Algoritmo Rápido de Estimação de Movimento Otimizado em Taxa-distorção para Codificação de Vídeo com Baixa Taxa de Bits

Marcos Moecke e Rui Seara

***Resumo*—O processo de estimação de movimento (EM) tem sido responsável por uma parte significativa do esforço computacional em codificadores de vídeo otimizados em taxa-distorção (R-D). Este trabalho propõe um algoritmo rápido de EM que se ajusta ao perfil típico de distribuição dos vetores de movimento, obtidos segundo a condição de otimização R-D. Tal algoritmo é comparado com algoritmos rápidos da literatura e seus resultados mostram um desempenho R-D superior para condições de operação equivalentes.**

***Palavras-Chave*—Algoritmos rápidos, codificação de vídeo, complexidade computacional, estimação de movimento, otimização em taxa-distorção.**

***Abstract*—In rate-distortion (R-D) optimized video coding the motion estimation (ME) process has been responsible for a large fraction of the computational effort. This paper proposes a fast ME algorithm that fits to the typical distribution profile of the motion vectors, obtained according to the R-D optimization condition. Such an algorithm is compared with fast algorithms of the literature and its results show a better R-D performance for the same operation conditions.**

***Keywords*—Fast algorithm, video coding, computational effort, motion estimation, rate-distortion optimization.**

I. INTRODUÇÃO

Algoritmos rápidos (ARs) de estimação de movimento (EM) que restringem o número de pontos de procura (NPP) são bastante utilizados em implementação de codificadores de vídeo. Algoritmos rápidos, como o TSS [1], NTSS [2] e o DSS [3] têm sido usados visando reduzir a complexidade computacional sem comprometer a qualidade, quando comparados com o algoritmo de busca exaustiva (*full search* - FS). Esses ARs, normalmente, são usados e avaliados apenas na EM otimizada em distorção [1]-[3]. Neste artigo, os ARs citados são integrados ao codificador JM7.3, que é otimizado em taxa-distorção (R-D) [4], e avaliados tanto em desempenho R-D quanto em complexidade computacional. Para a otimização R-D, a função custo Lagrangiano é usada devido ao seu excelente desempenho quando aplicada ao algoritmo FS [5].

Uma das principais causas de degradação de desempenho nos ARs citados é a estratégia de avaliação parcial (não-exaustiva) empregada na obtenção dos vetores de movimento (VMs). Os vetores obtidos estão limitados por posições predefinidas tanto no gabarito de procura (GP), usado na fase inicial, quanto naquele usado nas demais fases do processo de EM. Assim, nos ARs, existe uma forte tendência de se obterem VMs relacionados quase sempre a mínimos locais da função de otimização considerada. Apesar disso, em função da sua baixa complexidade, tais algoritmos são indicados para aplicações que requeiram limitada capacidade de processamento.

Neste artigo, é apresentada uma proposta de algoritmo rápido (AR) que se ajusta ao perfil típico de distribuição dos VMs, obtido com o codificador JM7.3 (baseado na recomendação H.264 [6]) otimizado em R-D. O AR proposto tem o gabarito de procura inicial (GPI) e os demais GPs usados na EM assemelhados a uma forma de diamante. Além disso, o tamanho dos GPs é definido e modificado em escala logarítmica. Por isso, esse algoritmo é denominado: procura logarítmica em formato de diamante (*logarithmic diamond shape search* – LDSS), com o objetivo de diferenciá-lo do conhecido algoritmo DSS que tem seu GP de tamanho fixo. A grande vantagem do algoritmo LDSS, em relação a seus concorrentes, é a sua menor susceptibilidade a mínimos locais, visto que, através do uso de seu GPI, a verificação de posições de baixa probabilidade de ocorrência é evitada.

