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

(21) BR 122022009663-8 A2

(22) Data do Depósito: 06/10/2020

(43) Data da Publicação Nacional: 28/06/2022

(54) Título: PREVENÇÃO DE SINALIZAÇÃO REDUNDANTE EM FLUXOS DE BITS DE VÍDEO DE MÚLTIPLAS CAMADAS

(51) Int. Cl.: H04N 19/52; H04N 7/12.

(30) Prioridade Unionista: 07/10/2019 US 62/911,808.

(71) Depositante(es): HUAWEI TECHNOLOGIES CO., LTD..

(72) Inventor(es): YE-KUI WANG.

(86) Pedido PCT: PCT US2020054450 de 06/10/2020

(87) Publicação PCT: WO 2021/022269 de 04/02/2021

(85) Data da Fase Nacional: 18/05/2022

(62) Pedido original do dividido: BR112022006392-0 - 06/10/2020

(57) Resumo: PREVENÇÃO DE SINALIZAÇÃO REDUNDANTE EM FLUXOS DE BITS DE VÍDEO DE MÚLTIPLAS CAMADAS. Um método de decodificação é fornecido. O método inclui receber um subfluxo de bits tendo um conjunto de parâmetros de sequência (SPS) com uma estrutura de sintaxe de buffer de imagem decodificada (DPB) para um conjunto de camadas de saída (OLS) com apenas uma camada; definir um número de camadas no OLS como igual a um com base no OLS tendo a apenas uma camada; obter a estrutura de sintaxe de DPB para o OLS com a apenas uma camada a partir do SPS quando o número de camadas no OLS tiver sido definido como igual a um; e decodificar uma imagem a partir da apenas uma camada usando a estrutura de sintaxe de DPB para obter uma imagem decodificada. Um método correspondente de codificação também é fornecido.

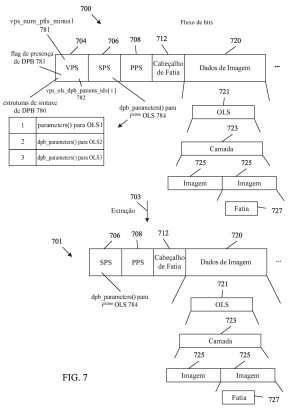


FIG. 7

“PREVENÇÃO DE SINALIZAÇÃO REDUNDANTE EM FLUXOS DE BITS DE VÍDEO DE MÚLTIPLAS CAMADAS”

Dividido do BR 112022006392-0, de 06/10/2020.

CAMPO TÉCNICO

[0001] Em geral, esta revelação descreve técnicas para fluxos de bits de vídeo de múltiplas camadas em codificação de vídeo. Mais especificamente, esta revelação descreve técnicas para eliminar redundância ao sinalizar parâmetros correspondentes a camadas em um fluxo de bits de múltiplas camadas em codificação de vídeo.

FUNDAMENTOS

[0002] A quantidade de dados de vídeo necessária para representar mesmo um vídeo relativamente curto pode ser substancial, o que pode resultar em dificuldades quando os dados devem ser transmitidos em fluxo contínuo ou comunicados de outra forma através de uma rede de comunicações com capacidade de largura de banda limitada. Assim, dados de vídeo geralmente são comprimidos antes de serem comunicados pelas redes de telecomunicações modernas. O tamanho de um vídeo também pode ser um problema quando o vídeo é armazenado em um dispositivo de armazenamento, pois os recursos de memória podem ser limitados. Dispositivos de compressão de vídeo geralmente usam software e/ou hardware na origem para codificar os dados de vídeo antes da transmissão ou armazenamento, diminuindo assim a quantidade de dados necessários para representar imagens de vídeo digital. Os dados comprimidos são então recebidos no destino por um dispositivo de descompressão de vídeo que decodifica os dados de vídeo. Com recursos de rede limitados e demandas cada vez maiores de maior qualidade de vídeo, técnicas aprimoradas de compressão e descompressão que melhoram a taxa de compressão com pouco ou nenhum sacrifício na qualidade de imagem são desejáveis.

SUMÁRIO

[0003] Um primeiro aspecto refere-se a um método de decodificação implementado por um decodificador de vídeo, compreendendo: receber, pelo decodificador de vídeo, um subfluxo de bits tendo um conjunto de parâmetros de sequência (SPS) com uma estrutura de sintaxe de buffer de imagem decodificada (DPB) para um conjunto de camadas de saída (OLS) com apenas uma camada; obter, pelo decodificador de vídeo, a estrutura de sintaxe de DPB

para o OLS com a única camada do SPS; e decodificar, pelo decodificador de vídeo, uma imagem a partir da apenas uma camada usando a estrutura de sintaxe de DPB para obter uma imagem decodificada.

[0004] O método fornece técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0005] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que a estrutura de sintaxe de DPB seja designada dpb_parameters().

[0006] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que a apenas uma camada é a $i^{ésima}$ camada, e em que o OLS é o $i^{ésimo}$ OLS.

[0007] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que a estrutura de sintaxe de DPB esteja presente em uma $i^{ésima}$ estrutura de sintaxe dpb_parameters() no SPS.

[0008] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o subfluxo de bits não inclua um conjunto de parâmetros de vídeo (VPS).

[0009] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece o armazenamento da imagem decodificada em um DPB antes de ser exibida.

[0010] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o SPS seja referido pela apenas uma camada no OLS.

[0011] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece a exibição da imagem decodificada em

um display de um dispositivo eletrônico.

[0012] Um segundo aspecto refere-se a um método de codificação implementado por um codificador de vídeo, o método compreendendo: gerar, pelo codificador de vídeo, um conjunto de parâmetros de vídeo (VPS) e um conjunto de parâmetros de sequência (SPS), em que o VPS inclui um lista de estruturas de sintaxe de buffer de imagem decodificada (DPB) para todos os conjuntos de camadas de saída (OLSs), e em que o SPS inclui a estrutura de sintaxe de DPB para cada OLS com apenas uma camada; codificar, pelo codificador de vídeo, o VPS e o SPS em um fluxo de bits de vídeo; e armazenar, pelo codificador de vídeo, o fluxo de bits de vídeo para comunicação em direção a um decodificador de vídeo.

[0013] O método fornece técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0014] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o VPS inclui um índice para a lista de estruturas de sintaxe de DPB, e em que o índice é designado vps_ols_dpb_params_idx[i].

[0015] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que um valor de vps_ols_dpb_params_idx[i] esteja em uma faixa de zero a um número total de estruturas de sintaxe de DPB no VPS.

[0016] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o VPS inclui vps_num_dpb_params_minus1, e em que vps_num_dpb_params_minus1 mais 1 especifica um número total de estruturas de sintaxe de DPB no VPS.

[0017] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece extração de um subfluxo de bits a partir do fluxo de bits, em que o subfluxo de bits inclui o SPS do fluxo de bits, mas não inclui o VPS do fluxo de bits.

[0018] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece a transmissão do subfluxo de bits em direção a um decodificador de vídeo.

[0019] Um terceiro aspecto refere-se a um dispositivo de decodificação, compreendendo: um receptor configurado para receber um subfluxo de bits tendo um conjunto de parâmetros de sequência (SPS) com uma estrutura de sintaxe de buffer de imagem decodificada (DPB) para um conjunto de camadas de saída (OLS) com apenas uma camada; e um processador acoplado à memória, o processador configurado para executar as instruções para fazer o dispositivo de decodificação: definir um número de camadas no OLS como igual a um com base no OLS tendo apenas uma camada; obter a estrutura de sintaxe de DPB para o OLS com a apenas uma camada a partir do SPS quando o número de camadas no OLS tiver sido definido como igual a um; e decodificar uma imagem a partir da apenas uma camada usando a estrutura de sintaxe de DPB para obter uma imagem decodificada.

[0020] O dispositivo de decodificação fornece técnicas para garantir que os parâmetros do buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0021] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que a estrutura de sintaxe de DPB seja designada dpb_parameters(), em que a apenas uma camada é a i^{ésima} camada, e em que o OLS é a i^{ésima} camada de OLS.

[0022] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que a estrutura de sintaxe de DPB esteja presente em uma i^{ésima} estrutura de sintaxe dpb_parameters() no SPS.

[0023] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o subfluxo de bits não inclua um conjunto de parâmetros de vídeo (VPS).

[0024] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o SPS seja referido pela apenas uma camada no OLS.

[0025] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece um display configurado para exibir a imagem decodificada.

[0026] Um quarto aspecto refere-se a um dispositivo de codificação, compreendendo: uma memória contendo instruções; um processador acoplado à memória, o processador configurado para implementar as instruções para fazer o dispositivo de codificação: gerar um conjunto de parâmetros de vídeo (VPS) e um conjunto de parâmetros de sequência (SPS), em que o VPS inclui uma lista de buffer de imagem decodificada (DPB) estruturas de sintaxe para todos os conjuntos de camadas de saída (OLSs), e em que o SPS inclui a estrutura de sintaxe de DPB para cada OLS tendo apenas uma camada; e codificar o VPS e o SPS em um fluxo de bits de vídeo; e um transmissor acoplado ao processador, o transmissor configurado para transmitir o fluxo de bits de vídeo em direção a um decodificador de vídeo.

[0027] O dispositivo de codificação fornece técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0028] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o VPS inclui um índice para a lista de estruturas de sintaxe de DPB, e em que o índice é designado vps_ols_dpb_params_idx[i].

[0029] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que um valor de vps_ols_dpb_params_idx [i] esteja em uma faixa de zero a um número total de estruturas de sintaxe de DPB no VPS.

[0030] Opcionalmente, em qualquer um dos aspectos anteriores, outra implementação do aspecto fornece que o VPS inclui vps_num_dpb_params_minus1, e em que vps_num_dpb_params_minus1 mais 1 especifica o número total das estruturas de sintaxe de DPB no VPS.

[0031] Um quinto aspecto refere-se a um aparelho de codificação. O aparelho de codificação inclui um receptor configurado para receber uma imagem para codificação ou para receber um fluxo de bits para decodificação; um transmissor acoplado ao receptor, o transmissor configurado para transmitir o fluxo de bits para um decodificador ou para transmitir uma imagem decodificada para um display; uma memória acoplada a pelo menos um dentre o receptor ou o transmissor, a memória configurada para armazenar instruções; e um processador acoplado à memória, o processador configurado para executar as instruções armazenadas na memória para realizar qualquer um dos métodos divulgados neste documento.

[0032] O aparelho de codificação fornece técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0033] Opcionalmente, em qualquer um dos aspectos anteriores,

outra implementação do aspecto fornece um display configurado para exibir uma imagem decodificada.

[0034] Um sexto aspecto refere-se a um sistema. O sistema inclui um codificador; e um decodificador em comunicação com o codificador, em que o codificador ou o decodificador inclui o dispositivo de decodificação, o dispositivo de codificação ou o aparelho de codificação divulgado neste documento.

[0035] O sistema fornece técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0036] Um sétimo aspecto refere-se a um meio para codificação. O meio para codificação inclui meio de recebimento configurado para receber uma imagem para codificação ou para receber um fluxo de bits para decodificação; meio de transmissão acoplado ao meio de recebimento, o meio de transmissão configurado para transmitir o fluxo de bits para um meio de decodificação ou para transmitir uma imagem decodificada para um meio de exibição; meio de armazenamento acoplado a pelo menos um do meio de recebimento ou meio de transmissão, o meio de armazenamento configurado para armazenar instruções; e meio de processamento acoplado ao meio de armazenamento, o meio de processamento configurado para executar as instruções armazenadas no meio de armazenamento para realizar qualquer um dos métodos divulgados neste documento.

[0037] O meio para codificação fornece técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de

parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0038] Para fins de clareza, qualquer uma das modalidades anteriores pode ser combinada com qualquer uma ou mais das outras modalidades anteriores para criar uma nova modalidade dentro do escopo da presente revelação.

[0039] Estes e outros recursos serão mais claramente compreendidos a partir da seguinte descrição detalhada tomada em conjunto com os desenhos e reivindicações anexas.

BREVE DESCRIÇÃO DOS DESENHOS

[0040] Para uma compreensão mais completa desta revelação, agora é feita referência à seguinte breve descrição, tomada em conexão com os desenhos anexos e a descrição detalhada, em que números de referência semelhantes representam partes semelhantes.

[0041] A Figura 1 é um fluxograma de um exemplo de método de codificação de um sinal de vídeo.

[0042] A Figura 2 é um diagrama esquemático de um exemplo de sistema de codificação e decodificação (codec) para codificação de vídeo.

[0043] A Figura 3 é um diagrama esquemático ilustrando um exemplo de codificador de vídeo.

[0044] A Figura 4 é um diagrama esquemático ilustrando um exemplo de decodificador de vídeo.

[0045] A Figura 5 ilustra um exemplo de codificação de múltiplas camadas para escalabilidade espacial.

[0046] A Figura 6 ilustra um exemplo de codificação de múltiplas camadas usando conjuntos de camadas de saída (OLSs).

[0047] A Figura 7 ilustra uma modalidade de um fluxo de bits de vídeo.

[0048] A Figura 8 é uma modalidade de um método de decodificação de um fluxo de bits de vídeo codificado.

[0049] A Figura 9 é uma modalidade de um método de codificação de

um fluxo de bits de vídeo codificado.

[0050] A Figura 10 é um diagrama esquemático de um dispositivo de codificação de vídeo.

[0051] A Figura 11 é um diagrama esquemático de uma modalidade de um meio para codificação.

DESCRIÇÃO DETALHADA

[0052] Deve ser entendido desde o início que, embora uma implementação ilustrativa de uma ou mais modalidades seja fornecida abaixo, os sistemas e/ou métodos revelados podem ser implementados usando qualquer número de técnicas, atualmente conhecidas ou existentes. A revelação não deve de forma alguma ser limitada às implementações, desenhos e técnicas ilustrativas ilustrados abaixo, incluindo os projetos e implementações exemplares ilustrados e descritos neste documento, mas pode ser modificada dentro do escopo das reivindicações anexas juntamente com seu escopo completo de equivalentes.

[0053] Os seguintes termos são definidos como segue, a menos que sejam usados em um contexto contrário aqui. Especificamente, as seguintes definições destinam-se a fornecer clareza adicional à presente revelação. No entanto, os termos podem ser descritos de forma diferente em contextos diferentes. Consequentemente, as seguintes definições devem ser consideradas como um suplemento e não devem ser consideradas como limitantes de quaisquer outras definições de descrições fornecidas para tais termos neste documento.

[0054] Um fluxo de bits é uma sequência de bits incluindo dados de vídeo que são comprimidos para transmissão entre um codificador e um decodificador. Um codificador é um dispositivo que é configurado para empregar processos de codificação para comprimir dados de vídeo em um fluxo de bits. Um decodificador é um dispositivo que é configurado para empregar processos de decodificação para reconstruir dados de vídeo a partir de um fluxo de bits para exibição. Uma imagem é um arranjo de amostras de luma e/ou um arranjo de amostras de croma que criam um quadro ou um campo do mesmo. Uma imagem que está sendo codificada ou decodificada pode ser chamada de imagem atual para maior clareza da discussão. Uma imagem de referência é uma imagem que contém amostras de referência que podem ser usadas ao

codificar outras imagens por referência de acordo com predição inter e/ou predição de camada inter. Uma lista de imagens de referência é uma lista de imagens de referência usadas para predição inter e/ou predição de camada inter. Alguns sistemas de codificação de vídeo utilizam duas listas de imagens de referência, que podem ser denotadas como lista de imagens de referência um e lista de imagens de referência zero. Uma estrutura de lista de imagens de referência é uma estrutura de sintaxe endereçável que contém múltiplas listas de imagens de referência.

