República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(21) BR 112020006232-4 A2

(22) Data do Depósito: 02/10/2018

(43) Data da Publicação Nacional: 13/10/2020

(54) Título: CODIFICAÇÃO DE INFORMAÇÃO DE MOVIMENTO DE PREDIÇÃO AFIM PARA CODIFICAÇÃO DE VÍDEO

(51) Int. Cl.: H04N 19/105; H04N 19/52; H04N 19/176; H04N 19/137; H04N 19/54.

(30) Prioridade Unionista: 01/10/2018 US 16/148,738; 03/10/2017 US 62/567,598.

(71) Depositante(es): QUALCOMM INCORPORATED.

(72) Inventor(es): KAI ZHANG; JIANLE CHEN; XIANG LI; WEI-JUNG CHIEN; YI-WEN CHEN; LI ZHANG; MARTA KARCZEWICZ.

(86) Pedido PCT: PCT US2018053936 de 02/10/2018

(87) Publicação PCT: WO 2019/070683 de 11/04/2019

(85) Data da Fase Nacional: 27/03/2020

(57) Resumo: Um dispositivo de exemplo para codificar dados de vídeo inclui uma memória configurada para armazenar dados de vídeo e um ou mais processadores implementados em conjuntos de circuitos e configurados para codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento, predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual e codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento. Predizer a segunda MVD a partir da primeira MVD pode reduzir a taxa de bits de um fluxo de bits, incluindo dados de vídeo codificados, além de melhorar a eficiência do processamento.

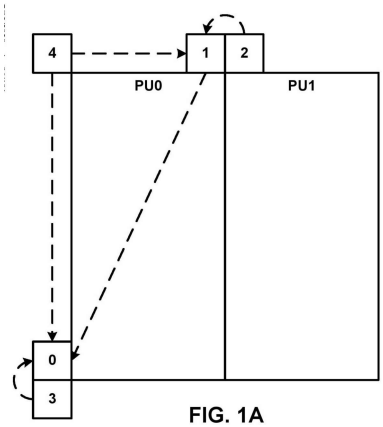


FIG. 1A

"CODIFICAÇÃO DE INFORMAÇÃO DE MOVIMENTO DE PREDIÇÃO AFIM PARA CODIFICAÇÃO DE VÍDEO"

[0001] Este pedido reivindica o benefício do Pedido de Patente Provisório Norte-Americano N° 62/567,598, depositado em 3 de Outubro de 2017 e o Pedido de Patente Norte-Americano N° 16/148,738, depositado em 1 de Outubro de 2018, cujos conteúdos totais de cada um são incorporados neste documento por referência.

CAMPO TÉCNICO

[0002] Esta descrição refere-se à codificação de vídeo e, mais particularmente, à codificação de informação de movimento de dados de vídeo.

FUNDAMENTOS

[0003] Os recursos de vídeo digital podem ser incorporados a uma ampla variedade de dispositivos, incluindo televisões digitais, sistemas de difusão direta digital, sistemas de difusão sem fio, assistentes digitais pessoais (PDAs), laptops ou computadores desktop, tablets, leitores de e-book, câmeras digitais, dispositivos de gravação digitais, reprodutores de mídia digital, dispositivos de videogame, consoles de videogame, telefones de rádio por satélite ou celular, os assim chamados "telefones inteligentes", dispositivos de teleconferência de vídeo, dispositivos de streaming de vídeo e similares. Os dispositivos de vídeo digitais implementam técnicas de codificação de vídeo, tais como as descritas nos padrões definidos por MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, parte 10, Codificação de Vídeo Avançada (AVC), o padrão de Codificação de Vídeo de Alta Eficiência (HEVC), ITU-T H.265/codificação de Vídeo de Alta Eficiência

(HEVC) e extensões desses padrões. Os dispositivos de vídeo podem transmitir, receber, codificar, decodificar e/ou armazenar informações de vídeo digital com mais eficiência, implementando essas técnicas de codificação de vídeo.

[0004] As técnicas de codificação de vídeo incluem predição espacial (intra-imagem) e/ou predição temporal (inter-imagem) para reduzir ou remover a redundância inerente às sequências de vídeo. Para codificação de vídeo baseada em bloco, uma fatia (slice) de vídeo (por exemplo, uma imagem de vídeo ou uma parte de uma imagem de vídeo) pode ser particionada em blocos de vídeo, que podem ser referidos também como unidades de árvore de codificação (CTUs), unidades de codificação (CUs) e/ou nós de codificação. Os blocos de vídeo em uma fatia intra-codificada (I) de uma imagem são codificados utilizando predição espacial em relação às amostras de referência em blocos vizinhos na mesma imagem. Os blocos de vídeo em uma fatia inter-codificada (P ou B) de uma imagem podem utilizar predição espacial em relação a amostras de referência em blocos vizinhos na mesma imagem ou predição temporal em relação às amostras de referência em outras imagens de referência. As imagens podem ser referidas como quadros, e as imagens de referência podem ser referidas como quadros de referência.

[0005] Predição espacial ou temporal resulta em um bloco preditivo para um bloco a ser codificado. Os dados residuais representam diferenças de pixel entre o bloco original a ser codificado e o bloco preditivo. Um bloco inter-codificado é codificado de acordo com um vetor de movimento que aponta para um bloco de amostras de

referência formando o bloco preditivo e os dados residuais indicando a diferença entre o bloco codificado e o bloco preditivo. Um bloco intra-codificado é codificado de acordo com um modo intra-codificado e os dados residuais. Para maior compressão, os dados residuais podem ser transformados a partir do domínio de pixel para um domínio de transformada, resultando em coeficientes de transformada residuais, que podem ser quantizados. Os coeficientes de transformada quantizados, inicialmente dispostos em um arranjo bidimensional, podem ser varridos para produzir um vetor unidimensional de coeficientes de transformada, e a codificação por entropia pode ser aplicada para obter ainda mais compressão.

SUMÁRIO

[0006] Em geral, esta descrição apresenta técnicas relacionadas à predição inter-imagens. Por exemplo, as técnicas desta descrição incluem codificar o vetor de movimento (codificação e/ou decodificação) para compensação de movimento afim na codificação de vídeo baseada em bloco. Essas técnicas podem ser aplicadas aos padrões de codificação de vídeo existentes e/ou futuros.

[0007] Em um exemplo, um método de codificação (por exemplo, codificação ou decodificação) de dados de vídeo inclui codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento, prevendo uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual e

codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

[0008] Em outro exemplo, um dispositivo para codificar (por exemplo, codificar ou decodificar) dados de vídeo inclui uma memória configurada para armazenar dados de vídeo e um ou mais processadores implementados em conjunto de circuitos e configurados para codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro MVP (preditor de vetor de movimento) para o primeiro vetor de movimento, predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual e codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

[0009] Em outro exemplo, um meio de armazenamento legível por computador tendo armazenado nele instruções que, quando executadas, fazem com que um processador codifique uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento, predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual e codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de

movimento.

[0010] Em outro exemplo, um dispositivo para codificar (por exemplo, codificar ou decodificar) dados de vídeo inclui meios para codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento, meios para predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual e meios para codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

[0011] Os detalhes de um ou mais exemplos são apresentados nos desenhos anexos e na descrição abaixo. Outras características, objetos e vantagens serão evidentes a partir da descrição e dos desenhos e a partir das reivindicações.

BREVE DESCRIÇÃO DOS DESENHOS

[0012] As FIGs. 1A e 1B são diagramas conceituais ilustrando exemplos de candidatos espaciais vizinhos para modos de mesclagem e predição de vetores de movimento avançada (AMVP) de Codificação de Vídeo de Alta Eficiência (HEVC).

[0013] A FIG. 2 é um diagrama conceitual ilustrando um vetor de movimento de dois pontos afim com quatro parâmetros afins para um bloco atual.

[0014] A FIG. 3 é um diagrama conceitual ilustrando um exemplo de modo de predição afim para um bloco atual.

[0015] As FIGs. 4A e 4B são diagramas conceituais ilustrando um exemplo de modo de mesclagem afim para um bloco atual.

[0016] A FIG. 5 é um diagrama conceitual ilustrando um modelo afim de exemplo com seis parâmetros (três vetores de movimento).

[0017] A FIG. 6 é um diagrama de blocos ilustrando um exemplo de sistema de codificação e decodificação de vídeo que pode utilizar técnicas desta descrição para codificar eficientemente informação de movimento para predição afim.

[0018] A FIG. 7 é um diagrama de blocos ilustrando um exemplo de um codificador de vídeo que pode implementar técnicas desta descrição para codificar informação de movimento de predição afim.

[0019] A FIG. 8 é um diagrama de blocos ilustrando um exemplo de decodificador de vídeo 30 que pode implementar técnicas desta descrição para decodificar informação de movimento de predição afim.

[0020] A FIG. 9 é um diagrama conceitual ilustrando um exemplo de predição de diferença de vetor de movimento (MVD) para predição de informação de movimento afim.

[0021] A FIG. 10 é um diagrama conceitual ilustrando um exemplo de predição MVD para predição afim com três vetores de movimento (predição afim de seis parâmetros).

[0022] A FIG. 11 é um fluxograma ilustrando um método de exemplo para codificar um bloco atual de dados de vídeo de acordo com as técnicas desta descrição.

[0023] A FIG. 12 é um fluxograma ilustrando um método de exemplo de decodificação de um bloco atual de dados de vídeo de acordo com as técnicas desta descrição.

DESCRIÇÃO DETALHADA

[0024] Os padrões de codificação de vídeo incluem ITU-T H.261, ISO/IEC MPEG-1 Visual, ITU-T H.262 ou ISO/IEC MPEG-2 Visual, ITU-T H.263, ISO/IEC MPEG-4 Visual e ITU -T H.264 (conhecido também como ISO/IEC MPEG-4 AVC), incluindo as extensões de Codificação de Vídeo Escalável (SVC) e de Codificação de Vídeo Multivista (MVC).

[0025] Recentemente, o projeto de um novo padrão de codificação de vídeo, denominado ITU-T H.265/Codificação de Vídeo de Alta Eficiência (HEVC), foi finalizado pela Equipe de Colaboração Conjunta em Codificação de Vídeo (JCT-VC) de ITCE-T Grupo de Experts em Codificação de Vídeo (VCEG) e ISO/IEC Grupo de Experts em Imagem de Movimento (MPEG). A última especificação de rascunho HEVC, e referida como HEVC WD a seguir, está disponível em phenix.int-evry.fr/jct/doc_end_user/documents/15_Geneva/wgll/JCTVC-O1003-v2.zip. As “Range Extensions” para HEVC, denominadas HEVC-Rext, também estão sendo desenvolvidas pelo JCT-VC. Um Rascunho de Trabalho (WD) das Range extension, referido a seguir como RExt WD6, está disponível em phenix.int-evry.fr/jct/doc_end_user/documents/16_San%20Jose/wgll/JCTVC-P1005-vl .zip.

[0026] Investigações de novas ferramentas de codificação para codificação de vídeo futura estão em andamento (por exemplo, conforme estudado na JVET- Equipe de Exploração Conjunta de Vídeo), e tecnologias que melhoram a eficiência de codificação para codificação de

vídeo foram propostas. Há evidências de que melhorias significativas na eficiência de codificação podem ser obtidas explorando as características do conteúdo de vídeo, especialmente para conteúdo de alta resolução como 4K, com novas ferramentas de codificação dedicadas além de H.265/HEVC. Empresas e organizações foram convidadas a enviar propostas na fase de exploração para possíveis melhorias da eficiência da codificação de vídeo.

[0027] No H.265/HEVC, para cada bloco, um conjunto de informações de movimento pode estar disponível. Um conjunto de informações de movimento pode conter informação de movimento para direções de predição para frente (forward) e para trás (backward). As direções de predição para frente e para trás são duas direções de predição de um modo de predição bidirecional e os termos "para frente" e "para trás" não têm necessariamente um significado geométrico, mas correspondem à lista de imagens de referência 0 (RefPicList0) e à lista de imagens de referência 1 (RefPicList1) de uma imagem atual. Quando apenas uma lista de imagens de referência está disponível para uma imagem ou fatia, apenas RefPicListO está disponível e a informação de movimento de cada bloco de uma fatia é sempre para frente.

[0028] Para cada direção de predição, de acordo com H.265/HEVC, a informação de movimento deve conter um índice de referência e um vetor de movimento. Em alguns casos, por simplicidade, um vetor de movimento pode ele próprio ser referido de uma maneira que se pressupõe que o vetor de movimento tenha um índice de referência associado. Um índice de referência é utilizado para identificar uma

imagem de referência na lista de imagens de referência atual (RefPicList0 ou RefPicList1). Um vetor de movimento tem uma componente horizontal e uma componente vertical.

[0029] A contagem de pedidos de imagens (POC) é amplamente utilizada nos padrões de codificação de vídeo para identificar a ordem de exibição de uma imagem. Embora existam casos em que duas imagens dentro de uma sequência de vídeo codificada podem ter o mesmo valor de POC, isto normalmente não ocorre dentro de uma sequência de vídeo codificada. Quando várias sequências de vídeo codificadas estão presentes em um fluxo de bits, as imagens com o mesmo valor de POC podem estar mais próximas umas das outras em termos de ordem de decodificação. Os valores POC das imagens geralmente são utilizados para a construção da lista de imagens de referência, derivação da imagem de referência definida como no HEVC e graduação de vetor de movimento.

[0030] No HEVC, a maior unidade de codificação de uma fatia é chamada de CTB (bloco de árvore de codificação). Um CTB contém uma quadtree, cujos nós são unidades de codificação. O tamanho de um CTB pode variar de 16x16 pixels até 64x64 pixels no perfil principal do HEVC (embora tecnicamente tamanhos de CTB 8x8 possam ser suportados). Uma unidade de codificação (CU) pode ter o mesmo tamanho de um CTB e ter apenas 8x8 pixels. Cada unidade de codificação pode ser codificada com um modo. Quando uma CU é codificada, ela pode ser particionada em duas ou mais unidades de predição (PUs) ou tornar-se apenas uma PU quando uma partição adicional não se aplicar. Quando duas PUs estão presentes em uma CU, elas podem ter

retângulos de meio tamanho ou dois retângulos com 1/4 ou 3/4 de tamanho da CU. Quando a CU é inter-codificada, um conjunto de informações de movimento está presente para cada PU. Além disso, cada PU é codificada com um modo de predição exclusivo para derivar o conjunto de informações de movimento. No HEVC, os menores tamanhos de PU são 8x4 e 4x8.

[0031] No HEVC, existem dois modos de inter-predição, denominados mesclagem (salto é considerado um caso especial de mesclagem) e modos avançados de predição de vetores de movimento (AMVP), para uma unidade de predição (PU). No modo AMVP ou no modo de mesclagem, uma lista de candidatos ao vetor de movimento (MV) é mantida para vários preditores de vetores de movimento. O(s) vetor(es) de movimento, bem como os índices de referência no modo de mesclagem, da PU atual são gerados retirando um candidato a partir da lista de candidatos MV.

[0032] A lista de candidatos MV, por HEVC, contém até 5 candidatos para o modo de mesclagem e apenas dois candidatos para o modo AMVP. Um candidato a mesclagem pode conter um conjunto de informações de movimento, por exemplo, vetores de movimento correspondentes a ambas as listas de imagens de referência (lista 0 e lista 1) e os índices de referência. Se um candidato a mesclagem é identificado por um índice de mesclagem, as imagens de referência são utilizadas para a predição dos blocos atuais, bem como são determinados os vetores de movimento associados. No entanto, sob o modo AMVP, para cada direção de predição em potencial a partir da lista 0 ou da lista 1, um índice de referência precisa ser explicitamente

sinalizado, juntamente com um índice preditor de vetor de movimento (MVM) para lista de candidatos MV, uma vez que o candidato AMVP contém apenas um vetor de movimento. No modo AMVP, os vetores de movimento preditos podem ser refinados ainda mais.

[0033] Conforme pode ser visto acima, um candidato a mesclagem corresponde a um conjunto completo de informações de movimento, enquanto um candidato AMVP contém apenas um vetor de movimento para uma direção de predição e um índice de referência específicos. Os candidatos para ambos os modos são derivados da mesma forma a partir dos mesmos blocos vizinhos espaciais e temporais.

[0034] As FIGs. 1A e 1B são diagramas conceituais ilustrando exemplos de candidatos vizinhos espaciais para os modos de mesclagem e AMVP de HEVC. Em particular, a FIG. 1A ilustra candidatos a vetores de movimento (MV) vizinhos espaciais para o modo de mesclagem, enquanto a FIG. 1B ilustra candidatos a MV vizinhos espaciais para o modo AMVP. De acordo com o HEVC, os candidatos MV espaciais são derivados dos blocos vizinhos mostrados nas FIGS. 1A e 1B, para uma PU específica (PU_0), embora as técnicas para gerar os candidatos a partir dos blocos sejam diferentes nos modos de mesclagem e AMVP.

[0035] No modo de mesclagem de HEVC, até quatro candidatos MV espaciais podem ser derivados com as ordens mostradas na FIG. 1A com números, e a ordem é a seguinte: esquerda (0), superior (1), superior direito (2), inferior esquerdo (3) e superior esquerdo (4), conforme mostrado na FIG. 1A.

[0036] No modo AVMP do HEVC, os blocos vizinhos

são divididos em dois grupos: um grupo esquerdo incluindo os blocos 0 e 1 e um grupo acima incluindo os blocos 2, 3 e 4, conforme mostrado na FIG. 1B. Para cada grupo, o candidato em potencial em um bloco vizinho que se refere à mesma imagem de referência que a indicada pelo índice de referência sinalizado tem a maior prioridade a ser escolhida para formar um candidato final do grupo. É possível que nenhum bloco vizinho contenha um vetor de movimento apontando para a mesma imagem de referência. Portanto, se esse candidato não puder ser encontrado, o primeiro candidato disponível será escalado para formar o candidato final. Assim, as diferenças de distância temporal podem ser compensadas.

[0037] A compensação de movimento no H.265/HEVC é utilizada para gerar um preditor para um bloco inter-codificado atual. Um vetor de movimento com precisão de um quarto de pixel pode ser utilizado e os valores de pixels em posições fracionárias podem ser interpolados utilizando valores de pixels inteiros vizinhos tanto para os componentes luma quanto para os componentes croma.

[0038] A FIG. 2 é um diagrama conceitual ilustrando um vetor de movimento de dois pontos afim com quatro parâmetros afins para um bloco atual. Nos padrões atuais de codec de vídeo existentes, apenas um modelo de movimento de tradução é aplicado para a predição de compensação de movimento (MCP). No entanto, no mundo real, existem muitos tipos de movimentos, por exemplo zoom in/out, rotação, movimentos de perspectiva e outros movimentos irregulares. Se apenas um modelo de movimento de translação para MCP for aplicado nessas sequências de teste

com movimentos irregulares, a precisão da predição será afetada e resultará em baixa eficiência de codificação. Por muitos anos, muitos especialistas em vídeo tentaram projetar muitos algoritmos para melhorar o MCP e aumentar a eficiência da codificação. Os modos de mesclagem afim e inter-afim (AMVP) foram propostos para lidar com modelos de movimento afim com 4 parâmetros como:

$$\begin{cases} V_x = ax + by + e \\ V_y = cx + dy + f \end{cases} \quad (1)$$

[0039] Na equação (1) acima, (vx_0, vy_0) é o vetor de movimento de ponto de controle no canto superior esquerdo do bloco atual da FIG. 2, e (vx_1, vy_1) é outro vetor de movimento de ponto de controle no canto superior direito do bloco atual da FIG. 2. O modelo afim se resume a:

$$\begin{cases} V_x = \frac{(v_{1x}-v_{0x})}{w}x + \frac{(v_{2x}-v_{0x})}{h}y + v_{0x} \\ V_y = \frac{(v_{1y}-v_{0y})}{w}x + \frac{(v_{2y}-v_{0y})}{h}y + v_{0y} \end{cases} \quad (2)$$

[0040] No software JEM atual, a predição de movimento afim é aplicada apenas a blocos quadrados. Como uma extensão natural, a predição de movimento afim pode ser aplicada a blocos não quadrados.

[0041] A FIG. 3 é um diagrama conceitual ilustrando um exemplo de modo de predição afim para um bloco atual. O bloco atual pode ser uma CU ou uma PU atual. Neste exemplo, o bloco atual inclui dois blocos denominados "V0" no canto superior esquerdo e "V1" no canto superior direito e os blocos vizinhos rotulados A, B, C, D e E. Em particular, o bloco "V0" bloqueia os blocos A, B e C, enquanto o bloco "V1" bloqueia os blocos D e E.