II. HISTOGRAMA DOS VETORES DE MOVIMENTO

Para se obter o perfil típico de distribuição (histograma bidimensional) dos VMs, foi considerada a média dos 100 primeiros quadros de seis seqüências clássicas de vídeo conferência, a saber: *Akiyo*, *News*, *Mother&Daughter*, *Silent*, *Foreman* e *Carphone*, todas em formato QCIF. Tais vetores foram determinados através do modelo de teste JM7.3, utilizando-se como critério de otimização tanto a distorção quanto a taxa-distorção. Os resultados obtidos são mostrados na Fig. 1. Nessa figura, uma escala logarítmica tem sido considerada para efeito de visualização das baixas freqüências de ocorrência de VMs. Para ambas as distribuições [Fig. 1(a) e (b)], observa-se uma forte concentração de VMs em torno

Marcos Moecke, Centro Federal de Educação Tecnológica de Santa Catarina - CEFET/SC, Curso de Telecomunicações, São José, SC. E-mail: moecke@sj.cefetsc.edu.br.

Rui Seara, LINSE - Laboratório de Circuitos e Processamento de Sinais, Departamento de Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, SC, E-mail: seara@linse.ufsc.br.

Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

do centro da janela de procura. Essa concentração pode ser atribuída aos seguintes fatores:

i) ausência de movimento em muitos blocos em relação aos quadros anteriores;

ii) movimento conjunto de blocos, em objetos visuais de grande extensão, que são preditos adequadamente a partir de VMs de blocos vizinhos causais.

Avaliando-se o histograma [Fig. 1(a)], obtido com a otimização da distorção, nota-se que a distribuição dos VMs é aproximadamente uma Laplaciana simétrica circular [7]. No entanto, quando é considerada a otimização R-D, ocorre uma concentração de VMs ao longo dos eixos *h* e *v* [ver Fig. 1(b)]. Tal concentração decorre da codificação independente dos componentes *h* e *v* dos VMs como também do emprego do critério custo de codificação na seleção dos VMs.

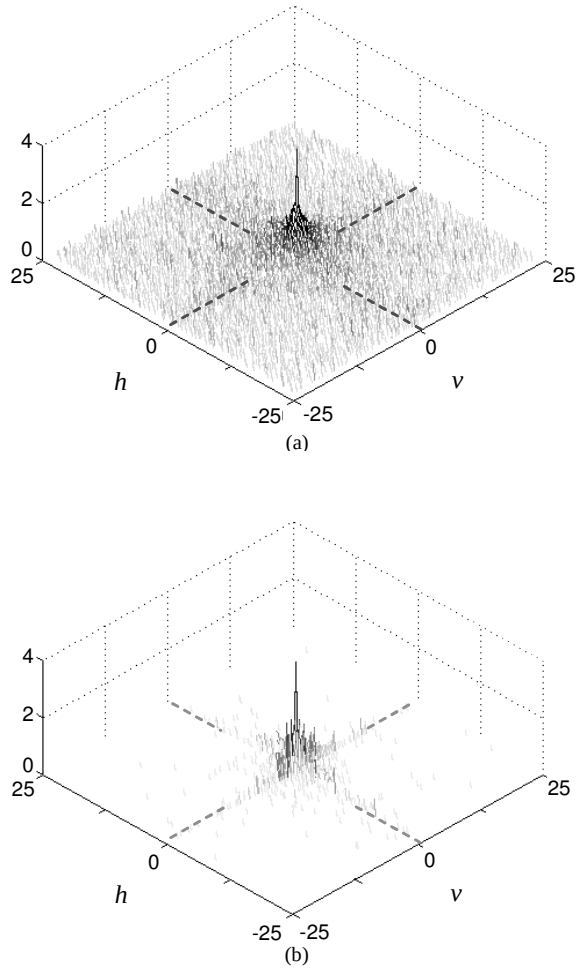


Fig. 1. Histograma bidimensional médio dos VMs das seqüências de vídeo selecionadas, obtidos através do codificador JM7.3 para baixas taxa de bits com $Q = 45$. (a) Otimizado em distorção. (b) Otimizado em taxa-distorção.