[0055] A predição inter é um mecanismo de codificação de amostras de uma imagem atual por referência a amostras indicadas em uma imagem de referência que é diferente da imagem atual em que a imagem de referência e a imagem atual estão na mesma camada. Uma entrada de estrutura de lista de imagens de referência é uma localização endereçável em uma estrutura de lista de imagens de referência que indica uma imagem de referência associada a uma lista de imagens de referência.

[0056] Um cabeçalho de fatia é uma parte de uma fatia codificada contendo elementos de dados pertencentes a todos os dados de vídeo dentro de um tile representado na fatia. Um conjunto de parâmetros de imagem (PPS) é um conjunto de parâmetros que contém dados referentes a uma imagem inteira. Mais especificamente, o PPS é uma estrutura de sintaxe contendo elementos de sintaxe que se aplicam a zero ou mais imagens codificadas inteiras conforme determinado por um elemento de sintaxe encontrado em cada cabeçalho de imagem. Um conjunto de parâmetros de sequência (SPS) é um conjunto de parâmetros que contém dados referentes a uma sequência de imagens. Uma unidade de acesso (AU) é um conjunto de uma ou mais imagens codificadas associadas ao mesmo tempo de exibição (por exemplo, a mesma contagem de ordem de imagem) para emissão a partir de um buffer de imagem decodificada (DPB) (por exemplo, para exibição a um usuário). Um delimitador de unidade de acesso (AUD) é um indicador ou estrutura de dados usado para indicar o início de uma AU ou a fronteira entre AUs. Uma sequência de vídeo decodificada é uma sequência de imagens que foram reconstruídas por um decodificador em preparação para exibição a um usuário.

[0057] Uma unidade de camada de abstração de rede (NAL) é uma estrutura de sintaxe contendo dados na forma de uma Carga Útil de Sequência

de Bytes Brutos (RBSP), uma indicação do tipo de dados e intercalada conforme desejado com bytes de prevenção de emulação. Uma unidade NAL de camada de codificação de vídeo (VCL) é uma unidade NAL codificada para conter dados de vídeo, como uma fatia codificada de uma imagem. Uma unidade não VCL NAL é uma unidade NAL que contém dados que não são de vídeo, como sintaxe e/ou parâmetros que suportam a decodificação dos dados de vídeo, desempenho de verificação de conformidade ou outras operações. Uma camada é um conjunto de unidades VCL NAL que compartilham uma característica especificada (por exemplo, uma resolução comum, taxa de quadros, tamanho de imagem etc.) e unidades não VCL NAL associadas. As unidades VCL NAL de uma camada podem compartilhar um valor particular de um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id). Uma imagem codificada é uma representação codificada de uma imagem compreendendo unidades VCL NAL com um valor particular de um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id) dentro de uma unidade de acesso (AU) e contendo todas as unidades de árvore de codificação (CTUs) da imagem. Uma imagem decodificada é uma imagem produzida pela aplicação de um processo de decodificação a uma imagem codificada.

[0058] Um conjunto de camadas de saída (OLS) é um conjunto de camadas para o qual uma ou mais camadas são especificadas como camada(s) de saída. Uma camada de saída é uma camada que é designada para emissão (por exemplo, para um display). Um zero^{ésimo} (0^{ésimo}) OLS é um OLS que contém apenas uma camada mais baixa (camada com um identificador de camada mais baixo) e, portanto, contém apenas uma camada de saída. O número de camadas em um OLS pode ser designado por um elemento de sintaxe designado NumLayersInOLS. Um conjunto de parâmetros de vídeo (VPS) é uma unidade de dados que contém parâmetros referentes a um vídeo inteiro. A predição de camada inter é um mecanismo de codificação de uma imagem atual em uma camada atual por referência a uma imagem de referência em uma camada de referência, em que a imagem atual e a imagem de referência estão incluídas na mesma AU e a camada de referência inclui um nuh_layer_id menor que a camada atual.

[0059] Um perfil é um conjunto definido de ferramentas de codificação usadas para criar um fluxo de bits compatível ou conforme. Cada perfil especifica

um subconjunto de recursos e limites algorítmicos que devem ser suportados por todos os decodificadores em conformidade com esse perfil. Tiers e níveis são restrições que definem um fluxo de bits em termos de taxa de bits máxima, taxa de amostra de luma máxima, tamanho máximo de imagem de luma, taxa de compressão mínima, número máximo de fatias permitidas e número máximo de tiles permitidos. Os tiers inferiores são mais restringidos do que os tiers superiores e os níveis inferiores são mais restringidos do que os níveis superiores. Os parâmetros de perfil, tier e nível (PTL) são uma estrutura de sintaxe que fornece informações de perfil, tier e/ou nível.

[0060] Um fluxo de bits escalável temporal é um fluxo de bits codificado em múltiplas camadas fornecendo resolução temporal/taxa de quadros variável (por exemplo, cada camada é codificada para suportar uma taxa de quadros diferente). Uma subcamada é uma camada escalável temporal de um fluxo de bits escalável temporal incluindo unidades VCL NAL com um valor de identificador temporal particular e unidades não VCL NAL associadas. Por exemplo, uma subcamada temporal é uma camada que contém dados de vídeo associados a uma taxa de quadros especificada. Uma representação de subcamada é um subconjunto do fluxo de bits contendo unidades NAL de uma subcamada particular e das subcamadas inferiores. Assim, uma ou mais subcamadas temporais podem ser combinadas para obter uma representação de subcamada que pode ser decodificada para resultar em uma sequência de vídeo com uma taxa de quadros especificada. Um índice de OLS é um índice que identifica exclusivamente um OLS correspondente. Um identificador (ID) temporal é um elemento de dados que indica que os dados correspondem à localização temporal em uma sequência de vídeo. Um processo de extração de subfluxo de bits é um processo que remove unidades NAL a partir de um fluxo de bits que não pertence a um conjunto alvo conforme determinado por um índice de OLS alvo e um ID temporal mais alto alvo. O processo de extração de subfluxo de bits resulta em um subfluxo de bits de saída contendo unidades NAL a partir do fluxo de bits que fazem parte do conjunto alvo.

[0061] Um decodificador de referência hipotético (HRD) é um modelo de decodificador operando em um codificador que verifica a variabilidade de fluxos de bits produzidos por um processo de codificação para verificar a conformidade com as restrições especificadas. Um teste de conformidade de

fluxo de bits é um teste para determinar se um fluxo de bits codificado está em conformidade com um padrão, como Codificação de Vídeo Versátil (VVC). Os parâmetros de HRD são elementos de sintaxe que inicializam e/ou definem as condições operacionais de um HRD. Os parâmetros de HRD de nível de sequência são parâmetros de HRD que se aplicam a uma sequência de vídeo codificada inteira. Um identificador (ID) temporal de HRD máximo (hrd_max_tid[i]) especifica o ID temporal da representação de subcamada mais alta para a qual os parâmetros de HRD estão contidos em um i^{ésimo} conjunto de parâmetros de OLS HRD. Uma estrutura de sintaxe de parâmetros de HRD gerais (general_hrd_parameters) é uma estrutura de sintaxe que contém parâmetros de HRD de nível de sequência. Um ponto de operação (OP) é um subconjunto temporal de um OLS que é identificado por um índice de OLS e um ID temporal mais alto. Um OP em teste (targetOp) é um OP selecionado para teste de conformidade em um HRD. Um OLS alvo é um OLS selecionado para extração de um fluxo de bits. Uma flag de presença de parâmetros de HRD de unidade de decodificação (decoding_unit_hrd_params_present_flag) é uma flag que indica se os parâmetros de HRD correspondentes operam em um nível de unidade de decodificação (DU) ou em um nível de AU. Um buffer de imagem codificada (CPB) é um buffer de primeiro a entrar, primeiro a sair em um HRD que contém imagens codificadas em ordem de decodificação para uso durante a verificação de conformidade de fluxo de bits. Um buffer de imagem decodificada (DPB) é um buffer para armazenar imagens decodificadas para referência, reordenação de emissão e/ou atraso de emissão.

[0062] Os parâmetros de buffer de imagem decodificada (DPB) são uma estrutura de sintaxe que fornece um tamanho de DPB e, opcionalmente, um número de reordenação de imagem máximo e informações de latência máxima. O número de reordenação de imagem máximo e as informações de latência máxima podem ser referidos coletivamente pela sigla MRML.

[0063] Os seguintes acrônimos são usados neste documento, Bloco de Árvore de Codificação (CTB), Unidade de Árvore de Codificação (CTU), Unidade de Codificação (CU), Sequência de Vídeo Codificada (CVS), Equipe Conjunta de Especialistas em Vídeo (JVET), Conjunto de Tiles com Movimento Restringido (MCTS), Unidade de Transferência Máxima (MTU), Camada de Abstração de Rede (NAL), Contagem de Ordem de Imagem (POC), Conjunto de

Parâmetros de Imagem (PPS), Carga Útil de Sequência de Byte Bruto (RBSP), Conjunto de Parâmetros de Sequência (SPS), Codificação de Vídeo Versátil (VVC) e Projeto de Trabalho (WD).

[0064] A Figura 1 é um fluxograma de um exemplo de método operacional 100 de codificação de um sinal de vídeo. Especificamente, um sinal de vídeo é codificado em um codificador. O processo de codificação comprime o sinal de vídeo empregando vários mecanismos para reduzir o tamanho de arquivo de vídeo. Um tamanho de arquivo menor permite que o arquivo de vídeo comprimido seja transmitido em direção a um usuário, reduzindo a sobrecarga de largura de banda associada. O decodificador então decodifica o arquivo de vídeo comprimido para reconstruir o sinal de vídeo original para exibição a um usuário final. O processo de decodificação geralmente espelha o processo de codificação para permitir que o decodificador reconstrua consistentemente o sinal de vídeo.

[0065] Na etapa 101, o sinal de vídeo é inserido no codificador. Por exemplo, o sinal de vídeo pode ser um arquivo de vídeo não comprimido armazenado na memória. Como outro exemplo, o arquivo de vídeo pode ser capturado por um dispositivo de captura de vídeo, como uma câmera de vídeo, e codificado para suportar transmissão em fluxo contínuo ao vivo do vídeo. O arquivo de vídeo pode incluir um componente de áudio e um componente de vídeo. O componente de vídeo contém uma série de quadros de imagem que, quando vistos em sequência, dão a impressão visual de movimento. Os quadros contêm pixels que são expressos em termos de luz, referidos neste documento como componentes de luma (ou amostras de luma), e cor, que é referida como componentes de croma (ou amostras de cor). Em alguns exemplos, os quadros também podem conter valores de profundidade para suportar a visualização tridimensional.

[0066] Na etapa 103, o vídeo é particionado em blocos. O particionamento inclui subdividir os pixels em cada quadro em blocos quadrados e/ou retangulares para compressão. Por exemplo, em Codificação de Vídeo de Alta Eficiência (HEVC) (também conhecida como H.265 e MPEG-H Part 2), o quadro pode ser dividido em unidades de árvore de codificação (CTUs), que são blocos de tamanho predefinido (por exemplo, sessenta e quatro pixels por sessenta e quatro pixels). As CTUs contêm amostras de luma e croma. Árvores

de codificação podem ser empregadas para dividir as CTUs em blocos e então subdividir recursivamente os blocos até que sejam alcançadas configurações que suportem codificação adicional. Por exemplo, os componentes de luma de um quadro podem ser subdivididos até que os blocos individuais contenham valores de iluminação relativamente homogêneos. Além disso, os componentes de croma de um quadro podem ser subdivididos até que os blocos individuais contenham valores de cor relativamente homogêneos. Assim, os mecanismos de particionamento variam dependendo do conteúdo dos quadros de vídeo.

[0067] Na etapa 105, vários mecanismos de compressão são empregados para comprimir os blocos de imagem particionados na etapa 103. Por exemplo, predição inter e/ou predição intra podem ser empregadas. A predição inter é projetada para aproveitar o fato de que objetos em uma cena comum tendem a aparecer em quadros sucessivos. Consequentemente, um bloco representando um objeto em um quadro de referência não precisa ser descrito repetidamente em quadros adjacentes. Especificamente, um objeto, como uma mesa, pode permanecer em uma posição constante em múltiplos quadros. Portanto, a mesa é descrita uma vez e os quadros adjacentes podem se referir ao quadro de referência. Mecanismos de correspondência de padrão podem ser empregados para combinar objetos em múltiplos quadros. Além disso, objetos em movimento podem ser representados em múltiplos quadros, por exemplo, devido ao movimento de objeto ou movimento de câmera. Como exemplo particular, um vídeo pode mostrar um automóvel que se move pela tela em múltiplos quadros. Vetores de movimento podem ser empregados para descrever tal movimento. Um vetor de movimento é um vetor bidimensional que fornece um deslocamento a partir das coordenadas de um objeto em um quadro para as coordenadas do objeto em um quadro de referência. Como tal, a predição inter pode codificar um bloco de imagem em um quadro atual como um conjunto de vetores de movimento indicando um deslocamento a partir de um bloco correspondente em um quadro de referência.

[0068] A predição intra codifica blocos em um quadro comum. A predição intra tira proveito do fato de que os componentes de luma e croma tendem a se agrupar em um quadro. Por exemplo, uma mancha verde em uma parte de uma árvore tende a ser posicionada adjacente a manchas verdes semelhantes. A predição intra emprega vários modos de predição direcional (por

exemplo, trinta e três em HEVC), um modo planar e um modo de corrente contínua (DC). Os modos direcionais indicam que um bloco atual é semelhante/igual às amostras de um bloco vizinho em uma direção correspondente. O modo planar indica que uma série de blocos ao longo de uma linha/coluna (por exemplo, um plano) pode ser interpolada com base em blocos vizinhos nas bordas da linha. O modo planar, na verdade, indica uma transição suave de luz/cor através de uma linha/coluna, empregando uma inclinação relativamente constante na mudança de valores. O modo DC é empregado para suavização de fronteira e indica que um bloco é semelhante/igual a um valor médio associado a amostras de todos os blocos vizinhos associados às direções angulares dos modos de predição direcional. Consequentemente, os blocos de predição intra podem representar blocos de imagem como vários valores de modo de predição relacional em vez dos valores reais. Além disso, os blocos de predição inter podem representar blocos de imagem como valores de vetor de movimento em vez dos valores reais. Em ambos os casos, os blocos de predição podem não representar exatamente os blocos de imagem em alguns casos. Quaisquer diferenças são armazenadas em blocos residuais. Transformadas podem ser aplicadas aos blocos residuais para comprimir ainda mais o arquivo.

[0069] Na etapa 107, várias técnicas de filtragem podem ser aplicadas. Em HEVC, os filtros são aplicados de acordo com um esquema de filtragem in-loop. A predição baseada em bloco discutida acima pode resultar na criação de imagens em bloco no decodificador. Além disso, o esquema de predição baseado em bloco pode codificar um bloco e, em seguida, reconstruir o bloco codificado para uso posterior como um bloco de referência. O esquema de filtragem in-loop aplica iterativamente filtros de supressão de ruído, filtros de desblocagem, filtros loop adaptativos e filtros de deslocamento adaptativo de amostra (SAO) aos blocos/quadros. Esses filtros atenuam esses artefatos de blocagem para que o arquivo codificado possa ser reconstruído com precisão. Além disso, esses filtros atenuam os artefatos nos blocos de referência reconstruídos para que os artefatos tenham menos probabilidade de criar artefatos adicionais nos blocos subsequentes que são codificados com base nos blocos de referência reconstruídos.