[0042] Para cada CU/PU cujo tamanho seja igual ou superior a 16x16, o modo de predição afim (modo AF_INTER)

pode ser aplicado como a seguir. Se a CU/PU atual estiver no modo AF_INTER, uma flag afim no nível de CU/PU poderá ser sinalizada no fluxo de bits. Uma lista de candidatos $\{(v_0, v_1)|v_0 = \{v_A, v_B, v_c\}, v_1 = \{v_D, v_E\}\}$ é construída utilizando os blocos válidos reconstruídos vizinhos.

[0043] Conforme mostrado na FIG. 3, a informação de movimento v_0 é selecionada a partir dos vetores de movimento dos blocos A, B e/ou C. O vetor de movimento a partir do bloco vizinho é dimensionado de acordo com a lista de referências e a relação entre a POC da referência para o bloco vizinho, a POC da referência para a CU/PU atual e a POC da CU/PU atual. E a abordagem para selecionar v_1 a partir dos blocos vizinhos D e E é similar. Se o número de lista de candidatos for menor que 2, os candidatos a AMVP serão atribuídos a v_0 e v_1. Um custo de otimização de taxa de distorção (RDO) da CU/PU atual é utilizado para determinar qual (v_0, v_1) é selecionada como a predição de vetor de movimento de ponto de controle (CPMVP) da CU/PU atual. E o índice para indicar a posição da CPMVP na lista de candidatos é sinalizado no fluxo de bits.

[0044] Depois que a CPMVP da CU/PU afim atual é determinada, a estimação de movimento afim é aplicada e a CPMV é encontrada. Em seguida, a diferença do CPMV e da CPMVP é codificada no fluxo de bits. A predição de compensação de movimento afim mencionada acima é aplicada para gerar os resíduos da CU/PU atual. Finalmente, os resíduos da CU/PU atual são transformados, quantizados e codificados no fluxo de bits, de acordo com procedimentos convencionais.

[0045] As FIGs. 4A e 4B são diagramas conceituais ilustrando um exemplo de modo de mesclagem afim para um bloco atual. O bloco atual pode ser uma CU ou uma PU atual. Neste exemplo, o bloco atual possui cinco blocos vizinhos, rotulados como A, B, C, D e E, conforme mostrado na FIG. 4A.

[0046] Quando a CU/PU atual é aplicada no modo de mesclagem afim (modo AF_MERGE), ela obtém o primeiro bloco codificado com o modo afim a partir dos blocos válidos reconstruídos vizinhos de A, B, C, D e E. A ordem de seleção para o bloco candidato é a partir da esquerda, superior, superior direito, inferior esquerdo, superior esquerdo, conforme mostrado na FIG. 4A. Por exemplo, se o bloco inferior esquerdo vizinho A for codificado no modo afim, conforme mostrado na FIG. 4B, os vetores de movimento v_2, v_3 e v_4 do canto superior esquerdo, do canto superior direito e do canto inferior esquerdo da CU/PU que contém o bloco A são derivados. O vetor de movimento v_0 do canto superior esquerdo na CU/PU atual é calculado de acordo com v_2, v_3 e v_4. De forma similar, o vetor de movimento v_1 do superior direito da CU/PU atual é calculado com base em V_2, V_3 e V_4.

[0047] Depois que as CPMV da CU/PU atual v_0 e v_1 são calculadas, de acordo com o modelo de movimento afim simplificado definido na equação (2) acima, é gerada a MVF da CU/PU atual. Em seguida, o MCP afim é aplicado. Para identificar se a CU/PU atual está codificada com o modo AF_MERGE, uma flag afim é sinalizada no fluxo de bits quando há pelo menos um bloco vizinho codificado no modo afim. Se nenhum bloco afim vizinho ao bloco atual existir

conforme mostrado na FIG. 4A, nenhuma flag afim é gravada no fluxo de bits.

[0048] No HEVC, a codificação aritmética binária adaptativa ao contexto (CABAC) inclui um processo de binarização que é utilizado para converter um símbolo em um valor binarizado. A binarização permite a codificação aritmética binária eficiente por meio de um mapeamento exclusivo de elementos de sintaxe não binário para uma sequência de bits, que são chamados de faixas. No software de referência JEM2.0, para o modo de mesclagem afim, apenas a flag afim é codificada e o índice de mesclagem é inferido como o primeiro modelo afim vizinho disponível na ordem de verificação predefinida A → B → C → D → E. Para o modo inter-afim, duas sintaxes de MVD são codificadas para cada lista de predição, indicando a diferença de vetor de movimento entre o vetor de movimento afim derivado e o vetor de movimento predito.

[0049] A FIG. 5 é um diagrama conceitual ilustrando um modelo afim de exemplo com seis parâmetros (três vetores de movimento). Em Zou et al., "PREDIÇÃO DE MOVIMENTO AFIM PARA CODIFICAÇÃO DE VÍDEO", pedido de Patente Norte-Americano N° 15/587.044, depositado em 4 de maio de 2017, foi descrito um esquema de predição de movimento afim selecionável. Um bloco com predição afim pode utilizar um modelo afim de quatro parâmetros ou um modelo afim de seis parâmetros adaptativamente. Um modelo afim com seis parâmetros pode ser definido como:

$$\begin{cases} mv_x = ax + by + e \\ mv_y = cx + dy + f \end{cases} \qquad (3)$$

[0050] Um modelo afim com 6 parâmetros possui

três pontos de controle. Em outras palavras, um modelo afim com seis parâmetros é determinado por três vetores de movimento (MV0, MV1 e MV2), por exemplo, conforme mostrado na FIG. 5. MV0 é o primeiro vetor de movimento de ponto de controle no canto superior esquerdo de um bloco atual, MV1 é o segundo vetor de movimento de ponto de controle no canto superior direito do bloco atual e MV2 é o terceiro vetor de movimento de ponto de controle no canto inferior esquerdo do bloco atual, conforme mostrado na FIG. 5. O modelo afim construído com os três vetores de movimento é calculado como:

$$\begin{cases} mv_x = \frac{(mv_{1x} - mv_{0x})}{w} x + \frac{(mv_{2x} - mv_{0x})}{w} y + mv_{0x} \\ mv_y = \frac{(mv_{1y} - mv_{0y})}{w} x + \frac{(mv_{2y} - mv_{0y})}{w} y + mv_{0y} \end{cases} \quad (4)$$

[0051] A equação (4) acima é para um bloco quadrado com lados iguais a w. Para um bloco não quadrado (por exemplo, um bloco retangular) tendo uma largura w e uma altura h, o seguinte modelo afim pode ser utilizado:

$$\begin{cases} mv_x = \frac{(mv_{1x} - mv_{0x})}{w} x + \frac{(mv_{2x} - mv_{0x})}{h} y + mv_{0x} \\ mv_y = \frac{(mv_{1y} - mv_{0y})}{w} x + \frac{(mv_{2y} - mv_{0y})}{h} y + mv_{0y} \end{cases} \quad (5)$$

[0052] De maneira similar à mesclagem afim, para derivar os vetores de movimento do canto superior esquerdo e do canto superior direito, conforme descrito em relação à FIG. 4 acima pode também ser utilizada para derivar os MVPs para o canto superior esquerdo, para o canto superior direito e para o canto inferior esquerdo. Exemplos adicionais são descritos em Chen et al., "PREDIÇÃO DE VETOR DE MOVIMENTO PARA O MODELO DE MOVIMENTO AFIM", pedido de Patente Provisório Norte-Americano N° 62/404.719, depositado em 5 de outubro de 2016.

[0053] A FIG. 6 é um diagrama de blocos ilustrando um exemplo de sistema de codificação e decodificação de vídeo 10 que pode utilizar técnicas desta descrição para codificar com eficiência informação de movimento para predição afim. Conforme mostrado na FIG. 6, o sistema 10 inclui um dispositivo de origem 12 que fornece dados de vídeo codificados para serem decodificados posteriormente por um dispositivo de destino 14. Em particular, o dispositivo de origem 12 fornece os dados de vídeo para o dispositivo de destino 14 através de um meio legível por computador 16. O dispositivo de origem 12 e o dispositivo de destino 14 podem compreender qualquer um dentre uma ampla variedade de dispositivos, incluindo computadores desktop, computadores notebooks (isto é, laptops), computadores tablets, set-top boxes, aparelhos de telefone, tais como os chamados telefones "inteligentes", os chamados pads "inteligentes", televisores, câmeras, dispositivos de exibição, players de mídia digital, consoles de videogame, dispositivo de streaming de vídeo ou similares. Em alguns casos, o dispositivo de origem 12 e o dispositivo de destino 14 podem ser equipados para comunicação sem fio.

[0054] O dispositivo de destino 14 pode receber os dados de vídeo codificados para serem decodificados através do meio legível por computador 16. O meio legível por computador 16 pode compreender qualquer tipo de meio ou dispositivo capaz de mover os dados de vídeo codificados a partir do dispositivo de origem 12 para o dispositivo de destino 14. Em um exemplo, o meio legível por computador 16 pode compreender um meio de comunicação para permitir que o

dispositivo de origem 12 transmita dados de vídeo codificados diretamente para o dispositivo de destino 14 em tempo real. Os dados de vídeo codificados podem ser modulados de acordo com um padrão de comunicação, tal como um protocolo de comunicação sem fio, e transmitidos para o dispositivo de destino 14. O meio de comunicação pode compreender qualquer meio de comunicação sem fio ou cabeado, tal como um espectro de radiofrequência (RF) ou uma ou mais linhas de transmissão físicas. O meio de comunicação pode fazer parte de uma rede baseada em pacotes, tal como uma rede de área local, uma rede de área ampla ou uma rede global tal como a Internet. O meio de comunicação pode incluir roteadores, comutadores, estações base ou qualquer outro equipamento que possa ser útil para facilitar a comunicação a partir do dispositivo de origem 12 para o dispositivo de destino 14.

[0055] Em alguns exemplos, os dados codificados podem ser enviados a partir da interface de saída 22 para um dispositivo de armazenamento. De forma similar, os dados codificados podem ser acessados a partir do dispositivo de armazenamento pela interface de entrada. O dispositivo de armazenamento pode incluir qualquer uma dentre uma variedade de meios de armazenamento de dados distribuídos ou acessados localmente, tais como um disco rígido, discos Blu-ray, DVDs, CD-ROMs, memória flash, memória volátil ou não volátil ou qualquer outro meio de armazenamento digital adequado para armazenar dados de vídeo codificados. Em um exemplo adicional, o dispositivo de armazenamento pode corresponder a um servidor de arquivos ou outro dispositivo de armazenamento intermediário que pode armazenar o vídeo

codificado gerado pelo dispositivo de origem 12. O dispositivo de destino 14 pode acessar dados de vídeo armazenados a partir do dispositivo de armazenamento por meio de streaming ou download. O servidor de arquivos pode ser qualquer tipo de servidor capaz de armazenar dados de vídeo codificados e transmitir esses dados de vídeo codificados para o dispositivo de destino 14. Os servidores de arquivos de exemplo incluem um servidor da Web (por exemplo, para um site), um servidor FTP, dispositivos de armazenamento conectado à rede (NAS) ou uma unidade de disco local. O dispositivo de destino 14 pode acessar os dados de vídeo codificados através de qualquer conexão de dados padrão, incluindo uma conexão com a Internet. Isto pode incluir um canal sem fio (por exemplo, uma conexão Wi-Fi), uma conexão com fio (por exemplo, DSL, modem a cabo etc.) ou uma combinação de ambos que seja adequada para acessar dados de vídeo codificados armazenados em um servidor de arquivos. A transmissão de dados de vídeo codificados a partir do dispositivo de armazenamento pode ser uma transmissão de streaming, uma transmissão de download ou uma combinação dos mesmos.

[0056] As técnicas desta descrição não estão necessariamente limitadas a aplicações ou configurações sem fio. As técnicas podem ser aplicadas à codificação de vídeo em suporte a qualquer uma dentre a variedade de aplicações multimídia, tais como difusões de televisão pelo ar, transmissões de televisão a cabo, transmissões de televisão por satélite, transmissões de vídeo por streaming na Internet, tal como streaming adaptável dinâmico por HTTP (DASH), vídeo digital codificado que é codificado em um

meio de armazenamento de dados, decodificação de vídeo digital armazenado em um meio de armazenamento de dados ou outras aplicações. Em alguns exemplos, o sistema 10 pode ser configurado para suportar transmissão de vídeo unidirecional ou bidirecional para suportar aplicações tais como streaming de vídeo, playback de vídeo, difusão de vídeo e/ou telefonia por vídeo.

[0057] No exemplo da FIG. 6, o dispositivo de origem 12 inclui a fonte de vídeo 18, o codificador de vídeo 20 e a interface de saída 22. O dispositivo de destino 14 inclui a interface de entrada 28, o decodificador de vídeo 30 e o dispositivo de exibição 32. De acordo com esta descrição, o codificador de vídeo 20 do dispositivo de origem 12 pode ser configurado para aplicar as técnicas para codificar eficientemente a informação de movimento para predição afim. Em outros exemplos, um dispositivo de origem e um dispositivo de destino podem incluir outros componentes ou arranjos. Por exemplo, o dispositivo de origem 12 pode receber dados de vídeo a partir de uma fonte de vídeo externa 18, tal como uma câmera externa. Da mesma forma, o dispositivo de destino 14 pode realizar interface com um dispositivo de exibição externo, em vez de incluir um dispositivo de exibição integrado.

[0058] O sistema ilustrado 10 da FIG. 6 é apenas um exemplo. Técnicas para codificar eficientemente a informação de movimento para predição afim podem ser realizadas por qualquer dispositivo de codificação e/ou decodificação de vídeo digital. Embora geralmente as técnicas desta descrição sejam realizadas por um

dispositivo de codificação de vídeo, as técnicas podem também ser realizadas por um codificador/decodificador de vídeo, normalmente referido como "CODEC". Além disso, as técnicas desta descrição podem também ser realizadas por um pré-processador de vídeo. O dispositivo de origem 12 e o dispositivo de destino 14 são meramente exemplos desses dispositivos de codificação nos quais o dispositivo de origem 12 gera dados de vídeo codificados para transmissão para o dispositivo de destino 14. Em alguns exemplos, os dispositivos 12, 14 podem operar de maneira substancialmente simétrica, de modo que cada um dos dispositivos 12, 14 inclua componentes de codificação e decodificação de vídeo. Portanto, o sistema 10 pode suportar transmissão de vídeo unidirecional ou bidirecional entre os dispositivos de vídeo 12, 14, por exemplo, para streaming de vídeo, playback de vídeo, difusão de vídeo ou telefonia por vídeo.

[0059] A fonte de vídeo 18 do dispositivo de origem 12 pode incluir um dispositivo de captura de vídeo, tal como uma câmera de vídeo, um arquivo de vídeo contendo vídeo capturado previamente e/ou uma interface de alimentação de vídeo para receber vídeo a partir de um provedor de conteúdo de vídeo. Como alternativa adicional, a fonte de vídeo 18 pode gerar dados baseados em gráficos de computador tal como o vídeo de origem ou uma combinação de vídeo ao vivo, vídeo arquivado e vídeo gerado por computador. Em alguns casos, se a fonte de vídeo 18 for uma câmera de vídeo, o dispositivo de origem 12 e o dispositivo de destino 14 poderão formar os chamados telefones com câmera ou telefones com vídeo. Conforme mencionado acima,

no entanto, as técnicas apresentadas nesta descrição podem ser aplicáveis à codificação de vídeo em geral e podem ser aplicadas a aplicações sem fio e/ou cabeadas. Em cada caso, o vídeo capturado, pré-capturado ou gerado por computador pode ser codificado pelo codificador de vídeo 20. As informações de vídeo codificadas podem então ser enviadas por meio da interface de saída 22 para um meio legível por computador 16.

[0060] O meio legível por computador 16 pode incluir meio transitório, tal como difusão sem fio ou transmissão de rede cabeada ou meio de armazenamento (ou seja, meio de armazenamento não transitório), tal como um disco rígido, um drive flash, disco compacto, disco de vídeo digital, Disco Blu-ray ou outro meio legível por computador. Em alguns exemplos, um servidor de rede (não mostrado) pode receber dados de vídeo codificados a partir do dispositivo de origem 12 e fornecer os dados de vídeo codificados para o dispositivo de destino 14, por exemplo, via transmissão de rede. De forma similar, um dispositivo de computação de uma instalação de produção de meio, tal como uma instalação de estampagem de disco, pode receber dados de vídeo codificados a partir do dispositivo de origem 12 e produzir um disco contendo os dados de vídeo codificados. Portanto, o meio legível por computador 16 pode ser entendido como incluindo um ou mais meios legíveis por computador de várias formas, em vários exemplos.

[0061] A interface de entrada 28 do dispositivo de destino 14 recebe informações a partir do meio legível por computador 16. As informações do meio legível por computador 16 podem incluir informações de sintaxe

definidas pelo codificador de vídeo 20, que também são utilizadas pelo decodificador de vídeo 30, que inclui elementos de sintaxe que descrevem características e/ou processamento de blocos e de outras unidades codificadas. O dispositivo de exibição 32 exibe os dados de vídeo decodificados para um usuário e pode compreender qualquer um dentre uma variedade de dispositivos de exibição, tais como um tubo de raios catódicos (CRT), um monitor de cristal líquido (LCD), um monitor de plasma, um monitor de diodo orgânico emissor de luz (OLED) ou outro tipo de dispositivo de exibição.

[0062] O codificador de vídeo 20 e o decodificador de vídeo 30 podem operar de acordo com um padrão de codificação de vídeo, tal como o padrão de Codificação de Vídeo de Alta Eficiência (HEVC), referido também como ITU-T H.265. Alternativamente, o codificador de vídeo 20 e o decodificador de vídeo 30 podem operar de acordo com outros padrões proprietários ou da indústria, tal como o padrão ITU-T H.264, alternativamente referido como MPEG-4, Parte 10, Codificação de Vídeo Avançada (AVC) ou extensões desses padrões. As técnicas desta descrição, no entanto, não estão limitadas a nenhum padrão de codificação específico. Outros exemplos de padrões de codificação de vídeo incluem MPEG-2 e ITU-T H.263. Embora não mostrado na FIG. 6, em alguns aspectos, o codificador de vídeo 20 e o decodificador de vídeo 30 podem ser cada um integrado com um codificador e um decodificador de áudio e podem incluir unidades MUX-DEMUX apropriadas, ou outro hardware e software, para lidar com a codificação tanto de áudio quanto de vídeo de maneira em um fluxo de dados comum

ou em fluxos de dados separados. Se aplicável, as unidades MUX-DEMUX podem estar em conformidade com o protocolo multiplexador ITU H.223 ou com outros protocolos, tal como o protocolo de datagrama de usuário (UDP).

[0063] O codificador de vídeo 20 e o decodificador de vídeo 30 podem ser implementados como qualquer um dentre uma variedade de conjuntos de circuitos codificadores adequados, tais como um ou mais microprocessadores, processadores de sinal digital (DSPs), circuitos integrados de aplicação específica (ASICs), arranjos de portas programáveis em campo (FPGAs), lógica discreta, software, hardware, firmware ou qualquer combinação dos mesmos. Quando as técnicas são implementadas parcialmente em software, um dispositivo pode armazenar instruções para o software em um meio legível por computador não transitório adequado e executar as instruções em hardware utilizando um ou mais processadores para realizar as técnicas desta descrição. Cada um dentre os codificadores de vídeo 20 e decodificadores de vídeo 30 pode ser incluído em um ou mais codificadores ou decodificadores, ou qualquer um dos quais pode ser integrado como parte de um codificador/decodificador combinado (CODEC) em um dispositivo respectivo.

[0064] Em geral, de acordo com a ITU-T H.265, uma imagem de vídeo pode ser dividida em uma sequência de unidades de árvore de codificação (CTUs) (ou unidades de codificação maiores (LCUs)) que podem incluir tanto amostras luma quanto amostras croma. Alternativamente, as CTUs podem incluir dados monocromáticos (isto é, apenas amostras luma). Os dados de sintaxe em um fluxo de bits

podem definir um tamanho para a CTU, que é a maior unidade de codificação em termos de número de pixels. Uma fatia inclui um número de CTUs consecutivas em ordem de codificação. Uma imagem de vídeo pode ser particionada em uma ou mais fatias. Cada CTU pode ser dividida em unidades de codificação (CUs) de acordo com um quadtree. Em geral, uma estrutura de dados quadtree inclui um nó por CU, com um nó raiz correspondente à CTU. Se uma CU for dividida em quatro sub-CUs, o nó correspondente à CU inclui quatro nós de folha, cada um dos quais corresponde a uma das sub-CUs.

