REDUÇÃO DA LARGURA DE BANDA DE MEMÓRIA UTILIZANDO COMPRESSÃO SEM PERDAS PARA QUADROS DE REFERÊNCIA NA CODIFICAÇÃO DE VÍDEOS DIGITAIS

SILVEIRA, Dieison[1]; GRELLERT, Mateus[1]; SANCHEZ, Gustavo[1]; POSSANI, Vinicius[1]; AGOSTINI, Luciano[2]

[1]UFPel, Curso de Ciência da Computação.
Email: {dssilveira, mgsilva, gfsanchez, vnpossani}@inf.ufpel.edu.br
[2]UFPel, Centro de Desenvolvimento Tecnológico.
Email: agostini@inf.ufpel.edu.br

1 INTRODUÇÃO

A transmissão e o armazenamento de vídeos digitais tornam-se um problema sério quando a resolução do vídeo é ampliada. Assim, a codificação de vídeo tornou-se um grande campo de pesquisa, onde os algoritmos e técnicas para a compressão de dados têm sido desenvolvidos. Através dessas técnicas, vídeos digitais podem ser representados com uma quantidade muito menor de dados e com perda de qualidade mínima ou imperceptível. Tendo em vista que vídeos digitais podem ser reproduzidos em diversos tipos de aparelhos, como computadores, celulares, televisores digitais, câmeras fotográficas e filmadoras, entre outros, a compressão de dados é requerida para viabilizar o armazenamento e a transmissão destes vídeos. Os vídeos digitais possuem muitas regiões semelhantes ou idênticas e a codificação de vídeo busca explorar a redundância nestas regiões, com o objetivo de reduzir o tamanho do vídeo sem diminuir a sua qualidade.

Neste contexto, o padrão H.264/AVC surge como o mais novo padrão de codificação de vídeos (RICHARDSON, 2003), gerando um aumento significativo na taxa de compressão para uma mesma qualidade, quando comparado a padrões anteriores. Isso se deve ao fato do padrão introduzir o uso de blocos de tamanhos variáveis, predição intra-quadro realizada no domínio espacial, utilização de múltiplos quadros de referência para a predição inter-quadros, entre outros (AGOSTINI, 2007). Um aspecto importante para motivar este trabalho é que o padrão H.264/AVC foi adotado pelo Sistema Brasileiro de TV Digital.

O codificador H.264/AVC, quando implementado em hardware, utiliza quantidades expressivas de memória externa para realizar a codificação. Sendo assim, o projeto do codificador deve considerar que um dos maiores gargalos deste tipo de sistema está nesta comunicação entre a memória externa e as unidades de processamento (AGOSTINI, 2007). Isso ocorre porque os quadros codificados são armazenados em memória externa para serem utilizados como referência na codificação. A comunicação com a memória é um gargalo porque, além de ser mais lenta do que o necessário, também demanda um elevado consumo de energia.

Este trabalho apresenta uma solução capaz de reduzir, em média, até 23% a quantidade de bits de memória externa e também reduz em 23% a quantidade de acessos à memória. Essa compressão é realizada sem inserir perdas de informação e conduz a um aumento da taxa de processamento e uma redução no consumo de energia do codificador. Esta solução utiliza o algoritmo de *Huffman* como base para fazer a compressão do quadro que irá para a memória externa. A Fig. 1 apresenta uma versão simplificada do diagrama dos módulos do codificador H.264/AVC, juntamente com o módulo de *Huffman* desenvolvido neste trabalho.

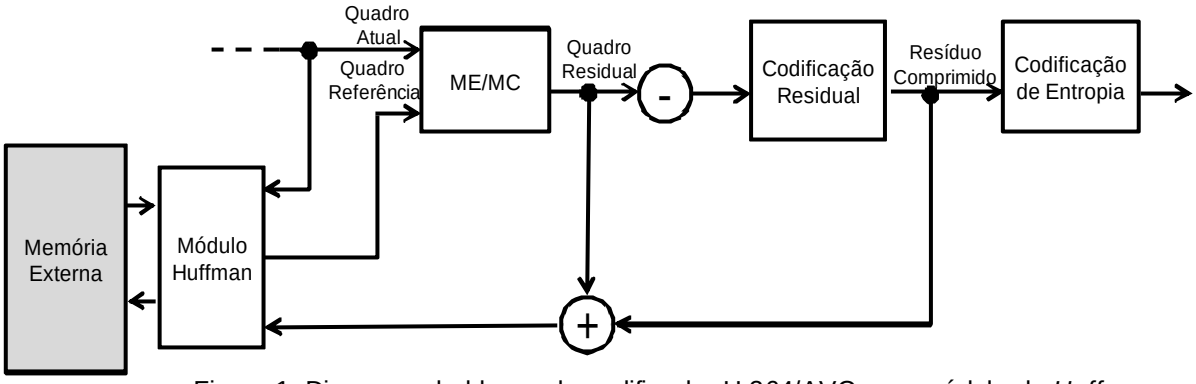

Figura 1. Diagrama de blocos do codificador H.264/AVC com módulo de *Huffman*.

