um pouco superiores para "Rushhour". Velocidade média de compressão de 15fps, com velocidade máxima de 20fps em baixas taxas, foi o melhor desempenho disponibilizado pela metodologia. Codificação rápida em baixas taxas era esperada pois os melhores modos para codificação dos macroblocos ficam agrupados

nas maiores partições para sequências HD e QPs elevados [78]. Apesar disso, essas velocidades podem ser aproveitadas em sistemas de videoconferência que usem computadores pessoais e câmeras capazes de realizar *trade-off* entre taxa de quadros capturados e resolução de quadro.

5.4 O CODIFICADOR X264

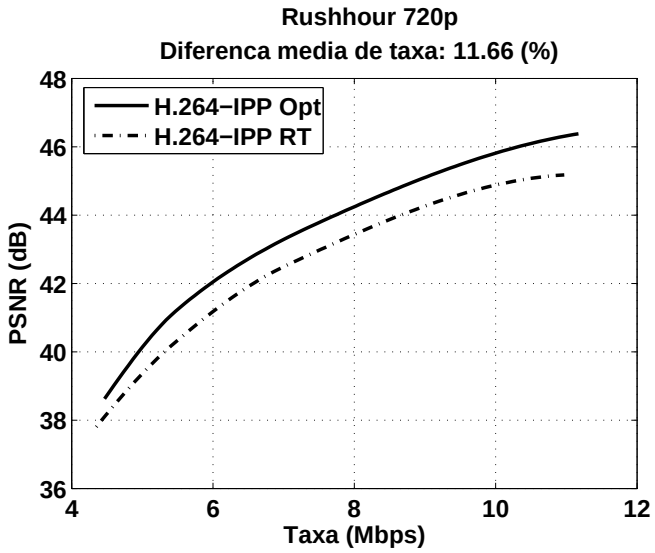
O codificador x264 é uma implementação do padrão H.264/AVC em código fonte aberto. Uma de suas grandes vantagens foi a implementação do perfil HIGH para sequências coloridas codificadas em YUV 4:2:0 [1]. Durante o desenvolvimento dessa realização, a otimização quanto à velocidade constituía uma importante restrição de projeto e, por volta de 2007, o x264 foi relatado 50 vezes mais rápido que a implementação de referência do padrão [76].

Naturalmente, o grande ganho em termos de tempo de computação foi resultado de um conjunto de fatores, dos quais destacamos:

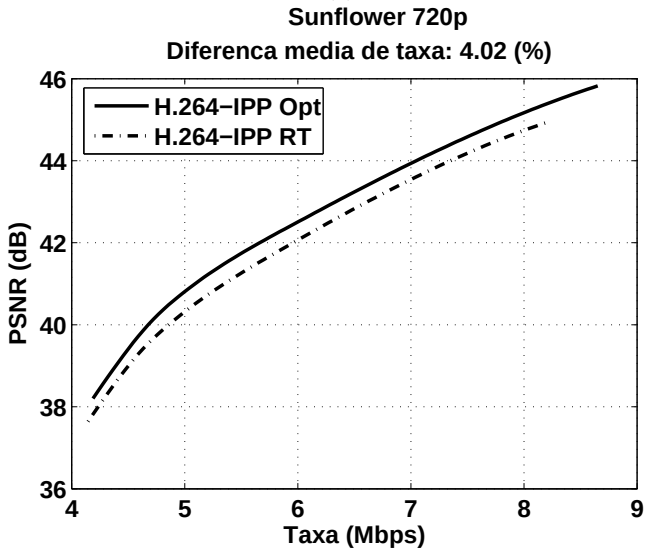
- uso de linguagem de montagem (*assembly*) na codificação de seções críticas de programa;
- programação *multithreaded*;
- implementação de técnicas modernas de estimação de movimentos;
- aplicação de heurísticas de paradas antecipadas na decisão do melhor modo de predição em termos de RD.

Entre as ferramentas disponibilizadas pelo x264, destacam-se:

- codificadores de entropia CAVLC e CABAC (demandado para perfil HIGH);
- possibilidade de ajustes personalizados das tabelas de quantização;
- controle de taxa de bits de canal;
- algoritmo de detecção de corte de cena;
- otimizações psicovisuais para retenção de detalhes das cenas.



(a)



(b)

Figura 5.4: Comparação de desempenho entre codificador de melhor desempenho e sua versão de velocidade de compressão de interesse prático: (a) “Rushhour” e (b) “Sunflower”. A redução de complexidade percebida nas curvas H.264-IPP RT foi de aproximadamente 80% para as duas sequências.

5.5 CONTROLE DE COMPLEXIDADE USANDO IMPLEMENTAÇÃO DE ALTO DESEMPENHO E EM CÓDIGO ABERTO: X264

5.5.1 Codificador de vídeo de complexidade controlada

A estratégia sugerida para controle de complexidade para codificação em tempo real com x264 também consiste no controle do esforço computacional durante a codificação de um sinal de vídeo. O controle é feito por meio de ajustes dos parâmetros do codificador para que as perdas de desempenho em termos de taxa$\times$distorção (RD) sejam as menores possíveis. Como medida de complexidade computacional, mede-se o tempo gasto na codificação de uma sequência de vídeo pelo codificador, que é processado em um computador pessoal.

5.5.2 Metodologia de otimização e controle de complexidade de codificação

A abordagem resume-se a um processo de otimização RDC. Para cada ajuste k de configuração do codificador, calculam-se a taxa total (R), a PSNR média (D) e a razão $C = \frac{T_k}{T_{\text{melhor } RD}}$ entre o tempo T_k para codificar a sequência de treinamento e o tempo $T_{\text{melhor } RD}$ gasto pelo codificador para a realização da mesma tarefa usando, contudo, todas as ferramentas disponibilizadas pelo padrão H.264/AVC[3]. Os pontos RDC coletados preencherão um espaço amostral que será submetido à seleção de pontos que ocupam o casco convexo inferior (confira a Figura 5.1). Encontrados os pontos que pertencem à frente de Pareto [81] deste critério de otimização, constrói-se tabela de valores a partir dos números de desempenho. Esta tabela lista pontos ótimos em termos de RDC para ajuste do codificador. Pontos intermediários de complexidade não encontrados numa consulta na tabela de valores são facilmente derivados por meio de interpolação de seus vizinhos de forma que a complexidade resultante seja muito próxima daquela solicitada.

Montada a tabela, prossegue-se com a calibração do sistema de forma que o codificador tenha associada a si determinada velocidade de compressão (um valor absoluto), medida como a quantidade de quadros por segundo (fps, *frames per seconds*) que o mesmo é capaz de processar. É relevante salientar que os valores de complexidade inicialmente derivados partem da razão entre as configurações ótimas RDC e o patamar de referência de desempenho alcançado pelo codificador com a maioria de suas ferramentas de análise habilitadas. Uma vez calibrado, o sistema é capaz de comprimir vídeo otimamente em termos

[3]Novamente, o melhor desempenho RD representa $C = \frac{T_{\text{melhor } RD}}{T_{\text{melhor } RD}} = 100\%$. Espera-se que outros pontos representem uma fração menor que 100%.

de RD e restrito a um valor de C ($C_{\text{usr.}}$, na Figura 5.5), demandado pelo usuário em fps e não mais em valores relativos de complexidade. Uma malha de controle é anexada ao *codec* de forma a garantir que a compressão não desrespeite determinada alocação de tempo destinada à codificação, conforme o ilustrado na Figura 5.5. O atuador de controle recebe diferenças entre a complexidade obtida e o valor $C_{\mathbf{usr}}$ indicado pelo usuário, a partir das quais decide alterar os parâmetros de configuração do codificador. Dessa forma, troca-se a velocidade de compressão por desempenho RD. Não se usa descarte de quadros (*frame skipping*) nessa estratégia.

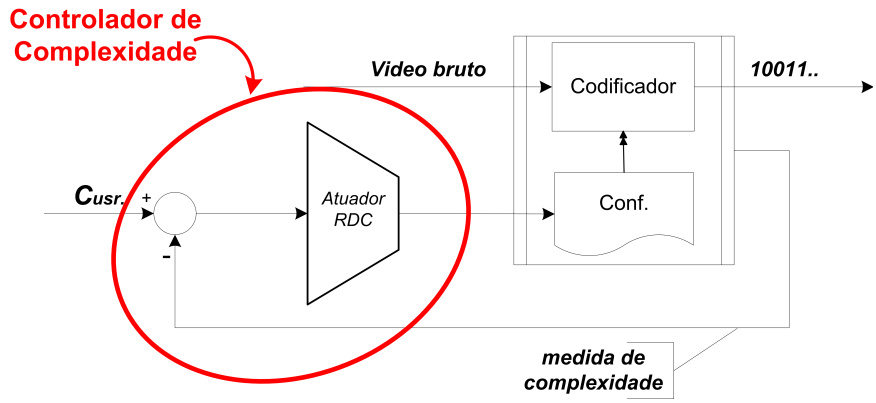


Figura 5.5: Controlador de complexidade. A complexidade demandada pelo usuário é representada por $C_{\mathbf{usr}}$.

5.5.3 Implementação do codificador x264 com controle de velocidade de codificação

5.5.3.1 Treinamento e otimização do codificador

Para a implementação do controlador de velocidade para o codificador x264, o sistema é submetido a uma etapa de treinamento para coleta de dados necessários no processo de otimização RDC. Partindo de sequências de treinamento nas resoluções CIF e 720p, executou-se o codificador x264 configurado com diferentes ajustes de parâmetros de seu módulo de predição. Como parâmetros fixos, optou-se pelo uso de estimação de UMHS, janela de estimação de movimentos de 64 *pixels*, dois quadros de referência, uma *thread* de codificação e quadros B desabilitados. Um segundo conjunto de parâmetros, aqui chamados de parâmetros para livre escolha, foi submetido à varredura durante o processo de treinamento. Esse grupo é formado por:

- a taxa de canal desejada (medida em bps);
- o uso de quantização otimizada dos coeficientes transformados;

- `subme`, a precisão dos vetores de movimentos no processo de estimação de movimentos:
  (a) vetores de movimentos inteiros;
  (b) uma iteração de estimação de movimento com vetores com precisão de um quarto de *pixel* e
  (c) estimação de movimento "completa" com vetores com precisão de um quarto de *pixel*.
- `trellis`, o nível de RDO (*Rate*$\times$*Distortion Optimization*) empregado na decisão das melhores partições[4]:
  (a) decisões apenas pela menor distorção;
  (b) decisão RDO por melhor partição apenas para os candidatos mais prováveis no final da codificação de um macrobloco e
  (c) decisão RDO por melhor partição em todos os testes de partições de um macrobloco.
- `partitions`[5], as possibilidades de partições de macrobloco, de acordo com as seguintes regras:
  (a) compensação de movimentos usando apenas blocos de 16×16 *pixels*;
  (b) compensação de movimentos usando blocos de tamanhos até 8×8 *pixels* e
  (c) compensação de movimentos usando blocos de tamanhos até 4×4 *pixels*.

A complexidade de cada execução da etapa de treinamento foi calculada pela razão entre o tempo total da codificação para a configuração de complexidade reduzida e o tempo gasto pelo codificador com todos os parâmetros de configuração ajustados para o melhor desempenho em termos de RD.

Os testes e resultados na resolução CIF foram executados em uma plataforma com dois processadores AMD® Opteron® de duplo núcleo enquanto os testes e resultados para 720p, em um processador Intel® Core i7® com quatro núcleos físicos e 8 núcleos lógicos[6].

Para cada combinação de valores do conjunto de parâmetros de configuração do codificador, o desempenho foi comparado ao do caso completo, correspondente a 100% de complexidade (9,0 fps para sequências CIF e 7,0 fps para sequências 720p). Médias de diferenças das taxas de canal e da PSNR entre as curvas de complexidade reduzida e de complexidade integral foram calculadas a partir de análises

[4] Atua em conjunto com o valor de *subme*.

[5] Permite a seleção das partições habilitadas durante a codificação. Durante a varredura, optou-se por sempre incluir partições Intra-quadros e, quanto às partições Inter-quadros, aplicou-se um esquema de corte *quadtree* semelhante ao originalmente fornecido pelo IPP no parâmetro *Sub-block split*, conforme descrito na Seção 5.3.1.

[6] Tecnologia Intel® *HyperThreading*.

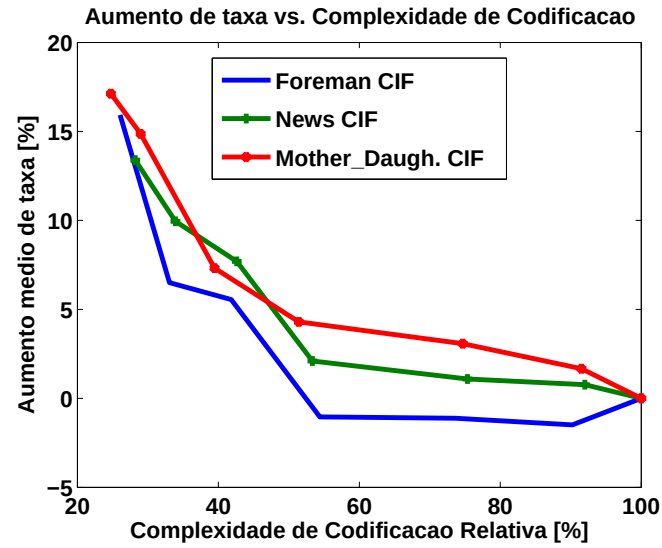
comparativas entre as curvas RD de acordo com a metodologia discutida na Seção 2.5 [61]. Esses valores foram plotados em um espaço tridimensional de eixos RDC e, em seguida, submetidos ao processo de otimização por RDC com a descoberta dos pontos pertencentes à frente de Pareto.

5.5.3.2 Avaliação de codificador otimizado e controlado por complexidade

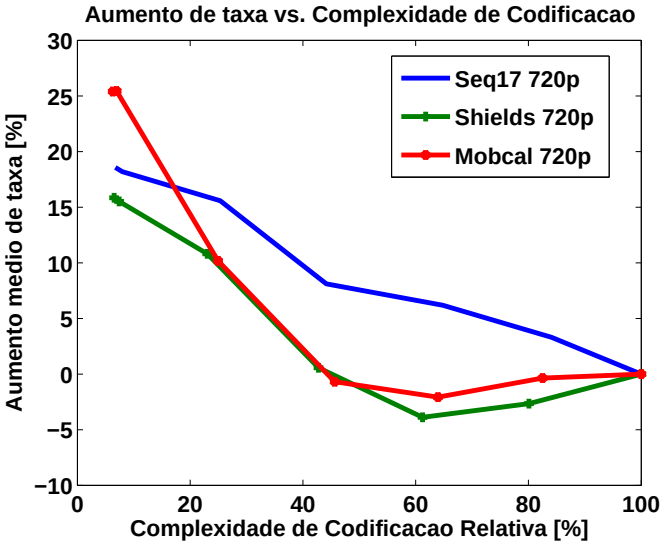
Depois da etapa de treinamento e otimização de parâmetros de codificação, executou-se o codificador otimizado em um conjunto de sequências de testes e obtiveram-se as curvas das Figuras 5.6(a) e (b), que ilustram a queda de desempenho do compressor em relação à configuração de complexidade integral (100% de complexidade) para o intervalo que se inicia em 20% de redução de complexidade (ou 80% de complexidade total). Num retorno à análise apresentada no Capítulo 2, nota-se que a Tabela 3.5, da Seção 3.3, aponta um limite inferior de complexidade em torno de 10% do total demandado pelo codificador x264. Contudo, para atingir esse patamar, é necessário remover completamente o estágio de predição do compressor de vídeo. Deste modo, o estágio de predição do codificador otimizado ainda contribui com uma pequena parcela de esforço computacional.

Nos gráficos apresentados, observa-se que o ponto ótimo de operação em termos de RD não é atingido pela configuração de 100% de complexidade para algumas das sequências em avaliação. Isso é consequência do fato de que a restrição do processo de otimização RD a um subconjunto de elementos sintáticos permite ao *codec* explorar melhor os custos dos mesmos durante a codificação entrópica além de evidenciar o fato de que a sequência usada nas etapas de treinamento não ser tão representativa como se esperava.

O compressor otimizado por RDC foi inserido no esquema de controle de complexidade mostrado na Figura 5.5. A partir de um valor de velocidade de compressão indicado pelo usuário em quadros por segundo (fps), o esquema de controle atua reconfigurando o codificador sempre que o valor medido de velocidade de compressão for diferente do especificado como patamar de referência. Se o valor de velocidade de compressão for maior que o especificado (saldo de complexidade), o atuador da Figura 5.5 ajusta o codificador para configurações que resultam em melhor desempenho em termos de RD, mas são mais demoradas; por outro lado, se o codificador estiver mais lento que o esperado (débito de complexidade), reconfigura-se o compressor de forma que ele simplifique seu estágio de predição, resultando em codificação mais rápida do sinal de vídeo. Dessa maneira, o controlador garante que,



(a)



(b)

Figura 5.6: Aumento médio de taxa vs. complexidade relativa para sequências (a) CIF e (b) 720p. A complexidade relativa C é a razão $\frac{T}{T_{\text{melhor } RD}}$ entre o tempo T para codificar a sequência de testes e o tempo $T_{\text{melhor } RD}$ gasto pelo codificador para a realização da mesma tarefa usando, contudo, todas as ferramentas disponibilizadas pelo padrão H.264/AVC.

na média, o patamar $C_{\mathbf{usr}}$ indicado pelo usuário será respeitado, com leves flutuações entre pontos pertencentes ao casco convexo inferior do espaço RDC.

As Figuras 5.7 e 5.8 apresentam curvas RD para o codificador ajustado em diferentes velocidades de compressão. Como esperado, o desempenho em termos de RD tende a ser apenado na medida em que a velocidade de codificação é aumentada. Todavia, as curvas são basicamente justapostas; mesmo no pior caso (60 fps para sequências CIF e 40 fps para 720p), o aumento da distorção gira em volta de 1dB.

Salienta-se que a curva rotulada por **Original** nos gráficos das Figuras 5.7 e 5.8 representa a configuração de complexidade $C = 100\%$, ou seja, a melhor curva em termos de RD para as sequências em discussão. Nessa curva, o controlador está desligado, permitindo que o codificador use toda a complexidade computacional disponível. Ao ajustar uma referência de complexidade num esquema de controle em malha fechada, fixando um valor para $C_{usr.}$, o controlador tentará seguir o valor demandado. Isso provoca o achatamento da curva de velocidade de compressão *vs.* taxa de bits.