[0065] Cada nó da estrutura de dados quadtree pode fornecer dados de sintaxe para a CU correspondente. Por exemplo, um nó no quadtree pode incluir uma flag de divisão, indicando se a CU correspondente ao nó está dividida em sub-CUs. Os elementos de sintaxe para uma CU podem ser definidos recursivamente e podem depender se a CU é dividida em sub-CUs. Se uma CU não for dividida ainda mais, será referida como CU folha. Nesta descrição, quatro sub-CUs de uma CU-folha serão também referidas como CU-folhas, mesmo se não houver uma divisão explícita da CU-folha original. Por exemplo, se uma CU no tamanho 16x16 não for dividida ainda mais, as quatro sub-CUs 8x8 serão também referidas como CUs folha, embora a CU 16x16 nunca tenha sido dividida.

[0066] Uma CU tem um propósito similar à de um macro bloco do padrão H.264, exceto que uma CU não tem uma distinção de tamanho. Por exemplo, uma CTU pode ser dividida em quatro nós filhos (referidos também como sub-CUs) e cada nó filho pode, por sua vez, ser um nó pai e ser dividido em outros quatro nós filhos. Um nó filho final não

dividido, referido como um nó folha de quadtree, compreende um nó de codificação, também chamado de CU-folha. Os dados de sintaxe associados com um fluxo de bits codificado podem definir um número máximo de vezes que uma CTU pode ser dividida, referido como profundidade máxima da CU e pode também definir um tamanho mínimo dos nós de codificação. Por conseguinte, um fluxo de bits pode também definir uma menor unidade de codificação (SCU). Esta descrição Utiliza o termo "bloco" para se referir a qualquer um dentre CU, unidade de predição (PU) ou unidade de transformada (TU), no contexto de HEVC, ou estruturas de dados similares no contexto de outros padrões (por exemplo, macro blocos e sub-blocos dos mesmos em H.264/AVC).

[0067] Uma CU inclui um nó de codificação e unidades de predição (PUs) e unidades de transformada (TUs) associadas com o nó de codificação. Um tamanho da CU corresponde ao tamanho do nó de codificação e geralmente tem uma forma quadrada. O tamanho da CU pode variar de 8x8 pixels até o tamanho da CTU com um tamanho máximo, por exemplo, 64x64 pixels ou superior. Cada CU pode conter uma ou mais PUs e uma ou mais TUs. Os dados de sintaxe associados com uma CU podem descrever, por exemplo, o particionamento da CU em uma ou mais PUs. Os modos de particionamento podem diferir entre se a CU é dividida ou codificada no modo direto, codificada no modo intra-predição ou codificada no modo inter-predição. As PUs podem ser particionadas para ter uma forma não quadrada. Os dados de sintaxe associados com uma CU também podem descrever, por exemplo, o particionamento da CU em uma ou mais TUs de acordo com uma quadtree. Uma TU pode ter forma quadrada ou

não quadrada (por exemplo, retangular).

[0068] O padrão HEVC permite transformações de acordo com as TUs, que podem ser diferentes para diferentes CUs. As TUs geralmente são dimensionadas com base no tamanho das PUs (ou partições de uma CU) dentro de uma determinada CU definida para uma CTU particionada, embora isso nem sempre seja o caso. As TUs são tipicamente do mesmo tamanho ou menores que as PUs (ou partições de uma CU, por exemplo, no caso de intra-predição). Em alguns exemplos, as amostras residuais correspondentes a uma CU podem ser subdivididas em unidades menores utilizando uma estrutura de quadtree conhecida como "quadtree residual" (RQT). Os nós folha da RQT podem ser referidos como unidades de transformada (TUs). Os valores de diferença de pixel associados com a TUs podem ser transformados para produzir coeficientes de transformada, que podem ser quantizados.

[0069] Uma CU-folha pode incluir uma ou mais unidades de predição (PUs) quando preditas utilizando inter-predição. Em geral, uma PU representa uma área espacial correspondente a toda ou a uma parte da CU correspondente e pode incluir dados para recuperar e/ou gerar uma amostra de referência para a PU. Além disso, uma PU inclui dados relacionados à predição. Quando a CU é codificada inter-modos, uma ou mais PUs da CU podem incluir dados que definem a informação de movimento, tal como um ou mais vetores de movimento, ou as PUs podem ser codificadas no modo de salto. Os dados que definem o vetor de movimento para uma PU podem descrever, por exemplo, uma componente horizontal do vetor de movimento, uma componente vertical

do vetor de movimento, uma resolução para o vetor de movimento (por exemplo, precisão de um quarto de pixel ou precisão de um oitavo pixel)), uma imagem de referência para a qual o vetor de movimento aponta e/ou uma lista de imagens de referência (por exemplo, Lista 0 ou Lista 1) para o vetor de movimento.

[0070] As CU-folhas podem também ser preditas intra-modo. Em geral, a intra-predição envolve predizer uma CU-folha (ou partições da mesma) utilizando um intra-modo. Um codificador de vídeo pode selecionar um conjunto de pixels vizinhos, codificados anteriormente, para a CU-folha para utilizar para predizer a CU-folha (ou partições das mesmas).

[0071] Uma CU-folha também pode incluir uma ou mais unidades de transformada (TUs). As unidades de transformada podem ser especificadas utilizando uma RQT (referida também como uma estrutura quadtree da TU), conforme discutido acima. Por exemplo, uma flag de divisão pode indicar se uma CU-folha é dividida em quatro unidades de transformada. Então, cada TU pode ser dividida ainda mais em sub-TUs. Quando uma TU não é dividida ainda mais, ela pode ser chamada de TU-folha. Geralmente, para intra-codificação, todas as TUs-folha pertencentes a uma CU-folha compartilham o mesmo modo de intra-predição. Ou seja, o mesmo modo de intra-predição é geralmente aplicado para calcular os valores preditos para todas as TUs de uma CU-folha. Para intra-codificação, um codificador de vídeo pode calcular um valor residual para cada TU-folha utilizando o modo de intra-predição, como uma diferença entre a porção da CU correspondente à TU e o bloco original. Uma TU não é

necessariamente limitada ao tamanho de uma PU. Assim, as TUs podem ser maiores ou menores que uma PU. Para intra-codificação, partições de uma CU, ou a própria CU, podem ser colocadas com uma TU-folha correspondente para a CU. Em alguns exemplos, o tamanho máximo de uma TU-folha pode corresponder ao tamanho da CU-folha correspondente.

[0072] Além disso, as TUs de CU-folhas podem também ser associadas às respectivas estruturas de dados quadtree, referidas como quadtrees residuais (RQTs). Ou seja, uma CU-folha pode incluir uma quadtree indicando como a CU-folha é particionada em TUs. O nó raiz de um quadtree de TU geralmente corresponde a uma CU-folha, enquanto o nó raiz de um quadtree de CU geralmente corresponde a uma CTU (ou LCU). As TUs da RQT que não são divididas são chamadas de TUs-folha. Em geral, esta descrição utiliza os termos CU e TU para se referir a CU-folha e TU-folha, respectivamente, a menos se indicado de outra forma.

[0073] Uma sequência de vídeo normalmente inclui uma série de quadros ou imagens de vídeo, começando com uma imagem de ponto de acesso aleatório (RAP). Uma sequência de vídeo pode incluir dados de sintaxe em um conjunto de parâmetros de sequência (SPS) que caracteriza a sequência de vídeo. Cada fatia de uma imagem pode incluir dados de sintaxe da fatia que descrevem um modo de codificação para a respectiva fatia. O codificador de vídeo 20 normalmente opera em blocos de vídeo dentro de fatias de vídeo individuais, a fim de codificar os dados de vídeo. Um bloco de vídeo pode corresponder a um nó de codificação dentro de uma CU. Os blocos de vídeo podem ter tamanhos fixos ou variáveis e podem diferir em tamanho de acordo com um

padrão de codificação especificado.

[0074] Como exemplo, a predição pode ser realizada para PUs de vários tamanhos. Assumindo que o tamanho de uma CU específica seja 2Nx2N, a intra-predição pode ser realizada em tamanhos de PU de 2Nx2N ou NxN e a inter-predição pode ser realizada em tamanhos de PU simétricos de 2Nx2N, 2NxN, Nx2N ou NxN. O particionamento assimétrico para inter-predição pode também ser realizado para tamanhos de PU de 2NxnU, 2NxnD, nLx2N e nRx2N. No particionamento assimétrico, uma direção de uma CU não é particionada, enquanto a outra direção é particionada 25% e 75%. A porção da CU correspondente à partição de 25% é indicada por um "n" seguido por uma indicação de "Superior (Up)", "Inferior (Down)", "Esquerda (Left)" ou "Direita (Right)”. Assim, por exemplo, “2NxnU" refere-se a uma CU 2Nx2N que é particionada horizontalmente com uma PU 2Nx0,5N na parte superior e uma PU 2Nxl,5N na parte inferior.

[0075] Nesta descrição, "NxN" e "N por N" podem ser utilizados de forma intercambiável para se referir às dimensões de pixel de um bloco de vídeo em termos de dimensões verticais e horizontais, por exemplo, 16x16 pixels ou 16 por 16 pixels. Em geral, um bloco 16x16 terá 16 pixels na direção vertical (y = 16) e 16 pixels na direção horizontal (x = 16). Da mesma forma, um bloco NxN geralmente possui N pixels na direção vertical e N pixels na direção horizontal, onde N representa um valor inteiro não negativo. Os pixels em um bloco podem ser organizados em linhas e colunas. Além disso, os blocos não precisam necessariamente ter o mesmo número de pixels na direção horizontal e na direção vertical. Por exemplo, os blocos

podem compreender NxM pixels, onde M não é necessariamente igual a N.

[0076] Após a codificação intra-preditiva ou inter-preditiva utilizando as PUs de uma CU, o codificador de vídeo 20 pode calcular dados residuais para as TUs da CU. As PUs podem compreender dados de sintaxe descrevendo um método ou modo de gerar dados de pixel preditivos no domínio espacial (referido também como domínio de pixel) e as TUs podem compreender coeficientes no domínio de transformada após a aplicação de uma transformada, por exemplo, uma transformada discreta de cosseno (DCT), uma transformada de número inteiro, uma transformada wavelet ou transformada conceitualmente similar para dados de vídeo residuais. Os dados residuais podem corresponder a diferenças de pixels entre os pixels da imagem não codificada e os valores de predição correspondentes às PUs. O codificador de vídeo 20 pode formar as TUs para incluir coeficientes de transformada quantizados representativos dos dados residuais para a CU. Isto é, o codificador de vídeo 20 pode calcular os dados residuais (na forma de um bloco residual), transformar o bloco residual para produzir um bloco de coeficientes de transformada e depois quantizar os coeficientes de transformada para formar coeficientes de transformada quantizados. O codificador de vídeo 20 pode formar uma TU incluindo os coeficientes de transformada quantizados, bem como outras informações de sintaxe (por exemplo, informações de divisão para a TU).

[0077] Conforme observado acima, após quaisquer transformações para produzir coeficientes de transformada, o codificador de vídeo 20 pode realizar a quantização dos

coeficientes de transformada. Quantização geralmente se refere a um processo no qual os coeficientes de transformada são quantizados para possivelmente reduzir a quantidade de dados utilizados para representar os coeficientes, fornecendo compressão adicional. O processo de quantização pode reduzir a profundidade de bits associada com alguns ou a todos os coeficientes. Por exemplo, um valor de *n*-bits pode ser arredondado para um valor de *m-bits* durante a quantização, em que *n* é maior que *m*.

[0078] Após a quantização, o codificador de vídeo pode varrer os coeficientes de transformada, produzindo um vetor unidimensional a partir da matriz bidimensional, incluindo os coeficientes de transformada quantizados. A varredura pode ser projetada para colocar coeficientes de energia mais altos (e, por conseguinte, frequência mais baixa) na frente da matriz e colocar coeficientes de energia mais baixos (e, por conseguinte, frequência mais alta) na parte de trás da matriz. Em alguns exemplos, o codificador de vídeo 20 pode utilizar uma ordem de varredura predefinida para varrer os coeficientes de transformada quantizados para produzir um vetor serializado que pode ser codificado por entropia. Em outros exemplos, o codificador de vídeo 20 pode realizar uma varredura adaptativa. Após a varredura dos coeficientes de transformada quantizados para formar um vetor unidimensional, o codificador de vídeo 20 pode codificar por entropia o vetor unidimensional, por exemplo, de acordo com a codificação de comprimento variável adaptativa ao contexto (CAVLC), a codificação aritmética binária

adaptativa ao contexto (CABAC), a codificação aritmética binária adaptativa ao contexto baseada em sintaxe (SBAC), a Codificação por Entropia de Particionamento de Intervalo de probabilidade (PIPE) ou outra metodologia de codificação por entropia. O codificador de vídeo 20 pode também codificar por entropia elementos de sintaxe de codificação associados com os dados de vídeo codificados para utilização pelo decodificador de vídeo 30 na decodificação dos dados de vídeo.

[0079] Para realizar CABAC, o codificador de vídeo 20 pode atribuir um contexto dentro de um modelo de contexto a um símbolo a ser transmitido. O contexto pode estar relacionado, por exemplo, se os valores vizinhos do símbolo são diferentes de zero ou não. Para realizar CAVLC, o codificador de vídeo 20 pode selecionar um código de comprimento variável para um símbolo a ser transmitido. Palavras código no VLC podem ser construídas de tal modo que códigos relativamente mais curtos correspondam a símbolos mais prováveis, enquanto códigos mais longos correspondem a símbolos menos prováveis. Dessa maneira, o uso de VLC pode obter uma economia de bits, por exemplo, utilizando palavras código de comprimentos iguais para cada símbolo a ser transmitido. A determinação da probabilidade pode ser com base em um contexto atribuído ao símbolo.

[0080] Em geral, o decodificador de vídeo 30 realiza um processo substancialmente similar, embora recíproco, ao realizado pelo codificador de vídeo 20 para decodificar dados codificados. Por exemplo, o decodificador de vídeo 30 quantiza inversamente e transforma os coeficientes inversos de uma TU recebida para reproduzir um

bloco residual. O decodificador de vídeo 30 utiliza um modo de predição sinalizado (intra ou inter-predição) para formar um bloco predito. Então o decodificador de vídeo 30 combina o bloco predito e o bloco residual (em uma base pixel por pixel) para reproduzir o bloco original. Processos adicionais podem ser realizados, como realizar um processo de desbloqueio para reduzir artefatos visuais ao longo dos limites do bloco. Além disso, o decodificador de vídeo 30 pode decodificar elementos de sintaxe utilizando CABAC de uma maneira substancialmente similar a, embora recíproca ao processo de codificação CABAC do codificador de vídeo 20.

[0081] Em geral, o codificador de vídeo 20 e o decodificador de vídeo 30 podem ser configurados para codificar mais eficientemente (codificar ou decodificar, respectivamente) informação de movimento para predição afim, de acordo com as técnicas desta descrição. O codificador de vídeo 20 e/ou decodificador de vídeo 30 pode ser configurado para aplicar qualquer uma dentre as várias técnicas discutidas abaixo, isoladamente ou em qualquer combinação.

[0082] Em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem utilizar a diferença de vetor de movimento (MVD) de um vetor de movimento (MV) para predizer a MVD de outro MV em um bloco predito com predição afim. A MVD pode ser definida como a diferença entre o MV e a predição de vetor de movimento (MVP): MVD = MV-MVP. Mais especificamente, se um vetor de movimento (MV_x, MV_y) é indicado por sua componente horizontal (MV_x) e sua componente vertical (MV_y), e o

preditor de vetor de movimento possui componentes (MVP_x, MVP_y), a componente horizontal (vertical) de MVD é definida como a diferença das componentes horizontais (verticais) de MV e MVP, respectivamente. Assim, a MVD pode ser definida como (MVD_x, MVD_y), onde $MVD_x = MV_x - MVP_x$ e $MVD_y = MV_y - MVP_y$.

[0083] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para utilizar a MVD do primeiro MV para predizer a(s) MVD(s) de um ou outros mais MVs na predição afim. A FIG. 9 é um diagrama conceitual ilustrando um exemplo dessa predição de MVD. Em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para utilizar a MVD de um primeiro MV para predizer a MVD de um segundo MV em predição afim (por exemplo, afim de 4 parâmetros). A FIG. 9 abaixo mostra um exemplo de predição de MVD para predição afim com dois vetores de movimento, em que MVD1 é predita por MVD0.

[0084] A FIG. 10 é um diagrama conceitual ilustrando um exemplo de predição de MVD para predição afim com três vetores de movimento (predição afim de seis parâmetros). Para a predição afim de seis parâmetros, o codificador de vídeo 20 e/ou decodificador de vídeo 30 podem utilizar a MVD do primeiro MV para predizer a MVD do segundo MV. Além disso, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem utilizar a MVD do primeiro MV para predizer a MVD do terceiro MV em predição afim com três vetores de movimento. A FIG. 10 mostra um exemplo de predição de MVD para predição afim com três vetores de

movimento, em que MVD1 é predita por MVD0 e MVD2 também é predita por MVD0.

[0085] Fazendo referência novamente à FIG. 6, em alguns exemplos, o codificador de vídeo 20 e o decodificador de vídeo 30 podem ser configurados de modo que o primeiro MV nos exemplos acima seja definido como o MV associado com o ponto de controle superior esquerdo indicado como "MV0" nas FIGS. 3, 9 e 10. Alternativamente, o codificador de vídeo 20 e o decodificador de vídeo 30 podem ser configurados para selecionar adaptativamente o primeiro ponto de controle associado com o primeiro MV. Por exemplo, o primeiro ponto de controle pode depender de informações codificadas, tal como formato do bloco. Alternativamente, o codificador de vídeo 20 e o decodificador de vídeo 30 podem derivar implicitamente o primeiro ponto de controle associado com o primeiro MV.

[0086] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para aplicar a predição de MVD entre quaisquer duas MVDs para predição afim. Por exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem predizer MVD0 a partir de MVD1 para predição afim com dois vetores de movimento. Em outro exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem predizer MVD1 a partir de MVD0 e predizer MVD2 a partir de MVD1, para predição afim com três vetores de movimento.

[0087] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para

predizer MVDa a partir de MVDb. O codificador de vídeo 20 pode calcular MVDa' = MVDa-MVDb e informações de código representando MVDa' como parte do fluxo de bits, de modo que o decodificador de vídeo 30 possa decodificar essas informações para determinar MVDa'. O decodificador de vídeo 30 pode então calcular MVDa = MVDa'+ MVDb. Em um exemplo, a = 1 e b = 0 para predição afim com quatro parâmetros.

[0088] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para predizer MVDa a partir de MVDb. O codificador de vídeo 20 pode então calcular MVDa' = MVDa-w*MVDb e codificar informações representando MVDa' como parte do fluxo de bits, de tal modo que o decodificador de vídeo 30 possa decodificar essas informações para determinar MVDa'. O decodificador de vídeo 30 pode então calcular MVDa = MVDa'+w*MVDb. Neste exemplo, *w* é um valor de ponderação, tal como 0,5. Em um exemplo, a = 1 e b = 0 para predição afim com dois parâmetros. Este exemplo pode ser implementado em um formato de número inteiro, como MVD1' = MVD1-((MVD0+1)>>1) quando *w* = 0,5, ou MVD1' = MVD1-((MVD0+2)>>2) quando *w* = 0,25. Em um exemplo, o codificador de vídeo 20 codifica dados representando *w* como parte do fluxo de bits, por exemplo, em um nível de sequência (tal como um conjunto de parâmetros de sequência (SPS)), um nível de imagem (tal como um conjunto de parâmetros de imagem (PPS)), um nível de fatia (tal como em um cabeçalho de fatia) ou nível de bloco (tal como em um cabeçalho de bloco). O decodificador de vídeo 30 pode extrair adicionalmente esta informação sinalizada a partir do nível

correspondente de informação.

[0089] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para predizer a MVD de um ponto de controle a partir de MVDs de mais de um outro ponto de controle. Por exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem predizer MVD2 a partir de MVD0 e MVD1 no modelo afim de seis parâmetros. Como um exemplo, o codificador de vídeo 20 pode calcular MVD2' = MVD2-(MVD0+MVD1)>>1) e codificar as informações representando a MVD2' como parte do fluxo de bits, de tal modo que o decodificador de vídeo 30 possa decodificar essas informações para determinar a MVD2'. O decodificador de vídeo 30 pode então utilizar essas informações para calcular MVD2 = MVD2'+((MVD0+MVD1)>>1).