2 METODOLOGIA (MATERIAIS E MÉTODOS)

O algoritmo utilizado neste trabalho é o algoritmo de *Huffman*. Este algoritmo trabalha com codificação estatística e é amplamente utilizado no núcleo de compressores de imagem e texto (SALOMON, 2004). O algoritmo proposto consiste em comprimir o quadro antes que ele seja armazenado na memória de referência e, quando o quadro for requerido, ele deve ser decodificado. Para atingir uma melhor taxa de compressão, foi utilizada uma estratégia de usar quatro tabelas estáticas no processo de codificação e decodificação do quadro.

Os valores contidos nestas tabelas foram gerados separadamente e eles correspondem à ocorrência média de cada valor no quadro, segundo a avaliação com dez vídeos. Os valores variam de 0 a 255, já que cada amostra é composta por 8 bits. Estes valores já devem estar presentes no codificador, pois seria inviável calculá-los em tempo de execução. Por isso são usadas quatro tabelas estáticas, para evitar a formação dinâmica das estatísticas de *Huffman* e também para melhorar a taxa de compressão. Assim é possível realizar uma codificação adaptativa ao contexto e, de acordo com o contexto, uma ou outra tabela é usada.

Durante o processamento de cada quadro pela estimação de movimento (*Motion Estimation* - ME) o quadro é dividido em blocos de tamanho 4x4. Este tamanho foi escolhido por ser o que atinge os melhores resultados de compressão. Para cada bloco é realizada uma verificação para definir a tabela que será usada na codificação do bloco antes deste ser armazenado na memória. A classificação é realizada da seguinte forma: (a) se a média aritmética das amostras tem valor entre 0 e 63, o bloco será codificado com a primeira tabela; (b) se o valor estiver entre 64 e 127, o bloco será codificado pela segunda tabela; (c) se a média estiver entre 128 e 191, a codificação do bloco usará a terceira tabela; e (d) se a média estiver entre 192 e 255, o bloco será codificado utilizando a quarta tabela.

Após o quadro ser processado pela ME, ele passará pelo codificador de *Huffman* e, em seguida, é enviado à memória. Quando o codificador buscar o quadro na memória para ser utilizado como referência na codificação do próximo quadro, este quadro terá que passar antes pelo decodificador de *Huffman*. A Fig. 2 apresenta o diagrama de blocos do módulo codificador/decodificador de *Huffman* e as suas respectivas tabelas. Como o quadro de referência é armazenado na memória de forma comprimida, um menor número de dados será gravado na memória. Assim, é possível reduzir a quantidade de memória necessária e, principalmente, será possível reduzir a quantidade de dados sendo transferidos do codificador para a memória e da memória para o codificador, aumentando o desempenho do codificador como um todo e reduzindo o consumo de energia.

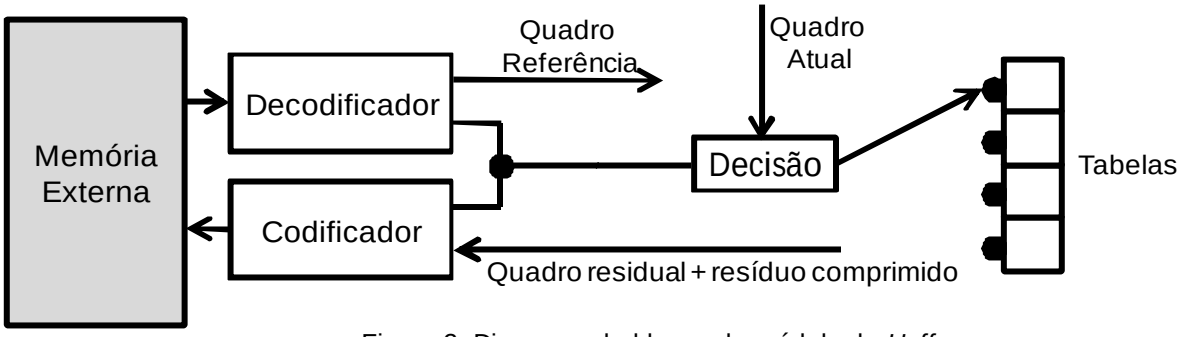

Figura 2. Diagrama de blocos do módulo de *Huffman*.