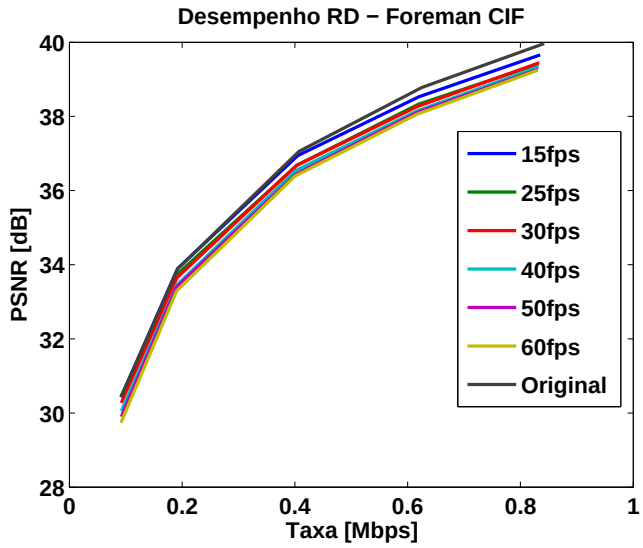
5.6 CONSIDERAÇÕES

Foi introduzido o conceito de otimização RDC neste capítulo, com o treinamento *off-line* para determinação do LCH. Duas estratégias de implementação para codificadores do padrão H.264/AVC foram discutidas. A primeira realização empregou biblioteca proprietária de alto desempenho específica para processadores Intel®. Apesar de resultados modestos, a metodologia serviu de prova de conceito na troca de complexidade computacional por desempenho RD na codificação em *software* de sequências de vídeo HD para aplicações práticas como videoconferência.

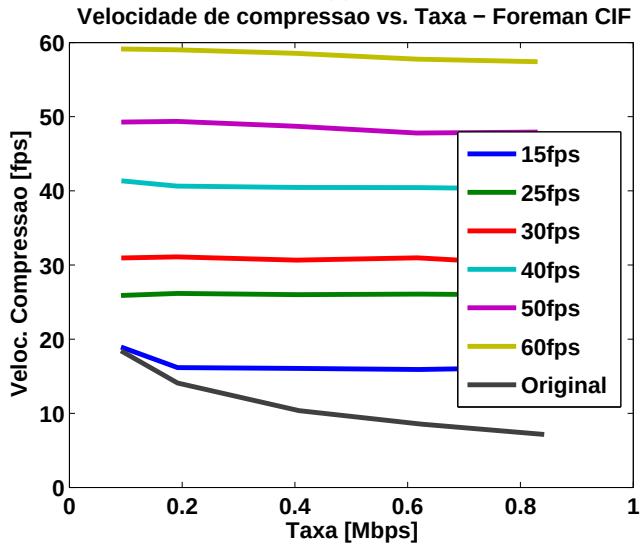
Cabe destacar que, por volta do final do ano de 2009, a biblioteca Intel® IPP passou a permitir a utilização de múltiplas *threads* para computação, o que permite a melhor utilização dos múltiplos núcleos de processamento presentes nos novos processadores. Com essa atualização técnica, espera-se que os números de velocidade de computação sejam melhorados, o que viabilizaria codificação em tempo real para conteúdos HD.

Acompanhando o lançamento das novas plataformas de processamento, convém comentar que esse fabricante disponibilizou, recentemente, uma versão de biblioteca para tratamento de informações multimeios capaz de comprimir vídeo digital em tempo real, a Intel® Media SDK 2012[7], capaz de usufruir

[7]Disponível em: http://software.intel.com/en-us/articles/vcsource-tools-media-sdk/

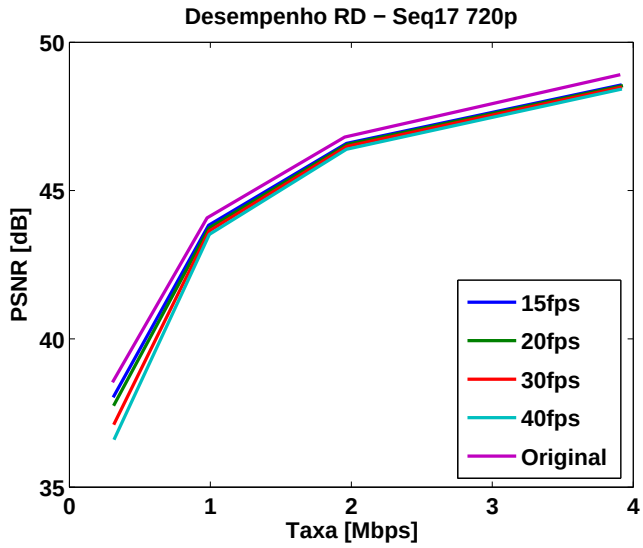


(a)

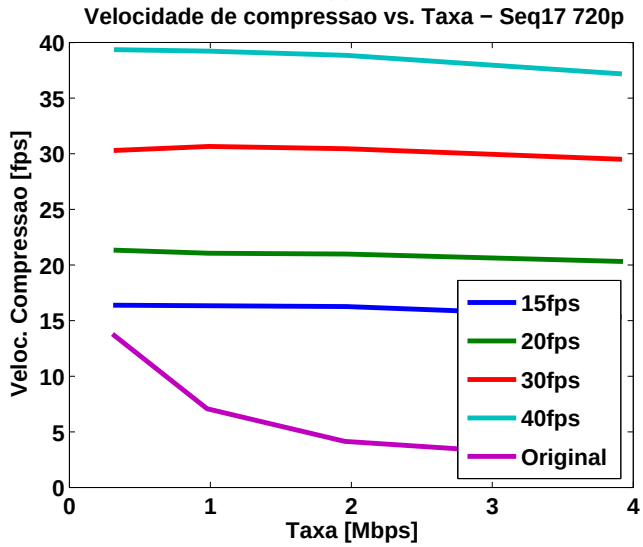


(b)

Figura 5.7: Comparação de desempenho entre codificador com complexidade integral e a versão de codificador com controle de velocidade para a sequência “Foreman”: (a) desempenho RD e (b) perfil de complexidade. “Original” representa o *codec* com complexidade integral, ou seja, $C = 100\%$.



(a)



(b)

Figura 5.8: Comparação de desempenho entre codificador com complexidade integral e a versão de codificador com controle de velocidade para a sequência “Seq17”: (a) desempenho RD e (b) perfil de complexidade. “Original” representa o *codec* com complexidade integral.

das últimas tecnologias dos novos processadores desse fabricante, além de permitir compressão com auxílio de cartões gráficos. As informações publicitárias sobre a biblioteca prometem desempenhos "extraordinários"; contudo, muitos números de velocidade de compressão foram calculados para atividades de transcodificação, uma tarefa menos intensa computacionalmente do que a codificação de vídeo a partir de dados brutos. Testes preliminares mostraram que o desempenho do codificador H.264/AVC disponibilizado por essa API é comparável com o desempenho do x264 para execução em processadores PC. Naturalmente, a disponibilidade de um cartão gráfico de alto desempenho pode elevar significativamente a velocidade de compressão da nova implementação de codificador da Intel®.

A segunda implementação propôs um arranjo para garantir codificação em tempo real de sequências de diferentes resoluções usando uma implementação otimizada do codificador H.264/AVC para a família de processadores x86. O princípio consiste em escolher otimamente um subconjunto de ferramentas de codificação restrito a determinada complexidade alvo. Nossa principal contribuição foi o projeto e a inserção de um módulo de controle de velocidade de codificação ao x264, velocidade esta ajustada por meio de valores absolutos em fps, diferencial em relação aos inúmeros trabalhos que usam valores relativos de complexidade para avaliações.

Os testes apresentados mostraram que o desempenho RD é levemente afetado pela estrutura de controle, que, ademais, não usa descarte de quadros para atingir velocidade de codificação desejada além de contar com controle de taxa de canal. Apesar das penalidades em desempenho, uma parcela significativa de esforço computacional pôde ser poupada.

Além da vantagem de que a contribuição pode ser usada como componente de um codificador de vídeo sobre plataforma PC e sem a necessidade de alterações na implementação do decodificador, o sistema pode beneficiar-se diretamente da disponibilidade de processadores mais poderosos para a arquitetura PC. O emprego de múltiplas *threads* e de processadores de múltiplos núcleos pode multiplicar os números de velocidade de imediato.

6 CODIFICAÇÃO DE VÍDEO OTIMIZADA POR ENERGIA

Este capítulo considera a energia elétrica consumida (E) como uma medida de esforço computacional. Usando um computador pessoal como plataforma de execução, propõe-se um processo de otimização RDE para o codificador H.264/AVC. Os resultados da metodologia proposta demonstram redução significativa na quantidade de energia consumida em tarefas de compressão.

6.1 INTRODUÇÃO

Historicamente, fabricantes de circuitos integrados responderam à demanda por maior poder computacional com o desenvolvimento de processadores mais velozes e com frequências de relógio aumentadas. A tecnologia CMOS, a mais disseminada na confecção de circuitos integrados, caracteriza-se fisicamente por vincular o consumo de energia e a frequência de relógio: o aumento de frequência implica aumento de consumo de energia e de dissipação de calor [82]. O processamento de imagens e vídeo digitais pode ser tomado como um dos grande motivadores dessa busca por capacidade computacional, pois as técnicas modernas de tratamento de sinais demandam um grande volume de computações. Não é surpresa que o padrão tomado como estado da arte na compressão de cenas reais, o H.264/AVC, seja uma aplicação de elevada complexidade computacional e extremamente demandante de recursos energéticos.

Todavia, as consequências ambientais das intervenções do homem na natureza têm provocado questionamentos acerca da necessidade de se aplicar recursos naturais de forma eficiente e, desejavelmente, sustentável. Em função dessa nova abordagem, conservação energética tornou-se uma restrição de projeto importante. Gastos com energia elétrica representam parcela significativa das despesas para o projeto e a operação de *datacenters*. Numa escala menor, o controle da dissipação térmica é fundamental em dispositivos móveis alimentados por baterias. Os governos têm intervido e proporcionado incentivos para a economia de energia, além de fomento para desenvolvimento e utilização de tecnologias sustentáveis [83].

Em função dessa mudança de paradigma, pessoas e empresas buscam produtos com selos de eficiência energética por questões de responsabilidade ambiental e, principalmente, para redução de despesas.

À iniciativa de se desenvolver e operar sistemas computacionais ambientalmente sustentáveis deu-se o nome de computação verde (do inglês, *Green Computing*) [84]. Seu objetivo é diminuir o impacto, no meio ambiente, da produção e do uso de computadores. O escopo deste trabalho se restringirá ao impacto ambiental do uso de computadores, especificamente no quesito de consumo energético.

A nova abordagem de projeto resultante de restrições ambientais fomentou o desenvolvimento de modernos algoritmos computacionalmente eficientes e o surgimento de tecnologias de eficiência energética que provêm subsistemas capazes de modular a frequência e a tensão de alimentação dos processadores para reduzir a potência demandada. Além da escalabilidade em frequência e tensão de alimentação, fabricantes de processadores podem desligar módulos de processadores que não estão sendo utilizados, resultando em mais economia de energia e menor dissipação térmica[1,2]. Isso permite correlação entre complexidade computacional e consumo energético nos modernos processadores.

Recentemente, pesquisadores da Rice University exploraram novos conceitos de otimização energética desde a fabricação de um chip, ao qual se pode agregar princípios básicos da teoria da informação passando a tratar o dispositivo analogamente a um canal de comunicação. Resultados experimentais [85] relataram economia de energia por meio da remoção de blocos funcionais pouco utilizados em chips, concomitante a um aumento significativo de desempenho computacional pela permissão de falhas durante computações. Ou seja, um chip de computação “inexata”.

Este capítulo apresenta uma nova estratégia para a economia de energia na codificação de vídeo em tempo real. Originalmente, complexidade pode ser tomada como uma medida de esforço na execução de uma tarefa computacional e pode ser calculada seja pela quantidade de memória necessária para a execução de um algoritmo, seja pelo número de operações necessárias para realizar um cálculo, seja mesmo pelo tempo necessário para conclusão de determinada atividade computacional [26]. Nós propomos calcular a energia demandada como medida de complexidade [24], haja vista que a energia é um recurso fundamental para as plataformas computacionais e que pode ser diretamente mapeada em custos operacionais.

Apresenta-se uma estratégia de otimização RDE (*Rate vs. Distortion vs. Energy*) para restringir a energia demandada por um codificador de vídeo em tempo real. Os parâmetros de codificação derivados pela otimização RDE são usados para avaliar um esquema de codificação de vídeo em tempo real e

[1] AMD® Cool’n Quiet®: http://www.amd.com/us/products/technologies/cool-n-quiet/Pages/cool-n-quiet.aspx

[2] Intel EIST®: http://www.intel.com/technology/product/demos/eist/demo.htm

otimizado em RDE. Para a implementação em *software* do esquema proposto, alteraram-se aspectos não normativos do estágio de predição do codificador H.264/AVC na implementação x264, apresentado na Seção 5.4. A metodologia apresentada pode ser prontamente aplicada a sistemas de comunicações móveis, os quais possuem eficiência energética como importante restrição [29].

O esquema de codificação H.264/AVC otimizado por RDE permite codificação em tempo real e restrita por energia. É apresentado um módulo de controle, que é anexado ao codificador e é capaz de fornecer a velocidade de compressão necessária ao usuário enquanto se gasta a menor quantidade de energia, com pequenas perdas de desempenho em termos de RD. O arranjo proposto não utiliza descarte de quadros nem mesmo mudança de resolução espacial dos quadros para adequar a velocidade de codificação.

6.2 TRABALHOS CORRELATOS

Conforme discutido em capítulos anteriores, e reapresentado aqui para clareza de explanação, há vários estudos que tratam da manipulação da complexidade do codificador H.264/AVC. Alguns exploram técnicas de predição com volume de computação reduzido às custas de pequenas perdas em desempenho RD [63, 66, 86]. Um trabalho recente relata redução significativa de complexidade [71], embora a escala de complexidade apresentada pelos autores não seja percebida quando a proposta é integrada em uma realização de alto desempenho do codificador H.264/AVC que lance mão de algoritmos rápidos, de bibliotecas otimizadas de primitivas de processamento de sinais e de recursos dependentes da plataforma computacional [72, 73]. Na literatura, encontram-se descritos alguns modelos de complexidade para codificadores [86, 75]. Os resultados temporais, todavia, são apresentados em números relativos, cuja linha de referência de comparação é o codificador de referência JM, reconhecido por seu baixo desempenho em termos de velocidade de processamento de quadros.

O relacionamento da codificação com o consumo de energia é tratado no trabalho [69], em que os autores propõem uma abordagem que possibilita codificação restrita por energia, mas altamente dependente de características específicas do processador analisado, o que restringe sua aplicabilidade a um dispositivo específico, de um fabricante específico. Ademais, usou-se descarte de quadros, além do fato de o codificador em estudo ser o H.263+, tecnologicamente ultrapassado pelo padrão H.264/AVC.

Até o conhecimento do autor, atualmente não há trabalhos na literatura que explorem a codificação otimizada por RDE, muito menos um sistema de controle capaz de receber um valor de provisão de

energia/potência para compressão de vídeo com o codificador H.264/AVC. Os níveis de escalabilidade apresentados neste trabalho foram obtidos por intervenções no módulo de predições do compressor em análise. Técnicas recorrentes na literatura para escalabilidade de complexidade, especificamente o descarte de quadros completos e mudança de resolução espacial dos quadros, não foram utilizadas neste trabalho; todavia, as contribuições aqui propostas podem ser aproveitadas em esquemas que usam desses artifícios de escalabilidade.

6.3 ENERGIA E COMPLEXIDADE

6.3.1 Economizando Energia numa plataforma PC

O princípio básico para o gerenciamento de energia numa plataforma PC é que, quanto menor o volume de computação, menor o consumo energético. Quando no estado *idle*[3], nossa plataforma (um PC com processador AMD Phenom® de 6 núcleos físicos, sem monitor de vídeo conectado) consome 80 W para manter o sistema operacional funcionando. Quando a carga de trabalho aumenta, a potência consumida também aumenta. O estado de processamento total[4], nossa plataforma demanda 180 W de potência ativa para fornecer as correntes elétricas para alimentar a comutação de transistores, o maior acesso às memórias, aos discos rígidos, barramentos e outros componentes.

Considera-se o cenário de codificação de video em tempo real a situação em que quadros são periodicamente disponibilizados para codificação a uma taxa f_a, por exemplo 30 Hz, ou 30 fps. Desta forma, dispõe-se de, no máximo, $T_a = 1/f_a$ segundos para codificar cada quadro, e esse é o período que restringirá o sistema de compressão. Usar apenas T_p segundos para codificar cada quadro permitirá que o escalonador do sistema operacional leve o processador ao estado *idle* no tempo restante $T_i = T_a - T_p$. Um perfil de potência é mostrado na Figura 6.1(a), em que $P_i = 80$ W é a potência demandada quando o sistema está no estado *idle*, enquanto $P_{fp} = 180$ W é a potência demandada pelo sistema quando o mesmo está em estado de processamento total. É interessante também definir a velocidade de processamento (ou de compressão) como $f_p = 1/T_p$, que indica a velocidade (em fps) na qual o codificador seria capaz de codificar os quadros se todos eles estivesse disponíveis de uma vez, por exemplo, no caso de codificação

[3]O estado *idle* é definido como o estado em que apenas as tarefas básicas do sistema operacional estão executando e o escalonador do sistema operacional mantém o processador quase sempre "dormindo".

[4]O estado de processamento total é definido como o estado em que o escalonador mantém o processador executando operações de computação intensiva e quase nunca permite que o processador "durma".

off-line. Da Figura 6.1(a), deve ser frisado que é possível economizar energia se reduzirmos T_p ou, de outro modo, se aumentarmos a velocidade de codificação f_p. Dessa forma, quanto mais rápido o codificador comprime um quadro de vídeo, mais tempo é permitido que o processador permaneça no estado *idle* (maior T_i).

Para esse modelo de utilização binário, em que o processador ou está quase inativo (estado *idle*) ou completamente atarefado (estado de processamento total), é possivel economizar energia aumentando a velocidade de codificação, isto é, reduzindo T_p, como na Figura 6.1(b). Aumentos na velocidade de codificação são tipicamente obtidos às custas de desempenho RD. Enquanto o perfil da Figura 6.1(b) demanda menos energia que o da Figura. 6.1(a), é possível também usar escalonamento dinâmico de tensão de alimentação ou de frequência de relógio do processador, o que diminui a velocidade de processamento do dispositivo e diminui o consumo energético do sistema de maneira semelhante ao observado na Figura 6.1(b), mas numa velocidade menor. Esse caso é ilustrado na Figura 6.1(c), em que o processador executaria a atividade de codificação por mais tempo, usando menos potência.