[0070] Uma vez que o sinal de vídeo tenha sido particionado, comprimido e filtrado, os dados resultantes são codificados em um fluxo de bits

na etapa 109. O fluxo de bits inclui os dados discutidos acima, bem como quaisquer dados de sinalização desejados para suportar a reconstrução adequada do sinal de vídeo no decodificador. Por exemplo, esses dados podem incluir dados de partição, dados de predição, blocos residuais e várias flags que fornecem instruções de codificação ao decodificador. O fluxo de bits pode ser armazenado na memória em direção a transmissão para um decodificador mediante solicitação. O fluxo de bits também pode ser transmitido por broadcast e/ou transmitido por multicast em direção a uma pluralidade de decodificadores. A criação do fluxo de bits é um processo iterativo. Consequentemente, as etapas 101, 103, 105, 107 e 109 podem ocorrer continuamente e/ou simultaneamente em muitos quadros e blocos. A ordem mostrada na Figura 1 é apresentada para clareza e facilidade de discussão e não se destina a limitar o processo de codificação de vídeo a uma ordem particular.

[0071] O decodificador recebe o fluxo de bits e inicia o processo de decodificação na etapa 111. Especificamente, o decodificador emprega um esquema de decodificação de entropia para converter o fluxo de bits na sintaxe e dados de vídeo correspondentes. O decodificador emprega os dados de sintaxe do fluxo de bits para determinar as partições para os quadros na etapa 111. O particionamento deve corresponder aos resultados da partição de bloco na etapa 103. A codificação/decodificação de entropia como empregada na etapa 111 é agora descrita. O codificador faz muitas escolhas durante o processo de compressão, como selecionar esquemas de particionamento de bloco a partir de várias opções possíveis com base no posicionamento espacial dos valores na(s) imagem(ns) de entrada. A sinalização das escolhas exatas pode empregar um grande número de bins. Conforme usado aqui, um bin é um valor binário que é tratado como uma variável (por exemplo, um valor de bit que pode variar dependendo do contexto). A codificação de entropia permite que o codificador descarte quaisquer opções que claramente não sejam viáveis para um caso particular, deixando um conjunto de opções permitidas. Cada opção permitida é então atribuída a uma palavra-código. O comprimento das palavras-código é baseado no número de opções permitidas (por exemplo, um bin para duas opções, dois bins para três a quatro opções, etc.) O codificador então codifica a palavra-código para a opção selecionada. Esse esquema reduz o tamanho das palavras-código, pois as palavras-código são tão grandes quanto desejado para

indicar exclusivamente uma seleção de um pequeno subconjunto de opções permitidas, em vez de indicar exclusivamente a seleção de um conjunto potencialmente grande de todas as opções possíveis. O decodificador então decodifica a seleção por determinar o conjunto de opções permitidas de maneira semelhante ao codificador. Ao determinar o conjunto de opções permitidas, o decodificador pode ler a palavra-código e determinar a seleção feita pelo codificador.

[0072] Na etapa 113, o decodificador realiza decodificação do bloco. Especificamente, o decodificador emprega transformadas reversas para gerar blocos residuais. Em seguida, o decodificador emprega os blocos residuais e os blocos de predição correspondentes para reconstruir os blocos de imagem de acordo com o particionamento. Os blocos de predição podem incluir blocos de predição intra e blocos de predição inter como gerados no codificador na etapa 105. Os blocos de imagem reconstruídos são então posicionados em quadros de um sinal de vídeo reconstruído de acordo com os dados de particionamento determinados na etapa 111. Sintaxe para a etapa 113 também pode ser sinalizada no fluxo de bits via codificação de entropia, conforme discutido acima.

[0073] Na etapa 115, a filtragem é realizada nos quadros do sinal de vídeo reconstruído de uma maneira semelhante à etapa 107 no codificador. Por exemplo, filtros de supressão de ruído, filtros de desblocagem, filtros loop adaptativos e filtros de SAO podem ser aplicados aos quadros para remover artefatos de blocagem. Uma vez que os quadros são filtrados, o sinal de vídeo pode ser enviado para um display na etapa 117 para visualização por um usuário final.

[0074] A Figura 2 é um diagrama esquemático de um exemplo de sistema de codificação e decodificação (codec) 200 para codificação de vídeo. Especificamente, o sistema codec 200 fornece funcionalidade para suportar a implementação do método operacional 100. O sistema codec 200 é generalizado para representar componentes empregados tanto em um codificador quanto em um decodificador. O sistema codec 200 recebe e particiona um sinal de vídeo como discutido em relação às etapas 101 e 103 no método de operação 100, que resulta em um sinal de vídeo particionado 201. O sistema codec 200 então comprime o sinal de vídeo particionado 201 em um fluxo de bits codificado ao atuar como um codificador conforme discutido em relação às etapas 105, 107 e

109 no método 100. Ao atuar como um decodificador, o sistema codec 200 gera um sinal de vídeo de saída a partir do fluxo de bits conforme discutido em relação às etapas 111, 113, 115 e 117 no método de operação 100. O sistema codec 200 inclui um componente de controle de codificador geral 211, um componente de escalonamento de transformada e quantização 213, um componente de estimação de imagem intra 215, um componente de predição de imagem intra 217, um componente de compensação de movimento 219, um componente de estimação de movimento 221, um componente de escalonamento e transformada inversa 229, um componente de análise de controle de filtro 227, um componente de filtros in-loop 225, um componente de buffer de imagem decodificada 223 e um componente de formatação de cabeçalho e codificação aritmética binária adaptativa ao contexto (CABAC) 231. Tais componentes são acoplados como mostrado. Na Figura 2, as linhas pretas indicam o movimento dos dados a serem codificados/decodificados enquanto as linhas tracejadas indicam o movimento dos dados de controle que controlam a operação de outros componentes. Os componentes do sistema codec 200 podem estar todos presentes no codificador. O decodificador pode incluir um subconjunto dos componentes do sistema codec 200. Por exemplo, o decodificador pode incluir o componente de predição de imagem intra 217, o componente de compensação de movimento 219, o componente de escalonamento e transformada inversa 229, o componente de filtros in-loop 225, e o componente de buffer de imagem decodificada 223. Esses componentes são agora descritos.

[0075] O sinal de vídeo particionado 201 é uma sequência de vídeo capturada que foi particionada em blocos de pixels por uma árvore de codificação. Uma árvore de codificação emprega vários modos de divisão para subdividir um bloco de pixels em blocos menores de pixels. Esses blocos podem então ser subdivididos em blocos menores. Os blocos podem ser referidos como nós na árvore de codificação. Os nós pais maiores são divididos em nós filhos menores. O número de vezes que um nó é subdividido é referido como a profundidade do nó/árvore de codificação. Os blocos divididos podem ser incluídos em unidades de codificação (CUs) em alguns casos. Por exemplo, uma CU pode ser uma subporção de uma CTU que contém um bloco de luma, bloco(s) de croma de diferença vermelha (Cr) e um(ns) bloco(s) de croma de diferença azul (Cb) juntamente com as instruções de sintaxe correspondentes

para a CU. Os modos de divisão podem incluir uma árvore binária (BT), árvore tripla (TT) e uma quadtree (QT) empregadas para particionar um nó em dois, três ou quatro nós filhos, respectivamente, de formas variadas, dependendo dos modos de divisão empregados. O sinal de vídeo particionado 201 é encaminhado para o componente de controle de codificador geral 211, o componente de escalonamento de transformada e quantização 213, o componente de estimação de imagem intra 215, o componente de análise de controle de filtro 227 e o componente de estimação de movimento 221 para compressão.

[0076] O componente de controle de codificador geral 211 é configurado para tomar decisões relacionadas à codificação das imagens da sequência de vídeo no fluxo de bits de acordo com as restrições de aplicação. Por exemplo, o componente de controle de codificador geral 211 gerencia a otimização da taxa de bits/tamanho de fluxo de bits versus qualidade de reconstrução. Tais decisões podem ser tomadas com base na disponibilidade de espaço de armazenamento/largura de banda e solicitações de resolução de imagem. O componente de controle de codificador geral 211 também gerencia a utilização de buffer à luz da velocidade de transmissão para mitigar questões de insuficiência (underrun) ou saturação (overrun) de buffer. Para gerenciar esses problemas, o componente de controle de codificador geral 211 gerencia o particionamento, predição e filtragem pelos outros componentes. Por exemplo, o componente de controle de codificador geral 211 pode aumentar dinamicamente a complexidade de compressão para aumentar a resolução e aumentar o uso de largura de banda ou diminuir a complexidade de compressão para diminuir a resolução e o uso de largura de banda. Assim, o componente de controle de codificador geral 211 controla os outros componentes do sistema codec 200 para equilibrar a qualidade de reconstrução do sinal de vídeo com preocupações de taxa de bits. O componente de controle de codificador geral 211 cria dados de controle, que controlam a operação dos outros componentes. Os dados de controle também são encaminhados para o componente de formatação de cabeçalho e CABAC 231 para serem codificados no fluxo de bits para sinalizar parâmetros para decodificação no decodificador.

[0077] O sinal de vídeo particionado 201 também é enviado para o componente de estimação de movimento 221 e o componente de compensação de movimento 219 para predição inter. Um quadro ou fatia do sinal de vídeo

particionado 201 pode ser dividido em múltiplos blocos de vídeo. O componente de estimação de movimento 221 e o componente de compensação de movimento 219 realizam codificação preditiva de modo inter do bloco de vídeo recebido em relação a um ou mais blocos em um ou mais quadros de referência para fornecer predição temporal. O sistema codec 200 pode realizar múltiplas passagens de codificação, por exemplo, para selecionar um modo de codificação apropriado para cada bloco de dados de vídeo.

[0078] O componente de estimação de movimento 221 e o componente de compensação de movimento 219 podem ser altamente integrados, mas são ilustrados separadamente para fins conceituais. A estimação de movimento, realizada pelo componente de estimação de movimento 221, é o processo de geração de vetores de movimento, que estima o movimento para blocos de vídeo. Um vetor de movimento, por exemplo, pode indicar o deslocamento de um objeto codificado em relação a um bloco preditivo. Um bloco preditivo é um bloco que corresponde ao bloco a ser codificado, em termos de diferença de pixel. Um bloco preditivo também pode ser referido como um bloco de referência. Essa diferença de pixel pode ser determinada pela soma de diferença absoluta (SAD), soma de diferença quadrada (SSD) ou outras métricas de diferença. HEVC emprega vários objetos codificados, incluindo uma CTU, blocos de árvore de codificação (CTBs) e CUs. Por exemplo, uma CTU pode ser dividida em CTBs, que podem então ser divididas em CBs para inclusão em CUs. Uma CU pode ser codificada como uma unidade de predição (PU) contendo dados de predição e/ou uma unidade de transformada (TU) contendo dados residuais transformados para a CU. O componente de estimação de movimento 221 gera vetores de movimento, PUs e TUs usando uma análise de taxa-distorção como parte de um processo de otimização de taxa-distorção. Por exemplo, o componente de estimação de movimento 221 pode determinar vários blocos de referência, vários vetores de movimento, etc. para um bloco/quadro atual e pode selecionar os blocos de referência, vetores de movimento, etc. com as melhores características de taxa-distorção. As melhores características de taxa-distorção equilibram a qualidade da reconstrução de vídeo (por exemplo, quantidade de perda de dados por compressão) com eficiência de codificação (por exemplo, tamanho da codificação final).

[0079] Em alguns exemplos, o sistema codec 200 pode calcular

valores para posições de pixel subinteiras de imagens de referência armazenadas no componente de buffer de imagem decodificada 223. Por exemplo, o sistema codec de vídeo 200 pode interpolar valores de um quarto de posições de pixel, um oitavo de posições de pixel ou outras posições de pixel fracionárias da imagem de referência. Portanto, o componente de estimação de movimento 221 pode realizar uma pesquisa de movimento em relação às posições de pixel completas e posições de pixel fracionárias e emitir um vetor de movimento com precisão de pixel fracionária. O componente de estimação de movimento 221 calcula um vetor de movimento para uma PU de um bloco de vídeo em uma fatia codificada de modo inter comparando a posição da PU com a posição de um bloco preditivo de uma imagem de referência. O componente de estimação de movimento 221 emite o vetor de movimento calculado como dados de movimento para componente de formatação de cabeçalho e CABAC 231 para codificação e movimento para o componente de compensação de movimento 219.

[0080] A compensação de movimento, realizada pelo componente de compensação de movimento 219, pode envolver buscar ou gerar o bloco preditivo com base no vetor de movimento determinado pelo componente de estimação de movimento 221. Novamente, o componente de estimação de movimento 221 e o componente de compensação de movimento 219 podem ser funcionalmente integrados, em alguns exemplos. Ao receber o vetor de movimento para a PU do bloco de vídeo atual, o componente de compensação de movimento 219 pode localizar o bloco preditivo para o qual o vetor de movimento aponta. Um bloco de vídeo residual é então formado subtraindo os valores de pixel do bloco preditivo a partir dos valores de pixel do bloco de vídeo atual que está sendo codificado, formando valores de diferença de pixel. Em geral, o componente de estimação de movimento 221 realiza estimação de movimento em relação aos componentes de luma, e o componente de compensação de movimento 219 usa vetores de movimento calculados com base nos componentes de luma para ambos os componentes de croma e componentes de luma. O bloco preditivo e o bloco residual são encaminhados para componente de escalonamento de transformada e quantização 213.

[0081] O sinal de vídeo particionado 201 também é enviado para o componente de estimação de imagem intra 215 e componente de predição de

imagem intra 217. Tal como acontece com o componente de estimação de movimento 221 e componente de compensação de movimento 219, componente de estimação de imagem intra 215 e componente de predição de imagem intra 217 podem ser altamente integrados, mas são ilustrados separadamente para fins conceituais. O componente de estimação de imagem intra 215 e o componente de predição de imagem intra 217 predizem de modo intra um bloco atual em relação aos blocos em um quadro atual, como uma alternativa à predição inter realizada pelo componente de estimação de movimento 221 e componente de compensação de movimento 219 entre quadros, como descrito acima. Em particular, o componente de estimação de imagem intra 215 determina um modo de predição intra a ser usado para codificar um bloco atual. Em alguns exemplos, o componente de estimação de imagem intra 215 seleciona um modo de predição intra apropriado para codificar um bloco atual a partir de múltiplos modos de predição intra testados. Os modos de predição intra selecionados são então encaminhados para o componente de formatação de cabeçalho e CABAC 231 para codificação.

[0082] Por exemplo, o componente de estimação de imagem intra 215 calcula os valores de taxa-distorção usando uma análise de taxa-distorção para os vários modos de predição intra testados, e seleciona o modo de predição intra com as melhores características de taxa-distorção entre os modos testados. A análise de taxa-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 (por exemplo, um número de bits) usada para produzir o bloco codificado. O componente de estimação de imagem intra 215 calcula as razões a partir das distorções e taxas para os vários blocos codificados para determinar qual modo de predição intra exibe o melhor valor de taxa-distorção para o bloco. Além disso, o componente de estimação de imagem intra 215 pode ser configurado para codificar blocos de profundidade de um mapa de profundidade usando um modo de modelagem de profundidade (DMM) com base na otimização de taxa-distorção (RDO).

