2
Arquiteturas Clássicas de Codificação Distribuída de Vídeo (*Distributed Video Coding* – DVC) e suas aplicações

A Codificação Distribuída de Vídeo, do inglês *Distributed Video Coding*, cujo acrônimo é DVC, baseia-se na codificação da fonte com informação lateral no decodificador, cenário este baseado na teoria da informação de Shannon e especializado nos Teoremas de Slepian-Wolf e de Wyner-Ziv (vide apêndice B sobre Teoria da Informação).

Diferentemente do que ocorre nos paradigmas tradicionais de codificação de vídeo, também conhecidos como codificadores híbridos, como os padrões ISO MPEG-1 [9], ITU-T H.263 [21] e ITU-T H.264 [22], onde o codificador requer um maior esforço computacional que o decodificador [23], na proporção de 5 a 10 vezes, principalmente devido à técnica de estimação de movimento, o paradigma DVC tem como princípio deslocar a complexidade para o decodificador devido aos requisitos técnicos do codificador, que normalmente está embarcado em dispositivos com menor capacidade de processamento ou poder computacional e com maiores limitações de consumo de bateria. Assim, uma das principais abordagens ao definir uma arquitetura de um *codec* DVC é deslocar a estimação de movimento para o decodificador.

Embora as teorias que são bases para os fundamentos do DVC remontem às décadas de cinquenta e setenta, os estudos para implementações práticas desse paradigma são recentes (e ainda estão em aberto), devido principalmente ao desenvolvimento de dispositivos eletrônicos com grande capacidade de mobilidade em rede e seus requisitos.

Os primeiros estudos e proposta de um codificador Wyner-Ziv foram descritos somente em 1999 [24], tratando da codificação assimétrica de uma fonte com informação lateral para fontes binárias e gaussianas, numa arquitetura chamada DISCUS, criada com o intuito de ser utilizada em redes de sensores. Depois disso, somente em 2002, foram publicados novos estudos, os quais deram origem às três principais arquiteturas consideradas até hoje "estado da arte" em DVC e base para novas pesquisas nesse ramo. Originados a partir dessas

publicações, os principais centros de estudo dessa tecnologia se estabeleceram na universidade de Stanford, na universidade de Berkeley e mais recentemente, um consórcio de universidades na Europa produziram os resultados mais atualizados. Assim, antes de se mostrar a abordagem DVC desta dissertação, se apresentarão as arquiteturas dessas três principais referências acima citadas.

2.1
Arquitetura DVC da Universidade de Stanford

A arquitetura de Stanford foi proposta inicialmente apenas para o domínio do *pixel* ou domínio espacial [25, 26] e depois expandida para o domínio da transformada [27]. Nas primeiras versões, foi implementada utilizando códigos turbo (*turbo codes*) [25, 26, 27] no codificador Slepian-Wolf como codificador de canal. Nos modelos mais recentes, o codificador de canal foi alterado, passando a utilizar código LDPC, acrônimo de *low-density parity-check codes* [28, 29].

A figura 2.1 ilustra o diagrama de blocos da arquitetura genérica do DVC de Stanford, ou seja, de um *codec* que suporta suas várias versões.

Inicialmente o sinal $X(i)$ é dividido em dois grupos de quadros. Os quadros ímpares são chamados quadros chave, definidos como $C(i) = X(2i + 1)$, $i = 0, 1, 2, ...$, e são codificados utilizando técnicas de codificação *intra-frame*. Estes quadros serão utilizados no decodificador para gerar a informação lateral. Os quadros pares são chamados quadros Wyner-Ziv, definidos como $W(i) = X(2i)$, $i = 1, 2, ...$, e são codificados com técnicas de codificação Wyner-Ziv, que é a utilização de um processo de quantização seguido de um codificador Slepian-Wolf.