Neste trabalho, não examinaremos este caso pois nos concentraremos na economia energética proporcionada pela estratégia da Figura 6.1(b) de aumento da velocidade de codificação.

6.3.1.1 Restrições de Implementação

Para as plataformas sob testes, foram necessárias soluções de compromisso a fim de se controlar o processo de codificação. Um dos primeiros itens de preocupação foi a frequência do escalonador de processos do sistema operacional e a sua influência nas medidas de tempo. Basicamente, o sistema operacional Linux utilizado nas simulações foi configurado de forma que seu escalonador apresentasse a frequência de atualização de 250 Hz. Isso insere algumas restrições quanto a medições de intervalos de tempo da ordem de 4 ms, pois os erros instrumentais esperados são da ordem de ± 2 ms. Como os processadores utilizados nas plataformas de testes (AMD Phenom® e Intel® Core® i7) são capazes de prover velocidades da ordem de 250 fps para sequências de vídeo SD e 720p, isso levou à questão de como adequar o comprimento do GOP (*Group of Pictures*) para uma codificação estável em termos de velocidade de compressão. Buscou-se também avaliar reflexo do comprimento do GOP quanto à forma do sinal de potência, que deve se comportar de acordo com o modelo de análise da Figura 6.1.

Nos ensaios, o processador é capaz de codificar a uma velocidade muito superior ao esperado de forma a garantir tempo real; por isso, o dispositivo “dorme” entre as codificações de GOPs para garantir a

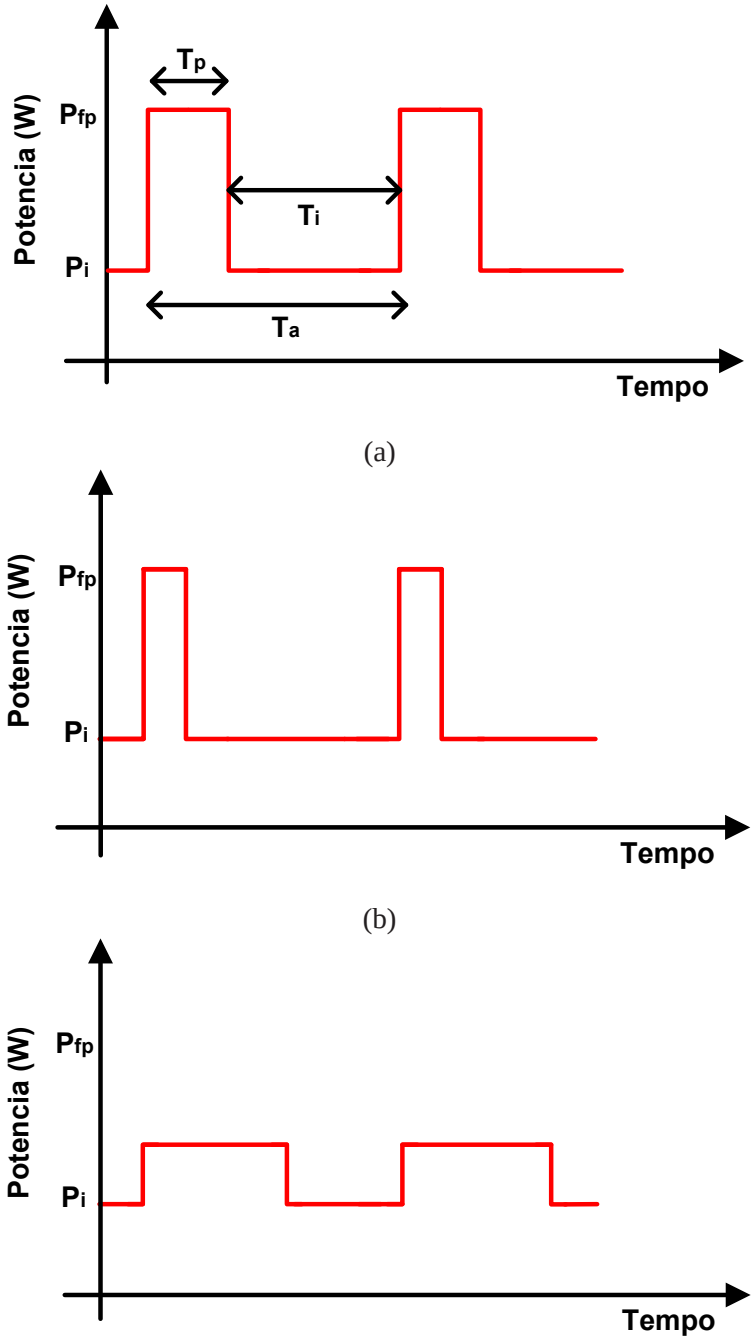


Figura 6.1: Perfil de potência (a) para codificação de vídeo. Quadros são disponibilizados em intervalos de T_a segundos. A potência demandada é maior quando o codificador está ocupando todo o tempo de atividade do processador do PC. Uma vez codificado o quadro em T_p segundos, o processador retorna ao estado *idle*, o que reduz o consumo de energia por $T_i = T_a - T_p$ até que um novo quadro seja disponibilizado. (b) Perfil para redução de consumo energético por aumento de velocidade de codificação. (c) Perfil para redução de consumo energético tornando o processador menos demandante e, também, mais lento.

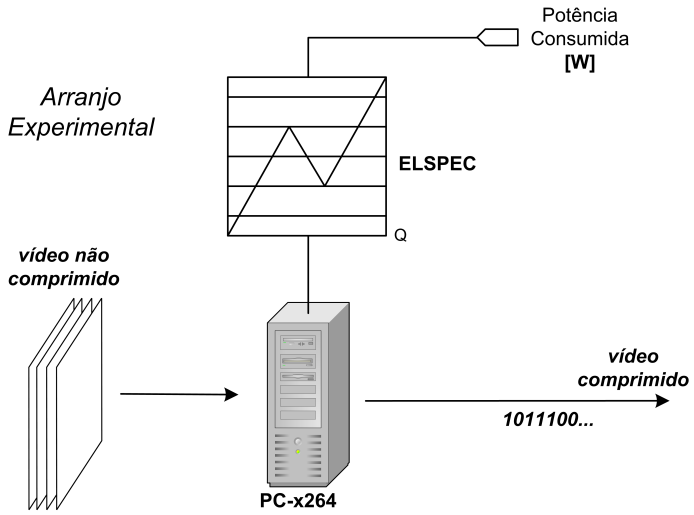


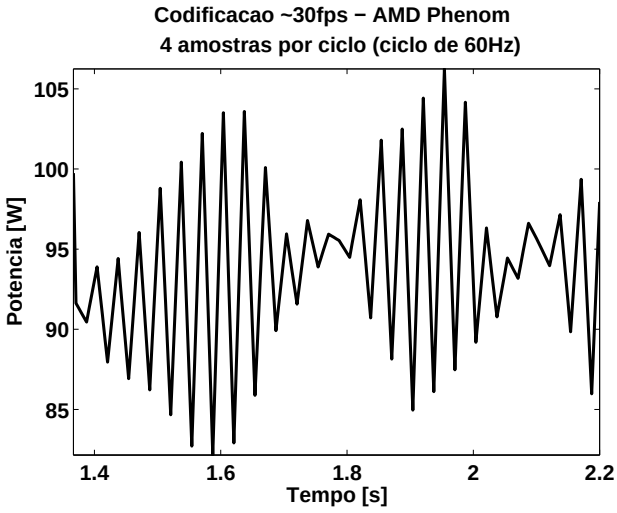
Figura 6.2: Esquema para ensaios de oscilografia de potência para um processador em plataforma PC codificando quadros de vídeo digital. O codec x264 recebe quadros de vídeo brutos e os comprime enquanto o equipamento ELSPEC realiza a monitoração dos valores de potência ativa demandados pela fonte de alimentação do computador pessoal que executa a aplicação. Não se usa monitor de vídeo nesse ensaio. O pequeno pentágono irregular da metade superior do esquema é usado para representação gráfica da sonda de medição de potência.

velocidade de compressão média não inferior a 30 fps. Mediu-se a potência AC consumida pela fonte de alimentação de um computador que executa o codificador x264 na compressão de uma sequência de vídeo 720p (gravada a 30 Hz) de 900 quadros. O instrumento de captura do sinal de potência foi o Elspec G4500 BLACKBOX[5] e o arranjo de medição é ilustrado na Figura 6.2. O ambiente de medidas contava com uma fonte de alimentação senoidal 5001ix da California Instruments[6]. A Figura 6.3 mostra como o comprimento do GOP afeta o sinal de potência.

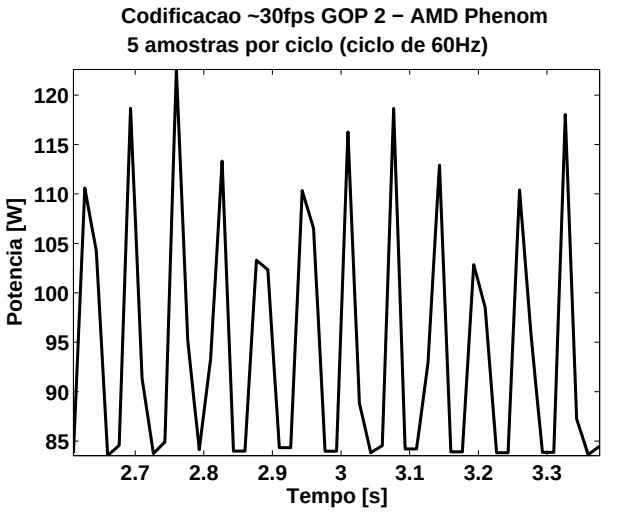
A Figura 6.3(a) apresenta o efeito, sobre a forma de onda de potência, das comutações muito rápidas no estado de atividade do processador. Nesse gráfico, verificam-se os picos de potência que representam as codificações de cada um dos 24 quadros. Todavia, do ponto de vista do sinal de potência medido na fonte de alimentação do computador pessoal, os momentos de "descanso" do processador não são bem discriminados, pois o sinal de potência não permanece por muito tempo no patamar de inativo (80 W de demanda energética para o processador do fabricante AMD®). A forma de onda observada mostra

[5]Especificações técnicas do produto disponíveis no sítio: http://www.elspec-ltd.com/?catid=%7B76FC81EE-B891-4239-AFE2-8454E8A2EFFC%7D. Último acesso dia 15/ 07/ 2012.

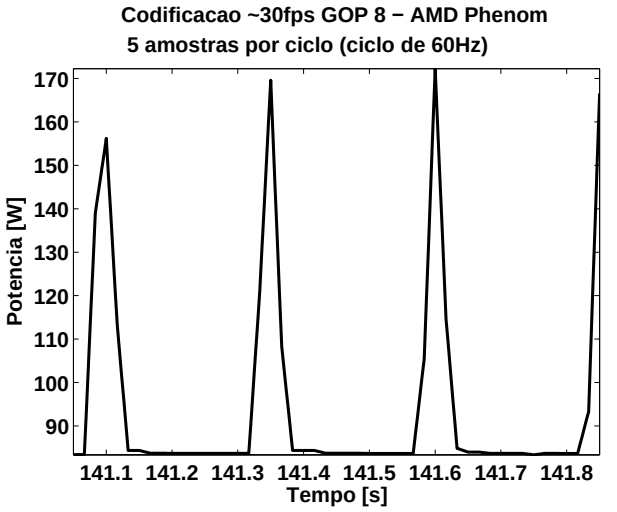
[6]Especificações técnicas do produto disponíveis no sítio: http://www.elgar.com/products/i-iX_Series_II/i-iX_Series_II_Overview.htm. Último acesso dia 15/ 07/ 2012.



(a)



(b)



(c)

Figura 6.3: Oscilografia de potência para um processador PC codificando 24 quadros de vídeo 720p a 30 fps, com GOP (*Group of Pictures*) de diferentes comprimentos: (a) um quadro por pausa; (b) dois quadros por pausa e (c) oito quadros por pausa. O aumento do GOP aproxima, gradativamente, a forma de onda das medições ao que é proposto no modelo.

uma função periódica de alta frequência cuja amplitude é modulada por uma outra função periódica, mas de menor frequência. Esse efeito é explicado por conta da própria arquitetura de dispositivos eletrônicos de alta escala de integração, em que capacitâncias são adicionadas aos módulos funcionais. Essas capacitâncias, quando agregadas às grandes capacitâncias da fonte de alimentação, são capazes de suprir local e rapidamente a demanda por carga elétrica [82], mascarando o efeito de demanda por carga quando se mede o consumo de energia na entrada da fonte de alimentação.

À medida que se aumenta o GOP, o sinal de potência conforma-se às formas de ondas esperadas da Figura 6.1, como pode ser observado quando se aumenta de um quadro por lote (GOP de comprimento 1, Figura 6.3(a)), para dois quadros por lote (GOP de comprimento 8, Figura 6.3(b)) e, em seguida, para 8 quadros por lote (GOP de comprimento 8, Figura 6.3(c)). O agrupamento dos quadros em GOP reduz a frequência de comutação que, por sua vez, permite que a demanda por potência seja melhor percebida pelas medições de consumo a partir da fonte de alimentação.

Para fins de implementação, optou-se por agrupar e codificar GOP de 50 quadros de forma a garantir estabilidade nas medidas de tempo de codificação de quadros e ter uma boa adequação da forma de onda com o modelo da Figura 6.1. A oscilografia do sinal de potência para esse ajuste de GOP de 50 quadros é apresentada na Figura 6.4. A tendência observada para o processador do fabricante AMD® repete-se também para o processador do fabricante Intel®.

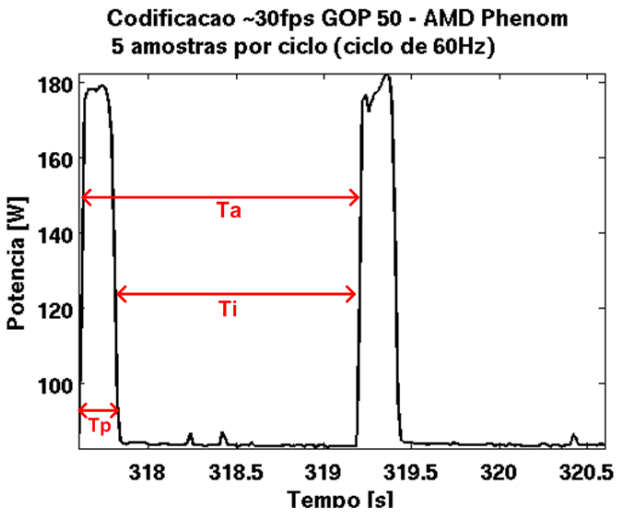


Figura 6.4: Oscilografia de potência para um processador PC codificando 100 quadros de vídeo 720p a 30 fps, com GOP de 50 quadros: dois GOPs são codificados na janela de tempo considerada. Em vermelho, destacam-se os comprimentos dos intervalos relativos a T_a, T_p e T_i do modelo da Figura 6.1.

6.3.2 Energia como medida de esforço computacional

É natural inferir que, quanto maior o número de operações para codificar um quadro, mais devagar um codificador irá executá-las. Contudo, medir complexidade por meio de estimativas de número de operações é uma abordagem muito imprecisa para aplicações complexas, pois elas costumam depender de recursos específicos da plataforma de execução. Da discussão anterior, pode-se inferir que o consumo energético é completamente definido por f_p.

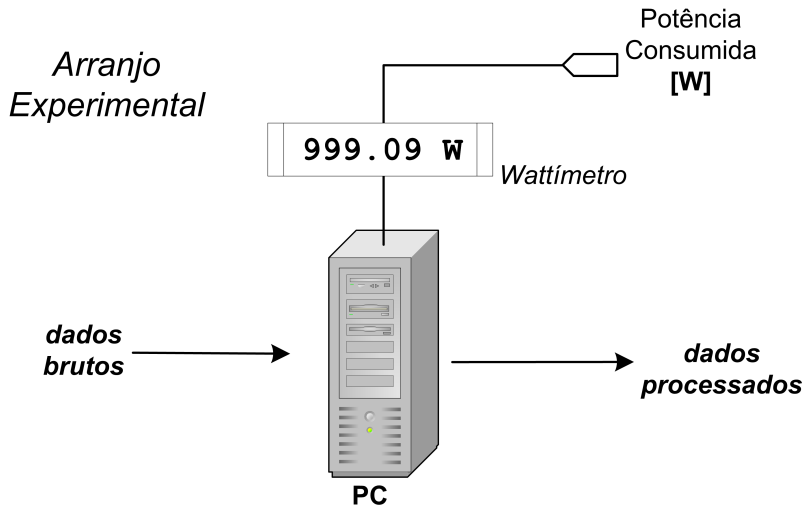


Figura 6.5: Esquema para ensaios de medições de potência para um processador em plataforma PC ao executar uma atividade computacional. Um computador pessoal recebe dados brutos e os processa enquanto um wattímetro realiza a medição dos valores de potência ativa demandados pela fonte de alimentação do equipamento. Não se usa monitor de vídeo nesse ensaio. O pequeno pentágono irregular da metade superior do esquema é usado para representação gráfica da sonda de medição de potência.

Em um experimento simples para avaliar as relações entre complexidade e energia, sorteia-se repetitivamente um número inteiro de 32 bits e verifica-se se o mesmo é um número primo usando a rotina `is_prime`, executada por um computador pessoal; simultaneamente, anota-se a demanda energética para todo o processo por meio de um wattímetro conectado à fonte de alimentação do computador pessoal (Figura 6.5). A rotina `is_prime` testa se um número inteiro é primo de forma simplista: aplicam-se sucessivas divisões do número sorteado por números inteiros partindo do número dois até que se encontre um divisor inteiro. Se nenhum divisor inteiro for encontrado, afirma-se que o número sorteado é primo. Com essa atividade intensa computacionalmente, é possível avaliar como o tempo de execução e como a ativação de múltiplos núcleos de processamento afetam o consumo energético numa arquitetura PC, por exemplo. Durante todo o experimento, o processador permanece ocupado executando a rotina.

Os resultados são apresentados na Figura 6.6, em que o número de *threads* simultâneas também foi variado para mostrar que, quanto maior o número de threads, melhor a eficiência energética (mais testes para a mesma quantidade de provisão de energia).