[0083] O componente de predição de imagem intra 217 pode gerar um bloco residual a partir do bloco preditivo com base nos modos de predição intra selecionados determinados pelo componente de estimação de imagem intra 215

quando implementado em um codificador ou ler o bloco residual a partir do fluxo de bits quando implementado em um decodificador. O bloco residual inclui a diferença de valores entre o bloco preditivo e o bloco original, representado como uma matriz. O bloco residual é então encaminhado para o componente de escalonamento de transformada e quantização 213. O componente de estimação de imagem intra 215 e o componente de predição de imagem intra 217 podem operar em ambos os componentes de luma e croma.

[0084] O componente de escalonamento de transformada e quantização 213 é configurado para comprimir adicionalmente o bloco residual. O componente de escalonamento de transformada e quantização 213 aplica uma transformada, tal como uma transformada de cosseno discreta (DCT), uma transformada de seno discreta (DST), ou uma transformada conceitualmente similar, ao bloco residual, produzindo um bloco de vídeo compreendendo valores de coeficiente de transformada residual. Transformadas wavelet, transformadas inteiras, transformadas de sub-banda ou outros tipos de transformadas também podem ser usadas. A transformada pode converter as informações residuais a partir de um domínio de valor de pixel para um domínio de transformada, como um domínio de frequência. O componente de escalonamento de transformada e quantização 213 também é configurado para escalonar as informações residuais transformadas, por exemplo, com base na frequência. Tal escalonamento envolve a aplicação de um fator de escala às informações residuais para que informações de frequência diferentes sejam quantizadas em diferentes granularidades, o que pode afetar a qualidade visual final do vídeo reconstruído. O componente de escalonamento de transformada e quantização 213 também é configurado para quantizar os coeficientes de transformada para reduzir adicionalmente a taxa de bits. O processo de quantização pode reduzir a profundidade de bits associada a alguns ou a todos os coeficientes. O grau de quantização pode ser modificado ajustando um parâmetro de quantização. Em alguns exemplos, o componente de escalonamento de transformada e quantização 213 pode então realizar uma varredura da matriz incluindo os coeficientes de transformada quantizados. Os coeficientes de transformada quantizados são encaminhados para o componente de formatação de cabeçalho e CABAC 231 para serem codificados no fluxo de bits.

[0085] O componente de escalonamento e transformada inversa 229

aplica uma operação reversa do componente de escalonamento de transformada e quantização 213 para suportar a estimação de movimento. O componente de escalonamento e transformada inversa 229 aplica escalonamento, transformação e/ou quantização inversos para reconstruir o bloco residual no domínio de pixel, por exemplo, para uso posterior como um bloco de referência que pode se tornar um bloco preditivo para outro bloco atual. O componente de estimação de movimento 221 e/ou componente de compensação de movimento 219 podem calcular um bloco de referência adicionando o bloco residual de volta a um bloco preditivo correspondente para uso na estimação de movimento de um bloco/quadro posterior. Os filtros são aplicados aos blocos de referência reconstruídos para mitigar os artefatos criados durante o escalonamento, a quantização e a transformada. Esses artefatos podem causar predição imprecisa (e criar artefatos adicionais) quando blocos subsequentes são preditos.

[0086] O componente de análise de controle de filtro 227 e o componente de filtros in-loop 225 aplicam os filtros aos blocos residuais e/ou aos blocos de imagem reconstruídos. Por exemplo, o bloco residual transformado do componente de escalonamento e transformada inversa 229 pode ser combinado com um bloco de predição correspondente a partir do componente de predição de imagem intra 217 e/ou componente de compensação de movimento 219 para reconstruir o bloco de imagem original. Os filtros podem então ser aplicados ao bloco de imagem reconstruído. Em alguns exemplos, os filtros podem ser aplicados aos blocos residuais. Tal como acontece com outros componentes na Figura 2, o componente de análise de controle de filtro 227 e o componente de filtros in-loop 225 são altamente integrados e podem ser implementados juntos, mas são representados separadamente para fins conceituais. Os filtros aplicados aos blocos de referência reconstruídos são aplicados a regiões espaciais particulares e incluem vários parâmetros para ajustar como esses filtros são aplicados. O componente de análise de controle de filtro 227 analisa os blocos de referência reconstruídos para determinar em que tais filtros devem ser aplicados e define os parâmetros correspondentes. Esses dados são encaminhados para o componente de formatação de cabeçalho e CABAC 231 como dados de controle de filtro para codificação. O componente de filtros in-loop 225 aplica tais filtros com base nos dados de controle de filtro. Os filtros

podem incluir um filtro de desblocagem, um filtro de supressão de ruído, um filtro de SAO e um filtro loop adaptativo. Tais filtros podem ser aplicados no domínio espacial/pixel (por exemplo, em um bloco de pixel reconstruído) ou no domínio da frequência, dependendo do exemplo.

[0087] Ao operar como um codificador, o bloco de imagem reconstruído filtrado, bloco residual e/ou bloco de predição são armazenados no componente de buffer de imagem decodificada 223 para uso posterior na estimação de movimento conforme discutido acima. Ao operar como um decodificador, o componente de buffer de imagem decodificada 223 armazena e encaminha os blocos reconstruídos e filtrados em direção a um display como parte de um sinal de vídeo de saída. O componente de buffer de imagem decodificada 223 pode ser qualquer dispositivo de memória capaz de armazenar blocos de predição, blocos residuais e/ou blocos de imagem reconstruídos.

[0088] O componente de formatação de cabeçalho e CABAC recebe os dados a partir dos vários componentes do sistema codec 200 e codifica esses dados em um fluxo de bits codificado para transmissão em direção a um decodificador. Especificamente, o componente de formatação de cabeçalho e CABAC 231 gera vários cabeçalhos para codificar dados de controle, como dados de controle gerais e dados de controle de filtro. Além disso, dados de predição, incluindo dados de predição intra e movimento, bem como dados residuais na forma de dados de coeficiente de transformada quantizados são todos codificados no fluxo de bits. O fluxo de bits final inclui todas as informações desejadas pelo decodificador para reconstruir o sinal de vídeo particionado original 201. Essas informações também podem incluir tabelas de índice de modo de predição intra (também referidas como tabelas de mapeamento de palavra-código), definições de contextos de codificação para vários blocos, indicações de modos de predição intra mais prováveis, uma indicação de informações de partição, etc. Tais dados podem ser codificados empregando codificação de entropia. Por exemplo, as informações podem ser codificadas empregando codificação de comprimento variável adaptativa ao contexto (CAVLC), CABAC, codificação aritmética binária adaptativa ao contexto baseada em sintaxe (SBAC), codificação de entropia de particionamento de intervalo de probabilidade (PIPE) ou outra técnica de codificação de entropia. Após a codificação de entropia, o fluxo de bits codificado pode ser transmitido para outro

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

[0089] A Figura 3 é um diagrama de blocos ilustrando um exemplo de codificador de vídeo 300. O codificador de vídeo 300 pode ser empregado para implementar as funções de codificação do sistema codec 200 e/ou implementar as etapas 101, 103, 105, 107 e/ou 109 do método operacional 100. Codificador 300 particiona um sinal de vídeo de entrada, resultando em um sinal de vídeo particionado 301, que é substancialmente similar ao sinal de vídeo particionado 201. O sinal de vídeo particionado 301 é então comprimido e codificado em um fluxo de bits por componentes do codificador 300.

[0090] Especificamente, o sinal de vídeo particionado 301 é encaminhado para um componente de predição de imagem intra 317 para predição intra. O componente de predição de imagem intra 317 pode ser substancialmente similar ao componente de estimação de imagem intra 215 e componente de predição de imagem intra 217. O sinal de vídeo particionado 301 também é encaminhado para um componente de compensação de movimento 321 para predição inter com base em blocos de referência em um componente de buffer de imagem decodificada 323. O componente de compensação de movimento 321 pode ser substancialmente similar ao componente de estimação de movimento 221 e componente de compensação de movimento 219. Os blocos de predição e blocos residuais do componente de predição de imagem intra 317 e o componente de compensação de movimento 321 são encaminhados para um componente de transformada e quantização 313 para transformada e quantização dos blocos residuais. O componente de transformada e quantização 313 pode ser substancialmente similar ao componente de escalonamento de transformada e quantização 213. Os blocos residuais transformados e quantizados e os blocos de predição correspondentes (juntamente com dados de controle associados) são encaminhados para um componente de codificação de entropia 331 para codificação em um fluxo de bits. O componente de codificação de entropia 331 pode ser substancialmente similar ao componente de formatação de cabeçalho e CABAC 231.

[0091] Os blocos residuais transformados e quantizados e/ou os blocos de predição correspondentes também são encaminhados a partir do componente de transformada e quantização 313 para um componente de

transformada inversa e quantização 329 para reconstrução em blocos de referência para uso pelo componente de compensação de movimento 321. O componente de transformada inversa e quantização 329 pode ser substancialmente similar ao componente de escalonamento e transformada inversa 229. Filtros in-loop em um componente de filtros in-loop 325 também são aplicados aos blocos residuais e/ou blocos de referência reconstruídos, dependendo do exemplo. O componente de filtros in-loop 325 pode ser substancialmente similar ao componente de análise de controle de filtro 227 e ao componente de filtros in-loop 225. O componente de filtros in-loop 325 pode incluir múltiplos filtros conforme discutido em relação ao componente de filtros in-loop 225. Os blocos filtrados são então armazenados em um componente de buffer de imagem decodificada 323 para uso como blocos de referência pelo componente de compensação de movimento 321. O componente de buffer de imagem decodificada 323 pode ser substancialmente similar ao componente de buffer de imagem decodificada 223.

[0092] A Figura 4 é um diagrama de blocos ilustrando um exemplo de decodificador de vídeo 400. O decodificador de vídeo 400 pode ser empregado para implementar as funções de decodificação do sistema codec 200 e/ou implementar as etapas 111, 113, 115 e/ou 117 do método de operação 100. O decodificador 400 recebe um fluxo de bits, por exemplo, de um codificador 300, e gera um sinal de vídeo de saída reconstruído com base no fluxo de bits para exibição a um usuário final.

[0093] O fluxo de bits é recebido por um componente de decodificação de entropia 433. O componente de decodificação de entropia 433 é configurado para implementar um esquema de decodificação de entropia, como CAVLC, CABAC, SBAC, codificação de PIPE ou outras técnicas de codificação de entropia. Por exemplo, o componente de decodificação de entropia 433 pode empregar informações de cabeçalho para fornecer um contexto para interpretar dados adicionais codificados como palavras-código no fluxo de bits. As informações decodificadas incluem quaisquer informações desejadas para decodificar o sinal de vídeo, como dados de controle geral, dados de controle de filtro, informações de partição, dados de movimento, dados de predição e coeficientes de transformada quantizados de blocos residuais. Os coeficientes de transformada quantizados são encaminhados para um componente de

transformada inversa e quantização 429 para reconstrução em blocos residuais. O componente de transformada inversa e quantização 429 pode ser similar ao componente de transformada inversa e quantização 329.

[0094] Os blocos residuais reconstruídos e/ou blocos de predição são encaminhados para o componente de predição de imagem intra 417 para reconstrução em blocos de imagem com base em operações de predição intra. O componente de predição de imagem intra 417 pode ser semelhante ao componente de estimação de imagem intra 215 e a componente de predição de imagem intra 217. Especificamente, o componente de predição de imagem intra 417 emprega modos de predição para localizar um bloco de referência no quadro e aplica um bloco residual ao resultado para reconstruir blocos de imagem preditos de modo intra. Os blocos de imagem preditos de modo intra reconstruídos e/ou os blocos residuais e os dados de predição inter correspondentes são encaminhados para um componente de buffer de imagem decodificada 423 através de um componente de filtros in-loop 425, que pode ser substancialmente similar ao componente de buffer de imagem decodificada 223 e componente de filtros in-loop 225, respectivamente. O componente de filtros in-loop 425 filtra os blocos de imagem reconstruídos, blocos residuais e/ou blocos de predição, e tais informações são armazenadas no componente de buffer de imagem decodificada 423. Blocos de imagem reconstruídos a partir do componente de buffer de imagem decodificada 423 são encaminhados para um componente de compensação de movimento 421 para predição inter. O componente de compensação de movimento 421 pode ser substancialmente similar ao componente de estimação de movimento 221 e/ou componente de compensação de movimento 219. Especificamente, o componente de compensação de movimento 421 emprega vetores de movimento a partir de um bloco de referência para gerar um bloco de predição e aplica um bloco residual ao resultado para reconstruir um bloco de imagem. Os blocos reconstruídos resultantes também podem ser encaminhados através do componente de filtros in-loop 425 para o componente de buffer de imagem decodificada 423. O componente de buffer de imagem decodificada 423 continua a armazenar blocos de imagem reconstruídos adicionais, que podem ser reconstruídos em quadros através das informações de partição. Esses quadros também podem ser colocados em sequência. A sequência é emitida em direção a um display como

um sinal de vídeo de saída reconstruído.

[0095] Mantendo o acima em mente, técnicas de compressão de vídeo realizam predição espacial (imagem intra) e/ou predição temporal (imagem inter) para reduzir ou remover redundância inerente em sequências de vídeo. Para codificação de vídeo baseada em bloco, uma fatia de vídeo (ou seja, uma imagem de vídeo ou uma parte de uma imagem de vídeo) pode ser particionada em blocos de vídeo, que também podem ser referidos como blocos de árvore, blocos de árvore de codificação (CTBs), unidades de árvore de codificação (CTUs), unidades de codificação (CUs) e/ou nós de codificação. Os blocos de vídeo em uma fatia codificada de modo intra (I) de uma imagem são codificados usando predição espacial com respeito a amostras de referência em blocos vizinhos na mesma imagem. Os blocos de vídeo em uma fatia codificada de modo inter (P ou B) de uma imagem podem usar predição espacial com respeito a amostras de referência em blocos vizinhos na mesma imagem ou predição temporal com respeito a amostras de referência em outras imagens de referência. As imagens podem ser chamadas de quadros e as imagens de referência podem ser chamadas de quadros de referência. A POC é uma variável associada a cada imagem que identifica exclusivamente a imagem associada entre todas as imagens na sequência de vídeo de camada codificada (CLVS), indica quando a imagem associada deve ser emitida a partir do DPB e indica a posição da imagem associada na ordem de emissão relativa às posições de ordem de emissão das outras imagens no mesmo CLVS que devem ser emitidas a partir do DPB. Uma flag é uma variável ou elemento de sintaxe de bit único que pode assumir um dos dois valores possíveis: 0 e 1.

[0096] Predição espacial ou temporal resulta em um bloco preditivo para um bloco a ser codificado. Dados residuais representam diferenças de pixel entre o bloco original a ser codificado e o bloco preditivo. Um bloco 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 codificado de modo intra é codificado de acordo com um modo de codificação intra e os dados residuais. Para compressão adicional, os dados residuais podem ser transformados a partir do domínio de pixel para um domínio de transformada, resultando em coeficientes de transformada residuais, que então

podem ser quantizados. Os coeficientes de transformada quantizados, inicialmente dispostos em um arranjo bidimensional, podem ser varridos a fim de produzir um vetor unidimensional de coeficientes de transformada e a codificação de entropia pode ser aplicada para obter ainda mais compressão.