A codificação é realizada no quadro reconstruído, que é gerado através da ME. Este quadro é dividido em blocos 4x4. Neste momento, a tabela de codificação que será usada para cada bloco do quadro já foi definida. A partir da tabela selecionada é construída a árvore de *Huffman*, a árvore é construída de baixo para cima, ficando os valores de menor frequência em baixo e os com frequências maiores a cima. A codificação gera uma cadeia de bits representando cada valor. Os valores com maior frequência são representados por um número menor de bits. Após realizar a codificação, será adicionado no início de cada bloco um identificador de qual tabela foi usada nesse processo. Cada identificador de tabela é composto por dois bits. Por fim, quando os blocos já estão codificados e devidamente associados a suas respectivas tabelas de codificação, eles são enviados para a memória.

A decodificação consiste em buscar na memória de referência o quadro codificado e aplicar o processo inverso à codificação. O quadro codificado estará disposto como uma lista de bits que será identificado como uma palavra. Uma palavra se refere a uma parte da lista de bits com tamanho de 32 bits. A leitura da lista de bits é realizada de forma gulosa, i.e., vai consumindo os bits da lista, lendo uma palavra por vez. Os dois bits iniciais da primeira palavra são usados como identificador da tabela usada para codificação do bloco e que deve ser usada na decodificação. Os outros bits são processados na árvore de *Huffman* até chegar à folha da árvore. Assim é possível obter o valor original do dado codificado. Caso a sequência de bits da palavra acabar e uma folha da árvore ainda não tenha sido atingida, a próxima palavra é lida da memória e a codificação continua até que uma folha da árvore seja encontrada.

3 RESULTADOS E DISCUSSÃO

Para avaliar os ganhos obtidos com esta solução, foi desenvolvido um software utilizando a linguagem de programação C++. Os resultados individuais alcançados para cada vídeo são apresentados na Tab.1. Além dos resultados individuais, é apresentada a média aritmética destes resultados.

Tabela 1. Resultados obtidos para dez vídeos.

Vídeos	Sem compressão – nº bits	Com compressão - nº bits	Porcentagem de ganho
Sunflower	16588800	12954903	28,05
Riverbed	16588800	13210042	25,57
Rolling	16588800	13500978	22,87
Tractor	16588800	13832010	19,93
Traffic	16588800	14001055	18,48
Média	**16588800**	**13499798**	**22,98**

Os resultados apresentados na Tab.1 mostram uma variação na taxa de compressão, que é em função do conteúdo do vídeo. Alguns vídeos possuem imagens mais homogêneas, facilitando a sua codificação e, assim, atingindo um ganho mais elevado de compressão. Todos os vídeos utilizados são de alta definição, com resolução de 1920x1080 pixels (também chamado de *Full HD*). Para a codificação foram utilizados 300 quadros por vídeo. O valor médio de compressão foi de aproximadamente 23%. Isso significa que o tempo necessário pela estimação de movimento para fazer acessos à memória pode ser reduzido em 23%. Além disso, o consumo de energia para realizar os acessos à memória também será reduzido nesta taxa.

4 CONCLUSÃO

Este trabalho apresentou uma solução para a redução da largura de banda de memória em um codificador de vídeo através da compressão de quadros de referência. A codificação de *Huffman*, que é uma compressão sem perdas, foi utilizada nesta solução. O objetivo do trabalho desenvolvido foi investigar soluções de baixa complexidade computacional capazes de gerar uma redução considerável no volume de dados que será armazenado na memória externa. O foco deste trabalho foi o codificador H.264/AVC, mas a solução desenvolvida é compatível com outros padrões de compressão de vídeo.

A solução desenvolvida utiliza quatro tabelas estáticas de *Huffman* para acelerar o processo e manter os ganhos de compressão. A decisão do uso da tabela é realizada através do valor médio das amostras do quadro. Com esta solução foi possível obter uma redução média de 23% no tamanho da memória externa e também na transferência de dados entre a memória externa e o codificador. Assim, foi possível um ganho expressivo na taxa de processamento e no consumo de energia do processo de codificação.

Como trabalhos futuros, o módulo proposto será implementado utilizando uma linguagem de descrição de hardware, como VHDL. Esta descrição será sintetizada para FPGAs e seu desempenho em hardware será avaliado.

5 REFERÊNCIAS

RICHARDSON, I. **H.264/AVC and MPEG-4 Video Compression – Video Coding for Next-Generation Multimedia**. Chichester: John Wiley and Sons, 2003.

SALOMON, David. **Data Compression – The Complete Reference**. New York: Springer, 2004.

AGOSTINI, Luciano. **Desenvolvimento de Arquiteturas de Alto Desempenho Dedicadas à Compressão de Vídeo Segundo o Padrão H.264/AVC.** Agosto de 2007. Doutorado em Computação, Universidade Federal do Rio Grande do Sul. Porto Alegre, agosto de 2007.