III. Algoritmos Rápidos de Estimação de Movimento

A maioria dos algoritmos rápidos que restringem o número de pontos de procura tem por base as seguintes premissas: (a) monotonicidade da função distorção; (b) distribuição centralizada dos seus mínimos [1]-[3]. Tais premissas podem ser consideradas válidas em seqüências de vídeo que empreguem uma câmera fixa e têm poucos objetos em movimento como, por exemplo, em aplicação de vídeo conferência e monitoração de ambientes. Nessas condições, os VMs obtidos por esses ARs apresentam um desempenho R-D similar ao obtido com o algoritmo FS. No entanto, em casos nos quais essas premissas não são válidas, podem ser selecionados VMs afastados do mínimo global, degradando significativamente o desempenho dos ARs. Assim, no que segue, discutiremos algumas estratégias de ARs, visando avaliar a influência da escolha do GP na complexidade e desempenho desses algoritmos quando aplicados em codificadores otimizados R-D.

A. Estratégias TSS e NTSS

A estratégia TSS (*three step search*) [1] é freqüentemente usada para codificadores otimizados em distorção devido à sua simplicidade e seu bom desempenho. A busca é efetuada em vários passos, usando-se um GP com formato quadrado. No primeiro passo, são avaliados o centro e as posições do GPI (ver Fig. 2). No passo seguinte, o centro do GP é deslocado para a posição de menor distorção e o GP tem seu tamanho reduzido à metade. O processo de EM pára quando a posição selecionada é o centro do GP e o seu tamanho é de 1 *pixel*.

A estratégia NTSS (*new three step search*) [2] é um aprimoramento do algoritmo TSS. No NTSS, as 8 posições vizinhas ao centro da janela de procura são agregadas ao GPI e uma regra de parada é considerada, favorecendo os VMs próximos ao centro do gabarito.

B. Estratégia DSS

A DSS (*diamond shape search*) [3] é a estratégia adotada nos modelos de teste de baixa complexidade do H.263 a partir do TMN-8 [8]. Ela emprega um GP em formato de diamante com tamanho unitário (ver Fig. 2). Esse GP é deslocado passo-a-passo até que a posição selecionada seja o próprio centro do gabarito. Essa estratégia apresenta um desempenho inferior tanto ao TSS quanto ao NTSS em cenas com movimentos mais amplos (ver Fig. 7).

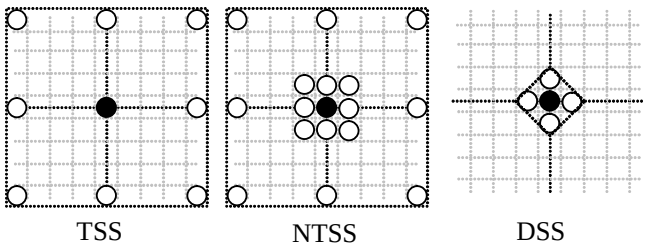


Fig. 2. Gabarito de procura inicial das estratégias TSS, NTSS e DSS para janela de procura de ±8 *pixels*. Os círculos indicam as posições avaliadas no primeiro passo de procura. Os pontos pretos denotam o centro da janela, que normalmente corresponde ao VM predito.

C. Otimização em Taxa-distorção dos Algoritmos Rápidos

Os ARs discutidos anteriormente foram integrados ao codificador de referência JM7.3 [4], visando avaliar seu desempenho em um processo de EM otimizado em R-D. Para alargar a janela de procura para ±16 *pixels*, foi necessário acrescentar um quarto passo de procura aos algoritmos TSS e NTSS. Após a seleção do VM ótimo com resolução de 1 *pixel*, é realizado um refinamento de ½ e ¼ de *pixel* [6]. Além disso, ainda são consideradas as seguintes condições de simulação: (a) a estrutura de quadros usada é *IPPP...P*; (b) para a memória de longo prazo [6] são usados, como referência na EM, os 5 quadros anteriores ao atual; (c) são habilitados todos os tamanhos de bloco e tipos de predição dos modos *Inter* e *Intra*, bem como o modo *Skip*; (d) a taxa de bits é controlada pelo parâmetro de quantização Q, variando-se o seu valor de 24 a 40; (e) são usados os 100 primeiros quadros das seqüências: *Akiyo*, *News*, *Silent*, *Carphone*, *Mother&Daughter*, *Foreman*, *Paris* e *Mobile&Calendar*, todas amostradas com taxa de quadros de 10 fps, sendo as 6 primeiras no formato QCIF e as 2 últimas, no formato CIF.

