REDUÇÃO DE COMPLEXIDADE DO ALGORITMO TEST ZONE SEARCH PARA ESTIMAÇÃO DE MOVIMENTO NO CODIFICADOR DE VÍDEO HEVC

MATEUS WACHHOLZ NORMEBERG; MARCELO SCHIAVON PORTO; GUILHERME RIBEIRO CORRÊA

Grupo de Arquiteturas e Circuitos Integrados - GACI
Universidade Federal de Pelotas – {mwnoremberg, porto, gcorrea}@inf.ufpel.edu.br

1. INTRODUÇÃO

Os vídeos digitais estão presentes nos mais diversos dispositivos. Além de estarem presentes nos televisores, os vídeos digitais estão presentes também nos *smartphones* e câmeras digitais e também na internet. Neste último, há um grande aumento na quantidade de conteúdo no formato de vídeo digital e o tráfego de dados gerado por este tipo de conteúdo é bastante grande. Estudos publicados em (CISCO, 2016a) mostram que há uma estimativa de que até 2020 cerca de 82% do tráfego de dados da internet será gerado por vídeos e os estudos publicados em (CISCO, 2016b) mostram que até 2020 cerca de 75% do tráfego gerado por dispositivos móveis corresponderá a este tipo de conteúdo.

A manipulação/transmissão de vídeos digitais só é possível devido ao processo de codificação de vídeos que é aplicado sobre estes. Especialmente em vídeos de alta definição, como os vídeos HD (1920x1080), a transmissão sem compressão é um problema, dada a grande quantidade de dados necessária para sua representação. Por isso, é necessária a utilização de estratégias que busquem remover as redundâncias de dados presentes nos vídeos digitais (RICHARDSON, 2002). Lançado em 2013, o padrão de codificação de vídeo estado-da-arte é o *High Efficiency Video Coding* (HEVC) (SULLIVAN et al., 2012).

Uma das etapas presentes nos codificadores de vídeo é a Estimação de Movimento (ME), a qual consiste em reduzir as redundâncias temporais, ou seja, visa reduzir a quantidade de dados repetidos nos diferentes quadros de um vídeo. Esta etapa é uma das que mais demanda tempo de execução em um codificador. No codificador HEVC, um dos motivos para esta grande complexidade é a utilização de 24 diferentes tamanhos de blocos na realização da ME.

A etapa de ME faz a busca pelo bloco mais similar ao bloco que está sendo codificado no quadro atual em um quadro de referência. Essa similaridade pode ser calculada através de diferentes métricas, mas a métrica mais comumente utilizada é a Soma das Diferenças Absolutas (SAD).

Existem diversos algoritmos para a realização da etapa de ME, dentre eles o algoritmo *Test Zone Search* (TZS), o qual está implementado no *software* de referência do padrão HEVC, o HEVC *Model* (HM). O algoritmo TZS consiste em, basicamente, quatro etapas: Predição, Busca Inicial, Busca Raster e Refinamento. A Predição busca apontar o melhor resultado ou então a região mais próxima ao possível melhor resultado. A etapa de Busca Inicial consiste em fazer uma busca em expansão até atingir o ponto de parada. Nesse momento, se a distância do melhor bloco candidato até o momento for maior que cinco com relação ao ponto de início da Busca Inicial, executa-se a etapa de Busca Raster (que consiste em uma busca semelhante à busca exaustiva, mas de forma sub-amostrada) e, a seguir, o Refinamento. Caso contrário, executa-se somente a etapa de Refinamento, a qual consiste em uma busca semelhante à Busca Inicial. Assim como na Busca Inicial, o Refinamento faz uma busca em expansão, mas o faz de forma iterativa, ou seja, cada iteração realiza a busca em expansão.

Este trabalho tem como objetivo propor uma solução que seja adaptativa aos diferentes tamanhos de bloco presentes no padrão HEVC, algo que ainda não foi explorado, reduzindo o custo computacional do algoritmo TZS.

2. METODOLOGIA

Em um primeiro momento foram realizadas três análises sobre o comportamento do algoritmo TZS. Para a realização destas análises foram utilizadas cinco sequências de vídeo: *BasketballDrive*, *Kimono*, *ParkScene*, *PeopleOnStreet* e *Traffic*. Também foram utilizados quatro valores de Parâmetro de Quantização (QP): 22, 27, 32 e 37.

A primeira análise visa identificar qual(is) das etapas do algoritmo TZS é(são) responsável(is) pela obtenção do melhor vetor de movimento em cada tamanho de bloco. Devido à extensão dos resultados e, também, para uma melhor visualização, este resumo apresenta apenas os resultados para uma sequência (*BasketballDrive*) e um valor de QP (32). Como pode ser visto na Figura 1, nos blocos de tamanho 64x64 até os blocos 8x4 (chamados de blocos simétricos), a maioria dos melhores vetores de movimento é resultante das etapas de Predição e Busca Inicial. Pode-se observar neste mesmo intervalo que, conforme ocorre a diminuição do tamanho do bloco, há um aumento na quantidade de vetores de movimento resultantes das duas primeiras etapas do algoritmo TZS. Já para os blocos de tamanho 32x24 até os de tamanho 4x16 (chamados de assimétricos) a maioria dos vetores de movimento resultam das etapas de Busca Raster e Refinamento. Em geral, os resultados para as demais sequências e valores de QP se mantiveram os mesmos. Em alguns casos, a quantidade de vetores de movimento resultantes das duas primeiras etapas fica próximo a 100%.