[0097] Compressão de imagem e vídeo experimentou um rápido crescimento, levando a vários padrões de codificação. Esses padrões de codificação de vídeo incluem ITU-T H.261, Organização Internacional para Padronização/Comissão Eletrotécnica Internacional (ISO/IEC) MPEG-1 Parte 2, ITU-T H.262 ou ISO/IEC MPEG-2 Parte 2, ITU-T H.263, ISO/IEC MPEG-4 Parte 2, Codificação de Vídeo Avançada (AVC), também conhecida como ITU-T H.264 ou ISO/IEC MPEG-4 Parte 10, e Codificação de Vídeo de Alta Eficiência (HEVC), também conhecida como ITU-T H.265 ou MPEG-H Parte 2. AVC inclui extensões como codificação de vídeo escalável (SVC), codificação de vídeo multivistas (MVC) e codificação de vídeo multivistas mais profundidade (MVC + D) e AVC 3D (3D- AVC). HEVC inclui extensões como HEVC escalável (SHVC), HEVC multivistas (MV-HEVC) e HEVC 3D (3D-HEVC).

[0098] Há também um novo padrão de codificação de vídeo, denominado Codificação de Vídeo Versátil (VVC), sendo desenvolvido pela equipe conjunta de especialistas em vídeo (JVET) da ITU-T e ISO/IEC. Embora o padrão VVC tenha vários projetos de trabalho, um Projeto de Trabalho (WD) de VVC em particular, a saber, B. Bross, J. Chen e S. Liu, "Versatile Video Coding (Draft 5)", JVET-N1001-v3, 13ª Reunião JVET, 27 de março de 2019 (VVC Draft 5) é referenciado neste documento.

[0099] A escalabilidade na codificação de vídeo geralmente é suportada pelo uso de técnicas de codificação de múltiplas camadas. Um fluxo de bits de múltiplas camadas compreende uma camada base (BL) e uma ou mais camadas de enriquecimento (ELs). Um exemplo de escalabilidade inclui escalabilidade espacial, escalabilidade de qualidade/sinal-ruído (SNR), escalabilidade de múltiplas vistas, etc. Quando uma técnica de codificação de múltiplas camadas é usada, uma imagem ou parte dela pode ser codificada (1) sem usar uma imagem de referência, ou seja, usando a predição intra; (2) referenciando imagens de referência que estão na mesma camada, ou seja, usando predição inter; ou (3) fazendo referência a imagens de referência que estão em outra(s) camada(s), ou seja, usando predição de camada inter. Uma

imagem de referência usada para predição de camada inter da imagem atual é chamada de imagem de referência de camada inter (ILRP).

[0100] A Figura 5 é um diagrama esquemático ilustrando um exemplo de predição baseada em camada 500, por exemplo, conforme realizado para determinar vetores de movimento (MVs) na etapa de compressão de bloco 105, etapa de decodificação de bloco 113, componente de estimação de movimento 221, componente de compensação de movimento 219, componente de compensação de movimento 321, e/ou componente de compensação de movimento 421. A predição baseada em camada 500 é compatível com predição inter unidirecional e/ou predição inter bidirecional, mas também é realizada entre imagens em diferentes camadas.

[0101] A predição baseada em camada 500 é aplicada entre as imagens 511, 512, 513 e 514 e as imagens 515, 516, 517 e 518 em diferentes camadas. No exemplo mostrado, as imagens 511, 512, 513 e 514 fazem parte da camada N+1 532 e as imagens 515, 516, 517 e 518 fazem parte da camada N 531. Uma camada, como a camada N 531 e/ou camada N+1 532, é um grupo de imagens que estão todas associadas a um valor similar de uma característica, como tamanho, qualidade, resolução, relação sinal-ruído, capacidade, etc. No exemplo mostrado, camada N+1 532 é associada a um tamanho de imagem maior do que a camada N 531. Assim, as imagens 511, 512, 513 e 514 na camada N+1 532 têm um tamanho de imagem maior (por exemplo, maior altura e largura e, portanto, mais amostras) do que imagens 515, 516, 517 e 518 na camada N 531 neste exemplo. No entanto, tais imagens podem ser separadas entre a camada N+1 532 e a camada N 531 por outras características. Enquanto apenas duas camadas, camada N+1 532 e camada N 531, são mostradas, um conjunto de imagens pode ser separado em qualquer número de camadas com base nas características associadas. A camada N+1 532 e a camada N 531 também podem ser indicadas por um ID de camada. Um ID de camada é um item de dados associado a uma imagem e indica que a imagem faz parte de uma camada indicada. Consequentemente, cada imagem 511-518 pode ser associada a uma ID de camada correspondente para indicar qual camada N+1 532 ou camada N 531 inclui a imagem correspondente.

[0102] As imagens 511-518 em diferentes camadas 531-532 são configuradas para serem exibidas em alternativa. Como tal, as imagens 511-518

em diferentes camadas 531-532 podem compartilhar o mesmo identificador (ID) temporal e podem ser incluídas na mesma AU. Conforme usado neste documento, uma AU é um conjunto de uma ou mais imagens codificadas associadas ao mesmo tempo de exibição para emissão a partir de um DPB. Por exemplo, um decodificador pode decodificar e exibir a imagem 515 no tempo de exibição atual se uma imagem menor for desejada ou o decodificador pode decodificar e exibir a imagem 511 no tempo de exibição atual se uma imagem maior for desejada. Como tal, as imagens 511-514 na camada superior N+1 532 contêm substancialmente os mesmos dados de imagem que as imagens correspondentes 515-518 na camada inferior N 531 (não obstante a diferença no tamanho de imagem). Especificamente, a imagem 511 contém substancialmente os mesmos dados de imagem que a imagem 515, a imagem 512 contém substancialmente os mesmos dados de imagem que a imagem 516, etc.

[0103] As imagens 511-518 podem ser codificadas por referência a outras imagens 511-518 na mesma camada N 531 ou N+1 532. Codificar uma imagem em referência a outra imagem na mesma camada resulta na predição inter 523, que é compatível com predição inter unidirecional e/ou predição inter bidirecional. A predição inter 523 é representada por setas de linha sólida. Por exemplo, a imagem 513 pode ser codificada empregando predição inter 523 usando uma ou duas das imagens 511, 512 e/ou 514 na camada N+1 532 como referência, em que uma imagem é referenciada para predição inter unidirecional e/ ou duas imagens são referenciadas para predição inter bidirecional. Além disso, a imagem 517 pode ser codificada empregando predição inter 523 usando uma ou duas das imagens 515, 516 e/ou 518 na camada N 531 como referência, em que uma imagem é referenciada para predição inter unidirecional e/ou duas imagens são referenciadas para predição inter bidirecional. Quando uma imagem é usada como referência para outra imagem na mesma camada ao realizar a predição inter 523, a imagem pode ser referida como uma imagem de referência. Por exemplo, a imagem 512 pode ser uma imagem de referência usada para codificar a imagem 513 de acordo com a predição inter 523. A predição inter 523 também pode ser referida como predição de camada intra em um contexto de múltiplas camadas. Como tal, a predição inter 523 é um mecanismo de codificação de amostras de uma imagem atual por referência a amostras indicadas em uma imagem de referência que são diferentes da imagem

atual onde a imagem de referência e a imagem atual estão na mesma camada.

[0104] As imagens 511-518 também podem ser codificadas por referência a outras imagens 511-518 em diferentes camadas. Este processo é conhecido como predição de camada inter 521 e é representado por setas tracejadas. A predição de camada inter 521 é um mecanismo de codificação de amostras de uma imagem atual por referência a amostras indicadas em uma imagem de referência em que a imagem atual e a imagem de referência estão em camadas diferentes e, portanto, têm IDs de camada diferentes. Por exemplo, uma imagem em uma camada inferior N 531 pode ser usada como uma imagem de referência para codificar uma imagem correspondente em uma camada superior N+1 532. Como um exemplo específico, a imagem 511 pode ser codificada por referência à imagem 515 de acordo com a predição de camada inter 521. Nesse caso, a imagem 515 é usada como uma imagem de referência de camada inter. Uma imagem de referência de camada inter é uma imagem de referência usada para predição de camada inter 521. Na maioria dos casos, a predição de camada inter 521 é restringida de modo que uma imagem atual, como a imagem 511, apenas pode usar imagem(ns) de referência de camada inter que estão incluídas na mesma AU e que estão em uma camada inferior, como a imagem 515. Quando várias camadas (por exemplo, mais de duas) estão disponíveis, a predição de camada inter 521 pode codificar/decodificar uma imagem atual com base em múltiplas imagens de referência de camada inter em níveis mais baixos do que a imagem atual.

[0105] Um codificador de vídeo pode empregar predição baseada em camada 500 para codificar imagens 511-518 através de muitas combinações e/ou permutações diferentes de predição inter 523 e predição de camada inter 521. Por exemplo, a imagem 515 pode ser codificada de acordo com predição intra. As imagens 516-518 podem então ser codificadas de acordo com a predição inter 523 usando a imagem 515 como uma imagem de referência. Além disso, a imagem 511 pode ser codificada de acordo com a predição de camada inter 521 usando a imagem 515 como uma imagem de referência de camada inter. As imagens 512-514 podem então ser codificadas de acordo com a predição inter 523 usando a imagem 511 como uma imagem de referência. Como tal, uma imagem de referência pode servir como uma imagem de referência de camada única e uma imagem de referência de camada inter para

diferentes mecanismos de codificação. Ao codificar imagens de camada superior N+1 532 com base em imagens de camada inferior N 531, a camada superior N+1 532 pode evitar o emprego de predição intra, que tem uma eficiência de codificação muito menor do que a predição inter 523 e a predição de camada inter 521. Como tal, a baixa eficiência de codificação da predição intra pode ser limitada às imagens de menor/mais inferior qualidade e, portanto, limitada à codificação da menor quantidade de dados de vídeo. As imagens usadas como imagens de referência e/ou imagens de referência de camada inter podem ser indicadas nas entradas da(s) lista(s) de imagens de referência contidas em uma estrutura de lista de imagens de referência.

[0106] Cada AU 506 na Figura 5 pode conter várias imagens. Por exemplo, uma AU 506 pode conter imagens 511 e 515. Outra AU 506 pode conter imagens 512 e 516. De fato, cada AU 506 é um conjunto de uma ou mais imagens codificadas associadas ao mesmo tempo de exibição (por exemplo, o mesmo ID temporal) para emissão a partir de um buffer de imagem decodificada (DPB) (por exemplo, para exibição a um usuário). Cada AUD 508 é um indicador ou estrutura de dados usado para indicar o início de uma AU (por exemplo, AU 508) ou a fronteira entre AUs.

[0107] Famílias de codificação de vídeo H.26x anteriores forneciam suporte para escalabilidade em perfil(is) separado(s) do(s) perfil(is) para codificação de camada única. Codificação de vídeo escalável (SVC) é a extensão escalável do AVC/H.264 que fornece suporte para escalabilidade espacial, temporal e de qualidade. Para SVC, uma flag é sinalizada em cada macrobloco (MB) em imagens de EL para indicar se o EL MB é predito usando o bloco colocalizado a partir de uma camada inferior. A predição a partir do bloco colocalizado pode incluir textura, vetores de movimento e/ou modos de codificação. Implementações de SVC não podem reusar diretamente implementações de H.264/AVC não modificadas em seu projeto. A sintaxe de macrobloco SVC EL e processo de decodificação diferem da sintaxe H.264/AVC e do processo de decodificação.

[0108] HEVC escalável (SHVC) é a extensão do padrão HEVC/H.265 que fornece suporte para escalabilidade espacial e de qualidade, HEVC de multivistas (MV-HEVC) é a extensão do HEVC/H.265 que fornece suporte para escalabilidade de múltiplas vistas, e 3D HEVC (3D-HEVC) é a extensão do

HEVC/H.264 que fornece suporte para codificação de vídeo tridimensional (3D) que é mais avançada e mais eficiente do que MV-HEVC. Observe que a escalabilidade temporal é incluída como parte integrante do codec HEVC de camada única. O projeto da extensão de múltiplas camadas de HEVC emprega a ideia de que as imagens decodificadas usadas para predição de camada inter vêm apenas da mesma unidade de acesso (AU) e são tratadas como imagens de referência de longo prazo (LTRPs), e são referência índices atribuídos na(s) lista(s) de imagens de referência junto com outras imagens de referência temporal na camada atual. Predição de camada inter (PLI) é alcançada no nível de unidade de predição (PU) por definir o valor do índice de referência para referir-se à(s) imagem(ns) de referência de camada inter na(s) lista(s) de imagens de referência.

[0109] Notavelmente, ambos recursos de reamostragem de imagem de referência e escalabilidade espacial exigem reamostragem de uma imagem de referência ou parte dela. Reamostragem de imagem de referência (RPR) pode ser realizada no nível de imagem ou no nível de bloco de codificação. No entanto, quando o RPR é referido como um recurso de codificação, é um recurso para codificação de camada única. Mesmo assim, é possível, ou mesmo preferível, do ponto de vista de projeto de codec, usar o mesmo filtro de reamostragem para tanto o recurso de RPR de codificação de camada única quanto o recurso de escalabilidade espacial para codificação de múltiplas camadas.

[0110] A Figura 6 ilustra um exemplo de predição baseada em camada 600 utilizando conjuntos de camadas de saída (OLSs), por exemplo, conforme realizado para determinar MVs na etapa de compressão de bloco 105, etapa de decodificação de bloco 113, componente de estimação de movimento 221, componente de compensação de movimento 219, componente de compensação de movimento 321, e/ou componente de compensação de movimento 421. A predição baseada em camada 600 é compatível com predição inter unidirecional e/ou predição inter bidirecional, mas também é realizada entre imagens em diferentes camadas. A predição baseada em camada da Figura 6 é semelhante à da Figura 5. Portanto, por uma questão de brevidade, uma descrição completa da predição baseada em camada não é repetida.

[0111] Algumas das camadas na sequência de vídeo codificada

(CVS) 690 da Figura 6 estão incluídas em um OLS. Um OLS é um conjunto de camadas para as quais uma ou mais camadas são especificadas como camadas de saída. Uma camada de saída é uma camada de um OLS que é emitido. A Figura 6 representa três OLSs diferentes, nomeadamente OLS 1, OLS 2 e OLS 3. Como mostrado, OLS 1 inclui a Camada N 631 e a Camada N+1 632. O OLS 2 inclui a Camada N 631, Camada N+1 632, Camada N+2 633 e Camada N+3 634. O OLS 3 inclui a Camada N 631. Ou seja, o OLS 3 inclui apenas uma camada única. Em aplicações práticas, outros OLSs também podem conter apenas uma camada única. Apesar de três OLSs serem mostrados, um número diferente de OLSs pode ser usado em aplicações práticas. Outras camadas, como a camada N+4 635, não estão incluídas nos OLSs ilustrados (por exemplo, OLS 1, 2 e 3), mas estão disponíveis para inclusão em outros OLSs.

[0112] Cada um dos diferentes OLSs pode conter qualquer número de camadas. Os diferentes OLSs são gerados em um esforço para acomodar os recursos de codificação de uma variedade de dispositivos diferentes com recursos de codificação variados. Por exemplo, OLS 3, que contém apenas uma camada, pode ser gerado para acomodar um telefone móvel com capacidades de codificação relativamente limitadas. Por outro lado, o OLS 2, que contém quatro camadas, pode ser gerado para acomodar uma televisão de tela grande, capaz de decodificar camadas mais altas que o telefone móvel. O OLS 1, que contém duas camadas, pode ser gerado para acomodar um computador pessoal, laptop ou tablet, que pode decodificar camadas mais altas do que o telefone móvel, mas não pode decodificar as camadas mais altas, como a televisão de tela grande.