A estimativa da complexidade computacional é obtida através da determinação do número de pontos de procura (NPP), o qual representa a média do número de cálculos de distorção por bloco.

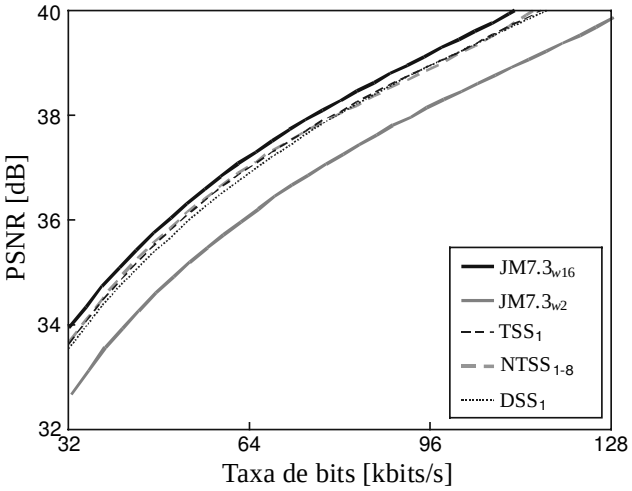


Fig. 3. Curvas R-D para a seqüência *Foreman* QCIF-10 Hz, usando os algoritmos FS (JM7.3$_{w16}$ e JM7.3$_{w2}$) e os algoritmos rápidos TSS, NTSS e DSS.

A Fig. 3 mostra as curvas R-D do algoritmo FS com janela de procura ±16 *pixels* (JM7.3$_{w16}$) e com janela de ±2 *pixels* (JM7.3$_{w2}$) obtidas para a seqüência *Foreman*. Essa figura também mostra o desempenho R-D dos ARs considerados. Nota-se que todos os ARs perdem em desempenho R-D para o JM7.3$_{w16}$; porém, apresentam uma significativa redução de complexidade computacional (95%), conforme indicado na Fig. 6(b). No entanto, tais algoritmos mostram um desempenho significativamente superior quando comparados com o JM7.3$_{w2}$, que possui uma complexidade similar. Para todas as seqüências avaliadas, os ARs que apresentaram o melhor desempenho R-D foram o NTSS e o TSS; por outro lado, o DSS sempre teve o pior desempenho. Tal perda de desempenho está fortemente relacionada ao tamanho do GP usado pelo DSS, o qual falha na detecção de movimentos amplos entre blocos vizinhos. Tal falha pode ser observada entre os quadros #170 e #230 da seqüência *Foreman* (ver Fig. 7).

IV. LDSS: Algoritmo Rápido Otimizado em R-D

Analisando-se os ARs anteriores em termos de ajuste de seu GPI ao perfil típico de distribuição dos VMs, observa-se que eles estão ajustados à distribuição (otimizada em distorção) mostrada na Fig. 1(a), porém não àquela (otimizada em R-D) apresentada na Fig. 1(b). Deve-se notar que o GPI com formato de diamante (usado no DSS) se ajusta melhor do que o de formato quadrado (usado no TSS e NTSS). Nesse último, existem posições situadas fora dos eixos para as quais a probabilidade de seleção é muito baixa, levando a uma perda considerável de recurso computacional. Para superar tal deficiência, é proposto neste trabalho o algoritmo LDSS, que utiliza diferentes GPIs ajustados ao perfil típico de distribuição dos VMs nos codificadores otimizados em R-D. O algoritmo proposto adianta (antecipa) a avaliação dos VMs de menor custo (com maior probabilidade de ocorrência), posicionando o GPI sobre os eixos.

A. Gabarito de Procura Inicial

O algoritmo LDSS prevê o uso de um conjunto diversificado de GPIs, todos eles com posições localizadas sobre os eixos. As posições são indicadas por índices descritos por 2^{α}, para $\alpha = 0, 1, 2, \ldots$. Esses índices representam a distância das posições até o centro do GPI.