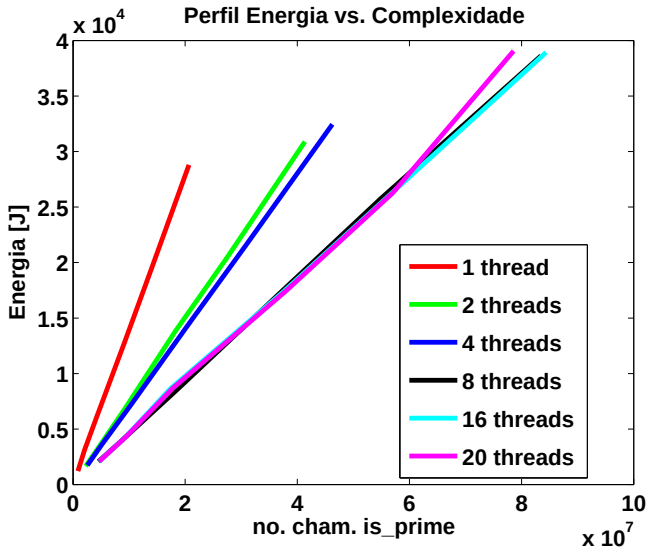
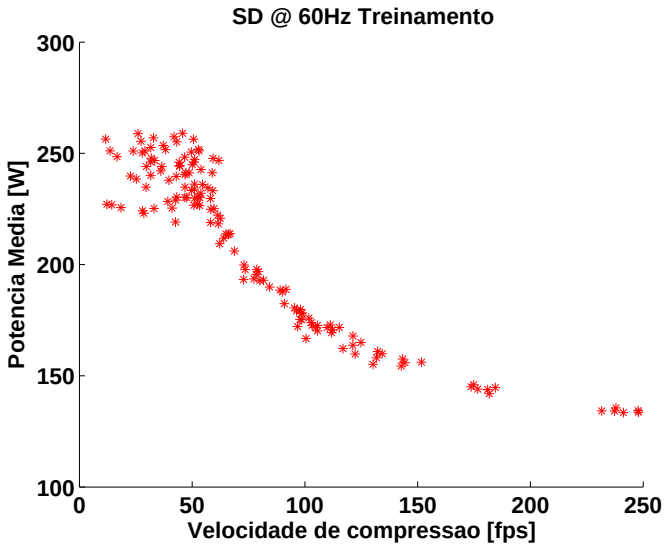
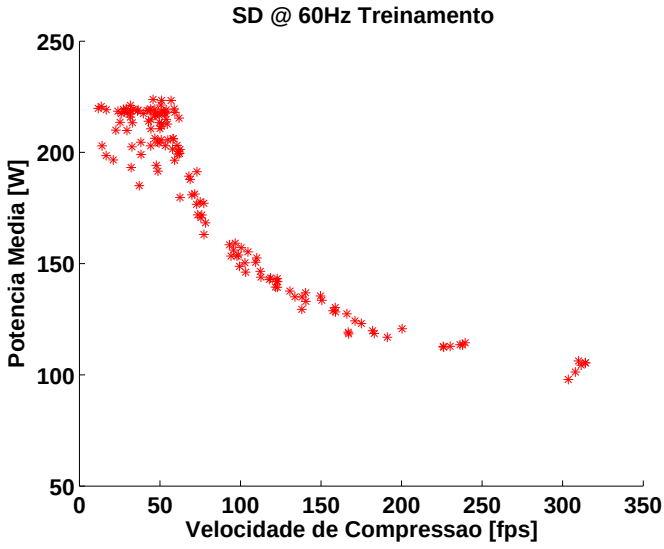


Figura 6.6: Perfis de energia para diferentes cargas de processamento. Quando o número de chamadas simultâneas à rotina is_prime é aumentado, a energia demandada cresce. Note que a instanciação de mais que 8 *threads* simultâneas não contribui de forma significativa com o número de chamadas à is_prime. O processador da plataforma é um Intel® Core® i7, com quatro núcleos físicos e oito núcleos lógicos.

Enquanto a correlação na Figura 6.6 é aproximadamente linear, para tarefas mais complexas a relação não é bem definida, como mostrado na Figura 6.7 para processadores PC dos fabricantes Intel® e AMD®. A Figura 6.7 mostra resultados típicos relacionando f_p e a potência média P_m (P_m consiste na razão E/T_a) para uma tarefa de codificação de vídeo, em que são computadas a velocidade de compressão e a potência demandada. Observa-se que a curva não é linear e há muitos pontos dispersos. Isso é devido ao fato de que o perfil de potência não é tão bem comportado como o da Figura 6.1, que não leva em consideração imperfeições e oscilações causadas pelos vários dispositivos eletrônicos envolvidos. Em função desse fato, decidiu-se medir a demanda de energia/potência ativa em vez de simplesmente estimá-la com algum modelo.



(a)



(b)

Figura 6.7: Correlação da potência demandada e a velocidade de compressão (f_p) para processador (a) Intel® (b) AMD®.

6.4 OTIMIZAÇÃO BASEADA EM ENERGIA

É comum que atividades de otimização tratem funções de custo ou taxas de sucesso. Tomemos um codificador em *software* que execute uma atividade para a qual nós desejamos, de alguma maneira, medir seu custo. Em compressão de sinais, costuma-se medir custos com medidas de qualidade, como a distorção (D), com medidas de taxa de bits (R) ou com uma combinação de ambas. Assume-se o processo de compressão como uma atividade parametrizada, i.e., é possível escolher os valores dos N parâmetros $\{P_i\}_{i=1,\ldots,N}$. Tomemos $\mathbf{P}$ como o vetor com todos os P_i. O codificador é executado sobre determinado conjunto Z de dados, que pode ser diferente a cada execução. Para cada escolha de $\mathbf{P}$ e Z, nós podemos ter uma medida C do custo do codificador. Basicamente, nós podemos ter o seguinte mapeamento

$$C = f(\mathbf{P}, Z).$$

Outro atributo que pode ser derivado a partir de cada execução é o esforço envolvido na execução da própria codificação, que pode ser medido como a energia demandada $E = g(\mathbf{P}, Z)$. Espera-se que alguns parâmetros como número de iterações, tamanho do conjunto de dados etc, influenciem a energia demandada enquanto outros não. A ideia central neste trabalho vem do fato de a correlação entre E e C ser afetada diferentemente ao se ajustar diferentes parâmetros. Usaremos essa propriedade para encontrar pontos que minimizem o consumo energético. A ideia é mostrada na Fig. 6.8, que apresenta uma nuvem de pontos, no espaço custo-energia, de todos os possíveis $\mathbf{P}$ para um determinado sistema e determinados dados de entrada. A figura enfatiza um conjunto de pontos encontrados após a otimização, o casco convexo inferior (LCH, do inglês *Lower Convex Hull*) de todos os pontos, representados por quadrados. Pontos que pertencem ao LCH representam codificações em que o sinal é comprimido usando a menor energia para determinado custo, e esta é a região em que se deseja operar. Outro subconjunto discriminado é composto por pontos que representam a varredura nos valores de determinado parâmetro P_i, mantendo-se todos os outros fixos, e é representado por estrelas vermelhas. A varredura no intervalo de definição de um parâmetro pode resultar um conjunto subótimo, diferente do LCH. Uma dentre muitas formas de definir o LCH, uma solução simples, que leva a um conjunto marginalmente não convexo, consiste em incluir um ponto no LCH somente se nenhum outro ponto possuir, simultaneamente, valores de C e E inferiores ao seu.

Apesar da explicação simples por meio do uso de um custo como variável escalar, em codificação de vídeo, o mapeamento é convenientemente tratado por meio de uma variável multidimensional $\mathbf{C} = [R, D]$,

onde R representa a taxa de bits e D, a distorção (por exemplo, o erro quadrático médio). Desta forma, $\mathbf{C} = f(\mathbf{P}, Z)$.

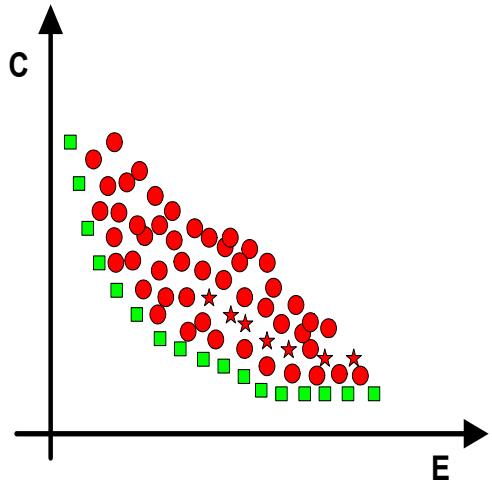


Figura 6.8: Nuvem de pontos no espaço energia vs. custo. Os pontos do LCH são indicados por quadrados verdes. Um conjunto subótimo determinado, por exemplo, pela variação de apenas um dos parâmetros é ilustrado pelas estrelas vermelhas.

6.5 OTIMIZAÇÃO DE FUNÇÕES DE CUSTO RDE

$\mathbf{P}$ e Z são mapeados para R, D and E, o que resulta na adição da dimensão da energia (E) ao problema de otimização por taxa e distorção[7]. Desejamos encontrar os parâmetros que nos permitam operar no LCH do espaço RDE. Desta forma, deve-se garantir que nenhuma outra configuração resulte em menor consumo energético para dado valor de custo. Por outro lado, garante-se que, para um determinado patamar de consumo energético, nenhuma outra configuração do codificador resultará em melhor desempenho. A Fig. 6.9 ilustra o LCH no espaço RDE.

Uma abordagem consiste em usar conjuntos de dados para treinamento. Tomemos $\{\mathbf{P}_k\}$ como o conjunto de todas as escolhas dos parâmetros, ordenados de modo conveniente. Considere, também, que $\mathbf{P}_k$ tenha elementos P_{kn}. Se usarmos um conjunto de dados representativo $\hat{Z}$, é possível varrer $\{\mathbf{P}_k\}$, calcular E, R e D e identificar os pontos que pertencem ao LCH do espaço $E \times R \times D$. Para o n-ésimo ponto pertencente ao LCH, registramos $\mathbf{Q}_n = [E_n, R_n, D_n, \mathbf{P}_n]$, que contém os pontos ótimos para o

[7]Medimos a potência ativa (em watts, W) a partir das leituras fornecidas por um wattímetro, das quais nós podemos derivar o consumo energético (em joules, J).

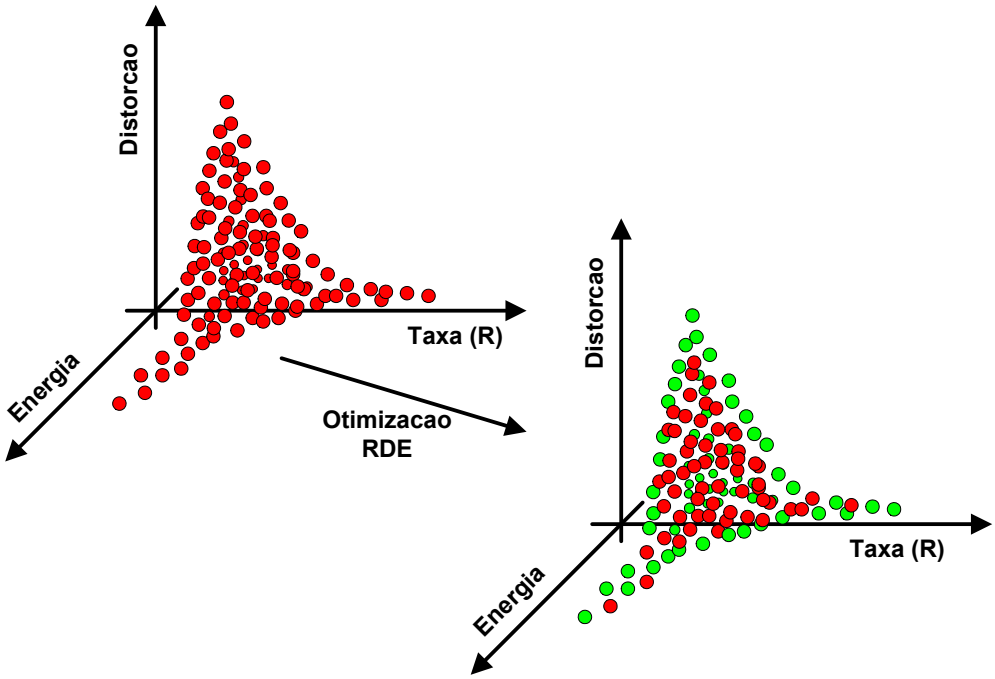


Figura 6.9: Ilustração do conjunto de pontos RDE que constituem a frente de Pareto. Os pontos verdes fazem parte do casco convexo inferior (LCH). Note que alguns pontos estão oclusos devido ao ponto de vista.

conjunto representativo $\hat{Z}$, que podem ser considerados bons o suficiente para outros dados. O algoritmo para treinamento *off-line* é:

1. Apresente um conjunto de dados representativos $\hat{Z}$ e crie uma lista vazia Q.
2. Para todos os k, calcule $E_k = g(\mathbf{P}_k, \hat{Z})$ e $[R_k, D_k] = f(\mathbf{P}_k, \hat{Z})$. Se o ponto pertencer ao LCH, insira $\mathbf{Q}_k = [E_k, R_k, D_k, \mathbf{P}_k]$ na lista Q.
3. Devolva a lista Q de pontos no LCH.

Depois de encontrar N_q pontos que pertençam ao LCH, orndena-se Q em ordem crescente de energia, i.e., E_l em Q é não decrescente. Ao se executar *on-line*, o algoritmo de busca de parâmetros é o seguinte: Inicie considerando a taxa de bits R^r (restrição do canal de transmissão) e o valor desejado de provisão de energia E^r. Então:

1. Apresente uma lista Q de pontos no LCH, o valor desejado de provisão de energia E^r e a taxa de bits alvo R^r. Crie uma lista vazia L.

2. Varra Q, para $k = 1, \ldots, N_q$. Se $|R_k - R^r| < \epsilon$ e insira $\mathbf{Q}_k$ na lista L.

3. Conte N_l, o número de itens em L. Observe que os itens da lista L estão ainda em ordem crescente de energia e assuma que todos os parâmetros resultem em taxa de bits muito próximas.

4. Varra a lista L, para $k = 1, \ldots, N_l$, até que $E_k \leq E^r \leq E_{k+1}$, então pare.

5. Encontre $\mathbf{P}'$ por interpolação de $\mathbf{P}_k$ e $\mathbf{P}_{k+1}$ da lista L.

6. Devolva o vetor de parâmetros $\mathbf{P}'$.

O conjunto de parâmetros $\mathbf{P}'$ passa a ser utilizado para comprimir o conjunto de dados Z. Utilizamos alvos energéticos E^r restritos a uma taxa de bits R^r, mas é trivial substituir a taxa alvo por um alvo de distorção D^r. Naturalmente, muitos parâmetros não assumem valores contínuos e alguns artifícios tem de ser usado para ajustá-los corretamente. Por exemplo, tome o caso para o m-ésimo parâmetro, em que é possível usar o valor P_{km} se $E^r - E_k < E_{k+1} - E^r$, caso contrário, usar o valor de $P_{k+1,m}$.

Se uma arquitetura de malha fechada for permitida, é possível monitorar o consumo de energia do sistema e ajustar, continuamente, os parâmetros. Discrepâncias entre Z e $\hat{Z}$ podem provocar um descasamento entre o consumo energético observado e o predito pelo modelo. Nesse caso, observa-se que $\hat{Z}$ não é tão representativo como foi assumido inicialmente. Esse descasamento pode ser devido a não-linearidades do mapeamento g. Uma solução é iniciar com uma provisão alvo E^r e medir, periodicamente, a energia instantânea $E(n)$. Em seguida, adaptam-se os parâmetros de forma a controlar o gasto energético (ou custo). Assumamos que, em determinado instante n, $\mathbf{P}'$ é tomado como uma interpolação de $\mathbf{P}_j$ and $\mathbf{P}_{j+1}$. Se $E(n) < E^r$, é necessário mover $\mathbf{P}'$ na direção de $\mathbf{P}_{j+1}$ ou mesmo $\mathbf{P}_{j+2}$. Por outro lado, se $E(n) > E^r$, deve-se mover na direção oposta, i.e., para $\mathbf{P}_j$ ou até mesmo para $\mathbf{P}_{j-1}$.

O laço de controle se beneficia de todas as propriedades de simples sistemas adaptativos e há muitas técnicas para escolher os mecanismos de adaptação e tratar de problemas de convergência. Nas próximas seções nós mostraremos exemplos da aplicabilidade do método proposto.

6.6 OTIMIZAÇÃO RDE DO CODIFICADOR H.264/AVC

Nossa estratégia para a construção de um codificador de vídeo em tempo real e restrito por energia consiste no controle da demanda de energia durante a codificação de uma sequência de vídeo por meio

do ajuste de parâmetros do codificador de forma que as perdas de desempenho em termos de RD, em comparação com o máximo desempenho alcançável, sejam as menores possíveis. A energia (E) é medida pelo tratamento das leituras de potência realizadas por um wattímetro conectado ao computador responsável pela execução do compressor de vídeo em *software*. Optou-se pela implementaçao x264 do padrão H.264/AVC devido a sua velocidade de codificação e bom desempenho em termos de RD [76]. Aplicando a metodologia discutida na Seção 6.5, nossa abordagem consiste numa otimização RDE.

O vetor $\mathbf{P}$ é composto pelos seguintes parâmetros relacionados a esforço computacional: a quantidade de quadros B ($\#B$), a quantidade de quadros de referência ($\#$Refs), a precisão dos vetores de movimentos (MVP) usados na compensação de movimentos, a seleção da heurística de decisão de modos de predição (MD), o parâmetro de quantização (QP) e o número de *threads* usadas na compressão ($\#$Thrds):

$$\mathbf{P} = \{\#\text{B}, \#\text{Refs}, \text{MVP}, \text{MD}, \text{QP}, \#\text{Thrds}\}\,.$$

O custo $\mathbf{C}$ é o vetor $[R, D]$, em que a taxa é medida em bits por segundo enquanto a distorção é calculada como o erro quadrático médio entre o sinal original e sua versão decodificada.

A primeira etapa da otimização do codificador H.264/AVC quanto à energia E de codificação consiste em determinar o conjunto representativo $\hat{Z}$ a partir do qual será derivada a frente de Pareto [81] de configurações. A análise da arquitetura proposta foi realizada para sequência de resolução SD (quadros de 704×576 *pixels*) gravadas a 60 Hz e de resolução 720p (quadros de 1280×720 *pixels*) gravadas a 30 Hz e a 50 Hz. As sequências de treinamento SD foram montadas pela concatenação das sequências “Harbour”, “Crew” e “Soccer”. O conjunto de treinamento HD de 50 Hz é composto pelas sequências “Parkrun,” “Stockholm” and “Tractor”[8]. O conjunto de treinamento HD de 30 Hz é composto pelas sequências de vídeo-conferência “Seq05”, “Seq06” e “Seq17”[9].