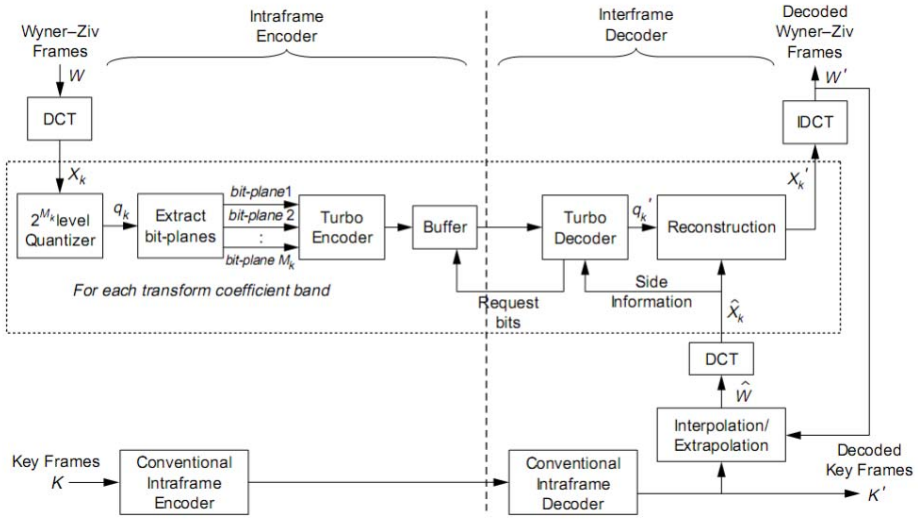

Figura 2.1 – *Codec* DVC – Arquitetura da Universidade de Stanford
Fonte: Ref [25]

O codificador Slepian-Wolf, que se baseia no teorema de mesmo nome, é um codificador considerado para situações sem perdas, ou seja, permite a reconstrução perfeita. Assim, podemos concluir que a função de quantização é a responsável pela distorção introduzida no *codec* Wyner-Ziv, cujo teorema de mesmo nome admite reconstruções com perdas em troca de uma redução considerável de taxa de transmissão.

Podemos definir o processo de codificação Wyner-Ziv da arquitetura DVC de Stanford da seguinte forma:

- Os quadros *W(i)* são quantizados através de um passo de quantização definido;
- O quadro quantizado é reordenado em uma estrutura chamada plano de bits, em inglês *bitplanes*. Um quadro dá origem a vários *bitplanes*. Esse processo será detalhado no codificador proposto nesta dissertação. A essa estrutura quantizada e reordenada de acordo com as definições das *bitplanes* chamaremos de X_q;

- X_q alimenta um codificador de canal sistemático utilizando alguma técnica de código corretor de erros, códigos turbo por exemplo [19], gerando o sinal concatenado [X_q |P_d], onde P_d são os bits de paridade adicionados pelo codificador de canal;
- Os bits de paridade P_d são enviados pelo canal, ao encontro da entrada do decodificador, à medida que sejam requeridos por este através do canal de retorno;
- Os bits de paridade são enviados até o *codec* conseguir a reconstrução exata do sinal quantizado, ou seja, $\hat{X}_q = X_q$.

A decodificação Wyner-Ziv da arquitetura DVC Stanford pode ser definida através dos seguintes passos:

- Os quadros chave são decodificados através de uma técnica de decodificação intra-frame e utilizados para gerar a informação lateral. O quadro chave anterior $C(i) = X(2i - 1)$ e o quadro chave posterior $C(i) = X(2i + 1)$ ao quadro Wyner-Ziv atual $W(i) = X(2i)$ geram o quadro interpolado $Y(i)$;
- $Y(i)$ é quantizado e reordenado para gerar Y_q, que é a nossa informação lateral;
- Y_q é concatenado aos bits de paridade recebidos, dando origem à [Y_q |P_d]. Para o decodificador do canal, Y_q é uma versão ruidosa de X_q após passar pelo canal;
- O decodificador de canal vai requerer, através do canal de retorno, a quantidade de bits de paridade necessários para reconstruir $\hat{X}_q$;
- $\hat{X}_q$ e Y são usados para reconstruir o quadro $\hat{W}(i)$ que é diferente de $W(i)$ através da máxima verossimilhança $\hat{W}(i) = E[W(i) \mid X_q, Y]$.