[0090] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para determinar se deve ou não predizer uma MVD com base em um valor de MVD. Por exemplo, o codificador de vídeo 20 pode determinar a predição de MVDa a partir de MVDb se $|MVDb^{x}+MVDb^{y}|<T$; caso contrário, o codificador de vídeo 20 não prediz MVDa a partir de MVDb. Em outro exemplo, o codificador de vídeo 20 pode determinar a predição de MVDa a partir de MVDb se $\max(|MVDb^{x}|, |MVDb^{y}|)<T$; caso contrário, o codificador de vídeo 20 não prediz MVDa a partir de MVDb. Ainda em outro exemplo, o codificador de vídeo 20 pode predizer o MVDa a partir de MVDb se MVDb if $|MVDb^{x}+MVDb^{y}|>T$; caso contrário, o codificador de vídeo 20 não prediz MVDa a partir de MVDb. Em ainda outro exemplo, o

codificador de vídeo 20 pode predizer o MVDa a partir de MVDb se $\min(|MVDb^x|, |MVDb^y|)>T$; caso contrário, o codificador de vídeo 20 não prediz MVDa a partir de MVDb. Nos exemplos acima, T representa um limite, que pode ser um número fixo ou sinalizado pelo codificador de vídeo 20 e decodificado pelo decodificador de vídeo 30. O codificador de vídeo 20 pode codificar dados representando se deve ou não predizer MVDa a partir de MVDb com base em qualquer uma dentre as determinações de exemplo acima, e o decodificador de vídeo 30 pode decodificar esses dados codificados para determinar se deve predizer a MVDa a partir de MVDb.

[0091] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para predizer as componentes horizontais(x) e/ou verticais y) de uma MVD de diferentes maneiras. Por exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem predizer apenas a componente x da MVDb a partir da componente x da MVDa, mas não predizer a componente y da MVDb a partir de outra MVD (por exemplo, MVDa).

[0092] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para determinar se deve predizer a MVD com base no valor do MVP. Por exemplo, o codificador de vídeo 20 pode determinar a predição de MVDa a partir de MVDb se $|MVPa^x-MVPb^x|+|MVPa^y-MVPb^y| < S$; caso contrário, o codificador de vídeo 20 não prediz MVDa a partir de MVDb. Em outro exemplo, o codificador de vídeo 20 pode determinar a predição de MVDa a partir de MVDb se $\max(||MVPa^x-MVPb^x|, |MVPa^y-MVPb^y|)<S$;

caso contrário, o codificador de vídeo 20 não prediz MVDa a partir de MVDb. Nos exemplos acima, S representa um limite, que pode ser um número fixo ou sinalizado pelo codificador de vídeo 20 e decodificado pelo decodificador de vídeo 30. O codificador de vídeo 20 pode codificar dados representando se deve ou não predizer MVDa a partir de MVDb com base em qualquer uma dentre as determinações de exemplo acima, e o decodificador de vídeo 30 pode decodificar esses dados codificados para determinar se deve predizer MVDa a partir de MVDb.

[0093] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para determinar se deve predizer uma MVD para um bloco predito utilizando predição afim com base em um método de predição de movimento. Por exemplo, se a MVP vier a partir do método de derivação MVP no JEM, conforme descrito acima em relação à FIG. 3, o codificador de vídeo 20 e o decodificador de vídeo 30 podem determinar não utilizar a predição de MVD. Como outro exemplo, se a MVP vier a partir do método de derivação do MVP similar à mesclagem afim, conforme descrito acima em relação ao Pedido Provisório Norte-Americano Nº 62/404.719, o codificador de vídeo 20 e o decodificador de vídeo 30 podem determinar a utilização de predição MVD. Adicionalmente ou alternativamente, o codificador de vídeo 20 e o decodificador de vídeo 30 podem determinar se deve utilizar a predição MVD com base em se a compensação de iluminação é utilizada para o bloco de origem do MVP.

[0094] Adicionalmente ou alternativamente, em

alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para determinar se deve predizer uma MVD com base no tamanho e/ou na forma de um bloco atual. Por exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem determinar o uso de predição MVD quando W*H>T, em que W representa uma largura do bloco atual, H representa uma altura do bloco atual e T representa um valor limite. T pode ser um número fixo ou sinalizado a partir do codificador de vídeo 20 para o decodificador de vídeo 30 no fluxo de bits. Em outro exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem determinar a utilização de predição MVD quando W*H<T.

[0095] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para codificar dados representando se deve ou não predizer uma MVD (ou uma componente da mesma) em um fluxo de bits. Ou seja, o codificador de vídeo 20 pode codificar dados representando a predição de uma ou de ambas as componentes (horizontal e vertical) de uma MVD no fluxo de bits, e o decodificador de vídeo 30 pode determinar se deve predizer uma ou ambas as componentes da MVD a partir dos dados codificados do fluxo de bits (pela decodificação dos dados).

[0096] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para codificar dados representando quais pontos de controle utilizar como referência para predição MVD no fluxo de

bits. Ou seja, o codificador de vídeo 20 pode codificar esses dados e o decodificador de vídeo 30 pode decodificar esses dados para determinar quais pontos de controle utilizar como referência para predição MVD.

[0097] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para gerar a MVP a partir de um MV a partir do MVD de outro MV em um bloco predito utilizando predição afim.

[0098] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para utilizar a MVD de um primeiro MV para gerar o(s) MVP(s) de um ou outros mais MVs em predição afim. Em um exemplo, o codificador de vídeo 20 e o decodificador de vídeo 30 podem utilizar a MVD do primeiro MV para gerar o MVP do segundo MV em predição afim (por exemplo, afim de quatro parâmetros). Em outro exemplo, para predição afim de seis parâmetros, o codificador de vídeo 20 e o decodificador de vídeo 30 podem utilizar a MVD do primeiro MV para gerar o MVP do segundo MV em predição afim com três vetores de movimento. Adicionalmente, o codificador de vídeo 20 e o decodificador de vídeo 30 podem utilizar a MVD do primeiro MV para gerar o MVP do terceiro MV em predição afim com três vetores de movimento. Alternativamente, o codificador de vídeo 20 e o decodificador de vídeo 30 podem utilizar a MVD do segundo MV para gerar o MVP do terceiro MV em predição afim com três vetores de movimento.

[0099] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o

decodificador de vídeo 30 podem ser configurados para gerar MVPO a partir de MVD1 para predição afim com dois vetores de movimento. Em outro exemplo, o codificador de vídeo 20 e/ou decodificador de vídeo 30 podem ser configurados para gerar MVP1 a partir de MVD0 e para gerar MVP2 a partir de MVD1, para predição afim com três vetores de movimento.

[0100] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para calcular MVPa de acordo com MVPa = MVP'a + MVDb. O MVP'a representa o MVP gerado na forma original sem considerar a MVDb conforme descrito acima para o modo AMVP e o modo de mesclagem para predição afim. Em um exemplo, a = 1 e b = 0 para predição afim com quatro parâmetros (dois vetores de movimento).

[0101] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para calcular o MVPa de acordo com MVPa = MVP'a + *w**MVDb, em que *w* é um valor de ponderação, tal como 0,5. Neste exemplo, MVP'a é o MVP gerado na forma original sem considerar a MVDb conforme descrito acima para o modo AMVP e modo de mesclagem para predição afim. Em um exemplo, a = 1 e b = 0 para predição afim com dois parâmetros. Este exemplo pode ser implementado em um formato inteiro, como MVP1 = MVP'1 + ((MVD0 + 1) >> 1) quando *w* = 0,5, ou MVP1 = MVP'1 + ((MVD0 + 2) >> 2) quando *w* = 0,25. Em um exemplo, o codificador de vídeo 20 determina *w* e sinaliza o valor de *w* no fluxo de bits no nível de sequência, no nível de imagem, no nível de fatia ou no nível de bloco. O decodificador de vídeo 30,

portanto, decodifica o valor de *w* a partir do nível apropriado de informação.

[0102] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para gerar o MVP de um ponto de controle a partir de MVDs de múltiplos outros pontos de controle. Por exemplo, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem gerar MVP2 a partir de MVD0 e MVD1 no modelo afim de seis parâmetros. O codificador de vídeo 20 e/ou decodificador de vídeo 30 podem calcular MVP2 como MVP2 = MVP'2 + ((MVD0 + MVD1) >> 1).

[0103] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para utilizar o MV de um ponto de controle para gerar o(s) MVP(s) do(s) MV(s) de um ou outros mais pontos de controle em um bloco predito utilizando predição afim. Em um exemplo, o codificador de vídeo 20 e/ou decodificador de vídeo 30 podem utilizar o primeiro MV para gerar o MVP do segundo MV em predição afim (por exemplo, afim de quatro parâmetros). Em outro exemplo, para a predição afim de seis parâmetros, o codificador de vídeo 20 e/ou decodificador de vídeo 30 podem utilizar o primeiro MV para gerar o MVP do segundo MV em predição afim com três vetores de movimento e utilizar o primeiro MV para gerar o MVP do terceiro MV em predição afim com três vetores de movimento. Alternativamente, para a predição afim de seis parâmetros, o codificador 20 e/ou decodificador de vídeo 30 podem utilizar o segundo MV para gerar o MVP do terceiro MV em

predição afim com três vetores de movimento.

[0104] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para calcular MVPa como MVPa = (MVP'a + MVb) >> 1. MVP'a é o MVP gerado na forma original, sem considerar o MVb conforme descrito acima nas seções discutindo AMVP e mesclagem para predição afim. Em um exemplo, a = 1 e b = 0 para predição afim com quatro parâmetros.

[0105] Adicionalmente ou alternativamente, em alguns exemplos, o codificador de vídeo 20 e/ou o decodificador de vídeo 30 podem ser configurados para calcular MVPa como MVPa = $w1$ * MVP'a + $w2$ * MVb. Neste exemplo, w1 e w2 são valores de ponderação que podem ter valores iguais ou diferentes, por exemplo, w1 = w2 = 0,5. MVP'a é o MVP gerado na forma original, sem considerar MVb, conforme descrito nas seções acima no modo AMVP e no modo de mesclagem para predição afim. Em um exemplo, a = 1 e b = 0 para predição afim com quatro parâmetros. Este exemplo pode ser implementado em um formato de número inteiro, como MVP1 = (3 * MVP'1 + MV0 + 2) >> 2 quando w1 = 0,75 e w2 = 0,25. Em um exemplo, o codificador de vídeo 20 codifica dados para w1 e para w2 no fluxo de bits em qualquer nível de sequência, nível de imagem, nível de fatia ou nível de bloco. Da mesma forma, o decodificador de vídeo 30 determinaria w1 e w2 decodificando esses dados no nível apropriado.

[0106] O codificador de vídeo 20 pode adicionalmente enviar dados de sintaxe, tais como dados de sintaxe com base em bloco, dados de sintaxe com base em

imagem e dados de sintaxe com base em sequência, para o decodificador de vídeo 30, por exemplo, em um cabeçalho de imagem, em um cabeçalho de bloco, em um cabeçalho de fatia, ou outros dados de sintaxe, tais como um conjunto de parâmetros de sequência (SPS), um conjunto de parâmetros de imagem (PPS) ou um conjunto de parâmetros de vídeo (VPS).

[0107] O codificador de vídeo 20 e o decodificador de vídeo 30 cada um pode ser implementado como qualquer um dentre uma variedade de conjuntos de circuitos de codificador ou decodificador adequados, conforme aplicável, tais como um ou mais microprocessadores, processadores de sinal digital (DSPs), circuitos integrados de aplicação específica (ASICs), arranjos de portas programáveis em campo (FPGAs), circuitos lógicos discretos, software, hardware, firmware ou qualquer combinação dos mesmos. Cada um dos codificadores de vídeo 20 e decodificadores de vídeo 30 pode ser incluído em um ou mais codificadores ou decodificadores, qualquer um dos quais pode ser integrado como parte de um codificador/decodificador de vídeo combinado (CODEC). Um dispositivo incluindo codificador de vídeo 20 e/ou decodificador de vídeo 30 pode compreender um circuito integrado, um microprocessador e/ou um dispositivo de comunicação sem fio, tal como um telefone celular.

[0108] A FIG. 7 é um diagrama de blocos ilustrando um exemplo de codificador de vídeo 20 que pode implementar técnicas desta descrição para codificar informação de movimento de predição afim. O codificador de vídeo 20 pode realizar intra e inter-codificação de blocos de vídeo dentro de fatias de vídeo. Intra-codificação

depende da predição espacial para reduzir ou remover a redundância espacial no vídeo em um determinado quadro ou imagem. Inter-codificação depende da predição temporal para reduzir ou remover a redundância temporal no vídeo em quadros adjacentes ou imagens de uma sequência de vídeo. Intra-modo (modo I) pode se referir a qualquer um dentre os vários modos de codificação com base espacial. Inter-modos, tal como predição unidirecional (modo P) ou bi-predição (modo B), podem se referir a qualquer um dos vários modos de codificação com base temporal.

[0109] Conforme mostrado na FIG. 7, o codificador de vídeo 20 recebe um bloco de vídeo atual dentro de um quadro de vídeo para ser codificado. No exemplo da FIG. 7, o codificador de vídeo 20 inclui a unidade de seleção de modo 40, a memória de imagem de referência 64 (que pode ser referida também como um buffer de imagem decodificada (DPB)), um somador (summer) 50, uma unidade de processamento de transformada 52, uma unidade de quantização 54 e uma unidade de codificação por entropia 56. A unidade de seleção de modo 40, por sua vez, inclui unidade de compensação de movimento 44, unidade de estimação de movimento 42, unidade de intra-predição 46 e unidade de partição 48. Para reconstrução de bloco de vídeo, o codificador de vídeo 20 inclui também a unidade de quantização inversa 58, a unidade de transformada inversa 60 e o somador 62. Um filtro de desbloqueio (não mostrado na FIG. 7) pode também ser incluído para filtrar os limites do bloco para remover artefatos de bloqueio a partir do vídeo reconstruído. Se desejado, o filtro de desbloqueio normalmente filtraria a saída do somador 62. Filtros

adicionais (em loop ou pós-loop) podem também ser utilizados além do filtro de desbloqueio. Esses filtros não são mostrados por questões de brevidade, mas, se desejado, podem filtrar a saída do somador 50 (como um filtro em loop).

[0110] Durante o processo de codificação, o codificador de vídeo 20 recebe um quadro ou fatia de vídeo para ser codificado. O quadro ou fatia pode ser dividido em múltiplos blocos de vídeo. A unidade de estimação de movimento 42 e a unidade de compensação de movimento 44 realizam a codificação inter-preditiva do bloco de vídeo recebido em relação a um ou mais blocos em um ou mais quadros de referência para fornecer predição temporal. A unidade intra-predição 46 pode, alternativamente, realizar codificação intra-preditiva do bloco de vídeo recebido em relação a um ou mais blocos vizinhos no mesmo quadro ou fatia que o bloco a ser codificado para fornecer predição espacial. O codificador de vídeo 20 pode realizar múltiplas passagens de codificação, por exemplo, para selecionar um modo de codificação apropriado para cada bloco de dados de vídeo.

[0111] Além disso, a unidade de partição 48 pode particionar blocos de dados de vídeo em sub-blocos, com base na avaliação de esquemas de particionamento anteriores em passagens de codificação anteriores. Por exemplo, a unidade de partição 48 pode inicialmente particionar um quadro ou uma fatia em CTUs e particionar cada uma das CTUs em sub-CUs com base na análise de taxa de distorção (por exemplo, otimização de taxa de distorção). A unidade de seleção de modo 40 pode adicionalmente produzir uma

estrutura de dados quadtree, indicativa de particionamento de uma CTU em sub-CUs. As CUs do nó folha da quadtree podem incluir uma ou mais PUs e uma ou mais TUs.

[0112] A unidade de seleção de modo 40 pode selecionar um entre os modos de predição, intra ou inter, por exemplo, com base em resultados de erro, e fornecer o bloco predito resultante para o somador 50 para gerar dados residuais e para o somador 62 para reconstruir o bloco codificado para utilização como quadro de referência. A unidade de seleção de modo 40 também fornece elementos de sintaxe, tais como vetores de movimento, indicadores intra-modo, informações de partição e outras informações de sintaxe, para a unidade de codificação por entropia 56.

[0113] A unidade de estimação de movimento 42 e a unidade de compensação de movimento 44 podem ser altamente integradas, mas são ilustradas separadamente para fins conceituais. A estimação de movimento, realizada pela unidade de estimação de movimento 42, é o processo de gerar vetores de movimento, que estimam o movimento para blocos de vídeo. Um vetor de movimento, por exemplo, pode indicar o deslocamento de uma PU de um bloco de vídeo dentro de um quadro ou imagem de vídeo atual em relação a um bloco preditivo dentro de um quadro de referência (ou outra unidade codificada) relativo ao bloco atual sendo codificado dentro de quadro atual (ou outra unidade codificada). Um bloco preditivo é um bloco que é encontrado para corresponder muito próximo do bloco a ser codificado, em termos de diferença de pixels, que pode ser determinada pela soma da diferença absoluta (SAD), pela soma da diferença quadrada (SSD) ou por outras métricas de

diferença. Em alguns exemplos, o codificador de vídeo 20 pode calcular valores para posições de pixels sub-inteiro de imagens de referência armazenadas na memória de imagem de referência 64. Por exemplo, o codificador de vídeo 20 pode interpolar valores de posições de um quarto de pixel, posições de um oitavo pixel ou outras posições de pixel fracionárias da imagem de referência. Portanto, a unidade de estimativa de movimento 42 pode realizar uma pesquisa de movimento em relação às posições completas de pixel e posições fracionárias de pixel e gerar um vetor de movimento com precisão fracionária de pixel.

[0114] A unidade de estimação de movimento 42 calcula um vetor de movimento para uma PU de um bloco de vídeo em uma fatia inter-codificada comparando a posição da PU com a posição de um bloco preditivo de uma imagem de referência. A imagem de referência pode ser selecionada a partir de uma primeira lista de imagens de referência (Lista 0) ou de uma segunda lista de imagens de referência (Lista 1), cada uma das quais identifica uma ou mais imagens de referência armazenadas na memória de imagens de referência 64. A unidade de estimação de movimento 42 envia o vetor de movimento calculado para a unidade de codificação por entropia 56 e para a unidade de compensação de movimento 44.

[0115] A compensação de movimento, realizada pela unidade de compensação de movimento 44, pode envolver buscar ou gerar o bloco preditivo com base no vetor de movimento determinado pela unidade de estimação de movimento 42. Novamente, a unidade de estimação de movimento 42 e a unidade de compensação de movimento 44

podem ser funcionalmente integradas, em alguns exemplos. Depois de receber o vetor de movimento para a PU do bloco de vídeo atual, a unidade de compensação de movimento 44 pode localizar o bloco preditivo para o qual o vetor de movimento aponta em uma das listas de imagens de referência. O somador 50 forma um bloco de vídeo residual subtraindo os valores de pixel do bloco preditivo dos valores de pixel do bloco de vídeo atual sendo codificado, formando valores de diferença de pixel, conforme discutido abaixo. Em geral, a unidade de estimação de movimento 42 realiza a estimação de movimento em relação às componentes luma, e a unidade de compensação de movimento 44 utiliza vetores de movimento calculados com base nas componentes luma tanto para componentes croma quanto para componentes luma. A unidade de compensação de movimento 44 pode gerar um bloco de predição utilizando os vetores de movimento, que pode incluir valores de interpolação ou de outro modo manipulação matemática dos blocos preditivos referidos pelos vetores de movimento. A unidade de seleção de modo 40 pode também gerar elementos de sintaxe associados com os blocos de vídeo e com a fatia de vídeo para utilização pelo decodificador de vídeo 30 na decodificação dos blocos de vídeo da fatia de vídeo.

[0116] O codificador de vídeo 20 pode ser configurado para realizar qualquer uma dentre as várias técnicas desta descrição discutidas acima em relação à FIG. 6. Por exemplo, a unidade de compensação de movimento 44 pode ser configurada para codificar informação de movimento para um bloco de dados de vídeo utilizando o modo AMVP ou modo de mesclagem de acordo com HEVC e/ou pode ser

configurado para codificar informação de movimento afim ou um bloco de dados de vídeo utilizando inter-modo afim ou modo de mesclagem afim de acordo com as técnicas desta descrição.

[0117] A unidade de intra-predição 46 pode intra-predizer um bloco atual, como uma alternativa para a inter-predição realizada pela unidade de estimação de movimento 42 e pela unidade de compensação de movimento 44, conforme descrito acima. Em particular, a unidade de intra-predição 46 pode determinar um modo de intra-predição para ser utilizado para codificar um bloco atual. Em alguns exemplos, a unidade de intra-predição 46 pode codificar um bloco atual utilizando vários modos de intra-predição, por exemplo, durante passagens de codificação separadas, e a unidade intra-predição 46 (ou a unidade de seleção de modo 40, em alguns exemplos) pode selecionar um modo de predição para utilizar nos modos testados.