As plataformas de análise são dois PCs (sem monitor de vídeo), um com processador Intel® Core® i7, outro com processador AMD Phenom®. Em ambos os equipamentos, o *codec* executa na mais alta prioridade de escalonamento. Usam-se duas plataformas para testar a robustez da técnica proposta quando exercitada em diferentes ambientes de computação.

Para cada resolução estudada, codificou-se o conjunto de treinamento e, para cada execução do *codec*,

[8] Esta sequência era 1080p originalmente e foi decimada para 720p.

[9] “Seq05” é uma cena em que há uma interlocutora sentada numa mesa (fundo de cena suave), por sua vez, na “Seq06”, há um interlocutor, o fundo de cena é mais detalhado (alta frequência). A “Seq17”, além de um fundo detalhado, verifica-se a chegada de um segundo interlocutor na cena, o que muda os padrões de movimento.

registrou-se a taxa de bits (R), a distorção (D) e a energia elétrica consumida[10]. O ambiente de testes, ilustrado na Figura 6.10, também coleta a velocidade de compressão, uma variável importante para a estratégia de restrição de energia do codificador otimizado, que deve operar em tempo real. Os valores de $\mathbf{P}_k$ que não são capazes de garantir $f_p \geq f_a$ são descartados. De outra forma, os parâmetros ótimos são restritos àqueles que permitam codificação em tempo real.

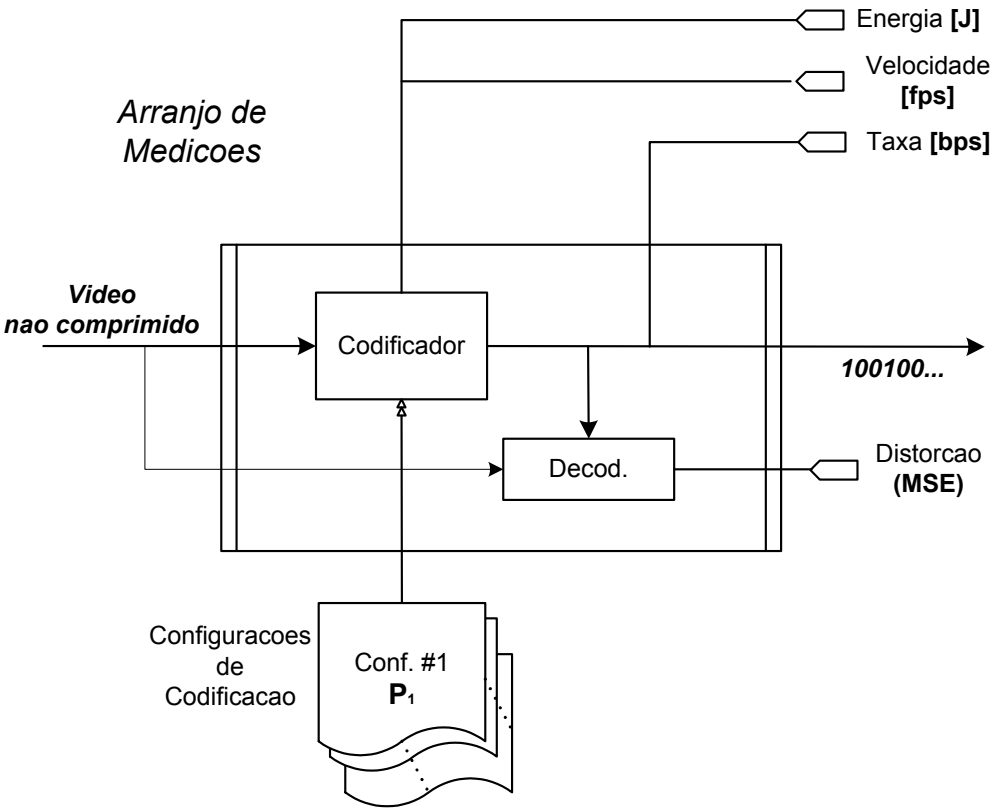


Figura 6.10: Arranjo de medições: para cada execução do codificador ($\mathbf{P}_k$), os custos $\mathbf{C} = [R, D]$ envolvidos (em que R é a taxa e D representa a distorção), a energia (E) e a velocidade de compressão são registrados. A partir desses pontos, o LCH de pontos no espaço RDE é determinado. Os pequenos pentágonos irregulares da metade superior do esquema são usados para representação gráfica de sondas de medição das grandezas descritas em seu lado direito.

A quantidade máxima de quadros de referência (#Refs) foi limitada a 5. A Figura 6.6 mostra que não vale a pena aumentar o número de *threads* de processamento simultâneo além do número de núcleos (lógicos) de processamento. Além disso, quando o número de *threads* cresce, o escalonador do sistema

[10] A energia E é medida pelas leituras de potência ativa fornecidas por um wattímetro acoplado ao PC que executa as atividades de compressão (Figura 6.5). As leituras obtidas correspondem à demanda de potência pela fonte de alimentação do equipamento. Gravam-se as leituras em um segundo computador que as transmite ao PC que executa a atividade de codificação.

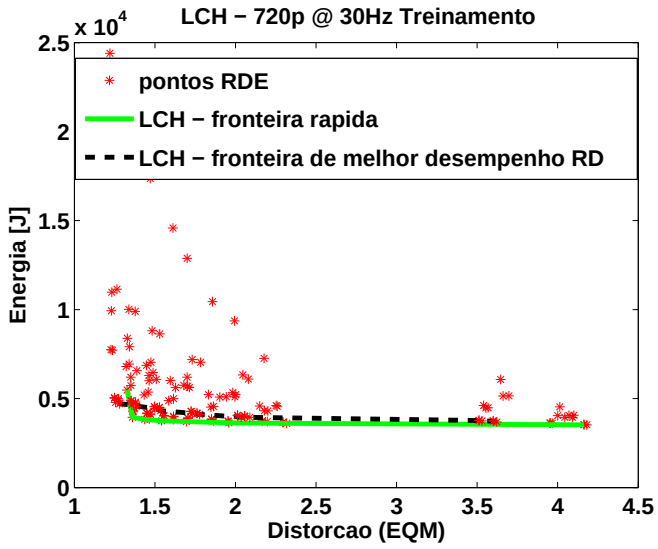
operacional pode desperdiçar tempo na alocação das fatias de tempo (do inglês, *time slots*) de cada tarefa, competindo pelos escarços recursos de CPU durante essa alocação. Optou-se por arbitrar o número de *threads* de codificação (#Thrds) como 8 de forma a garantir estabilidade nos valores de velocidade de compressão. O valor máximo de quadros B ($\#B$) foi determinado por características de implementação do próprio x264, que vincula o número máximo de quadros B entre quadros P ao número dos quadros de referência. Os outros componentes do vetor $\mathbf{P}$ (QP, MD[11] e MVP[12]) foram varridos em seus intervalos de definição. Em síntese, na etapa de treinamento, enfatizou-se a busca das configurações mais rápidas que proporcionassem baixo consumo de energia, com a garantia de que $f_p > f_a$. Durante a busca, variaram-se a precisão dos vetores de movimentos, a técnica de decisão pelo melhor modo de predição, o QP, o número de quadros de referência e o número de quadros B.

Os resultados das simulações fornecem uma nuvem de pontos RDE dos quais se deriva o LCH. A Figura 6.11 mostra uma projeção da nuvem de pontos RDE no plano ED, na qual destacam-se duas curvas: uma que representa a fronteira formada pelos pontos RDE das configurações mais rápidas e menos energéticas da superfície LCH; e uma que contém os pontos das configurações do *codec* de melhor desempenho em termos de RD, os quais, por sua vez, definem a fronteira mais energética da superfície LCH. Nas Figuras 6.11(a) e 6.12(a), mostram-se resultados para sequências 720p e, nas Figuras 6.11(b) e 6.12(b), para sequências SD. O controle adaptativo de energia demandada restringe a operação do codificador no LCH, uma superfície tridimensional que é delimitada pelas curvas de desempenho destacadas (Figuras 6.11 e 6.12).

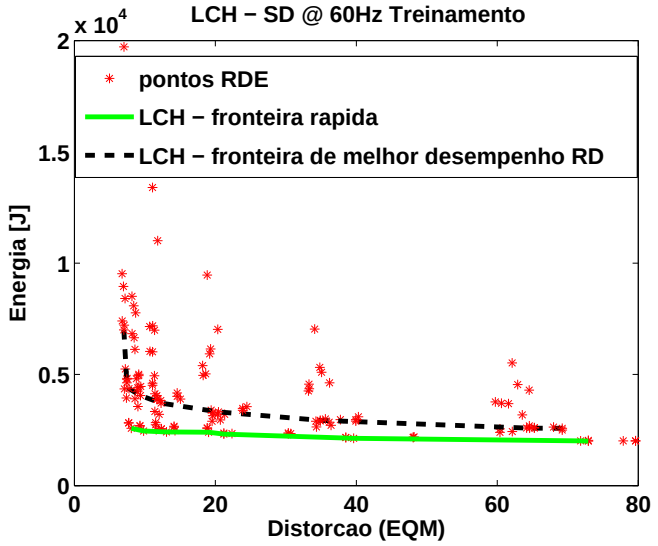
O desempenho do codificador otimizado pode ser avaliado nas Figuras 6.13 e 6.14, nas quais comprimiu-se as sequências de treinamento usando-se escalas intermediárias (e de intervalos regulares) de potência, entretanto ainda sem controle de potência. A Figura 6.13(a) apresenta as curvas RD ótimas para a sequência de treinamento 720p em diferentes níveis de potência e, portanto, de energia. Observa-se que, quanto maior a provisão de potência, melhor a sequência é comprimida em termos de RD. Na Figura 6.13(b), que mostra o perfil de potência de cada uma das curvas RD da Figura 6.13(a), verifica-se que a demanda de potência varia com a taxa de bits: cada curva em (a) é representada por um perfil do potência em (b) e o valor indicado nas legendas consiste na média das potências no intervalo das taxas. Observe que a demanda de potência não se mantém constante quando se variam as taxas. Tipicamente, altas taxas demandam mais potência (e energia) pois o volume de informações tratadas pelo quantizador

[11] Parâmetro *trellis* do x264. Veja a Seção 5.5.3.1.

[12] Parâmetro *subme* do x264. Veja a Seção 5.5.3.1.

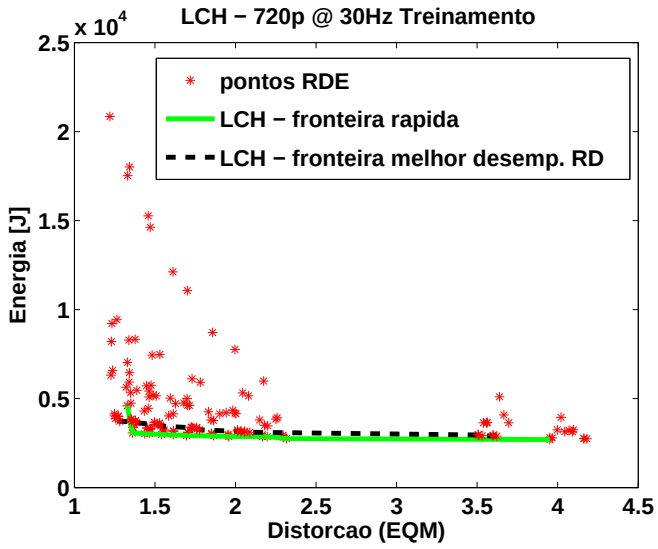


(a)

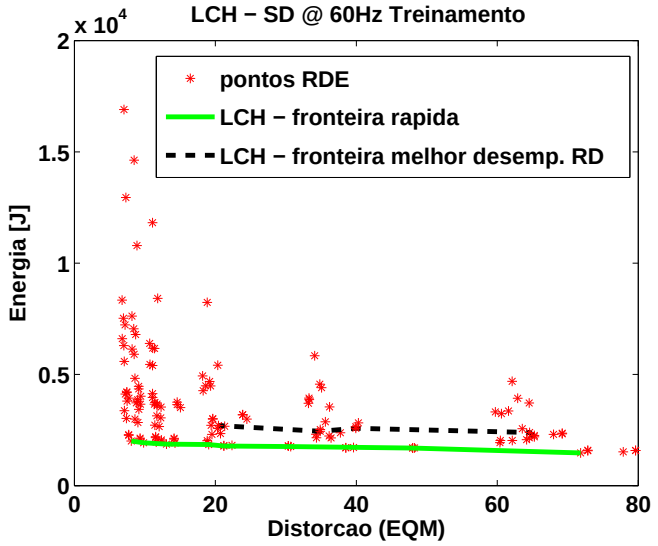


(b)

Figura 6.11: Nuvem de pontos RDE projetada no plano ED para sequências de treinamento (a) 720p e (b) SD. As curvas destacadas em verde e preto demarcam as fronteiras da superfície LCH para as configurações mais rápidas e de melhor desempenho RD, respectivamente. O processador é do fabricante Intel®.



(a)



(b)

Figura 6.12: Nuvem de pontos RDE projetada no plano ED para sequências de treinamento (a) 720p e (b) SD. As curvas destacadas em verde e preto demarcam as fronteiras da superfície LCH para as configurações mais rápidas e de melhor desempenho RD, respectivamente. O processador é do fabricante AMD®.

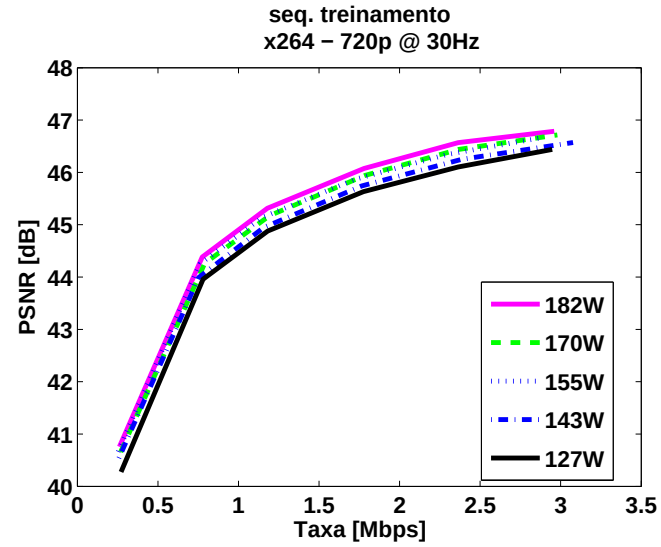
e pelo codificador de entropia aumenta, o que, por sua vez, demanda uso maior dos dispositivos de processamento na plataforma PC. Há, entretanto, uma região de variações abruptas devido ao próprio processo de otimização, que descartou pontos RDC mais potentes/energéticos no final do intervalo das taxas devido ao fato de esses não proporcionarem a velocidade mínima de codificação.

A mesma discussão é válida para a Figura 6.14(a). Para a Figura 6.14(b), há um súbito decréscimo na demanda de potência em baixas taxas e, em seguida, um aumento monotônico da demanda. A anomalia observada no início do intervalo das taxas na Figura 6.14(b) é devida a um esforço aumentado efetuado pelo controle de taxa de canal quando os parâmetros de quantização assumem grandes valores (baixas taxas de canal), uma característica particular do módulo de controle de taxas codificador x264, herdado do projeto ffmpeg [76]. Cabe ressaltar que os números de desempenho para o codificador otimizado para o processador do fabricante AMD® são similares.

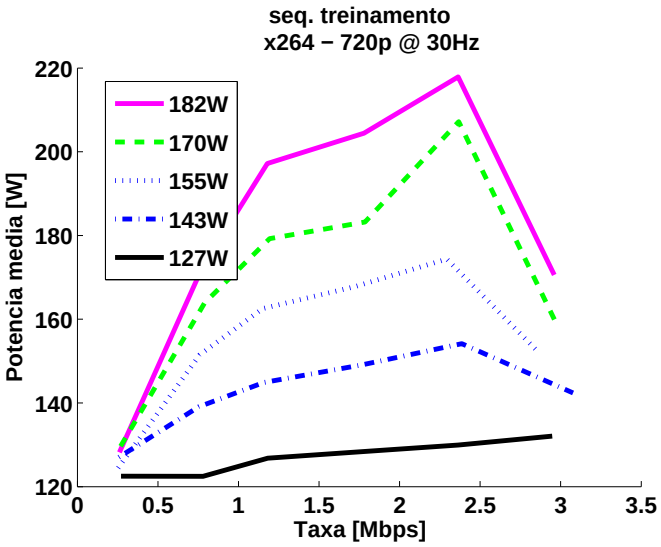
Uma vez determinado o LCH para as sequências representativas, derivam-se tabelas de consultas (*Look-up tables*) a partir das quais é possível controlar as demandas de energia do codificador. Essas tabelas são inseridas no esquema de controle, cujo diagrama é ilustrado na Figura 6.15. O alvo de energia indicado pelo usuário é alcançado pela combinação de configurações de codificador mais próximas do demandado. O controlador tenta propiciar a velocidade de codificação em tempo real e fornecer o melhor compromisso em termos de RDE.

Basicamente, o controlador em malha fechada da Figura 6.15 mede a energia gasta na codificação e ajusta o codificador de forma que a codificação seja feita em tempo real enquanto se gasta uma quantidade não superior à provisão energética indicada pelo usuário. O controlador manipula o perfil de potência do computador conforme o discutido na Seção 6.3.1. Assume-se que as flutuações de demanda no estado de processamento total não desviem de um patamar de potência ativa média durante a codificação de sequências de treinamento. Eventuais descasamentos são corrigidos por ações de controle da arquitetura de malha fechada. A ideia central consiste em escalar a razão T_p/T_a de forma a ajustar a energia demandada ao alvo indicado pelo usuário. O arranjo em malha fechada ajusta o *codec* para diferentes níveis de P_{fp} e P_i e garante a energia alvo solicitada. Se o codificador estiver gastando mais energia do que deveria, o módulo de controle ajusta os parâmetros de codificação para uma configuração menos energética/potente que, por sua vez, resultam na diminuição de desempenho RD. Se houver sobras energéticas, o codificador é ajustado para usar parâmetros mais energéticos/potentes, melhorando, desta forma, seu desempenho RD.