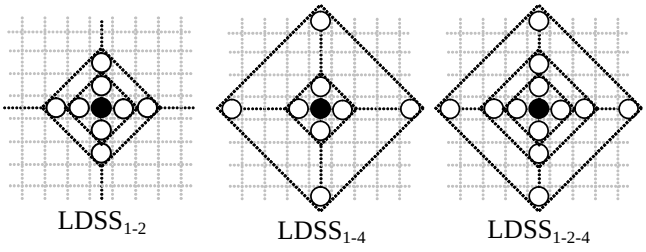


Fig. 4. Exemplos de gabarito de procura inicial.

No LDSS, o uso de diferentes GPIs possibilita uma adaptação da estratégia de EM ao conteúdo da cena e recurso computacional disponível. Em regiões da imagem com baixa quantidade de movimento, pode-se adotar um GPI com índices de posição baixos (exemplos: LDSS$_1$ e LDSS$_{1\text{-}2}$); enquanto que, em regiões com maior quantidade de movimento, pode-se usar um GPI com maiores índices (exemplo: LDSS$_{1\text{-}8}$). A predição da quantidade de movimento pode ser feita a partir dos quadros e blocos anteriores. De acordo com a janela de procura, pode-se também incluir no GPI índices 16, 32 e 64, ampliando sua abrangência e evitando assim que o algoritmo seja logrado por mínimos locais.

B. Critério de Seleção dos Vetores de Movimento

Para cada tamanho de bloco, os VMs são determinados através da minimização da função custo Lagrangiano, definida por $J = D + \lambda R$, onde R caracteriza a taxa de bits (considerando-se apenas os bits usados na codificação do VM) e D é a distorção medida entre os *pixels* do bloco codificado (compensado em movimento) e os correspondentes *pixels* do bloco original.

C. Fases da Estimação de Movimento

Na fase inicial da EM, são avaliadas todas as posições do GPI. Se a posição central da janela de procura é selecionada nessa fase e o GPI inclui o índice 1, então o processo de EM com resolução de 1 *pixel* é finalizado. Caso contrário, a posição selecionada é usada como centro do GP, cujo tamanho é determinado pelo distância d da posição selecionada ao centro da janela de procura. O GP com formato de diamante é denotado, neste texto, por GP_n, onde o índice n é dado por $n = \log_2(d)$, indicando o número de reduções sofridas pelo gabarito até a finalização do processo de procura. A Fig. 5 ilustra alguns exemplos de GP considerados no LDSS.

Na segunda fase, as posições definidas pelo GP são avaliadas. Se uma das posições do GP for selecionada, então o gabarito é deslocado para essa nova posição e a procura é repetida, mantendo-se o mesmo tamanho de gabarito. Quando a posição central do gabarito é selecionada, o GP é reduzido à metade, fazendo-se $n = n - 1$, e uma nova fase de procura é então iniciada. Dessa forma, o processo de EM prossegue até que o índice do GP seja zero e a posição central do GP, selecionada, conforme mostrado na Fig. 5.

Concluída a EM com resolução de 1 *pixel*, aplica-se a mesma forma de refinamento (½ e ¼ de *pixel*), usada nos outros ARs.