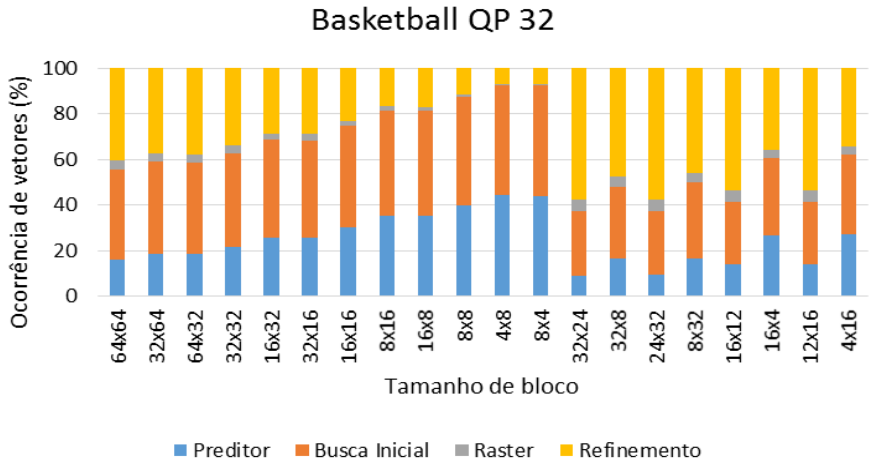

Figura 1. Etapa responsável pelo melhor vetor de movimento.

Na segunda análise foi avaliada a quantidade de tempo que cada uma das etapas do algoritmo TZS demanda com relação ao tempo total do algoritmo. À etapa de Busca Inicial foi adicionado o tempo da etapa de Predição, visto que o tempo demandado pela última é muito pequeno.

Os resultados obtidos nesta análise são apresentados na Figura 2, considerando a sequência *BasketballDrive*. Como pode ser visto, a etapa que mais demanda tempo de execução é a etapa de Busca Raster, mesmo essa resultando em poucos melhores vetores de movimento, como visto anteriormente. Quando somados os tempos das duas últimas etapas estes são responsáveis por mais da metade do tempo consumido pelo algoritmo TZS, isso para todas as sequências e valores de QP.

A terceira análise levou em conta os melhores valores de SAD quando os melhores vetores de movimento são gerados até a etapa de Busca Inicial e quando eles são gerados pelas duas últimas etapas do algoritmo TZS. Para tal, foram considerados todos os tamanhos de bloco. Neste resumo, são apresentados apenas os resultados obtidos para um tamanho de bloco (32x32) e um valor de QP (32) para a sequência *BasketballDrive*.

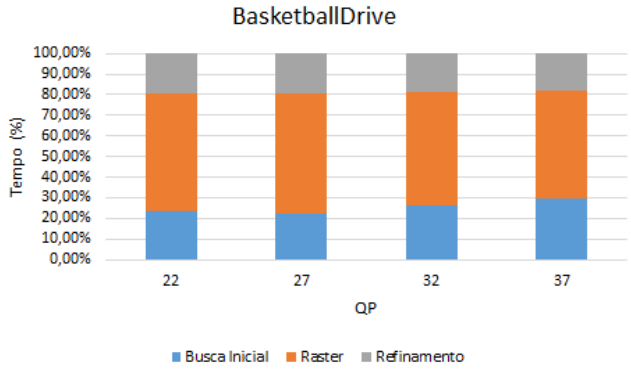

Figura 2. Tempo demandado por cada etapa do algoritmo TZS.

Os resultados obtidos na terceira análise são apresentados na Figura 3. Considerando por exemplo o intervalo de valores de SAD 2430-2440, pode-se ver que os melhores vetores de movimento resultantes com este valor de SAD são gerados nas duas primeiras etapas do algoritmo TZS (representado pela cor azul). Já no intervalo de valores de SAD 3770-3780, a maioria de melhores vetores de movimento são gerados pelas últimas duas etapas do TZS.