Quanto ao impacto do treinamento no esquema de codificação, salienta-se que a otimização do sistema

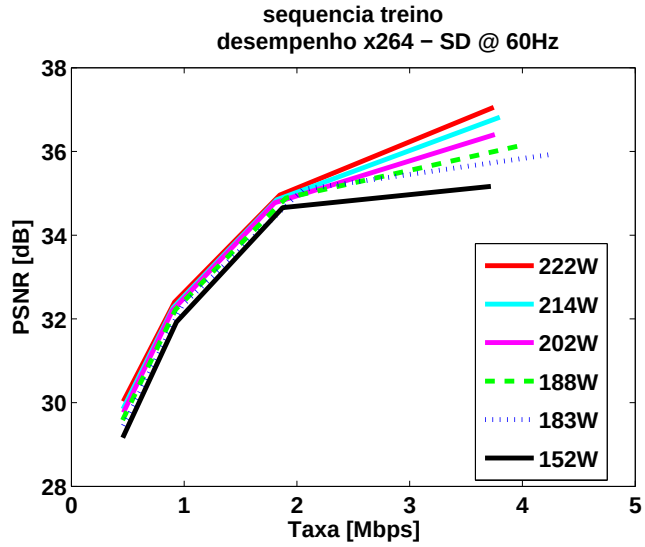


(a)

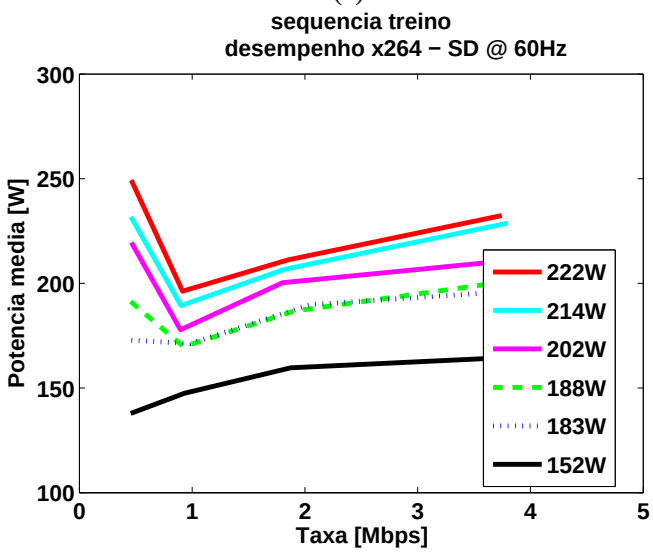


(b)

Figura 6.13: Resultados de codificação para as sequências de vídeo de treinamento 720p. O desempenho RD (a) varia na medida em que se escalam as demandas de potência (e de energia). Demandas de potência (b) para diferentes taxas de bits e para cada curva RD mostrada em (a): os valores nas legendas são a média de potências no intervalo das taxas. Para gerar as curvas em (a), escolheu-se uma escala de potência, contudo o controle de potência está desligado. O processador é do fabricante Intel®.



(a)



(b)

Figura 6.14: Resultados de codificação para as sequências de vídeo de treinamento SD.O desempenho RD (a) varia na medida em que se escalam as demandas de potência (e de energia). Demandas de potência (b) para diferentes taxas de bits e para cada curva RD mostrada em (a): os valores nas legendas são a média de potências no intervalo das taxas. Para gerar as curvas em (a), escolheu-se uma escala de potência, contudo o controle de potência está desligado. O processador é do fabricante Intel®.

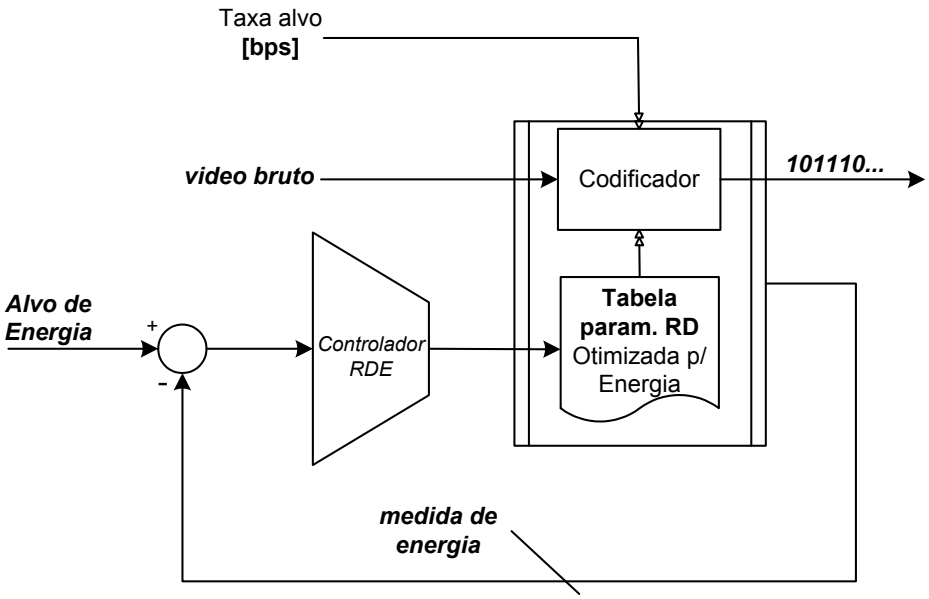


Figura 6.15: Esquema de controle de energia. A arquitetura em malha fechada garante ajustes que seguem a referência de energia indicada pelo usuário. Variações do valor requisitados são minimizados pelo esquema, que ajusta as configurações do *codec* para adequar a demanda energética.

em termos de RDE (a coleta da nuvem de pontos e a posterior otimização do codificador) é uma atividade que dura em torno de 24 h. Nesse intervalo, escolhem-se os parâmetros que serão usados na montagem da nuvem de pontos no espaço RDE. Em seguida, executa-se o codificador de vídeo para cada combinação possível do vetor de parâmetros $\mathbf{P}$ para as sequências de vídeo 720p e SD. Com os pontos da nuvem RDE, demarca-se o LCH (para cada resolução de video) e insere-se a tabela de parâmetros otimizados (derivada do LCH) no esquema de controle de energia/potência. Levando em consideração o treinamento precisa ser feito apenas uma vez com o devido conjunto de sequências de treinamento representativo, o custo de treinamento pode ser rapidamente diluído com a operação do sistema de compressão de vídeo restrito por demanda energética.

6.7 RESULTADOS

Os gráficos da Figura 6.16(a) ilustram o desempenho RD do controlador quando o mesmo tenta seguir o alvo de energia para várias taxas de bits. As curvas RD para codificação de sequência SD em diferentes níveis de energia são motradas na Figura 6.16(b) e na Figura 6.17. Gráficos similares são mostrados na

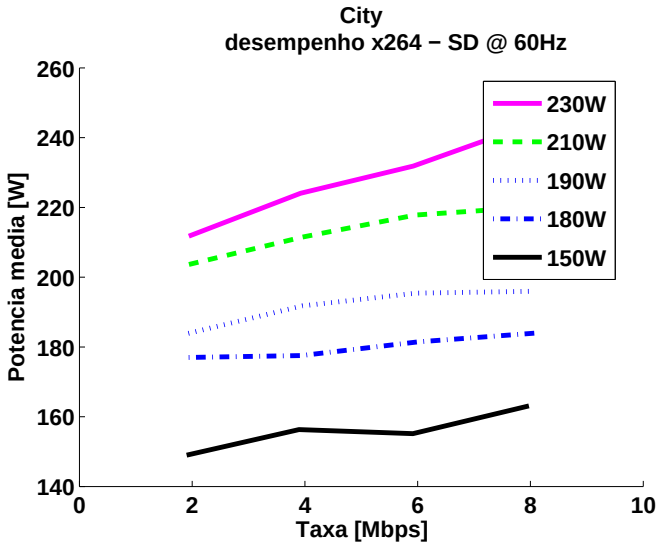
Figura 6.18 e 6.19 para sequências 720p gravadas a 50 Hz e 30 Hz, respectivamente. O controlador atua forçando a demanda energética a se adequar à provisão de energia. A velocidade de compressão mínima necessária para tratar sequências de 50 Hz e de 60 Hz resulta em demandas de potência maiores quando comparadas às observadas para a compressão de sequências de vídeo-conferência, gravadas a 30 Hz.

As curvas nas Figuras 6.16 a 6.19 são próximas umas das outras. De maneira a poder compará-las, é conveniente analisar as diferenças médias de PSNR entre duas curvas, conforme discutido na Seção 2.5. Para cada sequência, cada curva RD foi comparada com a configuração de melhor desempenho RD, que, por sua vez, é o que representa a configuração de maior consumo de potência. O consumo de potência será convenientemente representado por números relativos. Os resultados de comparação de desempenho médio são ilustrados na Figura 6.20(a) para sequências SD. O comportamento geral sugere que, ao se reduzir a potência disponível (e, portanto, a energia), as quedas em desempenho RD aumentam. Na Figura 6.20(b) os resultados mostrados são para sequências SD.

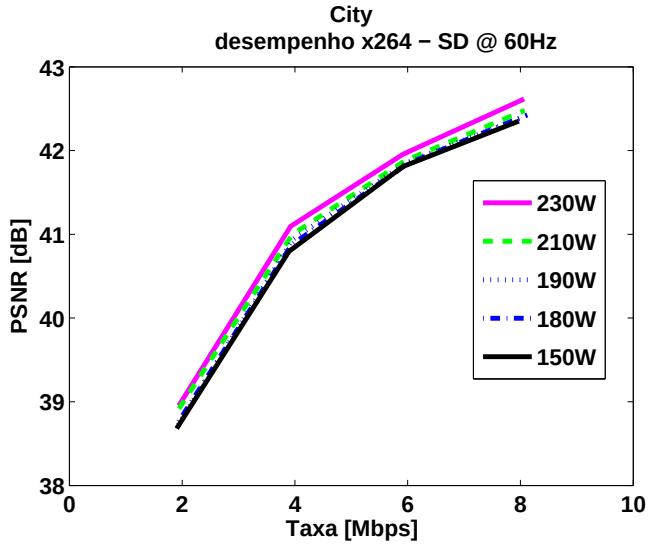
O uso de dois processadores nos ensaios surgiu da necessidade de se avaliar a sensibilidade da metodologia proposta a diferentes arquiteturas PCs. Apesar da diferença dos perfis de potência entre o processador do fabricante Intel® ($P_i = 105$ W e $P_{fp} = 260$ W) e o do fabricante AMD® ($P_i = 80$ W e $P_{fp} = 180$ W), verificou-se que os números de desempenho para o codificador otimizado para o processador do fabricante AMD® são muito similares aos apresentados para o processador da Intel®. A consolidação dos valores de acordo com a metodologia da Seção 2.5 aparece na Figura 6.21. Os valores de envoltória para escala de potência (33% de economia) são ligeiramente reduzidos pois as economias em potência média são inferiores às registradas para o processador do fabricante Intel® (35% de economia). Contudo, destaca-se que o processador do fabricante AMD® mostrou-se mais eficiente energeticamente.

O resultado principal é um esquema de compressão de vídeo em tempo real e restrito por energia que permite ao usuário escolher a provisão de energia segundo a qual deverá sem comprimida uma sequência de vídeo. Como esperado, o desempenho em termos de RD é apenado na medida em que as provisões de energia são diminuídas. Contudo, as curvas são muito próximas umas das outras e o pior caso é representado por curvas de cenas com muitos objetos em movimento ou com alto detalhamento (texturas e outras informações de alta frequência). Para sequências que demandam menor esforço de codificador ("Seq 15" e "Seq 21")[13], o esquema proposto reduz não mais que 1,3 dB em PSNR em média, enquanto resulta em 35% de economia de energia. Para as sequências SD, apesar de a velocidade mínima de compressão ser

[13]"Seq15" e "Seq21" são cenas em que há um casal de interlocutores: o fundo da sequência "Seq21" é mais detalhado que o fundo da sequência "Seq15". Ambas foram gravadas a 30 Hz e comprimidas a 30 fps.

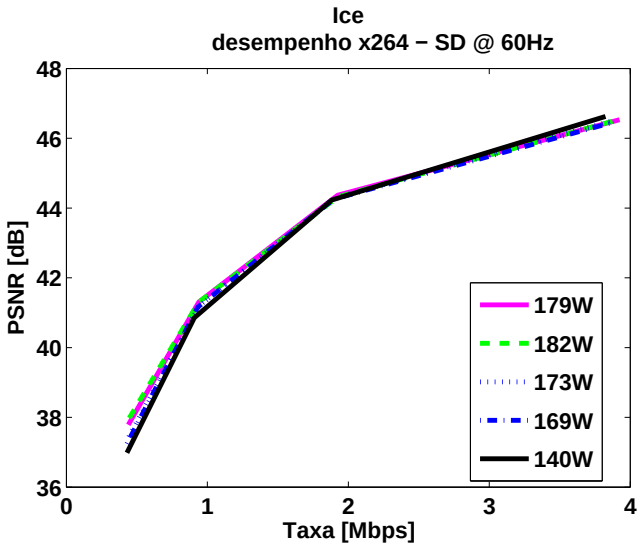


(a)

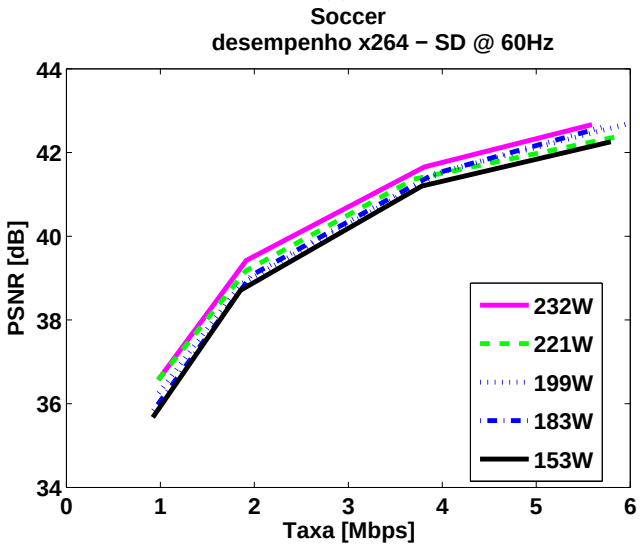


(b)

Figura 6.16: Escalonamento de energia para a compressão da sequência SD “City”. Uma margem de 10% de folga para flutuações é permitida tanto para a taxa como para a potência. (a) Potência demandada para várias taxas em diferentes níveis de potência alvo. (b) Curvas RD para compressão em tempo real para os alvos de potência ilustrados em (a). O processador é do fabricante Intel®.

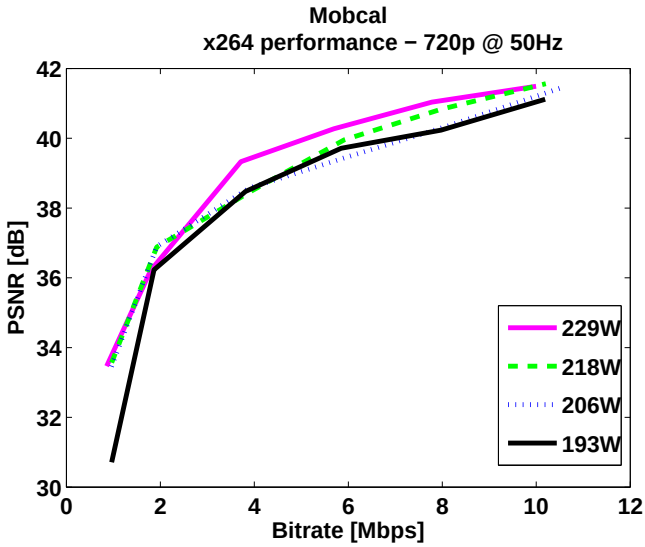


(a)

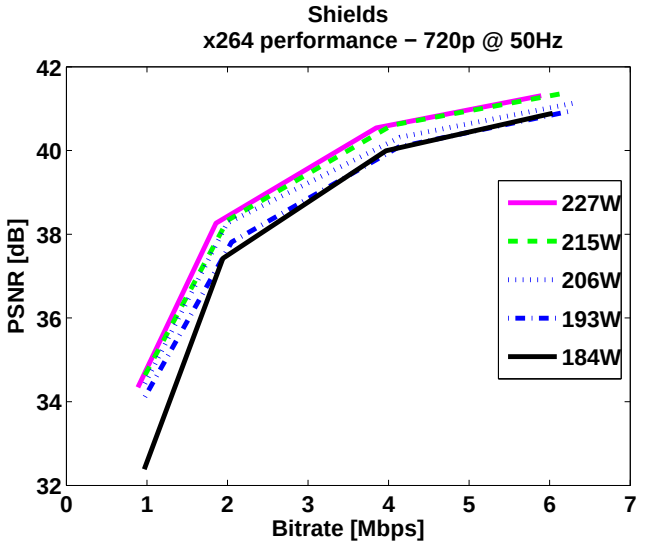


(b)

Figura 6.17: Curvas RD para as sequências (a) “Ice” e (b) “Soccer” codificadas em diferentes níveis de potência média. O processador é do fabricante Intel®.

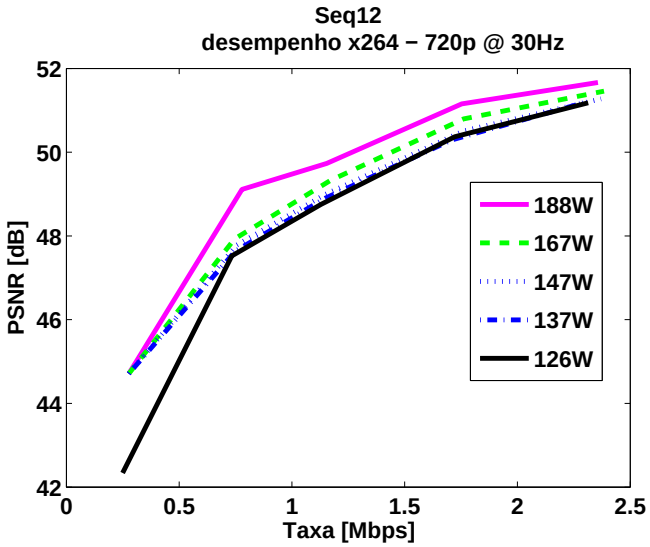


(a)

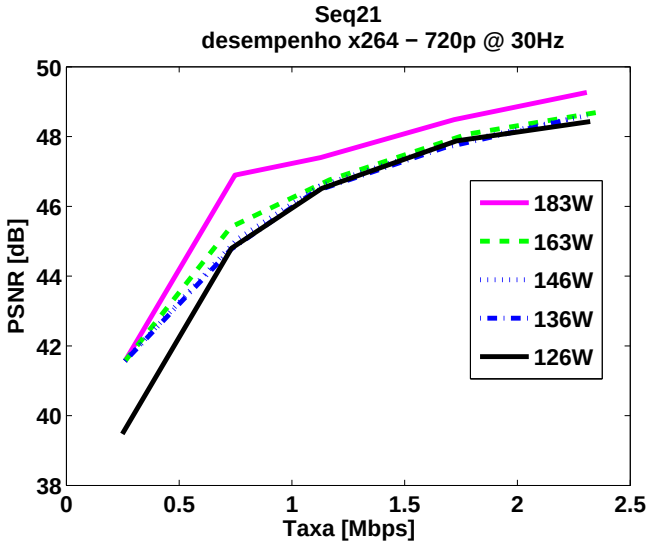


(b)

Figura 6.18: Curvas RD para as sequências (a) “Mobcal” e (b) “Shields” codificadas em diferentes níveis de potência média. O processador é do fabricante Intel®.