[0113] As camadas na Figura 6 podem ser todas independentes umas das outras. Ou seja, cada camada pode ser codificada sem usar a predição de camada inter (ILP). Nesse caso, as camadas são chamadas de camadas de simulcast. Uma ou mais das camadas na Figura 6 também podem ser codificadas usando ILP. Se as camadas são camadas de simulcast ou se algumas das camadas são codificadas usando ILP, isso é sinalizado por uma flag em um conjunto de parâmetros de vídeo (VPS), que será discutido mais detalhadamente abaixo. Quando algumas camadas utilizam ILP, a relação de dependência de camada entre as camadas também é sinalizada no VPS.

[0114] Em uma modalidade, quando as camadas são camadas de

simulcast, apenas uma camada é selecionada para decodificação e emissão. Em uma modalidade, quando algumas camadas usam ILP, todas as camadas (por exemplo, todo o fluxo de bits) são especificadas para serem decodificadas e certas camadas entre as camadas são especificadas para serem camadas de saída. A camada ou camadas de saída podem ser, por exemplo, 1) apenas a camada mais alta, 2) todas as camadas ou 3) a camada mais alta mais um conjunto de camadas inferiores indicadas. Por exemplo, quando a camada mais alta mais um conjunto de camadas inferiores indicadas são designadas para emissão por uma flag no VPS, a camada N+3 634 (que é a camada mais alta) e as camadas N 631 e N+1 632 (que são camadas inferiores) do OLS 2 são emitidas.

[0115] Como mostrado na Figura 6, cada camada é composta por qualquer número de imagens. Por exemplo, Camada N 631 inclui imagens 615-618, Camada N+1 632 inclui imagens 611-614, Camada N+2 633 inclui imagens 641-644, Camada N+3 634 inclui imagens 651-654 e Camada N+4 635 inclui as imagens 661-664.

[0116] Padrões de codificação de vídeo como HEVC, SHVC, MV-HEVC e VVC especificam e sinalizam um perfil, um tier e um nível. Perfis, tiers e níveis especificam restrições em fluxos de bits e, portanto, limitam as capacidades necessárias para decodificar os fluxos de bits. Perfis, tiers e níveis também podem ser usados para indicar pontos de interoperabilidade entre implementações de decodificador individuais. Um perfil é um conjunto definido de ferramentas de codificação usadas para criar um fluxo de bits compatível ou em conformidade. Cada perfil especifica um subconjunto de recursos e limites algorítmicos que devem ser suportados por todos os decodificadores em conformidade com esse perfil.

[0117] Um nível é um conjunto de restrições para um fluxo de bits (por exemplo, taxa de amostragem de luma máxima, taxa de bits máxima para uma resolução, etc.). Ou seja, um nível é um conjunto de restrições que indicam o desempenho do decodificador necessário para reproduzir um fluxo de bits do perfil especificado.

[0118] Os níveis são divididos em dois tiers: Principal e Alto. O tier principal é um tier inferior ao tier alto. Os tiers foram feitos para lidar com aplicações que diferem em termos de taxa de bits máxima. O tier principal foi

projetado para a maioria das aplicações, enquanto o tier alto foi projetado para aplicações muito exigentes. O conjunto de restrições e/ou tiers correspondentes aos níveis pode ser referido neste documento como informações de nível.

[0119] Cada nível de um tier especifica um conjunto de limites nos valores que podem ser tomados pelos elementos de sintaxe de um padrão particular (por exemplo, HEVC, VVC). O mesmo conjunto de definições de tier e nível é usado com todos os perfis, mas as implementações individuais podem dar suporte a um tier diferente e, dentro de um tier, um nível diferente para cada perfil suportado. Para qualquer perfil, um nível de um tier geralmente corresponde a uma capacidade de memória e carga de processamento de decodificador particular.

[0120] Usando o padrão HEVC, um exemplo representativo de níveis e tiers para diferentes perfis é fornecido na Tabela 1 abaixo.

Tabela 1

Nível	Taxa de bits máxima para perfil 4:4:4 12 principal (1000 bits/s)		Taxa de bits máxima para perfil intra 4:4:4 16 de alta taxa de transferência (1000 bits/s)		Exemplo de resolução de imagem na taxa de quadros mais alta (MaxDPBsize)
	Tier principal	Tier alto	Tier principal	Tier alto	
5.2	180.000	720.000	5.760.000	23.040.000	4.096×2.160@120,0
6	180.000	720.000	5.760.000	23.040.000	8.192×4.320@30,0
6.1	180.000	720.000	11.520.000	46.080.000	8.192×4.320@60,0
6.2	720.000	2.400.000	23.040.000	76.800.000	8.192×4.320@120,0

[0121] Os parâmetros de perfil, tier e nível (PTL) são uma estrutura de sintaxe (por exemplo, a i^{ésima} estrutura de sintaxe profile_tier_level()) que fornece perfil, tier e/ou nível conforme observado na tabela acima. Os parâmetros de buffer de imagem decodificada (DPB) são uma estrutura de sintaxe (por exemplo, a i^{ésima} estrutura de sintaxe dpb_parameters()) que fornece um tamanho de DPB e, opcionalmente, um número de reordenação de imagem máximo e informações de latência máxima. O número de reordenação de imagem máximo e as informações de latência máxima podem ser referidos coletivamente pela sigla MRML. Os parâmetros de HRD são uma estrutura de

sintaxe (por exemplo, i^{ésima} estrutura de sintaxe ols_timing_hrd_parameters()) que inicializa e/ou define as condições operacionais do HRD. Mais detalhes e especificidades sobre um ou mais dos parâmetros de PTL, os parâmetros de DPB e os parâmetros de HRD são fornecidos abaixo.

[0122] Em uma modalidade, o número de reordenação de imagem máximo é designado dpb_max_num_reorder_pics[i], que especifica o número permitido máximo de imagens do OLS que pode preceder qualquer imagem no OLS em ordem de decodificação e seguir essa imagem na ordem de emissão quando Htid é igual a i. O valor de dpb_max_num_reorder_pics[i] deve estar na faixa de 0 a dpb_max_dec_pic_buffering_minus1[i], inclusive. Quando i for maior que 0, dpb_max_num_reorder_pics[i] deve ser maior que ou igual a dpb_max_num_reorder_pics[i−1]. Quando dpb_max_num_reorder_pics[i] não está presente para i na faixa de 0 a MaxSubLayersMinus1 − 1, inclusive, devido a subLayerInfoFlag ser igual a 0, infere-se que seja igual a dpb_max_num_reorder_pics[MaxSubLayersMinus1]. Htid é uma variável que identifica a subcamada temporal mais alto a ser decodificada.

[0123] dpb_max_dec_pic_buffering_minus1[i] mais 1 especifica o tamanho requerido máximo do DPB em unidades de buffers de armazenamento de imagem quando Htid é igual a i. O valor de dpb_max_dec_pic_buffering_minus1[i] deve estar na faixa de 0 a MaxDpbSize − 1, inclusive, em que MaxDpbSize é conforme especificado na subcláusula A.4.2 do padrão VVC. Quando i for maior que 0, dpb_max_dec_pic_buffering_minus1[i] deve ser maior que ou igual a dpb_max_dec_pic_buffering_minus1[i−1]. Quando dpb_max_dec_pic_buffering_minus1[i] não está presente para i na faixa de 0 a MaxSubLayersMinus1 − 1, inclusive, devido a subLayerInfoFlag ser igual a 0, infere-se que seja igual a dpb_max_dec_pic_buffering_minus1[MaxSubLayersMinus1].

[0124] Em uma modalidade, as informações de latência máxima são designadas dpb_max_latency_increase_plus1[i]. dpb_max_latency_increase_plus1[i] diferente de 0 é usado para calcular o valor de MaxLatencyPictures[i], que especifica o número máximo de imagens no OLS que podem preceder qualquer imagem no OLS na ordem de emissão e seguir essa imagem na ordem de decodificação quando Htid é igual a i.

[0125] MaxLatencyPictures[i] = dpb_max_num_reorder_pics[i] + dpb_max_latency_increment_plus1[i] – 1.

[0126] As estruturas de sintaxe de DPB (também conhecidas como parâmetros de DPB) para os OLSs descritos acima estão incluídas no VPS do fluxo de bits. Infelizmente, sinalizar todas essas estruturas de sintaxe de DPB no VPS é ineficiente quando um OLS contém apenas uma camada única (por exemplo, OLS 3 na Figura 6).

[0127] São divulgadas neste documento técnicas que garantem que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0128] A Figura 7 ilustra uma modalidade de um fluxo de bits de vídeo 700. Conforme usado neste documento, o fluxo de bits de vídeo 700 também pode ser referido como um fluxo de bits de vídeo codificado, um fluxo de bits ou suas variações. Como mostrado na Figura 7, o fluxo de bits 700 inclui um conjunto de parâmetros de vídeo (VPS) 704, um conjunto de parâmetros de sequência (SPS) 706, um conjunto de parâmetros de imagem (PPS) 708, um cabeçalho de fatia 712 e dados de imagem 720. Cada um do VPS 704, o SPS 706 e o PPS 708 pode ser genericamente referido como um conjunto de parâmetros. Em uma modalidade, outros conjuntos de parâmetros e/ou cabeçalhos não mostrados na Figura 7 também podem ser incluídos no fluxo de bits 700.

[0129] O VPS 704 inclui dependência de decodificação ou informações para construção de conjunto de imagens de referência de camadas de enriquecimento. O VPS 704 fornece uma perspectiva geral ou visão de uma sequência escalável, incluindo quais tipos de pontos de operação são fornecidos, o perfil, tier e nível dos pontos de operação e algumas outras

propriedades de alto nível do fluxo de bits 700 que podem ser usadas como base para negociação de sessão e seleção de conteúdo, etc.

[0130] Em uma modalidade, o VPS 704 inclui uma lista das estruturas de sintaxe de DPB 780 para todos os OLSs. Em uma modalidade, cada uma das estruturas de sintaxe de DPB na lista de estruturas de sintaxe de DPB 780 é designada dpb_parameters(). O VPS 704 também inclui a estrutura de sintaxe vps_ols_dpb_params_idx[i] 782. A estrutura de sintaxe vps_ols_dpb_params_idx[i] 782 especifica o índice da estrutura de sintaxe de DPB que se aplica ao i^{ésimo} OLS. Ou seja, cada valor de índice corresponde a uma das estruturas de sintaxe de DPB na lista. A título de exemplo, o valor de índice 1 na lista corresponde ao dpb_parameters() para OLS 1, o valor de índice 2 corresponde a dpb_parameters() para OLS 2, e o valor de índice 3 corresponde a dpb_parameters() para OLS 3.

[0131] Quando presente, o valor de vps_ols_dpb_params_idx[i] está na faixa de 0 a VpsNumDpbParams − 1, inclusive.

[0132] O elemento de sintaxe vps_num_dpb_params_minus1 781 especifica o número de estruturas de sintaxe dpb_parameters() no VPS 704. O valor de vps_num_dpb_params_minus1 781 está em uma faixa de zero a menos que o número total de OLSs de múltiplas camadas, que pode ser designado como NumMultiLayerOlss.

[0133] A variável VpsNumDpbParams − 1 é derivada pelo decodificador de vídeo (ou seja, a variável não é sinalizada no fluxo de bits) com base no elemento de sintaxe vps_num_dpb_params_minus1 781, que é sinalizado no fluxo de bits.

[0134] Em uma modalidade, a variável VpsNumDpbParams – 1 é derivada da seguinte forma.

if(vps_each_layer_is_an_ols_flag)

VpsNumDpbParams = 0 (34)

else

VpsNumDpbParams = vps_num_dpb_params_minus1 + 1

[0135] A variável VpsNumDpbParams − 1, mais 1, especifica o número de estruturas de sintaxe de DPB no VPS 704. Por exemplo, quando o valor de VpsNumDpbParams − 1 é dois, o número de estruturas de sintaxe de DPB no VPS 704 é três (2+1 =3).

[0136] Em uma modalidade, o VPS 704 inclui uma flag de presença de DPB 783, que pode ser designada vps_sublayer_dpb_params_present_flag é usada para controlar a presença de elementos de sintaxe dpb_max_dec_pic_buffering_minus1[j], dpb_max_num_reorder_pics[j] e dpb_max_latency_increase_plus1[j] nos parâmetros de sintaxe dpb() no VPS para j na faixa de 0 a vps_dpb_max_tid[i] − 1, inclusive, quando vps_dpb_max_tid[i] for maior que 0. Quando não estiver presente, o valor de vps_sub_dpb_params_info_present_flag é inferido sendo como igual a 0.

[0137] O SPS 706 contém dados que são comuns a todas as imagens em uma sequência de imagens (SOP). O SPS 706 é uma estrutura de sintaxe contendo elementos de sintaxe que se aplicam a zero ou mais CLVSs inteiros conforme determinado pelo conteúdo de um elemento de sintaxe encontrado no PPS referido por um elemento de sintaxe encontrado em cada cabeçalho de imagem.

[0138] Em uma modalidade, os dpb_parameters() que se aplicam ao i^{ésimo} OLS 784 estão presentes no SPS 706 referido pela camada no i^{ésimo} OLS quando o i^{ésimo} OLS é um OLS de camada única. Ou seja, para um OLS de camada única (por exemplo, OLS 2 na Figura 6), a estrutura de sintaxe dpb_parameters() aplicável está presente no SPS referido pela camada no OLS.

[0139] Com base no anterior, deve-se considerar que quando um OLS é um OLS de camada única (por exemplo, o OLS contém apenas uma camada única), os dpb_parameters() que se aplicam ao i^{ésimo} OLS 784 estão presentes tanto no SPS 706 quanto no VPS 704 (ou seja, a estrutura de sintaxe de DPB que se aplica ao i^{ésimo} OLS é uma das estruturas de sintaxe de DPB incluídas nas estruturas de DPB 780). Além disso, os dpb_parameters() que se aplicam ao i^{ésimo} OLS 784 no SPS 706 são idênticos ao dpb_parameters() que se aplicam ao i^{ésimo} OLS 784 no VPS 704.

[0140] Em contraste com o SPS 706, o PPS 708 contém dados que são comuns a toda a imagem. O PPS 708 é uma estrutura de sintaxe que contém elementos de sintaxe que se aplicam a zero ou mais imagens codificadas inteiras. O cabeçalho de fatia 712 é uma parte de uma fatia codificada contendo elementos de dados pertencentes a todos os dados de vídeo dentro de um tile representado na fatia. O cabeçalho de fatia 712 contém parâmetros que são específicos para cada fatia em uma imagem. Assim, pode haver um cabeçalho

de fatia 712 por fatia na sequência de vídeo. O cabeçalho de fatia 712 pode conter informações de tipo de fatia, contagens de ordem de imagem (POCs), listas de imagens de referência (RPLs), pesos de predição, pontos de entrada de tile ou parâmetros de desblocagem. Um cabeçalho de fatia 712 também pode ser referido como um cabeçalho de grupo de tiles e/ou um cabeçalho de imagem (PH).

[0141] Em uma modalidade, o VPS 704 e/ou o SPS 706 contêm a seguinte sintaxe e semântica de SPS para implementar uma ou mais das modalidades divulgadas neste documento.