Analisando os processos de codificação e decodificação deste *codec*, podemos concluir que houve uma redução da complexidade do codificador à medida que nenhuma técnica de estimação de movimento é utilizada. Porém, o decodificador é mais complexo, pois na geração do quadro interpolado Y,

utilizamos algoritmos de interpolação que são tão custosos computacionalmente quanto às técnicas de estimação de movimento.

Os dois pontos críticos neste tipo de abordagem, para alavancar a qualidade do sinal recuperado, são a geração da informação lateral e o mecanismo utilizado na codificação de canal. Um dos pontos críticos e um problema em aberto nas arquiteturas modernas de DVC é a necessidade de termos um canal de retorno para conseguirmos uma boa qualidade e taxas consideradas razoáveis, como é o caso do DVC de Stanford.

Como visto nos resultados dos trabalhos publicados baseados nesta arquitetura [25, 26, 27], o desempenho do DVC de Stanford é superior ao padrão H.263+ com codificação intra, ou seja, codificação de quadros *I I I*..., e inferior quando o mesmo padrão utiliza codificação *I P I P*.... Comparando o DVC Stanford ao H.264/AVC o resultado é semelhante, com um resultado superior no caso da codificação do H.264/AVC *I I I*... e inferior no caso da codificação do H.264/AVC *I P I P*...

2.2
Arquitetura PRISM, da Universidade de Berkeley

A segunda arquitetura de referência para as pesquisas em DVC é a proposta pela Universidade de Berkeley, chamada de PRISM [30, 31, 32], acrônimo de *Power-efficient, robust, high-compression syndrome-based multimedia coding*, com uma abordagem diferente da utilizada no DVC de Stanford. Basicamente, o PRISM divide a imagem em macroblocos e aplica a cada um deles uma codificação distribuída baseada em *cosets* [32].

A técnica utilizada no codificador PRISM é através da divisão do quadro em macroblocos. Um macrobloco X de tamanho 8 x 8 ou 16 x 16 tem sua melhor estimativa Y calculada. A relação entre X e Y pode ser modelada como $Y = X + N$, onde N é um ruído e X e N são variáveis aleatórias independentes com distribuição Laplaciana. O macrobloco X é codificado e assumindo que o decodificador tem acesso a Y, o codificador procura por um codificador de canal para casar com o ruído de correlação N existente entre X e sua predição Y.

O processo de codificação do *codec* PRISM pode ser definido de acordo com os seguintes passos:

- O quadro *X(i)* é dividido em macroblocos e transformado através de uma DCT. Com base na estimação do ruído de correlação Wyner-Ziv, os macroblocos são classificados. Esta classificação determina a expectativa da qualidade do bloco decodificado e dependendo de seu resultado o codificador escolhe por codificar o macrobloco de forma *intra*;
- Após isso, os coeficientes transformados dos macroblocos são divididos em coeficientes de alta frequência e coeficientes de baixa frequência, de acordo com a classe do macrobloco, ou seja, de forma variável;

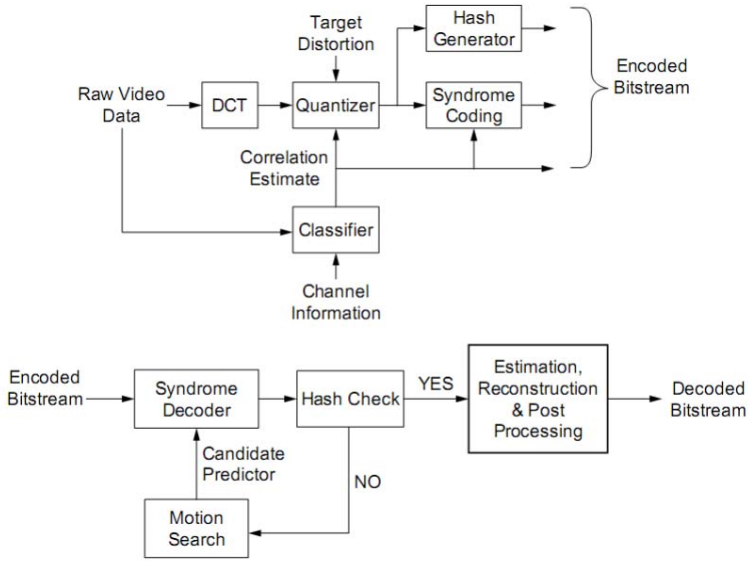

Figura 2.2 – *Codec* DVC – Arquitetura PRISM da Universidade de Berkeley
Fonte: Ref [32]