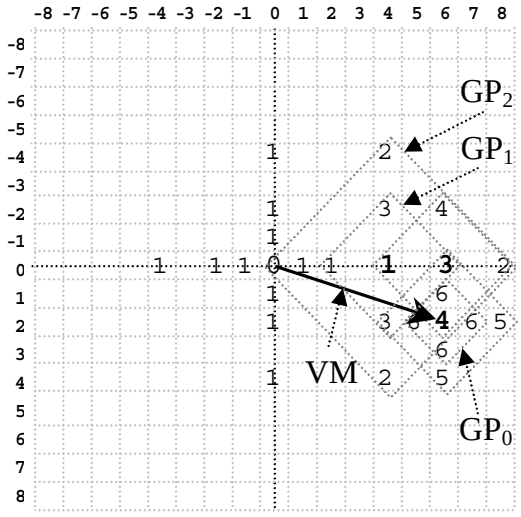
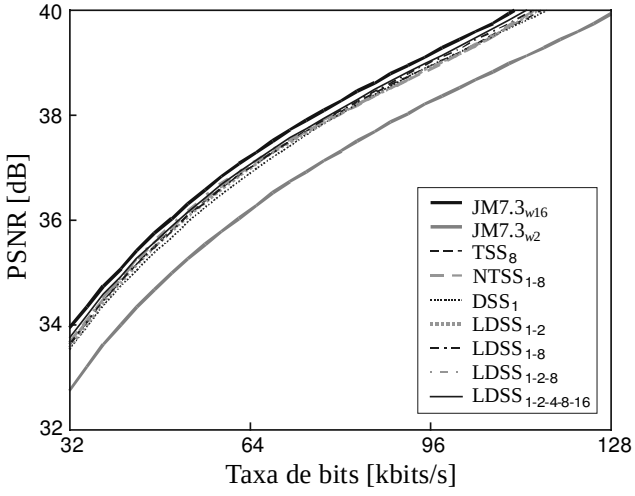


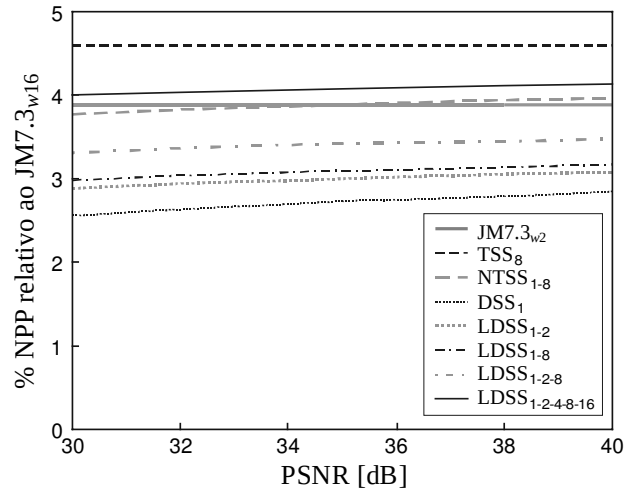
Fig. 5. Ilustração de posições com resolução 1 *pixel*, avaliadas pelo algoritmo $LDSS_{1\text{-}2\text{-}4}$ para uma janela de procura de ±16 *pixels*. Os números no interior do diagrama indicam as posições avaliadas a cada passo de procura. Em negrito, estão as posições selecionadas como centro do GP a cada passo.

V. RESULTADOS

São realizadas simulações empregando a estratégia LDSS sob as mesmas condições de avaliação e seqüências de vídeos consideradas na avaliação dos demais algoritmos rápidos da Seção III. Diferentes GPIs são usados para avaliar sua influência tanto em desempenho quanto em complexidade. O algoritmo LDSS apresentou um desempenho R-D superior ao da estratégia DSS e equivalente aos dos demais algoritmos rápidos aqui considerados. Para as seqüências de vídeo *Akiyo*, *News*, *Mother&Daughter*, *Silent* e *Paris*, a perda de desempenho R-D observada, relativa ao $JM7.3_{w16}$, é insignificante. A Fig. 6(a) mostra os resultados obtidos para a seqüência *Foreman*, na qual nota-se um melhor desempenho (em termos de PSNR) do algoritmo LDSS com respeito ao DSS, para toda a faixa de taxa de bits considerada.



(a)



(b)

Fig. 6. Desempenho R-D e complexidade computacional do algoritmos FS ($JM7.3_{w16}$ e $JM7.3_{w2}$), TSS, NTSS, DSS e LDSS com janela de procura de ±16 *pixels*, para a seqüência *Foreman*. (a) Curvas R-D. (b) Percentual de NPP relativo ao $JM7.3_{w16}$.

A complexidade computacional da estratégia LDSS está relacionada diretamente com o GPI usado, conforme mostrado na Fig. 6(b). Sua complexidade pode se ajustar entre a do algoritmo TSS (máxima) e a do DSS (mínima). Dentre os GPIs de menor complexidade, o melhor desempenho R-D é obtido com o $LDSS_{1\text{-}8}$, o qual é o melhor adaptado a cenas com movimentos amplos.