[0142] A sintaxe de VPS inclui o seguinte.

video_parameter_set_rbsp() {	Descritor
vps_video_parameter_set_id	u(4)
vps_max_layers_minus1	u(6)
vps_max_sub_layers_minus1	u(3)
if(vps_max_layers_minus1 > 0 && vps_max_sub_layers_minus1 > 0)	
vps_all_layers_same_num_sub_layers_flag	u(1)
if(vps_max_layers_minus1 > 0)	
vps_all_independent_layers_flag	u(1)
...	
vps_num_ptls	u(8)
for(i = 0; i < vps_num_ptls; i++) {	
if(i > 0)	
pt_present_flag[i]	u(1)
if(vps_max_sub_layers_minus1 > 0 && !vps_all_layers_same_num_sub_layers_flag)	
ptl_max_temporal_id[i]	u(3)
}	
while(!byte_aligned())	
vps_ptl_byte_alignment_zero_bit /* equal to 0 */	u(1)
for(i = 0; i < vps_num_ptls; i++)	
profile_tier_level(pt_present_flag[i], ptl_max_temporal_id[i])	

for(i = 0; i < TotalNumOlss; i++)	
if(NumLayersInOls[i] > 1 && vps_num_ptls > 1)	
ols_ptl_idx[i]	u(8)
if(!vps_all_independent_layers_flag)	
vps_num_dpb_params	ue(v)
if(vps_num_dpb_params > 0) {	
same_dpb_size_output_or_nonoutput_flag	u(1)
if(vps_max_sub_layers_minus1 > 0)	
vps_sub_layer_dpb_params_present_flag	u(1)
}	
for(i = 0; i < vps_num_dpb_params; i++) {	
dpb_size_only_flag[i]	u(1)
if(vps_max_sub_layers_minus1 > 0 && !vps_all_layers_same_num_sub_layers_flag)	
dpb_max_temporal_id[i]	u(3)
dpb_parameters(dpb_size_only_flag[i], dpb_max_temporal_id[i], vps_sub_layer_dpb_params_present_flag)	
}	
for(i = 0; i < vps_max_layers_minus1 && vps_num_dpb_params > 1; i++) {	
if(!vps_independent_layer_flag[i])	
layer_output_dpb_params_idx[i]	ue(v)
if(LayerUsedAsRefLayerFlag[i] && !same_dpb_size_output_or_nonoutput_flag)	
layer_nonoutput_dpb_params_idx[i]	ue(v)
}	
general_hrd_params_present_flag	u(1)
if(general_hrd_params_present_flag) {	
num_units_in_tick	u(32)
time_scale	u(32)
general_hrd_parameters()	
}	

vps_extension_flag	u(1)
if(vps_extension_flag)	
while(more_rbsp_data())	
vps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

[0143] A sintaxe de carga útil de sequência de bytes brutos (RBSP) de SPS inclui o seguinte.

seq_parameter_set_rbsp() {	Descritor
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_4bits	u(4)
sps_ptl_dpb_present_flag	u(1)
if(sps_ptl_dpb_present_flag)	
profile_tier_level(1, sps_max_sub_layers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
...	
log2_max_pic_order_cnt_lsb_minus4	ue(v)
poc_msb_in_rap_pics_flag	u(1)
if(poc_msb_in_rap_pics_flag > 0)	
poc_msb_len_minus1	ue(v)
if(sps_max_sub_layers_minus1 > 0)	
sps_sub_layer_dpb_params_flag	u(1)
if(sps_ptl_dpb_present_flag)	
dpb_parameters(0, sps_max_sub_layers_minus1, sps_sub_layer_dpb_params_flag)	
long_term_ref_pics_flag	u(1)
...	
sps_scaling_list_enabled_flag	u(1)
vui_parameters_present_flag	u(1)

if(vui_parameters_present_flag)	
vui_parameters()	
sps_extension_flag	u(1)
if(sps_extension_flag)	
while(more_rbsp_data())	
sps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

[0144] A sintaxe de DPB inclui o seguinte.

dpb_parameters(dpbSizeOnlyFlag, maxSubLayersMinus1, subLayerInfoFlag) {	Descritor
for(i = (subLayerInfoFlag ? 0 : maxSubLayersMinus1); i <= maxSubLayersMinus1; i++) {	
max_dec_pic_buffering_minus1[i]	ue(v)
if(!dpbSizeOnlyFlag) {	
max_num_reorder_pics[i]	ue(v)
max_latency_increase_plus1[i]	ue(v)
}	
}	
}	

[0145] A sintaxe de parâmetros de HRD inclui a sintaxe de parâmetros de HRD geral, a sintaxe de parâmetros de OLS HRD e a sintaxe de parâmetros de HRD de subcamada.

[0146] A sintaxe de parâmetros de HRD geral inclui o seguinte.

general_hrd_parameters() {	Descritor
general_nal_hrd_params_present_flag	u(1)
general_vcl_hrd_params_present_flag	u(1)
if(general_nal_hrd_params_present_flag || general_vcl_hrd_params_present_flag) {	
decoding_unit_hrd_params_present_flag	u(1)
if(decoding_unit_hrd_params_present_flag) {	
tick_divisor_minus2	u(8)
	u(1)

decoding_unit_cpb_params_in_pic_timing_sei_flag	
}	
bit_rate_scale	u(4)
cpb_size_scale	u(4)
if(decoding_unit_hrd_params_present_flag)	
cpb_size_du_scale	u(4)
}	
if(vps_max_sub_layers_minus1 > 0)	
sub_layer_cpb_params_present_flag	u(1)
if(TotalNumOlss > 1)	
num_ols_hrd_params_minus1	ue(v)
hrd_cpb_cnt_minus1	ue(v)
for(i = 0; i <= num_ols_hrd_params_minus1; i++) {	
if(vps_max_sub_layers_minus1 > 0 && !vps_all_layers_same_num_sub_layers_flag)	
hrd_max_temporal_id[i]	u(3)
ols_hrd_parameters(hrd_max_temporal_id[i])	
}	
if(num_ols_hrd_params_minus1 > 0)	
for(i = 1; i < TotalNumOlss; i++)	
ols_hrd_idx[i]	ue(v)
}	

[0147] A sintaxe de parâmetros de OLS HRD inclui o seguinte.

ols_hrd_parameters(hrdMaxTid) {	Descritor
firstSubLayer = sub_layer_cpb_params_present_flag ? 0: hrdMaxTid	
for(i = firstSubLayer; i <= hrdMaxTid; i++) {	
fixed_pic_rate_general_flag[i]	u(1)
if(!fixed_pic_rate_general_flag[i])	
fixed_pic_rate_within_cvs_flag[i]	u(1)
if(fixed_pic_rate_within_cvs_flag[i])	
elemental_duration_in_tc_minus1[i]	ue(v)
else if(hrd_cpb_cnt_minus1 = = 0)	

low_delay_hrd_flag[i]	u(1)
if(general_nal_hrd_params_present_flag)	
sub_layer_hrd_parameters(i)	
if(general_vcl_hrd_params_present_flag)	
sub_layer_hrd_parameters(i)	
}	
}	

[0148] A sintaxe de parâmetros de HRD da subcamada inclui o seguinte.

sub_layer_hrd_parameters(subLayerId) {	Descritor
for(j = 0; j <= hrd_cpb_cnt_minus1; j++) {	
bit_rate_value_minus1[subLayerId][j]	ue(v)
cpb_size_value_minus1[subLayerId][j]	ue(v)
if(decoding_unit_hrd_params_present_flag) {	
cpb_size_du_value_minus1[subLayerId][j]	ue(v)
bit_rate_du_value_minus1[subLayerId][j]	ue(v)
}	
cbr_flag[subLayerId][j]	u(1)
}	
}	

[0149] A semântica de VPS é a seguinte.

[0150] vps_max_layers_minus1 mais 1 especifica o número permitido máximo de camadas em cada CVS referente ao VPS.

[0151] vps_max_sub_layers_minus1 mais 1 especifica o número máximo de subcamadas temporais que podem estar presentes em cada CVS referente ao VPS. O valor de vps_max_sub_layers_minus1 deve estar na faixa de 0 a 6, inclusive.

[0152] vps_all_layers_same_num_sub_layers_flag igual a 1 especifica que o número de subcamadas temporais é o mesmo para todas as camadas em cada CVS referente ao VPS. vps_all_layers_same_num_sub_layers_flag igual a 0 especifica que as camadas em cada CVS referentes ao VPS podem ou não ter o mesmo número de subcamadas temporais. Quando não estiver presente, o valor de

vps_all_layers_same_num_sub_layers_flag é inferido como sendo igual a 1.

[0153] vps_all_independent_layers_flag igual a 1 especifica que todas as camadas no CVS são codificadas independentemente sem usar predição de camada inter. vps_all_independent_layers_flag igual a 0 especifica que uma ou mais camadas no CVS podem usar predição de camada inter. Quando não está presente, o valor de vps_all_independent_layers_flag é inferido como sendo igual a 1. Quando vps_all_independent_layers_flag é igual a 1, o valor de vps_independent_layer_flag[i] é inferido como sendo igual a 1. Quando vps_all_independent_layers_flag é igual a 0, o valor de vps_independent_layer_flag[0] é inferido como sendo igual a 1.

[0154] vps_direct_dependency_flag[i][j] igual a 0 especifica que a camada com índice j não é uma camada de referência direta para a camada com índice i. vps_direct_dependency_flag[i][j] igual a 1 especifica que a camada com índice j é uma camada de referência direta para a camada com índice i. Quando vps_direct_dependency_flag[i][j] não está presente para i e j na faixa de 0 a vps_max_layers_minus1, inclusive, infere-se que é igual a 0.

[0155] A variável DirectDependentLayerIdx[i][j], especificando a j^{ésima} camada dependente direta da i^{ésima} camada, e a variável LayerUsedAsRefLayerFlag[j], especificando se a camada com índice de camada j é usada como uma camada de referência por qualquer outra camada, são derivadas como segue:

```
for(i = 0; i  <=  vps_max_layers_minus1; i++)
LayerUsedAsRefLayerFlag[j] = 0
for(i = 1; i < vps_max_layers_minus1; i++)
if(!vps_independent_layer_flag[i])
   for(j = i − 1, k = 0; j  >=  0; j− −)
         if(vps_direct_dependency_flag[i][j]) {
               DirectDependentLayerIdx[i][k++] = j
               LayerUsedAsRefLayerFlag[j] = 1
         }
```

[0156] A variável GeneralLayerIdx[i], especificando o índice de camada da camada com nuh_layer_id igual a vps_layer_id[i], é derivada da seguinte forma:

```
for(i = 0; i <= vps_max_layers_minus1; i++)
```

GeneralLayerIdx[vps_layer_id[i]] = i

[0157] each_layer_is_an_ols_flag igual a 1 especifica que cada conjunto de camadas de saída contém apenas uma camada e cada camada em si no fluxo de bits é um conjunto de camadas de saída com a camada única incluída sendo a única camada de saída. each_layer_is_an_ols_flag igual a 0 especifica que um conjunto de camadas de saída pode conter mais de uma camada. Se vps_max_layers_minus1 for igual a 0, o valor de each_layer_is_an_ols_flag é inferido como sendo igual a 1. De outro modo, quando vps_all_independent_layers_flag for igual a 0, o valor de each_layer_is_an_ols_flag é inferido como sendo igual a 0.

[0158] ols_mode_idc igual a 0 especifica que o número total de OLSs especificados pelo VPS é igual a vps_max_layers_minus1 + 1, o $i^{ésimo}$ OLS inclui as camadas com índices de camada de 0 a i, inclusive, e para cada OLS apenas a camada mais alta no OLS é emitida. ols_mode_idc igual a 1 especifica que o número total de OLSs especificados pelo VPS é igual a vps_max_layers_minus1 + 1, o $i^{ésimo}$ OLS inclui as camadas com índices de camada de 0 a i, inclusive, e para cada OLS todas as camadas do OLS são emitidas. ols_mode_idc igual a 2 especifica que o número total de OLSs especificados pelo VPS é explicitamente sinalizado e para cada OLS a camada mais alta e um conjunto explicitamente sinalizado de camadas inferiores no OLS são emitidos. O valor de ols_mode_idc deve estar na faixa de 0 a 2, inclusive. O valor 3 de ols_mode_idc é reservado para uso futuro pelo ITU-T | ISO/IEC. Quando vps_all_independent_layers_flag é igual a 1 e each_layer_is_an_ols_flag é igual a 0, o valor de ols_mode_idc é inferido como sendo igual a 2.

[0159] num_output_layer_sets_minus1 mais 1 especifica o número total de OLSs especificados pelo VPS quando ols_mode_idc é igual a 2.

[0160] A variável TotalNumOlss, que especifica o número total de OLSs especificados pelo VPS, é derivada da seguinte forma:

```
if(vps_max_layers_minus1 == 0)
TotalNumOlss = 1
else if(each_layer_is_an_ols_flag || ols_mode_idc == 0 ||
ols_mode_idc == 1)
TotalNumOlss = vps_max_layers_minus1 + 1
else if(ols_mode_idc == 2)
```

TotalNumOlss = num_output_layer_sets_minus1 + 1

[0161] layer_included_flag[i][j] especifica se a j^{ésima} camada (ou seja, a camada com nuh_layer_id igual a vps_layer_id[j]) está incluída no i^{ésimo} OLS quando ols_mode_idc é igual a 2. layer_included_flag[i][j] igual a 1 especifica que a j^{ésima} camada está incluída no i^{ésimo} OLS. layer_included_flag[i][j] igual a 0 especifica que a j^{ésima} camada não está incluída no i^{ésimo} OLS.

[0162] A variável NumLayersInOls[i], especificando o número de camadas no i^{ésimo} OLS, e a variável LayerIdInOls[i][j], especificando o valor de nuh_layer_id da j^{ésima} camada no i^{ésimo} OLS, são derivadas como segue:

```
NumLayersInOls[0] = 1
LayerIdInOls[0][0] = vps_layer_id[0]
for(i = 1, i < TotalNumOlss; i++) {
if(each_layer_is_an_ols_flag) {
   NumLayersInOls[i] = 1
   LayerIdInOls[i][0] = vps_layer_id[i]
} else if(ols_mode_idc  = =  0  | |  ols_mode_idc  = =  1) {
   NumLayersInOls[i] = i + 1
   for(j = 0; j < NumLayersInOls[i]; j++)
         LayerIdInOls[i][j] = vps_layer_id[j]
} else if(ols_mode_idc  = =  2) {
   for(k = 0, j = 0; k  <=  vps_max_layers_minus1; k++)
         if(layer_included_flag[i][k])
               LayerIdInOls[i][j++] = vps_layer_id[k]
   NumLayersInOls[i] = j
}
}
```

[0163] A variável OlsLayeIdx[i][j], especificando o índice da camada de OLS da camada com nuh_layer_id igual a LayerIdInOls[i][j], é derivada da seguinte forma:

```
for(i = 0, i < TotalNumOlss; i++)
for j = 0; j < NumLayersInOls[i]; j++)
   OlsLayeIdx[i][LayerIdInOls[i][j]] = j
```

[0164] A camada mais baixa em cada OLS deve ser uma camada independente. Em outras palavras, para cada i na faixa de 0 a TotalNumOlss −

1, inclusive, o valor de vps_independent_layer_flag[GeneralLayerIdx[LayerIdInOls[i][0]]] deve ser igual a 1.