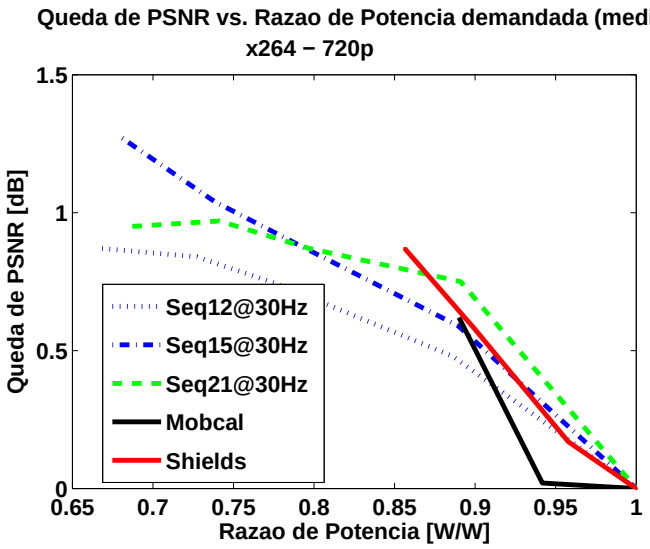


(a)

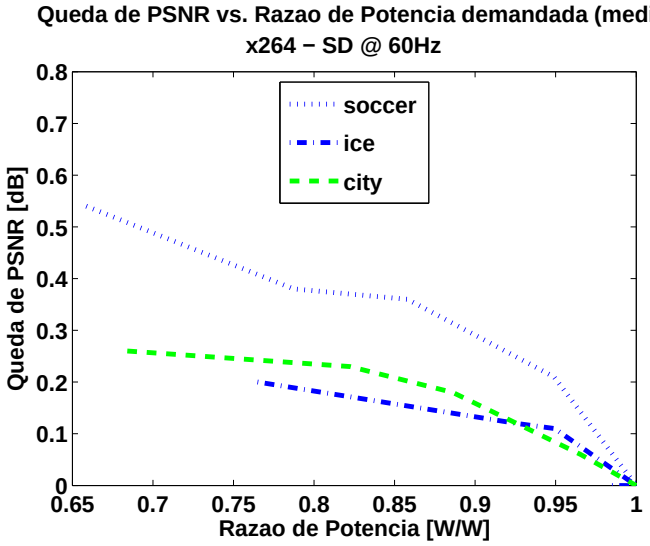


(a)

Figura 6.19: Curvas RD para as sequências (a) “Seq12” e (b) “Seq21” codificadas em diferentes níveis de potência média. O processador é do fabricante Intel®.

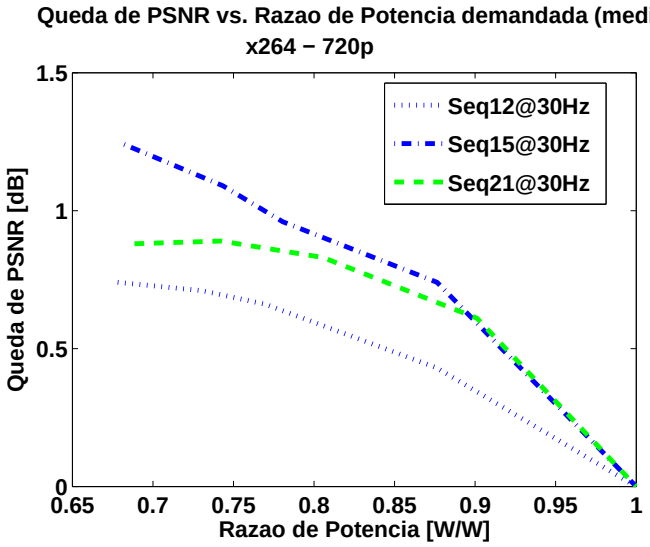


(a)

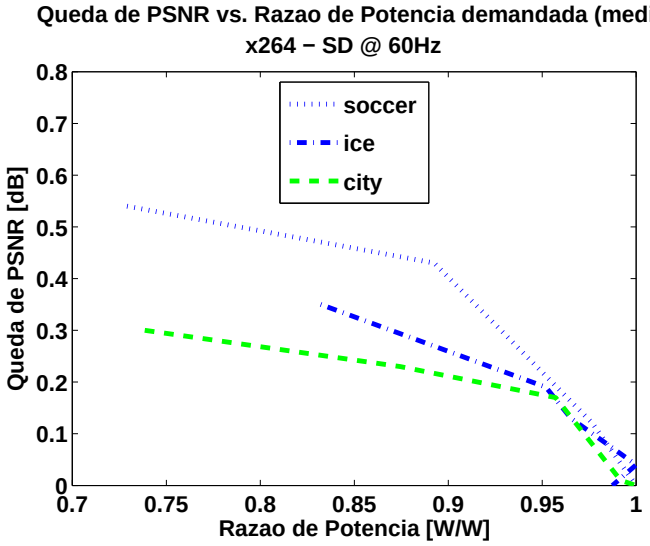


(b)

Figura 6.20: Queda em PSNR *vs.* razão de potência média para sequências de vídeo (a) SD e (b) 720p. A qualidade do vídeo comprimido aumenta na medida em que se aumenta as provisões de potência/energia. A razão de potência de 1.0W/W representa o caso de melhor desempenho RD para codificação em tempo real. O processador é do fabricante Intel®.



(a)



(b)

Figura 6.21: Queda em PSNR *vs.* razão de potência média para sequências de vídeo (a) SD e (b) 720p testadas em um processador AMD Phenom®. A qualidade do vídeo comprimido aumenta na medida em que se aumenta as provisões de potência/energia. A razão de potência de 1.0W/W representa o caso de melhor desempenho RD para codificação em tempo real. O processador é do fabricante AMD®.

maior (60 fps), as perdas em PSNR foram menores (abaixo de 0,6 dB) para valores similares de economia de energia, mesmo para sequências com alto nível de detalhes. O ajuste de conjuntos de treinamento pode ainda melhorar esses números.

Até então, o desempenho do codificador quanto à distorção foi avaliado por meio da métrica PSNR pois assim é feito na literatura na comparação entre diferentes codificadores. Conforme discutido na Seção 2.2.3, novas métricas foram propostas. Recentemente, destaca-se a aplicação da métrica SSIM na avaliação da distorção em imagens. Os valores objetivos de qualidade derivados da SSIM confinam-se no intervalo entre 0 e 1, com o valor 1 representando o caso em que os sinais de imagem comparados são idênticos.

A fim de avaliar os resultados obtidos pelo esquema de controle da codificação por provisões de energia quanto à distorção dos sinais de vídeo, calculou-se os valores de SSIM obtidos pelo esquema proposto para a componente de luminância dos sinais de vídeo codificados. Para cada codificação de um sequência de vídeo, o valor médio da SSIM dos quadros de luminância representará a medida de distorção da sequência de vídeo.

De forma a sintetizar os resultados, anotam-se quatro valores de SSIM para cada sequência, divididos em duas classes: a classe de codificação mais rápida (menor potência/energia demandada) e a classe de codificação que resulta em maior desempenho em termos de RD (maior potência/energia demandada). Para cada classe, anotam-se o valor mínimo de SSIM (baixas taxas de bits) e o valor máximo de SSIM (taxas de bits elevadas). Desta forma, determina-se a envoltória de qualidade da operação do sistema de compressão de vídeo restrito por demandas energéticas.

As Tabelas 6.1 e 6.2 exibem os valores de envoltória de qualidade para as sequências de vídeo usadas nos testes do codificador verde. Os resultados para as sequências de vídeo-conferência de 720p mostram valores elevados de SSIM (valor mínimo de $0,951$), indicando alta similaridade entre o sinal de vídeo original e o sinal de vídeo codificado. Como as sequências de vídeo-conferência possuem características peculiares de fluxo óptico, o codificador é capaz de extrair as redundâncias espaciais e temporais com baixa degradação do sinal.

Por sua vez, os valores para sequências SD são ligeiramente inferiores, com valor mínimo de 0,877 para a sequência “City”. Todavia, deve-se lembrar que o conteúdo (altas frequências) e o fluxo óptico das sequências SD é mais detalhado que o das sequência de vídeo-conferência, obrigando o codificador a tolerar maiores distorções na escala da demanda energética da codificação. Isso também é evidenciado

Tabela 6.1: Extremos de SSIM para sequências de vídeo 720p para codificação em tempo real e diferentes perfis de potência. Potência baixa refere-se à codificação que demanda a menor energia possível; por sua vez, potência alta, é a codificação que demanda a maior quantidade de energia para fornecer o melhor desempenho RD.

	Potência baixa		**Potência elevada**	
Sequência	**mín.**	**máx.**	**mín.**	**máx.**
“Seq12”	0,964	0,989	0,972	0,990
“Seq15”	**0,951**	0,987	0,962	0,988
“Seq21”	**0,951**	0,984	0,959	0,986

pelos valores de PSNR das sequências SD. Outro ponto que merece atenção é a envoltória da sequência SD “Ice”, extremamente reduzida quando comparada às outras sequências HD. Nesse caso, cabe comentar que seu conteúdo se enquadra num conjunto de sequências de vídeo de baixa complexidade devido ao seu suave fluxo óptico e baixo nível de detalhamento.

Tabela 6.2: Extremos de SSIM para sequências de vídeo SD para codificação em tempo real e diferentes perfis de potência. Potência baixa refere-se à codificação que demanda a menor energia possível; por sua vez, potência alta, é a codificação que demanda a maior quantidade de energia para fornecer o melhor desempenho RD.

	Potência baixa		**Potência elevada**	
Sequência	**mín.**	**máx.**	**mín.**	**máx.**
“City”	**0,877**	0,983	0,917	0,985
“Ice”	0,979	0,995	0,983	0,995
“Soccer”	0,894	0,986	0,912	0,987

6.8 CONSIDERAÇÕES

Propôs-se uma plataforma para codificação otimizada por complexidade e energia empregando implementação em *software* do codificador H.264/AVC que permitiu compressão em tempo real. Em vez de usar todas as modernas ferramentas de predição disponibilizadas pelo padrão H.264/AVC, decidiu-se por escolher de forma ótima um subconjunto dessas ferramentas, restritas de acordo com seu impacto em complexidade. Foi proposta, também, a avaliação da complexidade computacional em termos da energia elétrica demandada. Foi apresentado um módulo de controle capaz de prover a velocidade de codificação de vídeo requisitada por um usuário com o menor gasto de energia na compressão.

Os testes de avaliação mostram que o desempenho RD foi levemente afetado pelo módulo de controle. O esquema proposto controla a provisão de energia e a taxa de bits do canal de codificação. Além disso, não se usa descarte de quadros nem alterações na resolução das sequências de vídeo como ferramenta de atuação de controle. Apesar do impacto em desempenho, obteve-se grande escalabilidade em energia/potência demandada na codificação para processadores de plataformas PC de diferentes fabricantes. Até o conhecimento do autor, atualmente não há trabalhos na literatura que explorem a codificação otimizada por RDE, muito menos um sistema de controle capaz de receber um valor de provisão de energia/potência para compressão de vídeo com o codificador H.264/AVC.

A presente proposta é exemplo de realização de sistema de *green computing*, em que uma atividade é executada em um sistema gastando muito menos energia, às custas de pequena degradação de qualidade. A metodologia proposta pode ser rapidamente utilizada para desenvolver sistemas de codificação autônomos baseados na plataforma PC capazes de se ajustarem às condições RDC sem a necessidade de alterar a implementação do decodificador. Além disso, tecnologias como AMD® Cool'n Quiet® [14] e Intel EIST® [15] podem melhorar de maneira significativa o desempenho do esquema proposto, facilitando a detecção de momentos ociosos que permitam ao processador "dormir" enquanto espera por novos quadros a serem comprimidos, os quais chegarão a uma velocidade menor do que a de codificação alcançada pelo sistema.

Finalmente, dado que se determinam ajustes para prover demandas por taxas de canal e qualidade com máxima velocidade de execução consumindo a menor quantidade de energia possível, espera-se contribuir

[14] Disponível em: http://www.amd.com/us/products/technologies/cool-n-quiet/Pages/cool-n-quiet.aspx

[15] Disponível em: http://www.intel.com/technology/ product/demos/eist/demo.htm

para o desenvolvimento de sistemas de codificação ambientalmente “responsáveis” por meio da redução das emanações de gás carbônico nas atividades de codificação de vídeo digital.

7 CONCLUSÕES

7.1 CONSIDERAÇÕES INICIAIS

O presente trabalho tratou do estudo e da manipulação da complexidade computacional do codificador H.264/AVC. Partindo da análise do perfil de complexidade do codificador H.264/AVC em sua implementação de referência, observou-se que a maioria do esforço de computação empregado para a compressão de sequências de vídeo digital é demandada pelo módulo de predições.

Montado sobre uma estrutura DPCM híbrida, o desempenho do codificador H.264/AVC depende da qualidade das predições dos macroblocos geradas pelas suas ferramentas de compressão de sinais. Em função disso, o padrão normatizou um amplo conjunto de mecanismos de predição de sinais, capaz de explorar correlações no domínio espacial e no domínio temporal na busca pela melhor predição do sinal em termos de taxa e distorção. Em função do grande número de mecanismos, o módulo de predições do *codec* H.264/AVC agrega elevada complexidade computacional ao processo de predição.

A alteração do módulo de predições de macroblocos foi adotada como estratégia geral para a manipulação da complexidade computacional para a codificação de vídeo com o H.264/AVC em plataforma PC. A abordagem consiste em seletivamente descartar ferramentas de codificação com o intuito de escalonar o esforço computacional. Definida a estratégia geral, esta tese propôs três contribuições à codificação de vídeo escalonável em complexidade.

7.2 CONTRIBUIÇÕES DESTA TESE

Na primeira contribuição, um estudo estatístico da utilização das ferramentas de predição empregadas pelo codificador H.264/AVC em seu *software* de referência conduziu-nos a descartar, seletivamente, o testes de modos de predições menos frequentes a fim de escalonar a complexidade. Um esquema de treinamento *on-line* é empregado durante a compressão de uma sequência de vídeo de forma a derivar qual será o melhor subconjunto de ferramentas de predição para um patamar de complexidade.

A partir dos padrões de distribuições dos modos de predição em função da resolução, observou-se que as partições de macroblocos maiores são mais frequentes para sequências de maiores resoluções. Isso

revelou uma tendência de sequências de alta resolução incorrerem em menores penalidades em termos de taxa e distorção quando submetidas à escala de complexidade por meio de sorteio e descarte de modos. Contudo, a metodologia pôde ser aproveitada na escala de sequências de resoluções QCIF e CIF, embora com penalidades ligeiramente maiores em termos de taxa e distorção. Em termos de redução de complexidade, os valores atingidos foram superiores a 60% do demandado pela configuração de maior desempenho em termos de RD.

A tarefa de se escalonar a complexidade em implementações de alto desempenho de *codecs* é desafiadora, haja vista que muitos dos gargalos de complexidade computacional são devidamente inspecionados e otimizados. Todavia, a própria estrutura DPCM híbrida do codificador H.264/AVC restringe o processo de otimização da sua implementação, reservando muita complexidade a ser explorada. A segunda contribuição desta tese consiste em adotar uma abordagem de otimização RDC (*Rate vs. Distortion vs. Complexity*) para derivar esquemas de escalabilidade e controle de complexidade. Nessa atividade, foram objetos de estudos o codificador exemplo que acompanha a Intel® IPP™ (Intel Performance Primitives) e o codificador x264.

O primeiro codificador, referenciado apenas por IPP ao longo do texto, foi estudado e recebeu alterações em sua estrutura programática a fim de disponibilizar níveis finos de escalabilidade em complexidade. Resultados com apenas uma *thread* de codificação mostraram valores de escala de complexidade de até 80%, ou seja, um fator de $5\times$ adicionado à velocidade de compressão, às custas de aumento de taxa de transmissão inferiores a 10%. Os valores de velocidade de compressão passam a ser de interesse, pois são capazes suprir um sistema de codificação de vídeo em tempo real caso mais *threads* sejam habilitadas.

O codificador x264 também foi submetido à análise de complexidade e otimização RDC, derivando reduções de complexidade ainda superiores às relatadas para o IPP, naturalmente às custas de degradação de desempenho em termos de RD. O codificador x264 otimizado foi inserido num arranjo de controle de velocidade em malha fechada, esquema capaz de seguir uma referência de velocidade de compressão por meio de ajustes adaptativos nos parâmetros de predições do codificador. O controlador de velocidade otimizado é capaz de garantir a compressão mais rápida, demandando menor taxa de canal de transmissão e resultando em menor distorção do sinal codificado. Valores de velocidades para apenas uma *thread* de codificação mostraram que o sistema é capaz de prover codificação em tempo real de vídeo HD usando o H.264/AVC em implementação em *software*. Neste estudo, em particular, controlou-se a velocidade de compressão do x264 por meio de valores absolutos, indicados em fps. É raro encontrar referências a

valores absolutos de velocidade de codificação na literatura, pois os autores concentram-se em apresentar resultados de velocidades relativos e, quase sempre, omitem a linha de referência absoluta de comparação (o denominador da razão de velocidade).

A terceira contribuição desta tese consiste num sistema de codificação ciente de restrições ambientais. Em vez de somente otimizar um codificador de vídeo tendo em vista os eixos de taxas, distorção e complexidade, o *codec* foi analisado também sobre o eixo da energia elétrica envolvida no processo de compressão. Usando um computador pessoal (PC) como plataforma de testes, estudou-se o perfil de demanda energética desse sistema e propôs-se uma metodologia para a otimização em termos de RDE (*Rate vs. Distortion vs. Energy*) do codificador H.264/AVC. O objetivo é prover um sistema capaz de escalonar as demandas energéticas de uma implementação em *software* de um codificador de vídeo, de forma a garantir compressão de vídeo em tempo real, demandando a menor taxa de canal possível para determinado nível de distorção.