[0118] Por exemplo, a unidade intra-predição 46 pode calcular valores de taxa de distorção utilizando uma análise de taxa de distorção para os vários modos de predição testados e selecionar o modo de predição com as melhores características de taxa de distorção entre os modos testados. A análise de taxa de distorção geralmente determina uma quantidade de distorção (ou erro) entre um bloco codificado e um bloco não codificado original que foi codificado para produzir o bloco codificado, bem como uma taxa de bits (ou seja, um número de bits) utilizada para produzir o bloco codificado. A unidade de intra-predição 46 pode calcular relações a partir das distorções e das taxas para os vários blocos codificados para determinar qual modo

de intra-predição exibe o melhor valor de taxa de distorção para o bloco.

[0119] Depois de selecionar um modo de intra-predição para um bloco, a unidade de intra-predição 46 pode fornecer informação indicativa do modo de intra-predição selecionado para o bloco para a unidade de codificação por entropia 56. A unidade de codificação por entropia 56 pode codificar a informação indicando o modo de intra-predição selecionado. O codificador de vídeo 20 pode incluir no fluxo de bits transmitido os dados de configuração, que podem incluir uma pluralidade de tabelas de índice de modo de intra-predição e uma pluralidade de tabelas de índice de modo intra-predição modificada (referidas também como tabelas de mapeamento de palavras código), definições de contextos de codificação para vários blocos e indicações de um modo de intra-predição mais provável, uma tabela de índice de modo de intra-predição e uma tabela de índice de modo de intra-predição modificada para utilização em cada um dos contextos.

[0120] O codificador de vídeo 20 forma um bloco de vídeo residual subtraindo os dados de predição a partir da unidade de seleção de modo 40 do bloco de vídeo original que está sendo codificado. O somador 50 representa o componente ou os componentes que realizam essa operação de subtração. A unidade de processamento de transformada 52 aplica uma transformada, tal como uma transformada discreta de cosseno (DCT) ou uma transformada conceitualmente similar, ao bloco residual, produzindo um bloco de vídeo compreendendo valores de coeficiente de transformada. Transformadas Wavelet, transformadas de número inteiro,

transformadas de sub-banda, transformadas discretas de seno (DSTs) ou outros tipos de transformadas podem ser utilizados em vez de uma DCT. Em qualquer caso, a unidade de processamento de transformada 52 aplica a transformada ao bloco residual, produzindo um bloco de coeficientes de transformada. A transformada pode converter a informação residual a partir de um domínio de pixel para um domínio de transformada, tal como um domínio de frequência. A unidade de processamento de transformada 52 pode enviar os coeficientes de transformada resultantes para a unidade de quantização 54. A unidade de quantização 54 quantiza os coeficientes de transformada para reduzir ainda mais a taxa de bits. O processo de quantização pode reduzir a profundidade de bits associados com alguns ou com todos os coeficientes. O grau de quantização pode ser modificado ajustando um parâmetro de quantização.

[0121] Em seguida à quantização, a unidade de codificação por entropia 56 codifica por entropia os coeficientes de transformada quantizados. Por exemplo, a unidade de codificação por entropia 56 pode realizar codificação de comprimento variável adaptativa ao contexto (CAVLC), codificação aritmética binária adaptativa ao contexto (CABAC), codificação aritmética binária adaptativa ao contexto baseada em sintaxe (SBAC), codificação por entropia de particionamento de intervalo de probabilidade (PIPE) ou outra técnica de codificação por entropia. No caso da codificação por entropia baseada em contexto, o contexto pode ser baseado em blocos vizinhos. Em seguida à codificação por entropia pela unidade de codificação por entropia 56, o fluxo de bits codificado pode ser

transmitido para outro dispositivo (por exemplo, decodificador de vídeo 30) ou arquivado para transmissão ou recuperação posterior.

[0122] A unidade de quantização inversa 58 e a unidade de transformada inversa 60 aplicam quantização inversa e transformada inversa, respectivamente, para reconstruir o bloco residual no domínio de pixel. Em particular, o somador 62 adiciona o bloco residual reconstruído ao bloco de predição compensado por movimento produzido anteriormente pela unidade de compensação de movimento 44 ou pela unidade de intra-predição 46 para produzir um bloco de vídeo reconstruído para armazenamento na memória de imagem de referência 64. O bloco de vídeo reconstruído pode ser utilizado pela unidade de estimação de movimento 42 e pela unidade de compensação de movimento 44 como um bloco de referência para inter-codificar um bloco em um quadro de vídeo subsequente.

[0123] A FIG. 8 é um diagrama de blocos ilustrando um exemplo de decodificador de vídeo 30 que pode implementar técnicas desta descrição para decodificar informação de movimento de predição afim. No exemplo da FIG. 8, o decodificador de vídeo 30 inclui uma unidade de decodificação por entropia 70, uma unidade de compensação de movimento 72, uma unidade de intra-predição 74, uma unidade de quantização inversa 76, uma unidade de transformada inversa 78, uma memória de imagem de referência 82 e um somador 80. O decodificador de vídeo 30 pode, em alguns exemplos, realizar uma passagem de decodificação geralmente recíproca à passagem de codificação descrita em relação ao codificador de vídeo 20

(FIG. 7). A unidade de compensação de movimento 72 pode gerar dados de predição com base em vetores de movimento recebidos a partir da unidade de decodificação por entropia 70, enquanto a unidade de intra-predição 74 pode gerar dados de predição com base em indicadores de modo de intra-predição recebidos a partir da unidade de decodificação por entropia 70.

[0124] Durante o processo de decodificação, o decodificador de vídeo 30 recebe um fluxo de bits de vídeo codificado que representa blocos de vídeo de uma fatia de vídeo codificada e elementos de sintaxe associados a partir do codificador de vídeo 20. A unidade de decodificação por entropia 70 do decodificador de vídeo 30 decodifica por entropia o fluxo de bits para gerar coeficientes quantizados, vetores de movimento ou indicadores de modo de intra-predição e outros elementos de sintaxe. A unidade de decodificação por entropia 70 encaminha os vetores de movimento e outros elementos de sintaxe para a unidade de compensação de movimento 72. O decodificador de vídeo 30 pode receber os elementos de sintaxe no nível de fatia de vídeo e/ou no nível de bloco de vídeo.

[0125] Quando a fatia de vídeo é codificada como uma fatia intra-codificada (I), a unidade de intra-predição 74 pode gerar dados de predição para um bloco de vídeo da fatia de vídeo atual com base em um modo de predição sinalizado e dados a partir de blocos decodificados anteriormente do quadro ou da imagem atuais. Quando o quadro de vídeo é codificado como uma fatia inter-codificada (isto é, B ou P), a unidade de compensação de movimento 72 produz blocos preditivos para um bloco de

vídeo da fatia de vídeo atual com base nos vetores de movimento e outros elementos de sintaxe recebidos a partir da unidade de decodificação por entropia 70. Os blocos preditivos podem ser produzidos a partir de uma dentre as imagens de referência dentro de uma das listas de imagens de referência. O decodificador de vídeo 30 pode construir as listas de quadros de referência, Lista 0 e Lista 1, utilizando técnicas de construção padrão com base em imagens de referência armazenadas na memória de imagem de referência 82. A unidade de compensação de movimento 72 determina a informação de predição para um bloco de vídeo da fatia de vídeo atual analisando os vetores de movimento e outros elementos de sintaxe e utiliza a informação de predição para produzir os blocos preditivos para o bloco de vídeo atual que está sendo decodificado. Por exemplo, a unidade de compensação de movimento 72 utiliza alguns dos elementos de sintaxe recebidos para determinar um modo de predição (por exemplo, intra ou inter-predição) utilizado para codificar os blocos de vídeo da fatia de vídeo, um tipo de fatia inter-predição (por exemplo, fatia B ou fatia P), informação de construção para uma ou mais listas de imagens de referência para a fatia, vetores de movimento para cada bloco de vídeo codificado da fatia, status de inter-predição para cada bloco de vídeo codificado da fatia e outras informações para decodificar os blocos de vídeo na fatia de vídeo atual.

[0126] O decodificador de vídeo 30 pode ser configurado para realizar qualquer uma dentre as várias técnicas desta descrição discutidas acima em relação à FIG. 6. Por exemplo, a unidade de compensação movimento 72 pode

ser configurada para realizar a predição de vetor de movimento utilizando o modo AMVP ou modo de mesclagem de acordo com o HEVC e/ou pode ser configurado para realizar informação de movimento afim ou um bloco de dados de vídeo utilizando inter-modo ou modo de mesclagem afim de acordo com as técnicas desta descrição. A unidade de decodificação por entropia 70 pode decodificar um ou mais elementos de sintaxe representando como a informação de movimento (por exemplo, informação de movimento afim) é codificada para o bloco atual.

[0127] A unidade de compensação de movimento 72 pode também realizar interpolação com base em filtros de interpolação. A unidade de compensação de movimento 72 pode utilizar filtros de interpolação como utilizado pelo codificador de vídeo 20 durante a codificação dos blocos de vídeo para calcular valores interpolados para pixels sub-inteiros dos blocos de referência. Neste caso, a unidade de compensação de movimento 72 pode determinar os filtros de interpolação utilizados pelo codificador de vídeo 20 a partir dos elementos de sintaxe recebidos e utilizar os filtros de interpolação para produzir blocos preditivos.

[0128] A unidade de quantização inversa 76 quantiza inversamente, isto é, dequantiza os coeficientes de transformada quantizados fornecidos no fluxo de bits e decodificados pela unidade de decodificação por entropia 70. O processo de quantização inversa pode incluir a utilização de um parâmetro de quantização QP_Y calculado pelo decodificador de vídeo 30 para cada bloco de vídeo na fatia de vídeo para determinar um grau de quantização e, da mesma forma, um grau de quantização inversa que deve ser

aplicado.

[0129] A unidade de transformada inversa 78 aplica uma transformada inversa, por exemplo, uma DCT inversa, uma transformada de número inteiro inversa ou um processo de transformada inversa conceitualmente similar, aos coeficientes de transformada, a fim de produzir blocos residuais no domínio de pixel.

[0130] Depois que a unidade de compensação de movimento 72 gera o bloco preditivo para o bloco de vídeo atual com base nos vetores de movimento e outros elementos de sintaxe, o decodificador de vídeo 30 forma um bloco de vídeo decodificado pela soma dos blocos residuais da unidade de transformada inversa 78 com os blocos preditivos correspondentes gerados pela unidade de compensação de movimento 72. O somador 80 representa o componente ou os componentes que realizam essa operação de soma. Se desejado, um filtro de desbloqueio pode também ser aplicado para filtrar os blocos decodificados, com a finalidade de remover artefatos de bloqueio. Outros filtros de loop (ou no loop de codificação ou após o loop de codificação) podem também ser utilizados para suavizar as transições de pixel ou melhorar a qualidade de vídeo. Os blocos de vídeo decodificados em um determinado quadro ou imagem são então armazenados na memória de imagem de referência 82, que armazena imagens de referência utilizadas para a compensação de movimento subsequente. A memória de imagem de referência 82 armazena também vídeo decodificado para apresentação posterior em um dispositivo de exibição, tal como o dispositivo de exibição 32 da FIG. 6.

[0131] A FIG. 11 é um fluxograma ilustrando um

método de exemplo para codificar um bloco de dados de vídeo atual de acordo com as técnicas desta descrição. Para fins de exemplo e explicação, o método da FIG. 11 é explicado em relação ao codificador de vídeo 20 das FIGS. 6 e 7. No entanto, deve-se entender que outros dispositivos podem ser configurados para realizar este método ou um método similar.

[0132] Inicialmente, embora não mostrado na FIG. 11, a unidade de seleção de modo 40 pode determinar um modo de predição a ser utilizado para predizer um bloco atual. Neste exemplo, é assumido que a unidade de seleção de modo 40 selecione um modo de predição afim, que inclui a predição utilizando pelo menos dois vetores de movimento. Assim, a unidade de seleção de modo 40 faz com que a unidade de estimação de movimento 42 realize uma pesquisa de movimento para determinar um primeiro vetor de movimento e um segundo vetor de movimento (100). A unidade de seleção de modo 40 pode comparar os resultados de taxa de distorção entre uma variedade de métodos de predição, tal como intra-predição, inter-predição e predição afim, e determinar que a predição afim resulta nos melhores resultados de taxa de distorção entre os vários modos de predição testados.

[0133] Depois que a unidade de estimativa de movimento 42 determina o primeiro e o segundo vetores de movimento, o codificador de vídeo 20 pode calcular um primeiro preditor de vetor de movimento (MVP) (102) para o primeiro vetor de movimento. O primeiro MVP pode corresponder a um vetor de movimento de um bloco vizinho. O codificador de vídeo 20 pode então calcular uma primeira diferença de vetor de movimento (MVD) como a diferença

entre o primeiro vetor de movimento e o primeiro preditor de vetor de movimento (104). Em particular, o codificador de vídeo 20 pode calcular diferenças entre as componentes x e y do vetor de movimento e a MVD, respectivamente.

[0134] O codificador de vídeo 20 pode então determinar um segundo MVP (106) para o segundo vetor de movimento. O codificador de vídeo 20 pode adicionalmente calcular uma segunda MVD (108) como a diferença entre o segundo vetor de movimento e a segunda MVD.

[0135] A unidade de compensação de movimento 44 também pode predizer o bloco atual (110) utilizando predição afim para gerar um bloco de predição para o bloco atual. Embora dois vetores de movimento tenham sido discutidos com a finalidade de exemplo, deve-se entender que três vetores de movimento podem ser utilizados para predição afim para gerar o bloco de predição. Da mesma forma, o codificador de vídeo 20 pode gerar uma terceira MVD e um terceiro MVP de acordo com as técnicas desta descrição para um terceiro vetor de movimento, conforme discutido acima.

[0136] Depois de gerar o bloco de predição, o codificador de vídeo 20 pode calcular um bloco residual (112) representando diferenças de pixel por pixel entre o bloco atual e o bloco de predição. Em particular, o somador 50 pode calcular as diferenças de pixel por pixel entre o bloco atual e o bloco de predição. O codificador de vídeo 20 pode então codificar a primeira MVD, a segunda MVD e o bloco residual (114) para codificar o bloco atual. Ou seja, de acordo com as técnicas desta descrição, o codificador de vídeo 20 pode codificar a primeira MVD utilizando, por

exemplo, técnicas de predição de vetor de movimento, tal como modo de mesclagem ou modo AMVP, e codificar a segunda MVD, prevendo a segunda MVD a partir da primeira MVD. Assim, para codificar a segunda MVD, o codificador de vídeo 20 pode codificar dados representativos de uma diferença entre a primeira MVD e a segunda MVD, como diferenças entre as componentes x e y da primeira MVD e da segunda MVD. Para codificar o bloco residual, a unidade de processamento de transformada 52 pode transformar o bloco residual, a unidade de quantização 54 pode quantizar os coeficientes de transformada do bloco de transformada resultante e a unidade de codificação por entropia 56 pode codificar por entropia os coeficientes de transformada quantizados resultantes.

[0137] Desta maneira, o método da FIG. 11 representa um exemplo de um método que inclui codificar/decodificar (ou seja, codificar) uma primeira diferença de vetor de movimento (MVD) que representa uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento; predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual; e codificar/decodificar (ou seja, codificar) bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento. Realizando o método da FIG. 11, o codificador de vídeo 20 pode gerar um fluxo de bits que é mais eficiente em largura de banda do que quando realiza as técnicas anteriores, porque os dados representando a segunda MVD

podem ser menores porque a segunda MVD é predita a partir da primeira MVD.

[0138] A FIG. 12 é um fluxograma ilustrando um método de exemplo de decodificação de um bloco de dados de vídeo atual de acordo com as técnicas desta descrição. O método da FIG. 12 é explicado em relação ao decodificador de vídeo 30 das FIGS. 6 e 8 para fins de exemplo. No entanto, deve ser entendido que outros dispositivos podem ser configurados para realizar as técnicas deste método ou de um método similar.

[0139] O decodificador de vídeo 30 pode decodificar uma primeira diferença de vetor de movimento (MVD), uma segunda MVD e um bloco residual (120) de um bloco atual. Ou seja, de acordo com as técnicas desta descrição, o decodificador de vídeo 30 pode decodificar a primeira MVD utilizando, por exemplo, técnicas de predição de vetor de movimento, tais como modo de mesclagem ou modo AMVP, e decodificar a segunda MVD, prevendo a segunda MVD a partir da primeira MVD. Assim, para decodificar a segunda MVD, o decodificador de vídeo 30 pode decodificar dados representativos de uma diferença entre a primeira MVD e a segunda MVD, tais como diferenças entre as componentes x e y da primeira MVD e da segunda MVD. Para decodificar o bloco residual, a unidade de decodificação por entropia 70 pode decodificar coeficientes de transformada quantizados, a unidade de quantização inversa 76 pode quantizar inversamente os coeficientes de transformada e a unidade de transformada inversa 78 pode transformar inversamente os coeficientes de transformada para reproduzir o bloco residual.

[0140] A unidade de compensação de movimento 72 pode então determinar um primeiro preditor de vetor de movimento (MVP) (122) para um primeiro vetor de movimento do bloco atual e calcular o primeiro vetor de movimento (124) a partir do primeiro MVP. Em particular, a unidade de compensação de movimento 72 pode adicionar a primeira MVD ao primeiro MVP para calcular o primeiro vetor de movimento. A unidade de compensação de movimento 72 pode determinar de forma similar um segundo MVP (126) para um segundo vetor de movimento do bloco atual e calcular o segundo vetor de movimento (128) a partir do segundo MVP. Em particular, a unidade de compensação de movimento 72 pode adicionar a segunda MVD ao segundo MVP para calcular o segundo vetor de movimento. Em alguns exemplos, um terceiro vetor de movimento pode ser incluído, caso em que a unidade de decodificação por entropia 70 pode decodificar por entropia dados representando uma diferença entre, por exemplo, a primeira MVD e uma terceira MVD para o terceiro vetor de movimento, e a unidade de compensação de movimento 72 pode calcular o terceiro vetor de movimento a partir da terceira MVD e um terceiro MVP de maneira similar.

[0141] A unidade de compensação de movimento 72 pode então predizer o bloco atual (130), por exemplo, de acordo com a predição de movimento afim utilizando o primeiro e o segundo (e potencialmente o terceiro) vetores de movimento. O decodificador de vídeo 30 pode então decodificar o bloco atual (132), por exemplo, fazendo com que o somador 80 adicione os valores do bloco de predição aos valores do bloco residual em uma base pixel por pixel.

[0142] Desta maneira, o método da FIG. 12

representa um exemplo de um método que inclui codificar/decodificar (ou seja, decodificar) uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento; predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual; e codificar/decodificar (ou seja, decodificar) bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e o segundo vetor de movimento. Realizando o método da FIG. 12, o decodificador de vídeo 30 pode decodificar um fluxo de bits que seja mais eficiente em largura de banda do que na realização de técnicas anteriores, porque os dados representando a segunda MVD podem ser menores porque a segunda MVD é predita a partir da primeira MVD.

[0143] Deve ser reconhecido que, dependendo do exemplo, certos atos ou eventos de qualquer uma das técnicas descritas neste documento podem ser realizados em uma sequência diferente, podem ser adicionados, mesclados ou deixados de fora (por exemplo, nem todos os atos ou eventos descritos são necessário para a prática das técnicas). Além disso, em certos exemplos, atos ou eventos podem ser realizados simultaneamente, por exemplo, através de processamento multitarefa (multi-threaded), processamento de interrupção ou múltiplos processadores, em vez de sequencialmente.

[0144] Em um ou mais exemplos, as funções descritas podem ser implementadas em hardware, software,

firmware ou qualquer combinação dos mesmos. Se implementadas em software, as funções podem ser armazenadas ou transmitidas como uma ou mais instruções ou código em um meio legível por computador e executadas por uma unidade de processamento baseada em hardware. O meio legível por computador pode compreender o meio de armazenamento legível por computador, que corresponde a um meio tangível tal como um meio de armazenamento de dados, ou um meio de comunicação, incluindo qualquer meio que facilita a transferência de um programa de computador de um lugar para outro, por exemplo, de acordo com um protocolo de comunicação. Desta forma, o meio legível por computador geralmente pode corresponder a (1) um meio de armazenamento legível por computador tangível que é não transitório ou (2) um meio de comunicação tal como um sinal ou uma onda portadora. O meio de armazenamento de dados pode ser qualquer meio disponível que pode ser acessado por um ou mais computadores ou um ou mais processadores para recuperar instruções, códigos e/ou estruturas de dados para a implementação das técnicas apresentadas nesta descrição. Um produto de programa de computador pode incluir um meio legível por computador.