- Os coeficientes de alta frequência são quantizados e codificados utilizando codificação de entropia, sendo o passo de quantização definido pela classificação do macrobloco;
- Os coeficientes de baixa frequência são quantizados em dois passos, o primeiro da mesma forma que os coeficientes de alta frequência e

o segundo através da análise da distorção permitida. A diferença dos dois passos é enviada ao codificador de entropia enquanto o resultado do primeiro passo é utilizado para gerar o código de síndrome;

- O codificador PRISM também gera um *hash* para cada macrobloco utilizando CRC, acrônimo de *cyclic redundancy check*;
- Após a geração de todos esses dados, eles são enviados para o decodificador. Assim, a *bit stream* do codificador PRISM é composta pelos coeficientes quantizados de alta frequência, a diferença dos coeficientes de baixa frequência, o CRC de cada macrobloco e a síndrome.

A codificação por síndrome [19] é um tipo de codificação de canal onde no lugar de enviar a paridade é enviada o que chamamos de síndrome do código. Seja X uma fonte com G sendo a matriz de paridade de um código por bloco linear, então, S é a síndrome do código, sendo $S = X.G$. A funcionalidade da síndrome é similar à da paridade, sendo que ambas são usadas para corrigir erros de transmissão. A síndrome gera o que chamamos de *coset*. Se a fonte X possui um alfabeto χ, ao dividir esse alfabeto em subconjuntos por regiões, cada subconjunto é o que chamamos de *coset*. A síndrome indica a qual subconjunto ou *coset* uma determinada informação pertence.

A decodificação no PRISM segue o seguinte fluxo:

- Blocos candidatos à informação lateral no quadro decodificado anteriormente são selecionados mediante uma técnica de estimação de movimento modificada, usando os bits residuais do refinamento da quantização para realizar a procura dos blocos candidatos;
- No primeiro candidato, são substituídos os coeficientes de alta frequência pelos existentes no *bit stream*, gerando assim a informação lateral ou versão ruidosa da informação original;
- A informação lateral é corrigida com o uso da síndrome seguida de uma verificação usando o CRC recebido. Se a verificação é incorreta, o processo é reiniciado retornando ao passo anterior com um novo candidato e assim sucessivamente até obter-se uma

verificação válida. Se todas as verificações forem incorretas após percorrer todos os candidatos, então o candidato que gerou a informação mais próxima à correta é selecionado.

Um detalhe importante nesta arquitetura é que não é necessário um canal de retorno, o que tem um lado positivo. Porém, como não existe a comunicação proporcionada por este canal, que fornece complementos da informação até atingir-se a reconstrução adequada, esse processo tem que ser feito através da estimação do ruído. O problema é que ainda não foi proposto um modelo eficiente de estimação do ruído. Assim, teoricamente o *codec* PRISM tem como premissa o conhecimento no codificador e no decodificador da informação exata do ruído inserido pelo canal na geração da informação lateral, o que do ponto de vista prático não é possível, inviabilizando uma implementação desse *codec* em dispositivos de mercado.

2.3
Arquitetura DISCOVER

A arquitetura mais recente em termos de desenvolvimento e resultados sobre DVC e que tem o melhor suporte e publicação tanto da parte documental quanto da parte computacional é a desenvolvida por um consórcio de universidades européias chamada DISCOVER, acrônimo para DIStributed COding for Video sERvices [48].