O arranjo de codificação proposto é capaz de seguir uma referência de potência média demandada indicada pelo usuário, enquanto codifica o vídeo digital em tempo real. Nessa implementação, o controle de taxa de bits de canal estava ativado. A comparação entre os desempenhos RD de diferentes patamares de potência foram feitos mediante a comparação da diferença média de PSNR entre as curvas de desempenho RD. Resultados consolidados mostraram reduções de até 35% de potência média (e, consequentemente, de energia demandada) às custas de degradações de PSNR inferiores a 1.3 dB no pior dos casos. O sistema obtido é capaz de codificar vídeos de alta resolução em tempo real a partir da implementação em *software* x264.

Diferenciais deste trabalho em relação ao que é encontrado na literatura de sistemas de codificação quanto a escalonamento de complexidade são:

1. testes e validação das contribuições em processadores de diferentes fabricantes (não-vinculação a característica específica de modelo de processador);
2. o tratamento da complexidade computacional em termos absolutos (a velocidade de compressão é medida em fps);
3. o não uso de descarte de quadros para garantir a velocidade de compressão desejada;
4. o tratamento de sequências de vídeo em alta resolução no contexto de otimização RDC; e
5. o uso de energia demandada no processo de otimização RDC.

O componente criado aqui, um módulo de controle de complexidade “ciente” de consumo energético, pode ser aplicado a outros problemas computacionais e é de interesse do autor analisar sua aplicação nestes. A avaliação da integração do controlador em sistemas de larga escala é promissora, especialmente na verificação de seu comportamento no auxílio a heurísticas de escalonamento de atividade e ao balanceamento de cargas.

7.3 PERSPECTIVAS DE TRABALHOS FUTUROS

Trabalhos futuros poderão estender a linha de pesquisa aqui explorada nas seguintes direções:

- Explorar as estatísticas dos modos de predição por meio de sorteio com ordenação em sistemas de transcodificação, a fim de se reduzir a complexidade do processo de conversão entre diferentes padrões de codificação de vídeo.
- Estender a abordagem de otimização RDE apresentada para sistemas de maior porte, em que mais de um equipamento ou processo seja responsável pela codificação de vídeo. Um exemplo de pequeno porte seria um gravador “verde” de vídeo digital para vigilância com múltiplos canais de aquisição; por sua vez, uma fazenda de servidores (*server farm*) representaria um sistema de grande porte.
- Especializar a abordagem de otimização RDE apresentada para sistemas móveis, para os quais poderão ser disponibilizados chips especializados (mas configuráveis) para compressão de vídeo digital.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] RICHARDSON, I. E. G. *H.264 and MPEG-4 Video Compression*. [S.l.]: John Wiley & Sons Ltd, 2003.

[2] RANGANATHAN, P. From microprocessors to nanostores: Rethinking system building blocks for the data-centric era. *Computer*, IEEE, n. 99, p. 39–48, 2011.

[3] OPPENHEIM, A. V.; SCHAFER, R. W.; BUCK, J. R. *Discrete-Time Signal Processing*. [S.l.]: Prentice Hall, 1998.

[4] SAYOOD, K. *Introduction to Data Compression*. [S.l.]: Morgan Kauffmann Publishers, 2000.

[5] JUANG, B. Quantification and transmission of information and intelligence – history and outlook. *Signal Processing Magazine, IEEE*, v. 28, n. 4, p. 90–101, 2011.

[6] ITU-T. *Video codec for audiovisual services at p x 64 kbit/s*. [S.l.], Nov. 1990.

[7] ISO-IEC JTC1. *Coding of Moving Pictures and Associated Audio for Storage Media at up to About 1.5Mbps - Part 2: Visual*. [S.l.], 1993.

[8] ITU-T and ISO/IEC JTC 1 - ISO/IEC 13818-2 (MPEG-2). *Generic coding of moving pictures and associating audio information - Part 2: Video*. [S.l.], Nov. 1994.

[9] HASKEL, B. G.; PURI, A.; NETRAVALLI, A. N. *Digital Video: An Introduction to MPEG-2*. [S.l.]: Chapmann and Hall, 1997.

[10] ITU-T. *ITU-T Recommendation H.263, Video coding for low bit rate communication*. [S.l.], Nov. 2000.

[11] JTC1, I.-I. *ISO/IEC 14496-2 (MPEG-4 visual version 1), Coding of Audio Visual Objects - Part 2: Visual*. [S.l.], Nov. 1999.

[12] WIEGAND, T. et al. Rate-Constrained Coder Control and Comparison of Video Coding Standards. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, n. 7, p. 688–703, July 2003.

[13] JVT of ISO/IEC MPEG and ITU-T VCEG. *Advanced Video Coding for Generic Audiovisual Services*. [S.l.], March 2005.

[14] OSTERMANN, J. et al. Video coding with H.264/AVC: tools, performance, and complexity. *IEEE Circuits and Systems Magazine*, p. 7–28, Jan-Mar 2004.

[15] HUANG, Y.-Y. et al. Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 16, n. 4, p. 507–522, Apr. 2006.

[16] SULLIVAN, G. J.; TOPIWALA, P.; LUTHRA, A. The H.264/AVC advanced video coding standard: Overview and introduction to the fidelity range extensions. *Proc. SPIE Conference on Applications of Digital Image Processing XXVII*, v. 5558(1), p. 454–474, Aug. 2004.

[17] SULLIVAN, G. et al. New standardized extensions of mpeg4-avc/h. 264 for professional-quality video applications. In: IEEE. *Image Processing, 2007. ICIP 2007. IEEE International Conference on*. [S.l.], 2007. v. 1, p. 13–16.

[18] SCHWARZ, H.; MARPE, D.; WIEGAND, T. Overview of the scalable video coding extension of the h. 264/avc standard. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 17, n. 9, p. 1103–1120, 2007.

[19] SULLIVAN, G. Standards-based approaches to 3d and multiview video coding. In: *Proceedings of SPIE Conference on Applicationsof Digital Image Processing*. [S.l.: s.n.], 2009. v. 7443.

[20] CHEN, Y. et al. The emerging mvc standard for 3d video services. *EURASIP Journal on Applied Signal Processing*, Hindawi Publishing Corp., v. 2009, p. 8, 2009.

[21] SULLIVAN, G.; OHM, J. Meeting report of the first meeting of the joint collaborative team on video coding (jct-vc), dresden, de, 15-23 april, 2010. *document JCTVC-A200 of JCT-VC*, 2010.

[22] SULLIVAN, G.; OHM, J. Recent developments in standardization of high efficiency video coding (HEVC). *SPIE Applications of Digital Image Processing XXXIII, Proc. SPIE*, v. 7798, 2010.

[23] BONDI, A. Characteristics of scalability and their impact on performance. In: ACM. *Proceedings of the 2nd international workshop on Software and performance*. [S.l.], 2000. p. 195–203.

[24] HARTMANIS, J.; STEARNS, R. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, v. 117, n. 5, p. 285–306, 1965.

[25] YAMADA, H. Real-time computation and recursive functions not real-time computable. *Electronic Computers, IRE Transactions on*, IEEE, n. 6, p. 753–760, 1962.

[26] FORTNOW, L.; HOMER, S. A short history of computational complexity. *Bulletin of the EATCS*, v. 80, p. 95–133, 2003.

[27] BARROSO, L. A. The price of performance. *Queue*, ACM, v. 3, n. 7, p. 48–53, 2005.

[28] ALBERS, S. Energy-efficient algorithms. *Communications of the ACM*, ACM, v. 53, n. 5, p. 86–96, 2010.

[29] SILVEN, O.; JYRKKÄ, K. Observations on power-efficiency trends in mobile communication devices. *EURASIP Journal on Embedded Systems*, Hindawi Publishing Corp., v. 2007, n. 1, p. 17–27, 2007.

[30] ALLEBACH, J. P. Image scanning, sampling and interpolation. In: BOVIK, A. (Ed.). *Handbook of Image and Video Processing*. [S.l.]: Academic Press, 2000. p. 629–644.

[31] DUBOIS, E. Video sampling and interpolation. In: BOVIK, A. (Ed.). *Handbook of Image and Video Processing*. [S.l.]: Academic Press, 2000. p. 645–654.

[32] DINIZ, P. S. R.; SILVA, E. A. B. da; NETTO, S. L. *Digital Signal Processing*. [S.l.]: Cambridge University Press, 2006. (Series in Imaging Science and Technology: System Analysis and Design).

[33] SHARMA, G. Color fundamentals for digital imaging. In: SHARMA, G. (Ed.). *Digital Color Imaging Handbook*. [S.l.]: CRC Press, 2003. p. 1–114.

[34] HUNT, R. W. G. *The Reproduction of Colour*. England: John Wiley and Sons, 2004. (Series in Imaging Science and Technology).

[35] Recommendation ITU-T BT.601-5. *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*. [S.l.], 1995.

[36] WU, H.; RAO, K. *Digital video image quality and perceptual coding*. [S.l.]: CRC press, 2006.

[37] WANG, Z. et al. Image quality assessment: From error measurement to structural similarity. *IEEE transactions on image processing*, Citeseer, v. 13, n. 4, p. 600–612, 2004.

[38] PRATT, W. Spatial transform coding of color images. *Communication Technology, IEEE Transactions on*, IEEE, v. 19, n. 6, p. 980–992, 1971.

[39] TESCHER, A.; COX, R. V. An adaptive transform coding algorithm. In: *in Proc. of IEEE XI ICC*. [S.l.: s.n.], 1976. p. 47–20–47–25.

[40] AHMED, N.; NATARAJAN, T.; RAO, K. R. On image processing and a discrete cosine transform. *IEEE Trans. on Computers*, IEEE, C-23, n. 1, p. 90–93, 1974.

[41] PENNEBAKER, W. B.; MITCHELL, J. L. *JPEG: Still Image Data Compression Standard*. New York, USA: Van Nostrand Reinhold, 1993.

[42] PRATT, W. K. *Digital Image Processing: PIKS Inside*. California, USA: Wiley-Interscience, 2001.

[43] WU, H. R.; RAO, K. R. Critical issues and challenges. In: WU, H. R.; RAO, K. R. (Ed.). *Digital Video Image Quality and Perceptual Coding*. [S.l.]: CRC Press, 2006. p. 543–573.

[44] HUNG, E. M. *Compensação de Movimento utilizando Blocos Multi-escala e Forma Variável em um CODEC de Vídeo Híbrido*. [S.l.]: Dissertação de Mestrado em Engenharia Elétrica com ênfase em Telecomunicações, Publicação PPGENE.DM 304/07, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 73p., 2007.

[45] TAUBMAN, D. S.; MARCELLIN, M. W. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. [S.l.]: Kluwer Academic, 2002.

[46] ISO/IEC. *Image Coding System: Motion JPEG 2000 (JPEG2000 Part 3)*. [S.l.], September 2003.

[47] MOUNTS, F. W. Frame-to-frame digital processing of tv pictures to remove redundancy. In: GORDON AND BREACH, NEW YORK, NY. *Proc. 1969 Symp. Picture Bandwidth Compression*. [S.l.], 1972. p. 653–673.

[48] ROCCA, F. Television bandwidth compression utilizing frame-to-frame correlation and movement compensation. In: GORDON AND BREACH, NEW YORK, NY. *Proc. 1969 Symp. Picture Bandwidth Compression*. [S.l.], 1972.

[49] BROFFERIO, S. et al. Redundancy reduction of video signals using movement compensation. *Alta Frequenza*, v. 43, p. 836–842, 1974.

[50] TAKI, Y.; HATORI, M.; TANAKA, S. Interframe coding that follows the motion. In: IECEJ. *Proc. Institute of Electronics and Communication Engineers of Japan*. [S.l.], 1974. p. 1263.

[51] KOGA, T. et al. Motion compensated interframe coding for video conferencing. In: *Proc. Nat. Telecommun. Conf*. [S.l.: s.n.], 1981. v. 5, n. 3, p. 1–5.

[52] MARPE, D.; SCHWARZ, H.; WIEGAND, T. Context-based adaptative binary arithmetic coding in H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, p. 620–636, July 2003.

[53] QUEIROZ, R. L. de et al. Fringe benefits of the H.264/AVC. In: *Proc. International Telecommunication Symposium*. [S.l.: s.n.], 2006. p. 208–212.

[54] XU, X.; HE, Y. Improvements on fast motion estimation strategy for h. 264/avc. *Circuits and Systems for Video Technology, IEEE Transactions on*, IEEE, v. 18, n. 3, p. 285–293, 2008.

[55] MALVAR, H. S. et al. Low-Complexity transform and quantization in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, n. 13, p. 590–603, July 2003.

[56] LIST, P. et al. Adaptive deblocking filter. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, p. 614–619, July 2003.

[57] COTE, G. et al. H.263+: Video coding at low bit rates. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 8, p. 849–866, Nov. 1998.

[58] LUTHRA, A.; SULLIVAN, G. J.; WIEGAND, T. H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 13, n. 7, July 2003.

[59] ITU-T. *Amendment 1: Support of additional colour spaces and removal of the High 4:4:4 Profile*. [S.l.], June 2006.

[60] WANG, H. et al. *4G wireless video communications*. [S.l.]: Wiley, 2009.

[61] BJONTEGAARD, G. Calculation of average PSNR differences between RD-curves. *VCEG-M33*, April 2001.

[62] TOURAPIS, H.-Y. C.; TOURAPIS, A. M.; TOPIWALA, P. Fast Motion Estimation within the JVT Codec. *JVT-E023, 5th Meeting: Geneva, Switzerland*, October 2002.

[63] CHEN, Z.; ZHOU, P.; HE, Y. Hybrid unsymmetrical-cross multi-hexagon-grid search strategy for integer pel motion estimation in H.264. *Proc. Picture Coding Symposium*, Apr. 2003.

[64] TOURAPIS, H.-Y. C.; TOURAPIS, A. M.; TOPIWALA, P. Fast motion estimation within the H.264 Codec. *Proc. of Intl. Conf. on Multimedia and Expo. ICME*, v. 3, p. 517–520, July 2003.

[65] KUO, T.-Y.; CHAN, C.-H. Fast variable block size motion estimation for H.264 using likelihood and correlation of motion field. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 16, n. 10, p. 1185–1195, Oct. 2006.

[66] KIM, B.; SONG, S.-K.; CHO, C.-S. Efficient inter-mode decision based on contextual prediction for the P-slice in H.264/AVC video coding. *Proc. IEEE International Conference on Image Processing*, p. 1333–1336, September 2006.

[67] LA, B.; EOM, M.; CHOE, Y. Fast mode decision for intra prediction in H.264/AVC encoder. *Proc. IEEE International Conference on Image Processing*, V, p. 321–324, September 2007.

[68] HWANG, C.; ZHUANG, S.; LAI, S.-H. Efficient intra mode selection using image structure tensor for H.264/AVC. *Proc. IEEE International Conference on Image Processing*, V, p. 289–292, September 2007.

[69] HE, Z. et al. Power-rate-distortion analysis for wireless video communication under energy constraints. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 15, n. 5, p. 645–658, May 2005.

[70] AKYOL, E.; MUKHERJEE, D.; LIU, Y. Complexity Control for Real-Time Video Coding. *Proc. IEEE International Conference on Image Processing*, I, p. 77–80, September 2007.

[71] MOECKE, M.; SEARA, R. Sorting rates in video encoding process for complexity reduction. *IEEE Transactions on Circuits and Systems for Video Technology*, Institute of Electrical and Electronics Engineers, v. 20, n. 1, p. 88–101, 2010.

[72] PARK, I. et al. Efficient design and implementation of visual computing algorithms on the gpu. In: IEEE. *Image Processing (ICIP), 2009 16th IEEE International Conference on*. [S.l.], 2009. p. 2321–2324.

[73] INTEL. Intel Integrated Performance Primitives. *http://software.intel.com/en-us/intel-ipp/*.

[74] KANNANGARA, C. et al. Complexity control of H.264/AVC based on mode-conditional cost probability distributions. *IEEE Transactions on Multimedia*, IEEE, v. 11, n. 3, p. 433–442, 2009.

[75] SU, L. et al. Complexity-constrained H.264 video encoding. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 19, n. 4, p. 477–490, 2009.

[76] MERRITT, L.; VANAM, R. Improved rate control and motion estimation for H.264 encoder. In: *Proc. IEEE International Conference on Image Processing*. [S.l.: s.n.], 2007. v. 5.

[77] KHAN, A. et al. Video quality prediction models based on video content dynamics for h. 264 video over umts networks. *International Journal of Digital Multimedia Broadcasting*, Hindawi Publishing Corporation, v. 2010, 2010.

[78] FONSECA, T. A.; QUEIROZ, R. L. de. Complexity reduction techniques for the compression of high-definition video. *Journal of Communications and Information Systems*, v. 24, n. 1, p. 1–9, Apr. 2009.

[79] SWAROOP, K.; RAO, K. Performance analysis and comparison of jm 15.1 and intel ipp h. 264 encoder and decoder. In: IEEE. *System Theory (SSST), 2010 42nd Southeastern Symposium on*. [S.l.], 2010. p. 371–375.

[80] SULLIVAN, G. J.; WIEGAND, T. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, v. 15, n. 6, p. 74–90, Nov. 1998.

[81] OSBORNE, M.; RUBINSTEIN, A. *A course in game theory*. [S.l.]: The MIT press, 1994.

[82] SEDRA, A.; SMITH, K. *Microelectronic circuits*. [S.l.]: Oxford University Press, USA, 1998.

[83] BRYNJOLFSSON, E.; HOFMANN, P.; JORDAN, J. Cloud computing and electricity: beyond the utility model. *Communications of the ACM*, ACM, v. 53, n. 5, p. 32–34, 2010.

[84] MURUGESAN, S. Harnessing green it: Principles and practices. *IT professional*, IEEE, v. 10, n. 1, p. 24–33, 2008.

[85] BOYD, J. *Computing experts unveil superefficient "inexact" chip*. http://news.rice.edu/2012/05/17/computing-experts-unveil-superefficient-inexact-chip/, último acesso em 02/07/2012.

[86] KANNANGARA, C. S. et al. Complexity control of H.264 based on a Bayesian framework. *Proc. Picture Coding Symposium*, Lisbon, Portugal, Nov. 2007.