[0145] A título de exemplo e não de limitação, tal meio de armazenamento legível por computador pode incluir RAM, ROM, EEPROM, CD-ROM ou outro armazenamento em disco ótico, armazenamento em disco magnético, ou outros dispositivos de armazenamento magnéticos, memória flash, ou qualquer outro meio que pode ser utilizado para armazenar o código do programa desejado na forma de instruções ou estruturas de dados e que pode ser acessado por um

computador. Ainda, qualquer conexão é corretamente denominada como um meio legível por computado. Por exemplo, se as instruções forem transmitidas a partir de um site, de um servidor ou de outra fonte remota utilizando um cabo coaxial, cabo de fibra óptica, par trançado, linha de assinante digital (DSL) ou tecnologias sem fio tais como infravermelho, rádio e micro-ondas, então o cabo coaxial, o cabo de fibra óptica, o par trançado, a DSL ou as tecnologias sem fio tais como infravermelho, rádio e micro-ondas estão incluídos na definição de meio. Deve ser entendido, no entanto, que meio de armazenamento legível por computador e meio de armazenamento de dados não compreendem conexões, ondas portadoras, sinais ou outros meios de comunicação transitórios, mas em vez disso são direcionados para os meios de armazenamento tangíveis não transitórios. Disco (disk) e disco (disc), conforme utilizado neste documento, inclui disco compacto (CD), disco laser, disco ótico, disco versátil digital (DVD), disquetes e discos Blu-ray, onde discos (disks) normalmente reproduzem dados magneticamente, enquanto discos (discs) reproduzem dados oticamente com lasers. As combinações desses últimos também devem ser incluídas no âmbito do meio legível por computador.

[0146] As instruções podem ser executadas por um ou mais processadores, tais como um ou mais processadores de sinal digital (DSPs), microprocessadores de propósito geral, circuitos integrados de aplicação específica (ASICs), arranjos de portas programáveis em campo (FPGAs) ou outros conjuntos de circuitos equivalente de lógica integrada ou discreta. Nesse sentido, o termo

"processador", conforme utilizado neste documento pode se referir a qualquer um dentre as estruturas acima ou qualquer outra estrutura adequada para a aplicação das técnicas descritas neste documento. Adicionalmente, em alguns aspectos, a funcionalidade descrita neste documento pode ser fornecida dentro de módulos de hardware e/ou software dedicados configurados para codificação e decodificação, ou incorporados em um codec combinado. Além disso, as técnicas podem ser totalmente implementadas em um ou mais circuitos ou elementos de lógica.

[0147] As técnicas desta descrição podem ser implementadas em uma ampla variedade de dispositivos ou aparelhos, incluindo um aparelho de telefone sem fio, um circuito integrado (IC) ou um conjunto de ICs (por exemplo, um conjunto de chips). Vários componentes, módulos ou unidades são apresentados nesta descrição para enfatizar os aspectos funcionais dos dispositivos configurados para realizar as técnicas descritas, mas que não necessariamente exigem a realização por diferentes unidades de hardware. Pelo contrário, conforme descrito acima, diversas unidades podem ser combinadas em uma unidade de hardware de codec ou fornecidas por uma coleção de unidades de hardware interoperativas, incluindo um ou mais processadores, conforme descrito acima, em conjunção com o software e/ou com o firmware adequado.

[0148] Vários exemplos foram descritos. Estes e outros exemplos estão dentro do escopo das reivindicações a seguir.

REIVINDICAÇÕES

1. Método para codificar dados de vídeo, o método compreendendo:

codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco atual de dados de vídeo predito utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento;

predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual; e

codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

2. Método, de acordo com a reivindicação 1, em que a primeira MVD inclui uma componente horizontal (MVDx1) e uma componente vertical (MVDy1), o primeiro MVP inclui uma componente horizontal (MVPx1) e uma componente vertical (MVPy1), o primeiro vetor de movimento inclui uma componente horizontal (MVx1) e uma componente vertical (MVy1), MVDx1 = MVx1-MVPx1 e MVDy1 = MVy1-MVPy1.

3. Método, de acordo com a reivindicação 1, em que codificar o bloco atual compreende codificar o bloco atual de acordo com um modelo afim de quatro parâmetros $\begin{cases} V_x = ax + by + e \\ V_y = cx + dy + f \end{cases}$.

4. Método, de acordo com a reivindicação 1, compreendendo adicionalmente predizer uma terceira MVD a partir de pelo menos uma entre a primeira MVD ou a segunda MVD para um terceiro vetor de movimento do bloco atual, em

que codificar o bloco atual compreende codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento, com o segundo vetor de movimento e com o terceiro vetor de movimento.

5. Método, de acordo com a reivindicação 4, em que codificar o bloco atual compreende codificar o bloco atual de acordo com um modelo afim de seis parâmetros

$$\begin{cases} mv_x = \frac{(mv_{1x} - mv_{0x})}{w} x + \frac{(mv_{2x} - mv_{0x})}{w} y + mv_{0x} \\ mv_y = \frac{(mv_{1y} - mv_{0y})}{w} x + \frac{(mv_{2y} - mv_{0y})}{w} y + mv_{0y}. \end{cases}$$

6. Método, de acordo com a reivindicação 1, em que o primeiro vetor de movimento se origina a partir de um canto superior esquerdo do bloco atual e em que o segundo vetor de movimento se origina a partir de um canto superior direito do bloco atual.

7. Método, de acordo com a reivindicação 1, compreendendo adicionalmente codificar dados definindo pontos de controle para o primeiro vetor de movimento e para o segundo vetor de movimento.

8. Método, de acordo com a reivindicação 1, compreendendo adicionalmente determinar os pontos de controle para o primeiro vetor de movimento e para o segundo vetor de movimento com base em uma forma do bloco atual.

9. Método, de acordo com a reivindicação 1, compreendendo adicionalmente derivar implicitamente pontos de controle para o primeiro vetor de movimento e para o segundo vetor de movimento.

10. Método, de acordo com a reivindicação 1, compreendendo adicionalmente codificar dados representando MVD'2 para a segunda MVD, em que MVD'2 representa um valor

residual para a segunda MVD em relação à primeira MVD.

11. Método, de acordo com a reivindicação 10, em que a primeira MVD compreende MVD1, a segunda MVD compreende MVD2, w compreende um valor de ponderação e MVD'2 = MVD1-w*MVD2.

12. Método, de acordo com a reivindicação 10, em que a primeira MVD compreende MVD1, a segunda MVD compreende MVD2 e MVD’2 = MVD2-((MVD1+1)>>1) para um valor de ponderação de 0,5.

13. Método, de acordo com a reivindicação 10, em que a primeira MVD compreende MVD1, a segunda MVD compreende MVD2 e MVD’2 = MVD2-((MVD1+2)>>2) para um valor de ponderação de 0,25.

14. Método, de acordo com a reivindicação 1, compreendendo adicionalmente determinar a predição da segunda MVD a partir da primeira MVD, em que a predição da segunda MVD a partir da primeira MVD compreende predizer a segunda MVD a partir da primeira MVD em resposta à determinação da predição da segunda MVD a partir da primeira MVD.

15. Método, de acordo com a reivindicação 14, em que determinar a predição da segunda MVD a partir da primeira MVD compreende determinar a predição da segunda MVD a partir da primeira MVD com base em uma forma do bloco atual.

16. Método, de acordo com a reivindicação 14, em que determinar a predição da segunda MVD a partir da primeira MVD compreende determinar a predição da segunda MVD a partir da primeira MVD com base em um método de predição de movimento para o bloco atual.

17. Método, de acordo com a reivindicação 16, em que determinar a predição da segunda MVD a partir da primeira MVD compreende determinar que o método de predição de movimento é um modo de mesclagem afim.

18. Método, de acordo com a reivindicação 1, em que a segunda MVD inclui uma componente horizontal ($MVD2^x$) e uma componente vertical ($MVD2^y$), e em que a predição da segunda MVD compreende a predição de $MVD2^x$ diferentemente da predição de $MVD2^y$.

19. Método, de acordo com a reivindicação 1, compreendendo adicionalmente gerar um terceiro MVP para um terceiro vetor de movimento do bloco atual a partir de pelo menos uma entre a primeira MVD ou a segunda MVD.

20. Método, de acordo com a reivindicação 1, em que a primeira MVD compreende MVD1, o método compreendendo adicionalmente:

determinar uma segunda MVP intermediária (MVP'2) para o segundo vetor de movimento a partir de um vetor de movimento de um ou mais blocos vizinhos ao bloco atual; e

gerar uma segunda MVP (MVP2) para o segundo vetor de movimento a partir de MVP'2 e de MVD1.

21. Método, de acordo com a reivindicação 1, compreendendo adicionalmente gerar uma segunda MVP para o segundo vetor de movimento a partir da primeira MVD e uma terceira MVD para um terceiro vetor de movimento do bloco atual, em que a segunda MVP compreende MVP2, a primeira MVD compreende MVDl, a terceira MVD compreende MVD3, o método compreendendo adicionalmente determinar uma segunda MVP intermediária (MVP'2) para o segundo vetor de movimento a partir de um vetor de movimento de um ou mais blocos

vizinhos ao bloco atual, em que gerar MPV2 compreende gerar MVP2 como MVP2 = MVP'2+((MVD1+MVD3)>>1).

22. Método, de acordo com a reivindicação 1, em que codificar o bloco atual compreende decodificar o bloco atual, compreendendo:

adicionar a primeira MVD ao primeiro MVP para reconstruir o primeiro vetor de movimento;

determinar um segundo MVP para o segundo vetor de movimento;

reconstruir a segunda MVD utilizando a predição a partir da primeira MVD;

adicionar a segunda MVD ao segundo MVP para reconstruir o segundo vetor de movimento;

formar um bloco de predição para o bloco atual utilizando o primeiro vetor de movimento e o segundo vetor de movimento;

decodificar um bloco residual para o bloco atual; e

adicionar o bloco residual e o bloco de predição para reconstruir o bloco atual.

23. Método, de acordo com a reivindicação 1, em que codificar o bloco atual compreende codificar o bloco atual, compreendendo:

subtrair o primeiro MVP a partir do primeiro vetor de movimento para gerar a primeira MVD;

determinar um segundo MVP para o segundo vetor de movimento;

subtrair o segundo vetor de movimento a partir do segundo MVP para gerar a segunda MVD;

codificar a primeira MVD;

codificar dados representativos da segunda MVD predita a partir da primeira MVD;

formar um bloco de predição para o bloco atual utilizando o primeiro vetor de movimento e o segundo vetor de movimento;

subtrair o bloco de predição a partir do bloco atual para gerar um bloco residual; e

codificar o bloco residual.

24. Dispositivo para codificar dados de vídeo, o dispositivo compreendendo:

uma memória configurada para armazenar dados de vídeo; e

um ou mais processadores implementados em conjunto de circuitos e configurados para:

codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco atual de dados de vídeo predito utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento;

predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual; e

codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

25. Dispositivo, de acordo com a reivindicação 24, em que os um ou mais processadores estão configurados para:

adicionar a primeira MVD ao primeiro MVP para

reconstruir o primeiro vetor de movimento;

determinar um segundo MVP para o segundo vetor de movimento;

reconstruir a segunda MVD utilizando a predição a partir da primeira MVD;

adicionar a segunda MVD ao segundo MVP para reconstruir o segundo vetor de movimento;

formar um bloco de predição para o bloco atual utilizando o primeiro vetor de movimento e o segundo vetor de movimento;

decodificar um bloco residual para o bloco atual; e

adicionar o bloco residual e o bloco de predição para reconstruir o bloco atual.

26. Dispositivo, de acordo com a reivindicação 24, em que os um ou mais processadores estão configurados para:

subtrair o primeiro MVP a partir do primeiro vetor de movimento para gerar a primeira MVD;

determinar um segundo MVP para o segundo vetor de movimento;

subtrair o segundo vetor de movimento a partir do segundo MVP para gerar a segunda MVD;

codificar a primeira MVD;

codificar dados representativos da segunda MVD predita a partir da primeira MVD;

formar um bloco de predição para o bloco atual utilizando o primeiro vetor de movimento e o segundo vetor de movimento;

subtrair o bloco de predição do bloco atual para

gerar um bloco residual; e

codificar o bloco residual.

27. Dispositivo, de acordo com a reivindicação 24, compreendendo adicionalmente um display configurado para exibir dados de vídeo decodificados.

28. Dispositivo, de acordo com a reivindicação 24, em que o dispositivo compreende um ou mais dentre uma câmera, um computador, um dispositivo móvel, um dispositivo receptor de difusão ou um set-top box.

29. Meio de armazenamento legível por computador tendo armazenado nele instruções que, quando executadas, fazem um processador de um dispositivo para codificar dados de vídeo:

codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo atual predito utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento;

predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual; e

codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

30. Dispositivo para codificar dados de vídeo, o dispositivo compreendendo:

meios para codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo atual predito utilizando predição afim e um primeiro

preditor de vetor de movimento (MVP) para o primeiro vetor de movimento;

meios para predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual; e

meios para codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento.

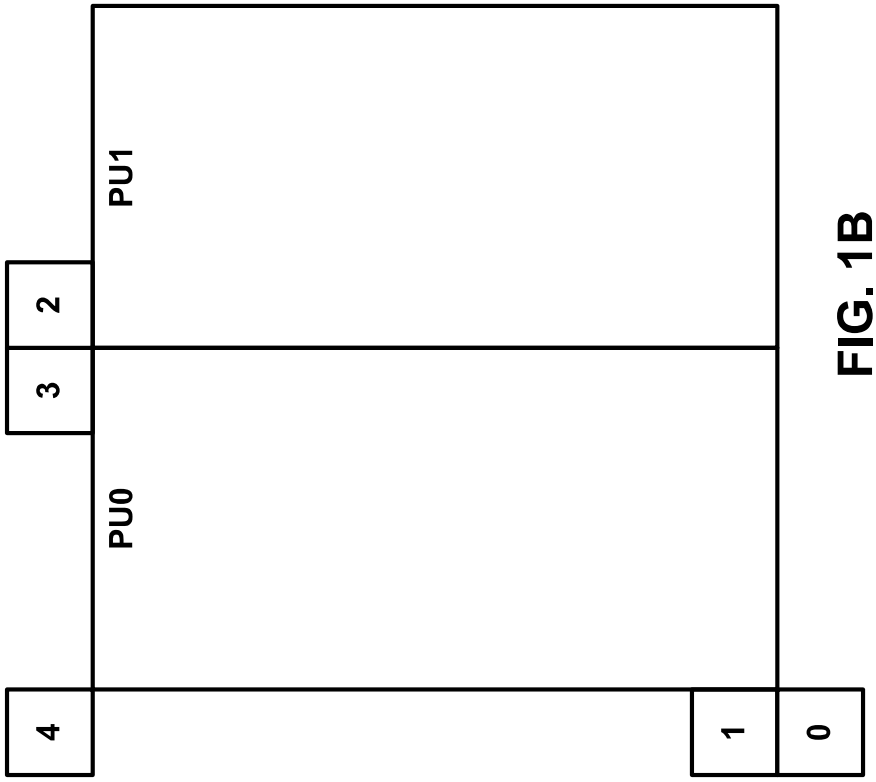


FIG. 1B

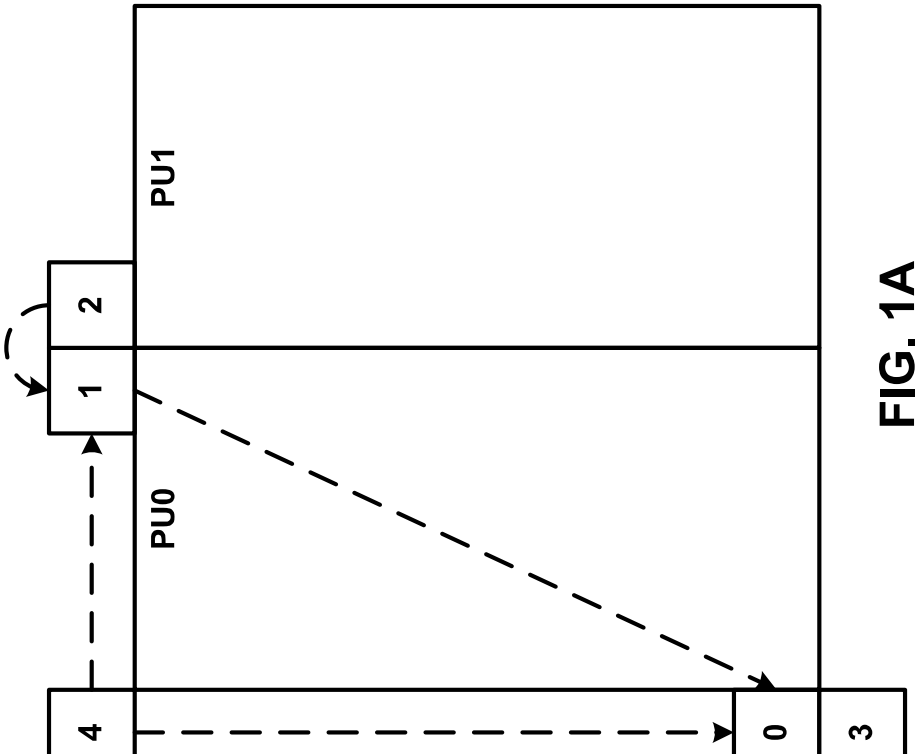
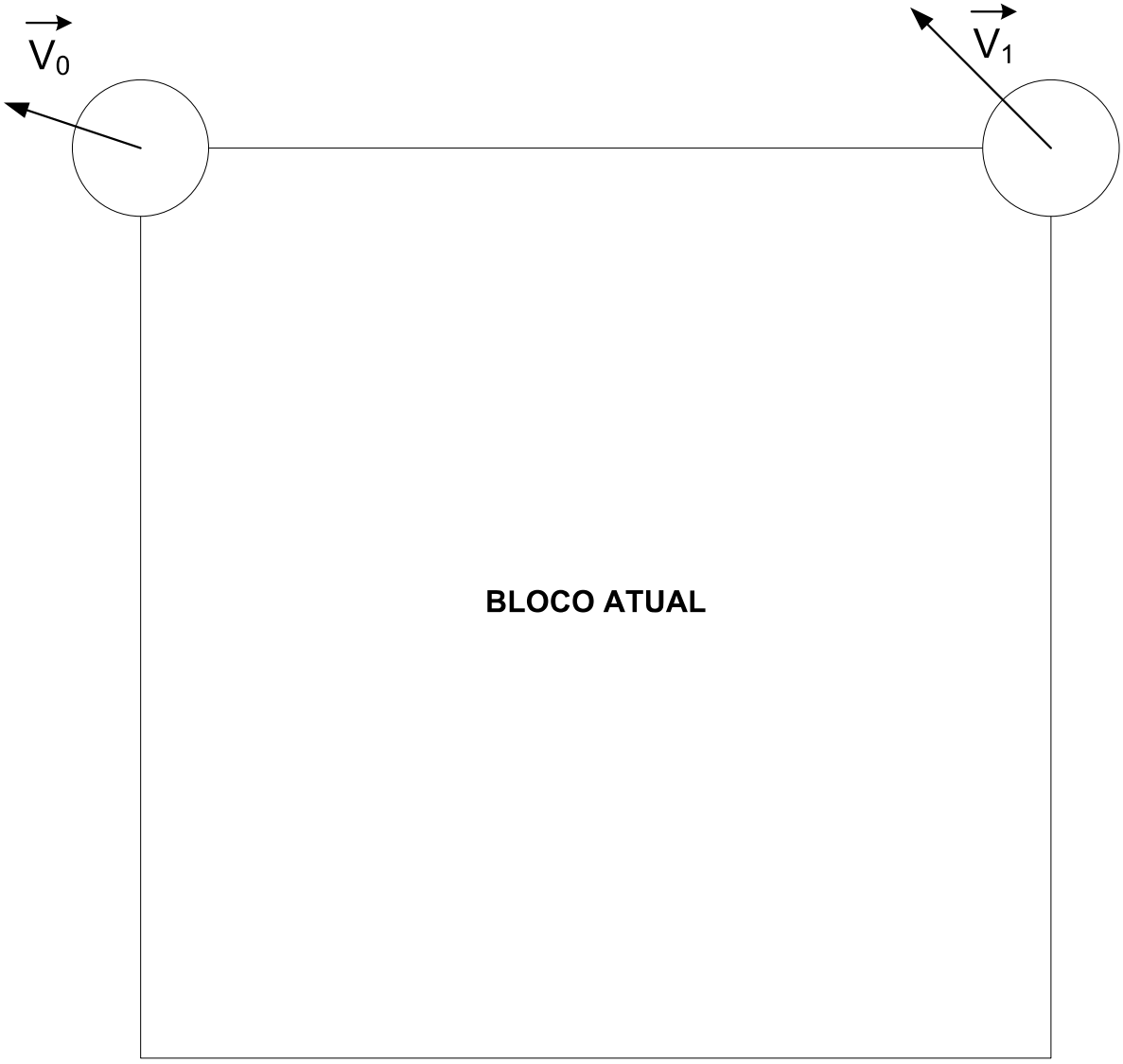
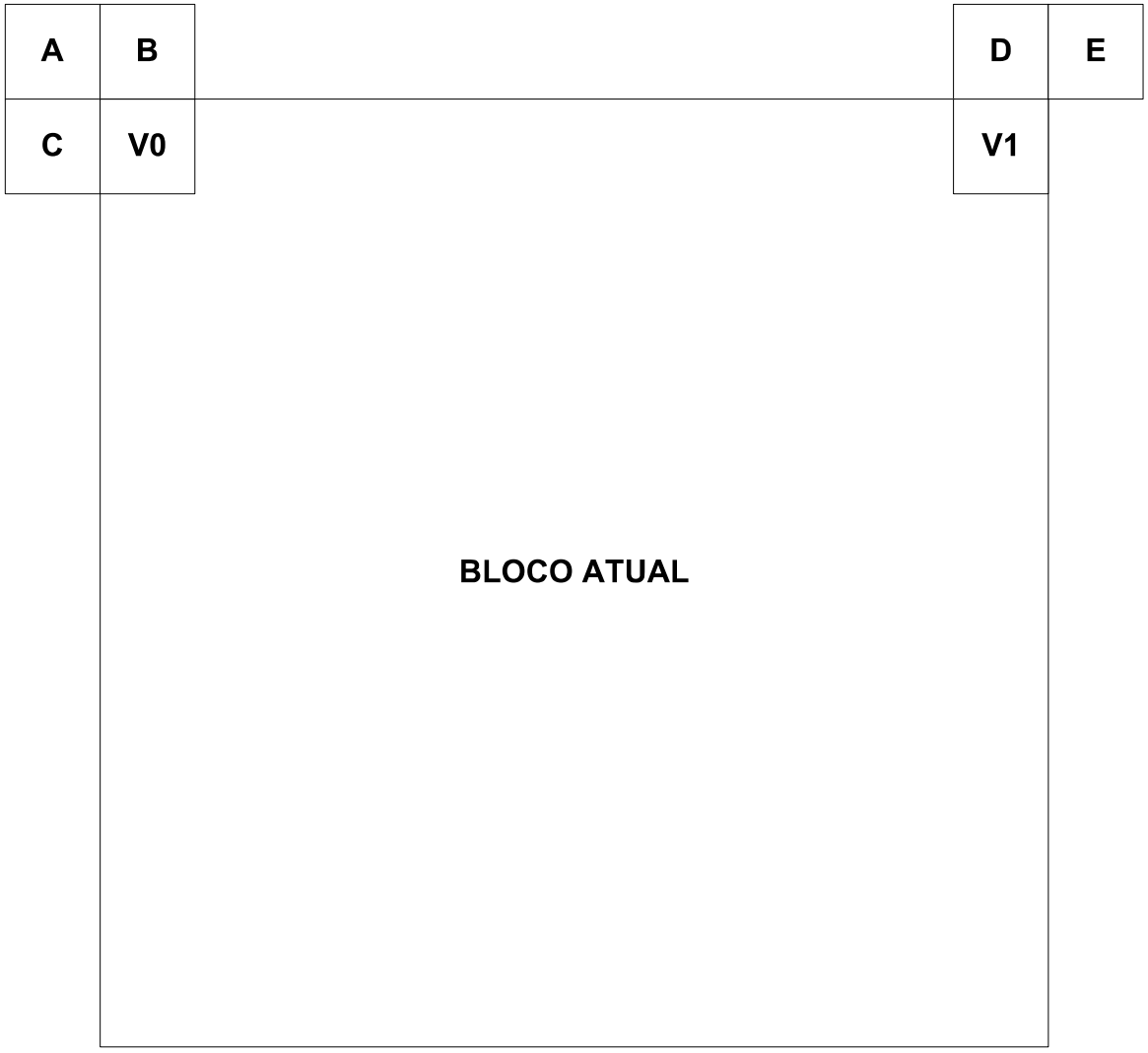