A avaliação quadro-a-quadro da seqüência *Foreman* (ver Fig. 7) mostra que o algoritmo $LDSS_{1\text{-}8}$ é aquele que possui a menor perda de desempenho R-D para todos os quadros da seqüência considerada. Além disso, nas proximidades do quadro #207, o algoritmo $LDSS_{1\text{-}8}$ apresenta um ganho de desempenho, com respeito ao DSS, em torno $1,23\,\text{dB}$.

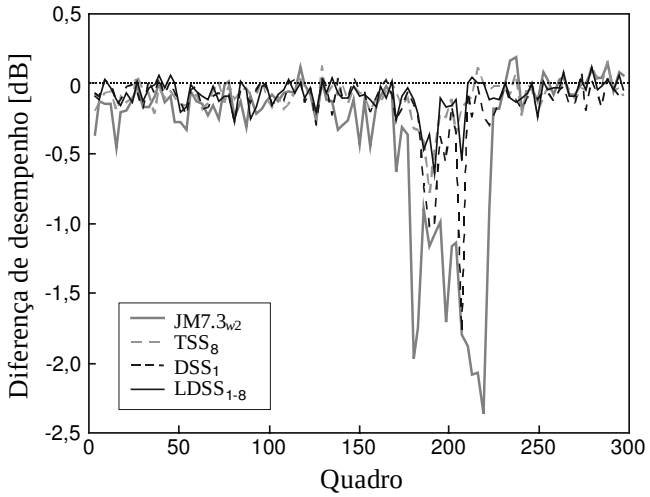


Fig. 7. Variação de desempenho R-D quadro-a-quadro, relativo ao $JM7.3_{w16}$, para os algoritmos $JM7.3_{w2}$, TSS, DSS e $LDSS_{1\text{-}8}$.

VI. CONCLUSÕES

Para codificadores de vídeo otimizados em taxa-distorção, sob condições de baixa atividade de movimento, os algoritmos rápidos mostraram um desempenho de qualidade próximo aquele obtido com o algoritmo FS. Além disso, os ARs proporcionam uma significativa redução de complexidade computacional em relação ao FS, podendo atingir mais de 97% de redução de complexidade para a mesma janela de procura. Em seqüências nas quais existem uma maior quantidade de movimento, o algoritmo DSS apresentou uma perda considerável de desempenho. Por outro lado, nas mesmas condições, o algoritmo LDSS proporcionou, para a mesma complexidade de procura, um desempenho de qualidade superior aos demais ARs, mostrando que a estratégia de ajuste do GPI ao perfil típico de distribuição dos VMs é apropriada.

REFERÊNCIAS

[1] T. Koga, K. Inuma, A. Hirano et al., "Motion compensated interframe coding for video conferencing," *Proc. Nat'l Telecomm. Conf.*, New Orleans, Nov. 1981, pp. G5.3.1-5.3.5.

[2] R. Li, B. Zeng, and M. L. Liou, "A new three step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438-442, Aug. 1994.

[3] M. Gallant, G. Cote, and F. Kossentini, "An efficient computation-constrained block-based motion estimation algorithm for low bit rate video coding," *IEEE Trans. Image Processing*, vol. 8, no. 12, pp. 1816-1823, Dec. 1999.

[4] JVT JM73 implementation. Disponível para *download* em http:/bs.hhi.de/~suehring/tml/download/jm73.zip.

[5] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 688-703, July 2003.

[6] ITU-T Recommendation H.264, "*Advanced video coding for generic audiovisual services*," Int'l Telecommunications Union, Geneva, Switzerland, 2003.

[7] B. Zeng, R. Li, and M. L. Liou, "Optimization of fast block motion estimation algorithms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 6, pp. 833-844, Dec. 1997.

[8] ITU-T, *Video Codec Test Model, Near-Term, Version 8 (TMN8)*, Int'l Telecommunications Union, Portland, USA, June 1997.