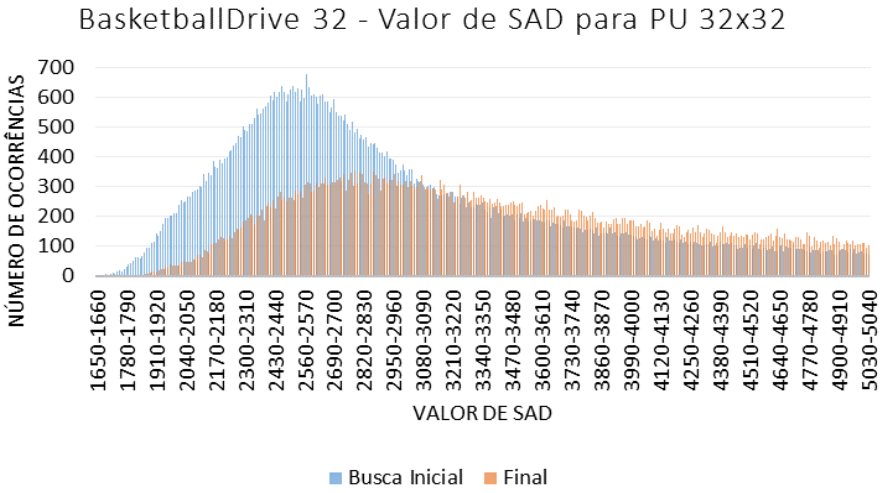

Figura 3. Melhores valores de SAD após Busca Inicial e ao final do TZS.

A solução proposta considera os valores de SAD que representam o ponto onde há a inversão da predominância de vetores de movimento resultantes das duas primeiras etapas do TZS para a predominância de vetores de movimento resultando das duas últimas etapas do TZS. Foram considerados apenas os blocos de tamanho simétrico na solução proposta. Dessa forma, considerando as cinco sequências e os quatro valores de QP, foram coletados 160 valores de SAD que representam pontos de inversão de predominância, os quais foram utilizados para o treinamento de um modelo que permite a geração automática do limiar. Para a realização do treinamento foi utilizado o *software* WEKA, utilizando o método de regressão linear. A Equação 1 apresenta a equação resultante do treinamento, o qual levou em consideração o tamanho de bloco, valor de QP e o valor da inversão de predominância, mencionado anteriormente.

$$Limiar = 71{,}1675 * QP + 2{,}438 * Tamanho - 1629{,}4271 \quad (1)$$

Ao algoritmo TZS original foi aplicada, para os blocos simétricos, a parada de execução levando em consideração o valor de Limiar obtido pela aplicação da Equação 1 para cada tamanho de bloco.

3. RESULTADOS E DISCUSSÃO

Para a obtenção dos resultados apresentados a seguir, foram utilizadas quatro sequências diferentes das utilizadas para o treinamento. São elas: *NebutaFestival*, *SteamLocomotiveTrain*, *BQTerrace* e *Cactus*. Também foram utilizados os quatro valores de QP: 22, 27, 32 e 37.

A Tabela 1 apresenta os resultados obtidos para a aplicação do método proposto. Como pode ser visto, o método proposto permite uma redução média de 2,47% no tempo total do codificador e uma redução média de 17,34% no tempo total do algoritmo TZS. Além disso, os resultados mostram que as variações de BD-rate (variação na taxa de bits ao manter-se uma mesma qualidade visual), para todas as sequências se mantiveram praticamente nulas.

Tabela 1. Resultados obtidos com o método proposto.

Sequência	**BD-rate (%)**	**Tempo Total (%)**	**Tempo TZS (%)**
BQTerrace	-0,04	-3,20	-25,81
Cactus	-0,04	-2,69	-14,02
NebutaFestival	0,04	-1,03	-9,07
SteamLocomotive	-0,59	-2,97	-20,46
Média	**-0,16**	**-2,47**	**-17,34**

Finalmente, foram testados deslocamentos no valor de Limiar, implementados como multiplicações desses valores por um fator. Estes experimentos foram realizados com o intuito de avaliar a aplicabilidade da solução em um controlador de complexidade futuro. Os fatores utilizados foram 1,1, 1,2 e 1,5. Os resultados obtidos para tais aplicações mostraram que conforme ocorre o aumento no valor de limiar há uma maior redução dos tempos total do codificador e do algoritmo TZS. Além disso, para os deslocamentos mencionados, a variação em BD-rate se manteve ainda próxima a 0%.

4. CONCLUSÕES

Neste trabalho, foi apresentada uma solução que reduz o tempo de execução do algoritmo TZS em 17,34% e, por conseguinte, de codificadores que o utilizam, sem impactos significativos em termos de eficiência de compressão. Foi apresentada também uma análise prévia do método considerando uma futura implementação de um controlador de complexidade.

5. REFERÊNCIAS BIBLIOGRÁFICAS

CISCO, Global mobile data traffic forecast update, 2015 – 2020. **Cisco visual networking index**, 2016a.

CISCO, Forecast and methodology, 2015 – 2020. **Cisco visual networking index**, 2016b.

RICHARDSON, I.E. **Video Codec Design: Developing Image and Video Compression Systems.** Chichester: John Wiley & Sons ltd., 2002. 1ed.

SULLIVAN, G. J.; OHM, J.-R.; HAN, W.-J.; WIEGAND, T. Overview of the High Efficiency Video Coding (HEVC) Standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v.22, n.12, p.1649 –1668, 2012.