FIG. 1A


$\vec{V}_0$
$\vec{V}_1$
BLOCO ATUAL


A
B
D
E
C
V0
V1
BLOCO ATUAL

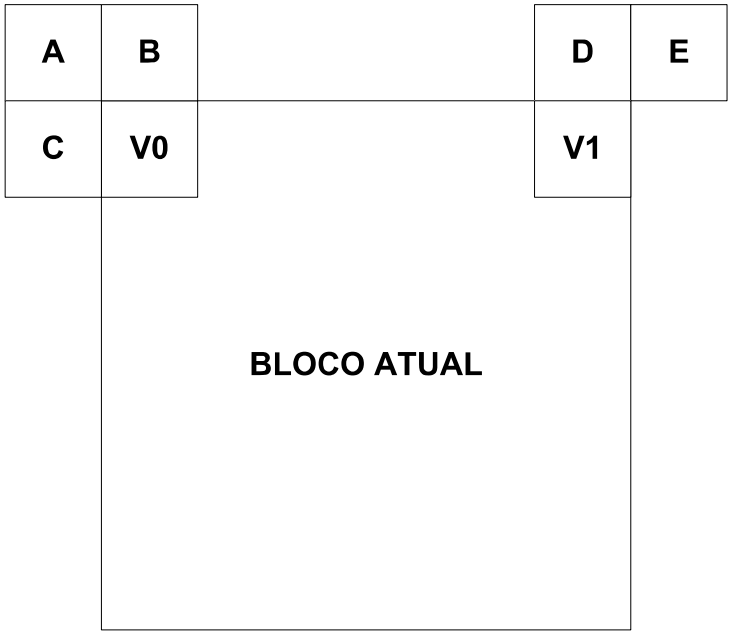
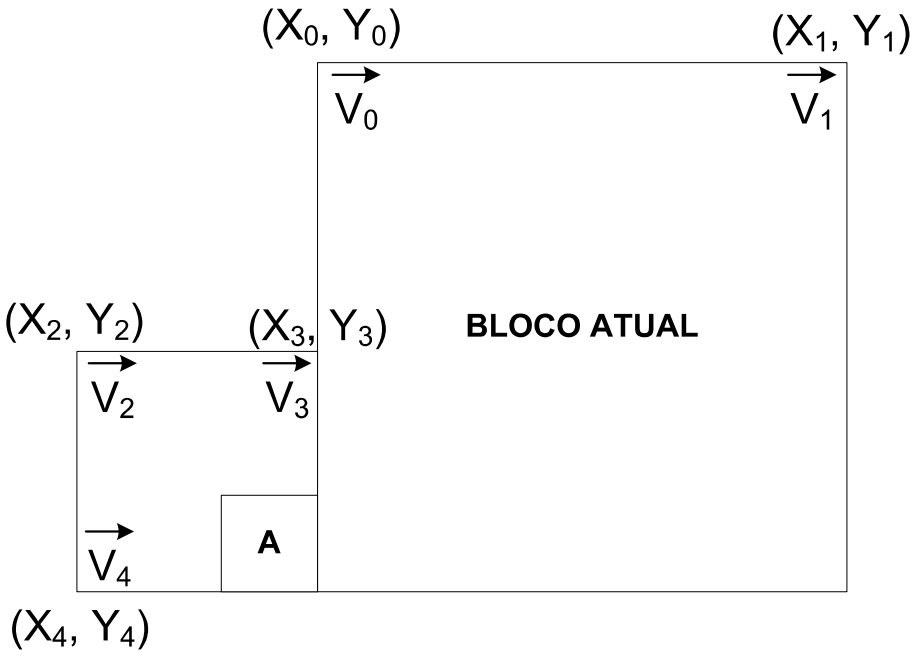

A
B
D
E
C
V0
V1
BLOCO ATUAL

FIG. 4A


(X0, Y0)
(X1, Y1)
V0
V1
(X2, Y2)
(X3, Y3)
BLOCO ATUAL
V2
V3
A
V4
(X4, Y4)

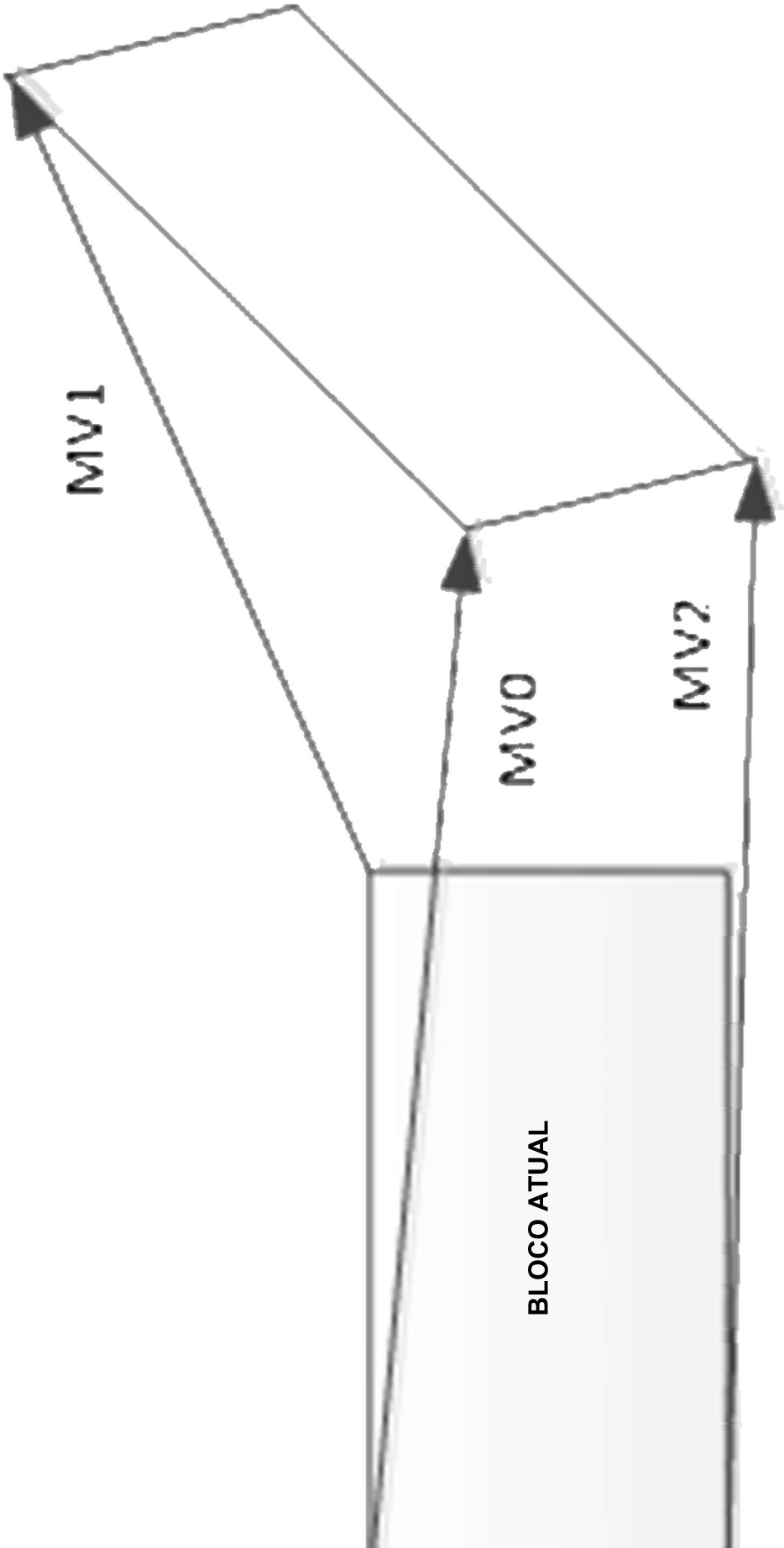
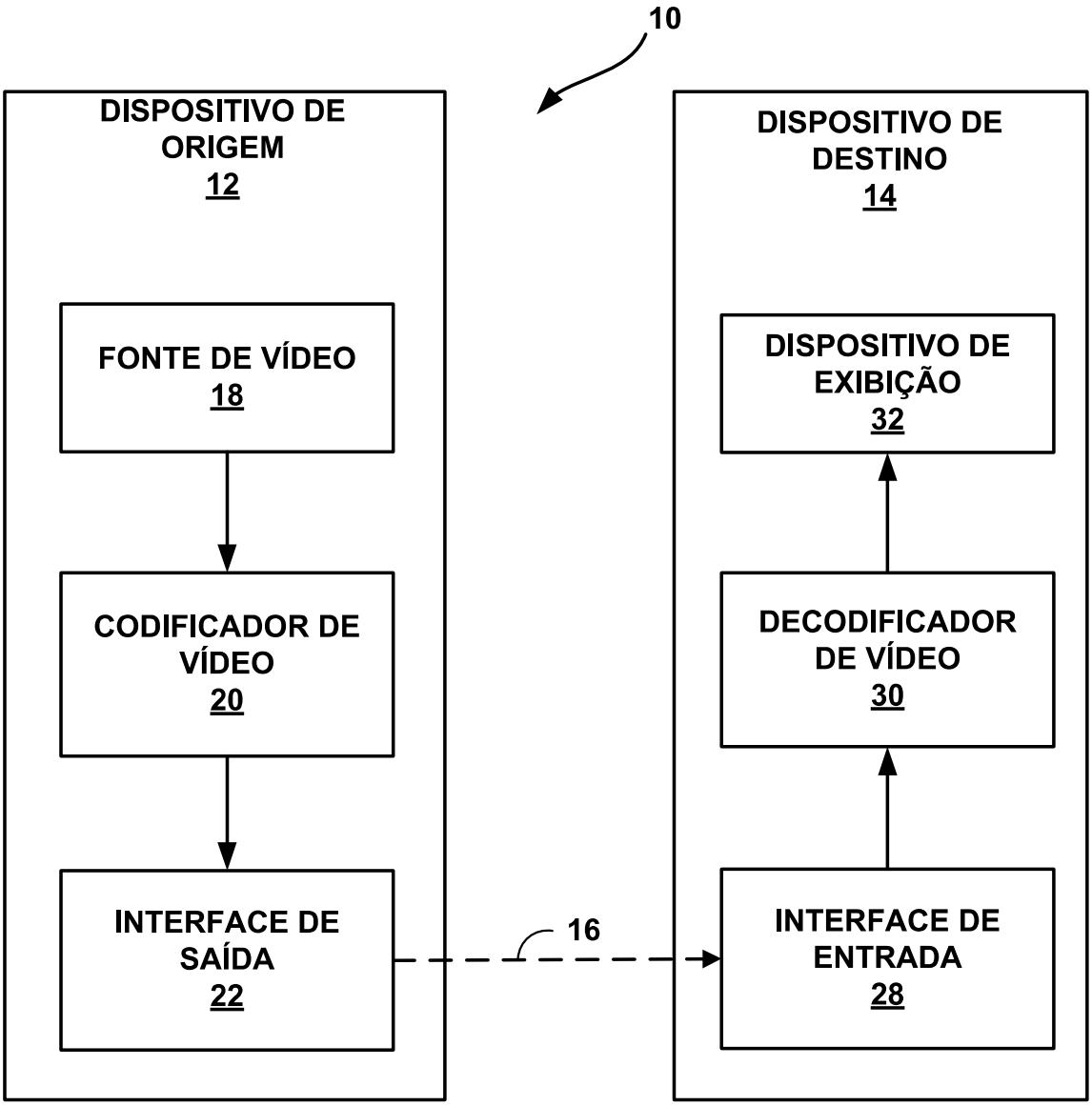


FIG. 5


10
DISPOSITIVO DE ORIGEM
12
FONTE DE VÍDEO
18
CODIFICADOR DE VÍDEO
20
INTERFACE DE SAÍDA
22
16
DISPOSITIVO DE DESTINO
14
DISPOSITIVO DE EXIBIÇÃO
32
DECODIFICADOR DE VÍDEO
30
INTERFACE DE ENTRADA
28

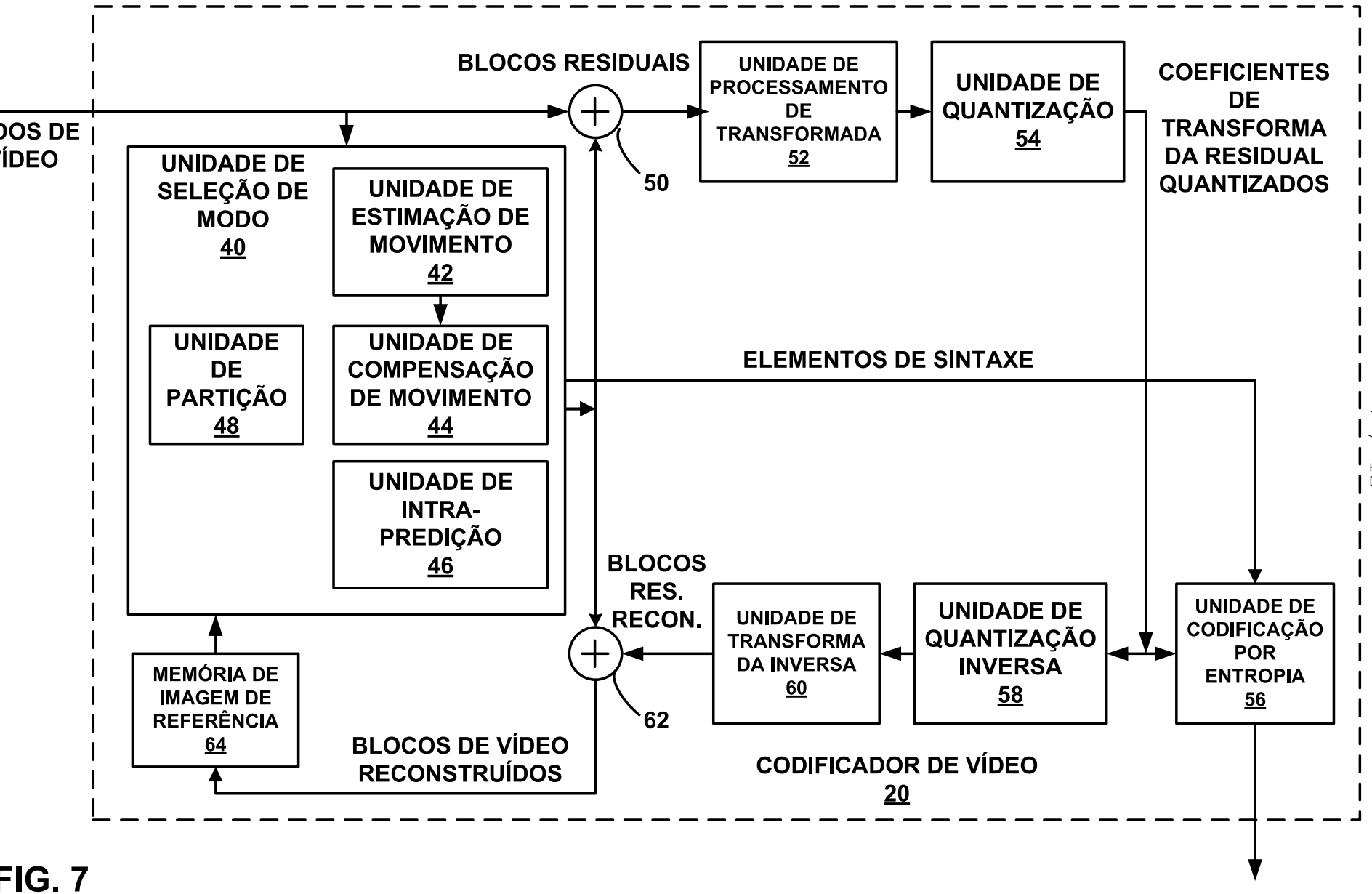

DOS DE
ÍDEO
BLOCOS RESIDUAIS
50
UNIDADE DE PROCESSAMENTO DE TRANSFORMADA
52
UNIDADE DE QUANTIZAÇÃO
54
COEFICIENTES DE TRANSFORMA DA RESIDUAL QUANTIZADOS
UNIDADE DE SELEÇÃO DE MODO
40
UNIDADE DE ESTIMAÇÃO DE MOVIMENTO
42
UNIDADE DE PARTIÇÃO
48
UNIDADE DE COMPENSAÇÃO DE MOVIMENTO
44
UNIDADE DE INTRA-PREDIÇÃO
46
ELEMENTOS DE SINTAXE
BLOCOS RES. RECON.
62
UNIDADE DE TRANSFORMADA INVERSA
60
UNIDADE DE QUANTIZAÇÃO INVERSA
58
UNIDADE DE CODIFICAÇÃO POR ENTROPIA
56
MEMÓRIA DE IMAGEM DE REFERÊNCIA
64
BLOCOS DE VÍDEO RECONSTRUÍDOS
CODIFICADOR DE VÍDEO
20