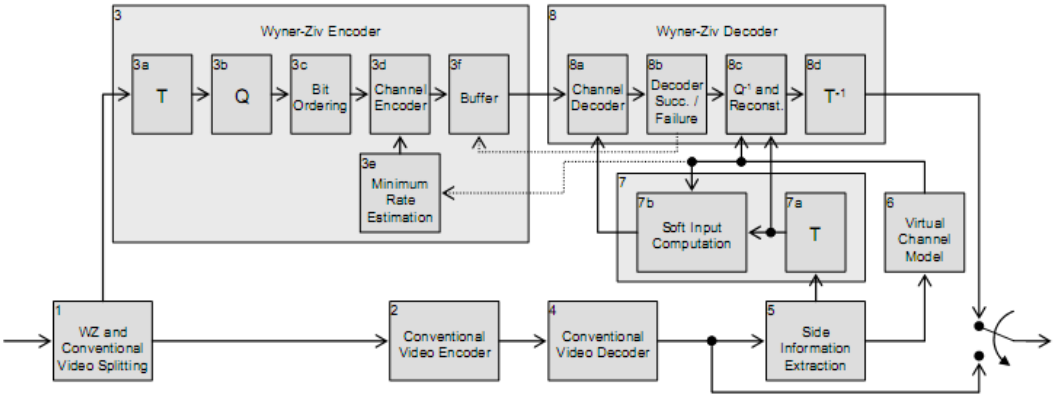

Figura 2.3 – *Codec* DVC – Arquitetura DISCOVER, consórcio de universidades européias
Fonte: Ref [48]

O esforço de estudo no desenvolvimento desta arquitetura foi coordenado pela *Universitat Politècnica de Catalunya* (Espanha) com o apoio de outras instituições, a saber, Instituto Superior Técnico (Portugal), *Ecole Polytechnique Fédérale de Lausanne* (Suíça), *Leibniz Universität Hannover* (Alemanha), *Institut National de Recherche en Informatique et en Automatique* (França) e *Università di Brescia* (Itália).

A arquitetura do DISCOVER, cujo diagrama está ilustrado na figura 2.3, é baseada no esquema proposto pelo Professor Bernd Girod da Universidade de Stanford e sua equipe [29], apresentado na seção 2.1 deste capítulo. No entanto, várias técnicas foram melhoradas ou adicionadas ao esquema de Stanford, que teve seu formato final definido em 2005. Tais estudos começaram na Europa no final de 2007 e se estenderam até o final de 2010, quando foram publicados os últimos resultados. Entre as principais melhorias das técnicas podemos citar a otimização do desempenho dos blocos básicos de construção e a solução para problemas de estimação de parâmetros em tempo de execução (on-line).

Além de algumas contribuições ao modelo de Stanford, outro grande reconhecimento que deve ser feito ao projeto DISCOVER, foi a diversidade de experimentos e resultados que foram realizados, diferentes e em maior número do que para as arquiteturas DVC anteriores. Por exemplo, foram feitos testes com uma grande variedade e disposição de GOPs (*Group of Pictures*), testes de complexidade computacional, testes com modificações de técnicas de geração de informação lateral, testes de desempenho em relação a mudanças no canal de retorno e testes comparativos entre codificadores de canais.

Assim, a forma em que foram conduzidos o desenvolvimento e os estudos do projeto DISCOVER, o torna uma bem organizada fonte de consulta tanto sobre a arquitetura de Stanford quanto sobre a própria arquitetura do *codec* DISCOVER.

2.4
Conclusões

Neste capítulo, foram apresentadas as duas arquiteturas que deram início aos estudos sobre DVC, a arquitetura de Stanford e a arquitetura PRISM de Berkeley, que até hoje são consideradas "estado da arte" e literatura obrigatória para quem inicia estudos sobre a tecnologia.

Além disso, mostramos a importante contribuição do projeto DISCOVER, implementação mais recente de DVC baseada na arquitetura de Stanford com algumas otimizações e solução de problemas. O projeto se desenvolveu do 2° semestre de 2007 até o 2° semestre de 2010 e é interessante do ponto de vista do registro e organização das informações, além de mostrar uma grande diversidade de testes e resultados.