FIG. 7

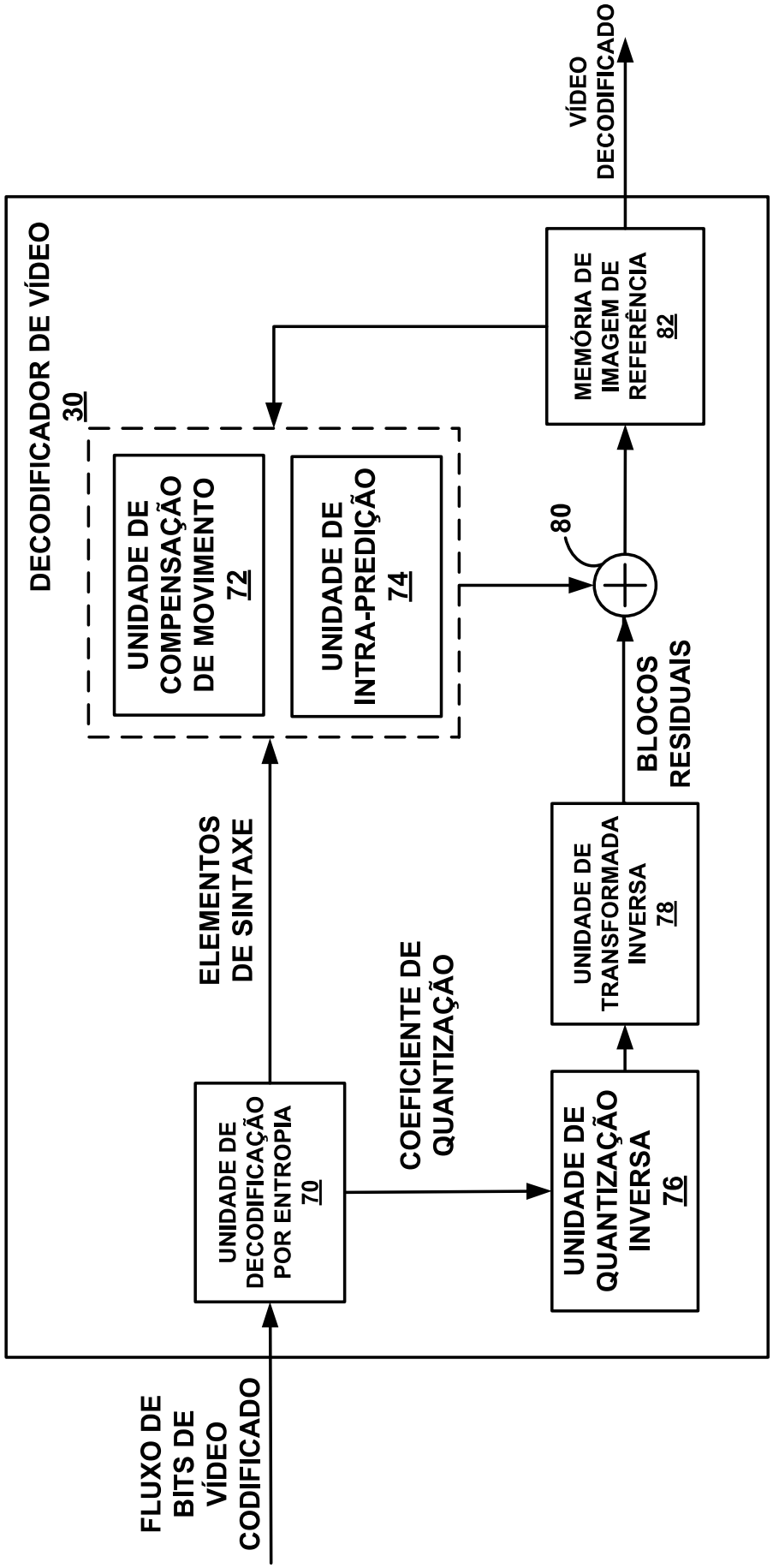

FLUXO DE BITS DE VÍDEO CODIFICADO
DECODIFICADOR DE VÍDEO
30
UNIDADE DE DECODIFICAÇÃO POR ENTROPIA 70
UNIDADE DE COMPENSAÇÃO DE MOVIMENTO 72
UNIDADE DE INTRA-PREDIÇÃO 74
ELEMENTOS DE SINTAXE
COEFICIENTE DE QUANTIZAÇÃO
UNIDADE DE QUANTIZAÇÃO INVERSA 76
UNIDADE DE TRANSFORMADA INVERSA 78
BLOCOS RESIDUAIS
80
MEMÓRIA DE IMAGEM DE REFERÊNCIA 82
VÍDEO DECODIFICADO

FIG. 8

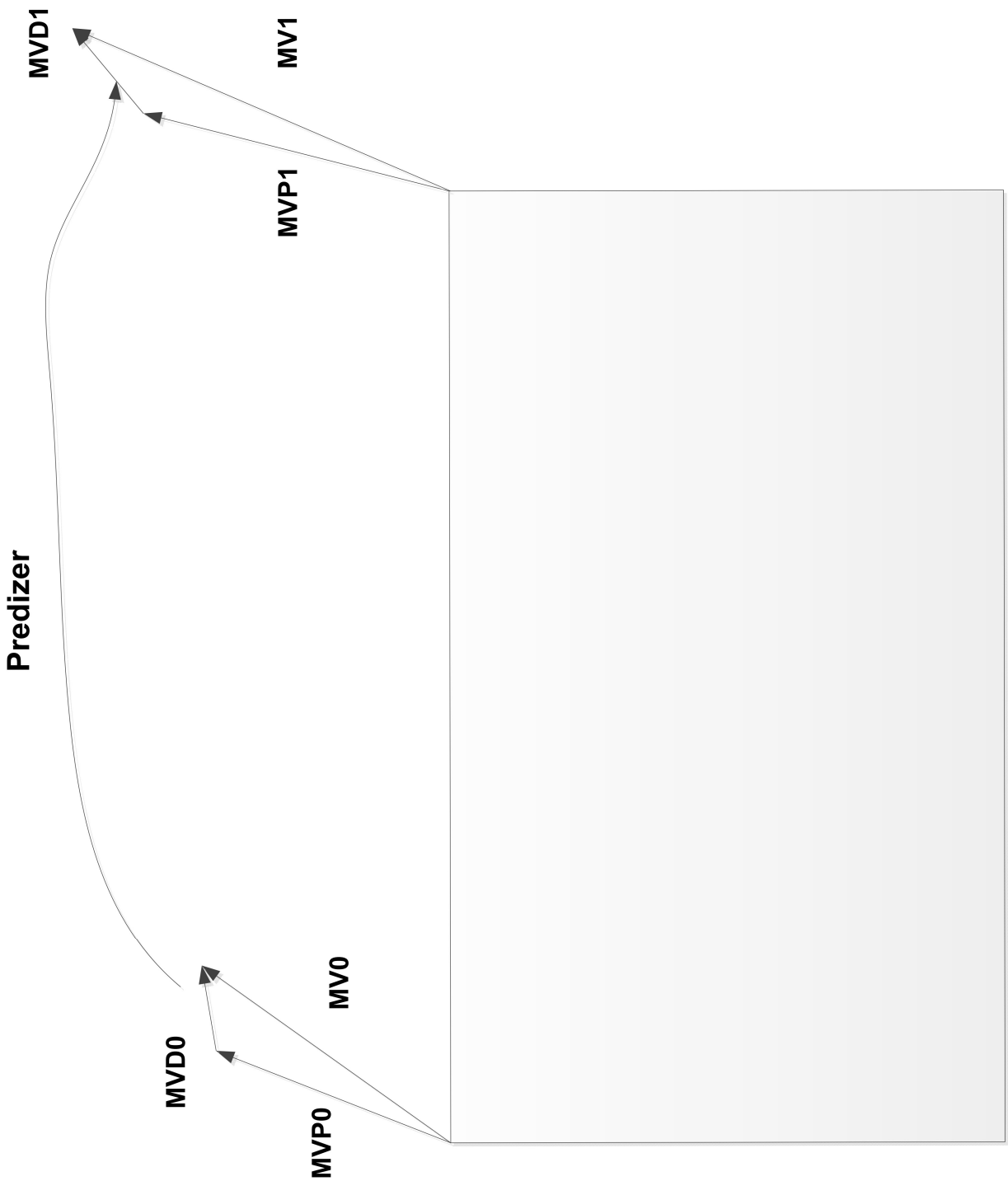


FIG. 9

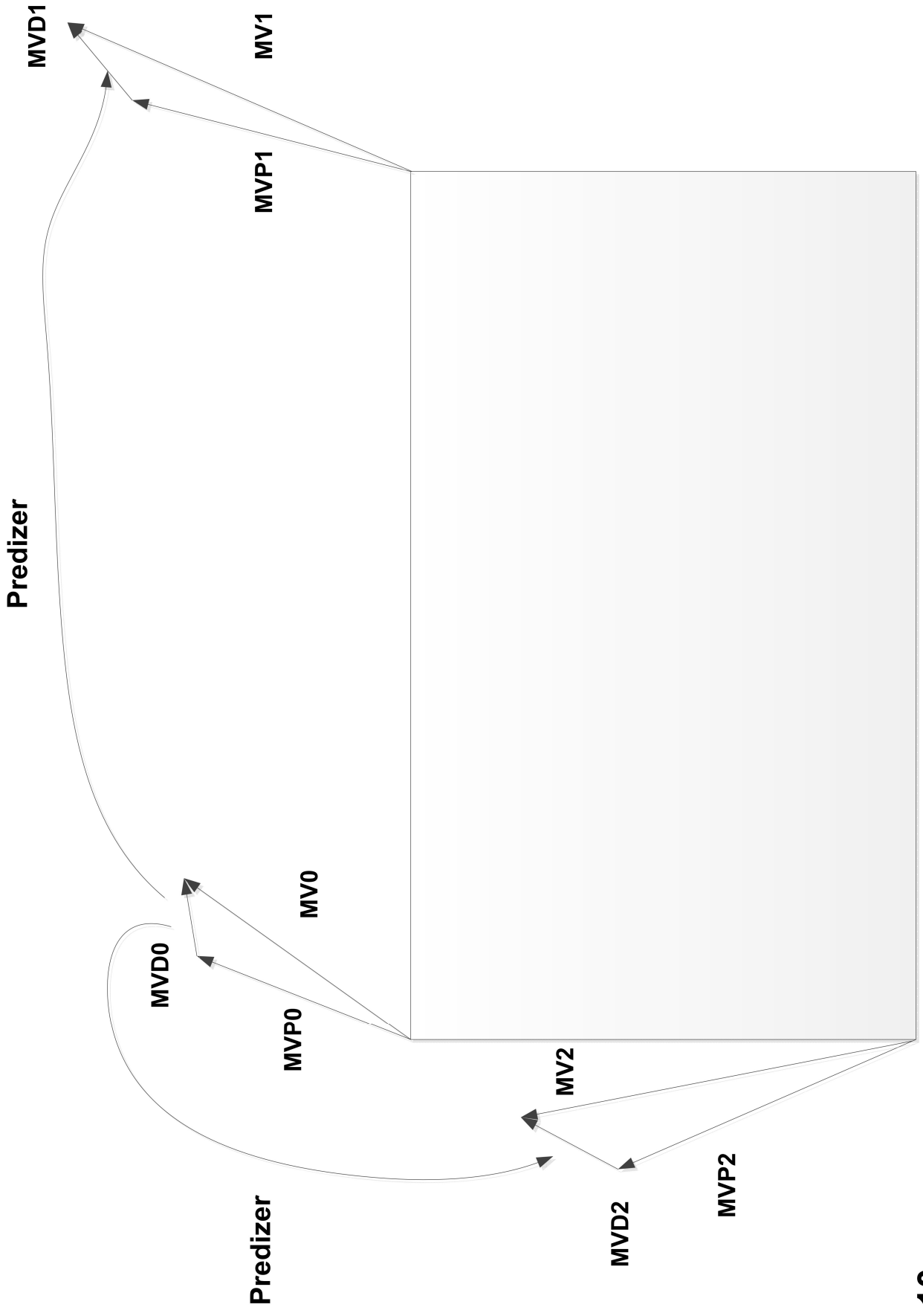
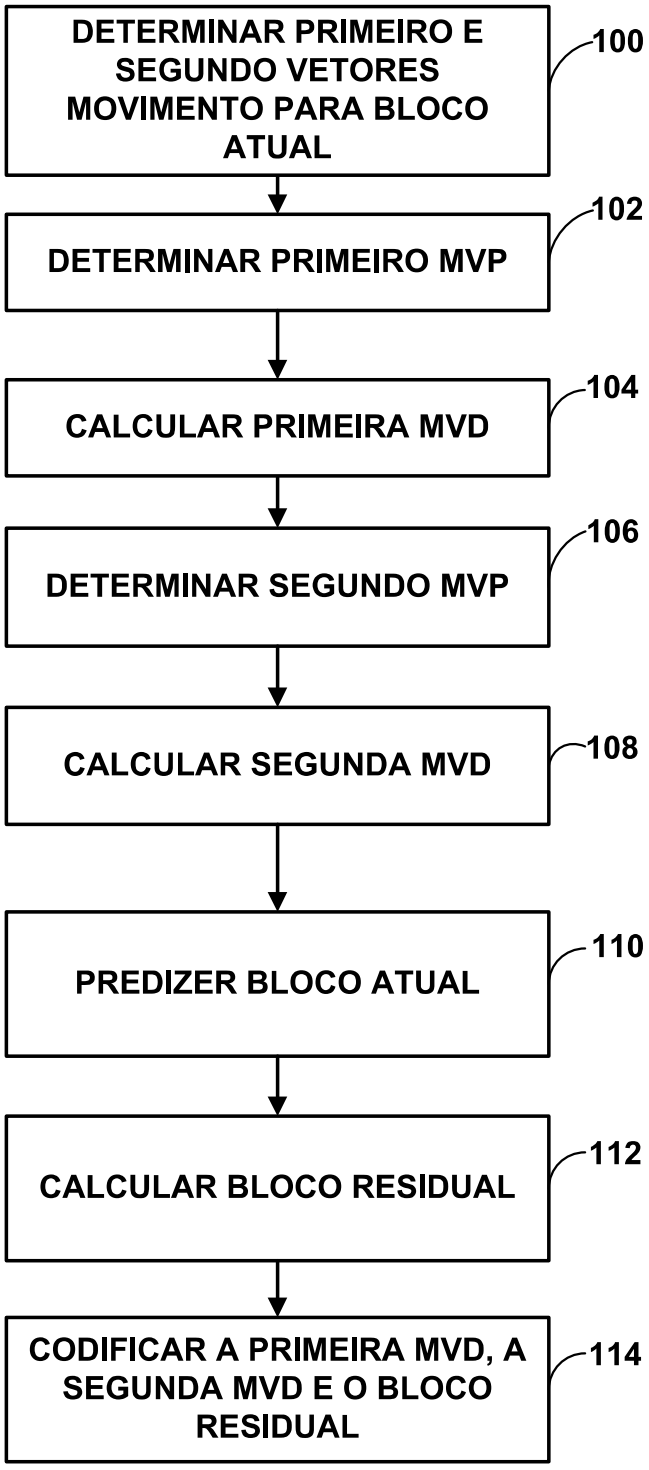
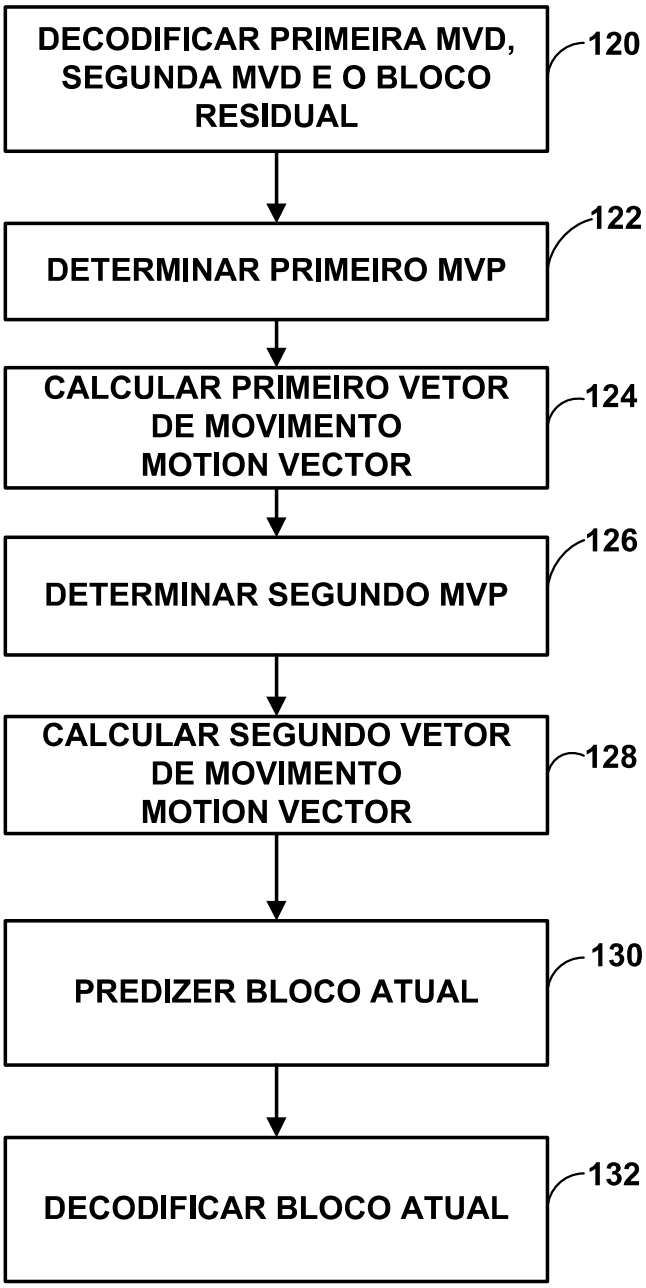


FIG. 10


DETERMINAR PRIMEIRO E SEGUNDO VETORES MOVIMENTO PARA BLOCO ATUAL
100
DETERMINAR PRIMEIRO MVP
102
CALCULAR PRIMEIRA MVD
104
DETERMINAR SEGUNDO MVP
106
CALCULAR SEGUNDA MVD
108
PREDIZER BLOCO ATUAL
110
CALCULAR BLOCO RESIDUAL
112
CODIFICAR A PRIMEIRA MVD, A SEGUNDA MVD E O BLOCO RESIDUAL
114


DECODIFICAR PRIMEIRA MVD, SEGUNDA MVD E O BLOCO RESIDUAL
120
DETERMINAR PRIMEIRO MVP
122
CALCULAR PRIMEIRO VETOR DE MOVIMENTO MOTION VECTOR
124
DETERMINAR SEGUNDO MVP
126
CALCULAR SEGUNDO VETOR DE MOVIMENTO MOTION VECTOR
128
PREDIZER BLOCO ATUAL
130
DECODIFICAR BLOCO ATUAL
132

RESUMO

"CODIFICAÇÃO DE INFORMAÇÃO DE MOVIMENTO DE PREDIÇÃO AFIM PARA CODIFICAÇÃO DE VÍDEO"

Um dispositivo de exemplo para codificar dados de vídeo inclui uma memória configurada para armazenar dados de vídeo e um ou mais processadores implementados em conjuntos de circuitos e configurados para codificar uma primeira diferença de vetor de movimento (MVD) representando uma diferença entre um primeiro vetor de movimento de um bloco de dados de vídeo predito atual utilizando predição afim e um primeiro preditor de vetor de movimento (MVP) para o primeiro vetor de movimento, predizer uma segunda MVD a partir da primeira MVD para um segundo vetor de movimento do bloco atual e codificar o bloco atual utilizando predição afim de acordo com o primeiro vetor de movimento e com o segundo vetor de movimento. Predizer a segunda MVD a partir da primeira MVD pode reduzir a taxa de bits de um fluxo de bits, incluindo dados de vídeo codificados, além de melhorar a eficiência do processamento.