[0165] Cada camada deve ser incluída em pelo menos um OLS especificado pelo VPS. Em outras palavras, para cada camada com um valor particular de nuh_layer_id nuhLayerId, igual a um de vps_layer_id[k] a k na faixa de 0 a vps_max_layers_minus1, inclusive, deve haver pelo menos um par de valores de i e j, em que i está na faixa de 0 a TotalNumOlss − 1, inclusive, ej está na faixa de NumLayersInOls[i] − 1, inclusive, de modo que o valor de LayerIdInOls[i][j] seja igual a nuhLayerId.

[0166] Qualquer camada em um OLS deve ser uma camada de saída do OLS ou uma camada de referência (direta ou indireta) de uma camada de saída do OLS.

[0167] vps_output_layer_flag[i][j] especifica se a j^{ésima} camada no i^{ésimo} OLS é emitida quando ols_mode_idc é igual a 2. vps_output_layer_flag[i] igual a 1 especifica que a j^{ésima} camada no i^{ésimo} OLS é emitida. vps_output_layer_flag[i] igual a 0 especifica que a j^{ésima} camada no i^{ésimo} OLS não é emitida. Quando vps_all_independent_layers_flag é igual a 1 e each_layer_is_an_ols_flag é igual a 0, o valor de vps_output_layer_flag[i] é inferido como sendo igual a 1.

[0168] A variável OutputLayerFlag[i][j], para a qual o valor 1 especifica que a j^{ésima} camada no i^{ésimo} OLS é emitida e o valor 0 especifica que a j^{ésima} camada no i^{ésimo} OLS não é emitida, é derivado da seguinte forma:

```
for(i = 0, i < TotalNumOlss; i++) {
OutputLayerFlag[i][NumLayersInOls[i] − 1] = 1
for(j = 0; j < NumLayersInOls[i] − 1; j++)
   if(ols_mode_idc[i]  = =  0)
         OutputLayerFlag[i][j] = 0
   else if(ols_mode_idc[i]  = =  1)
         OutputLayerFlag[i][j] = 1
   else if(ols_mode_idc[i]  = =  2)
         OutputLayerFlag[i][j] = vps_output_layer_flag[i][j]
}
```

[0169] NOTA – O 0^{ésimo} OLS contém apenas a camada mais baixa (ou seja, a camada com nuh_layer_id igual a vps_layer_id[0]) e para o 0^{ésimo} OLS a

única camada incluída é emitida.

[0170] vps_num_ptls especifica o número de estruturas de sintaxe profile_tier_level() no VPS.

[0171] pt_present_flag[i] igual a 1 especifica que as informações de perfil, tier e restrições gerais estão presentes na i^{ésima} estrutura de sintaxe profile_tier_level() no VPS. pt_present_flag[i] igual a 0 especifica que as informações de perfil, tier e restrições gerais não estão presentes na i^{ésima} estrutura de sintaxe profile_tier_level() no VPS. O valor de pt_present_flag[0] é inferido como sendo igual a 0. Quando pt_present_flag[i] é igual a 0, as informações de perfil, tier e restrições gerais para a i^{ésima} estrutura de sintaxe profile_tier_level() no VPS são inferidas como sendo as mesmas que para a (i−1)^{ésima} estrutura de sintaxe profile_tier_level() no VPS.

[0172] ptl_max_temporal_id[i] especifica o TemporalId da representação de subcamada mais alta para a qual as informações de nível estão presentes na i^{ésima} estrutura de sintaxe profile_tier_level() no VPS. O valor de ptl_max_temporal_id[i] deve estar na faixa de 0 a vps_max_sub_layers_minus1, inclusive. Quando vps_max_sub_layers_minus1 é igual a 0, o valor de ptl_max_temporal_id[i] é inferido como sendo igual a 0. Quando vps_max_sub_layers_minus1 é maior que 0 e vps_all_layers_same_num_sub_layers_flag é igual a 1, o valor de ptl_max_temporal_id[i] é inferido como sendo igual a vps_max_sub_layers_minus1.

[0173] vps_ptl_byte_alignment_zero_bit deve ser igual a 0.

[0174] ols_ptl_idx[i] especifica o índice, para a lista de estruturas de sintaxe profile_tier_level() no VPS, da estrutura de sintaxe profile_tier_level() que se aplica ao i^{ésimo} OLS. Quando presente, o valor de ols_ptl_idx[i] deve estar na faixa de 0 a vps_num_ptls − 1, inclusive.

[0175] Quando NumLayersInOls[i] é igual a 1, a estrutura de sintaxe profile_tier_level() que se aplica ao i^{ésimo} OLS está presente no SPS referido pela camada no i^{ésimo} OLS.

[0176] vps_num_dpb_params especifica o número de estruturas de sintaxe dpb_parameters() no VPS. O valor de vps_num_dpb_params deve estar na faixa de 0 a 16, inclusive. Quando não estiver presente, o valor de vps_num_dpb_params é inferido como sendo igual a 0.

[0177] same_dpb_size_output_or_nonoutput_flag igual a 1 especifica que não há elemento de sintaxe layer_nonoutput_dpb_params_idx[i] presente no VPS. same_dpb_size_output_or_nonoutput_flag igual a 0 especifica que pode ou não haver elementos de sintaxe layer_nonoutput_dpb_params_idx[i] presentes no VPS.

[0178] vps_sub_layer_dpb_params_present_flag é usado para controlar a presença de elementos de sintaxe max_dec_pic_buffering_minus1[], max_num_reorder_pics[] e max_latency_increase_plus1[] nas estruturas de sintaxe dpb_parameters() no VPS. Quando não estiver presente, vps_sub_dpb_params_info_present_flag é inferido como sendo igual a 0.

[0179] dpb_size_only_flag[i] igual a 1 especifica que os elementos de sintaxe max_num_reorder_pics[] e max_latency_increase_plus1[] não estão presentes na i^{ésima} estrutura de sintaxe dpb_parameters() no VPS. dpb_size_only_flag[i] igual a 1 especifica que os elementos de sintaxe max_num_reorder_pics[] e max_latency_increase_plus1[] podem estar presentes na i^{ésima} estrutura de sintaxe dpb_parameters() no VPS.

[0180] dpb_max_temporal_id[i] especifica o TemporalId da representação de subcamada mais alta para a qual os parâmetros de DPB podem estar presentes na i^{ésima} estrutura de sintaxe dpb_parameters() no VPS. O valor de dpb_max_temporal_id[i] deve estar na faixa de 0 a vps_max_sub_layers_minus1, inclusive. Quando vps_max_sub_layers_minus1 é igual a 0, o valor de dpb_max_temporal_id[i] é inferido como sendo igual a 0. Quando vps_max_sub_layers_minus1 é maior que 0 e vps_all_layers_same_num_sub_layers_flag é igual a 1, o valor de dpb_max_temporal_id[i]1 é inferido como sendo igual a vps_max_sub_layers_minus1.

[0181] layer_output_dpb_params_idx[i] especifica o índice, para a lista de estruturas de sintaxe dpb_parameters() no VPS, da estrutura de sintaxe dpb_parameters() que se aplica à i^{ésima} camada quando é uma camada de saída em um OLS. Quando presente, o valor de layer_output_dpb_params_idx[i] deve estar na faixa de 0 a vps_num_dpb_params − 1, inclusive.

[0182] Se vps_independent_layer_flag[i] for igual a 1, a estrutura de sintaxe dpb_parameters() que se aplica à i^{ésima} camada quando é uma camada de saída é a estrutura de sintaxe dpb_parameters() presente no SPS referido

pela camada.

[0183] De outro modo (vps_independent_layer_flag[i] é igual a 0), o seguinte se aplica:

[0184] Quando vps_num_dpb_params é igual a 1, o valor de layer_output_dpb_params_idx[i] é inferido como sendo igual a 0.

[0185] É um requisito de conformidade de fluxo de bits que o valor de layer_output_dpb_params_idx[i] seja tal que dpb_size_only_flag[layer_output_dpb_params_idx[i]] seja igual a 0.

[0186] layer_nonoutput_dpb_params_idx[i] especifica o índice, para a lista de estruturas de sintaxe dpb_parameters() no VPS, da estrutura de sintaxe dpb_parameters() que se aplica à i^{ésima} camada quando é uma camada de não saída em um OLS. Quando presente, o valor de layer_nonoutput_dpb_params_idx[i] deve estar na faixa de 0 a vps_num_dpb_params − 1, inclusive.

[0187] Se same_dpb_size_output_or_nonoutput_flag for igual a 1, o seguinte se aplica:

[0188] Se vps_independent_layer_flag[i] for igual a 1, a estrutura de sintaxe dpb_parameters() que se aplica à i^{ésima} camada quando é uma camada de não saída é a estrutura de sintaxe dpb_parameters() presente no SPS referido pela camada.

[0189] De outro modo (vps_independent_layer_flag[i] é igual a 0), o valor de layer_nonoutput_dpb_params_idx[i] é inferido como sendo igual a layer_output_dpb_params_idx[i].

[0190] De outro modo (same_dpb_size_output_or_nonoutput_flag é igual a 0), quando vps_num_dpb_params é igual a 1, o valor de layer_output_dpb_params_idx[i] é inferido como sendo igual a 0.

[0191] vps_extension_flag igual a 0 especifica que nenhum elemento de sintaxe vps_extension_data_flag está presente na estrutura de sintaxe de VPS RBSP. vps_extension_flag igual a 1 especifica que existem elementos de sintaxe vps_extension_data_flag presentes na estrutura de sintaxe de VPS RBSP.

[0192] vps_extension_data_flag pode ter qualquer valor. Sua presença e valor não afetam a conformidade do decodificador com os perfis especificados nesta versão deste Relatório Descritivo. Os decodificadores em

conformidade com esta versão deste Relatório Descritivo devem ignorar todos os elementos de sintaxe vps_extension_data_flag.

[0193] A semântica de SPS RBSP é a seguinte.

[0194] Um SPS RBSP deve estar disponível para o processo de decodificação antes de ser referenciado, incluído em pelo menos uma unidade de acesso com TemporalId igual a 0 ou fornecido através de meios externos, e a unidade SPS NAL contendo o SPS RBSP deve ter nuh_layer_id igual ao menor valor de nuh_layer_id das unidades PPS NAL que se referem a ele.

[0195] Todas as unidades SPS NAL com um valor particular de sps_seq_parameter_set_id em um CVS devem ter o mesmo conteúdo.

[0196] sps_decoding_parameter_set_id, quando maior que 0, especifica o valor de dps_decoding_parameter_set_id para o DPS referido pelo SPS. Quando sps_decoding_parameter_set_id é igual a 0, o SPS não se refere a um DPS e nenhum DPS é referido ao decodificar cada CLVS referente ao SPS. O valor de sps_decoding_parameter_set_id deve ser o mesmo em todos os SPSs referenciados por imagens codificadas em um fluxo de bits.

[0197] sps_video_parameter_set_id, quando maior que 0, especifica o valor de vps_video_parameter_set_id para o VPS referido pelo SPS. Quando sps_video_parameter_set_id é igual a 0, o SPS não se refere a um VPS e nenhum VPS é referido ao decodificar cada CLVS referente ao SPS, e o valor de GeneralLayerIdx[nuh_layer_id] é inferido como sendo igual a 0, e o valor de vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é inferido como sendo igual a 1.

[0198] Quando vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é igual a 1, o SPS referido por um CLVS com um valor de nuh_layer_id particular nuhLayerId deve ter nuh_layer_id igual a nuhLayerId.

[0199] sps_max_sub_layers_minus1 mais 1 especifica o número máximo de subcamadas temporais que podem estar presentes em cada CLVS referente ao SPS. O valor de sps_max_sub_layers_minus1 deve estar na faixa de 0 a vps_max_sub_layers_minus1, inclusive.

[0200] sps_reserved_zero_4bits deve ser igual a 0 em fluxos de bits em conformidade com esta versão deste Relatório Descritivo. Outros valores para sps_reserved_zero_4bits são reservados para uso futuro pelo ITU-T |

ISO/IEC.

[0201] sps_ptl_dpb_present_flag igual a 1 especifica que uma estrutura de sintaxe profile_tier_level() e uma estrutura de sintaxe dpb_parameters() estão presentes no SPS. sps_ptl_dpb_present_flag igual a 0 especifica que nenhuma estrutura de sintaxe profile_tier_level() e nenhuma estrutura de sintaxe dpb_parameters() estão presentes no SPS. O valor de sps_ptl_dpb_present_flag deve ser igual a vps_independent_layer_flag[nuh_layer_id].

[0202] Se vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] for igual a 1, a variável MaxDecPicBuffMinus1 será definida como igual a max_dec_pic_buffering_minus1[sps_max_sub_layers_minus1] na estrutura de sintaxe dpb_parameters() no SPS. De outro modo, MaxDecPicBuffMinus1 é definido como igual a max_dec_pic_buffering_minus1[sps_max_sub_layers_minus1] na estrutura de sintaxe layer_nonoutput_dpb_params_idx[GeneralLayerIdx[nuh_layer_id]]ésimo dpb_parameters() no VPS.

[0203] gdr_enabled_flag igual a 1 especifica que imagens de atualização de decodificação gradual (GDR) podem estar presentes em CLVSs referentes ao SPS. gdr_enabled_flag igual a 0 especifica que as imagens GDR não estão presentes em CLVSs referentes ao SPS.

[0204] sps_sub_layer_dpb_params_flag é usado para controlar a presença de elementos de sintaxe max_dec_pic_buffering_minus1[i], max_num_reorder_pics[i] e max_latency_increase_plus1[i] na estrutura de sintaxe dpb_parameters() no SPS. Quando não estiver presente, sps_sub_dpb_params_info_present_flag é inferido como sendo igual a 0.

[0205] long_term_ref_pics_flag igual a 0 especifica que nenhum LTRP é usado para predição inter de qualquer imagem codificada no CLVS. long_term_ref_pics_flag igual a 1 especifica que LTRPs podem ser usados para predição inter de uma ou mais imagens codificadas no CLVS.

[0206] A semântica de perfil, tier e nível geral é a seguinte.

[0207] Uma estrutura de sintaxe profile_tier_level() fornece informações de nível e, opcionalmente, informações de perfil, tier, subperfil e restrições gerais (denotadas como informações de PT).

[0208] Quando a estrutura de sintaxe profile_tier_level() é incluída em um DPS, o OlsInScope é o OLS que inclui todas as camadas em todo o fluxo de bits que se refere ao DPS. Quando a estrutura de sintaxe profile_tier_level() é incluída em um VPS, o OlsInScope é um ou mais OLSs especificados pelo VPS. Quando a estrutura de sintaxe profile_tier_level() é incluída em um SPS, o OlsInScope é o OLS que inclui apenas a camada que é a mais baixa dentre as camadas que se referem ao SPS, que deve ser uma camada independente.

[0209] general_profile_idc indica um perfil ao qual OlsInScope está em conformidade conforme especificado no Anexo A do padrão VVC. Os fluxos de bits não devem conter valores de general_profile_idc além dos especificados no Anexo A . Outros valores de general_profile_idc são reservados para uso futuro pelo ITU-T | ISO/IEC.

[0210] general_tier_flag especifica o contexto de tier para a interpretação de general_level_idc conforme especificado no Anexo A .

[0211] num_sub_profiles especifica o número dos elementos de sintaxe general_sub_profile_idc[i].

[0212] general_sub_profile_idc[i] indica os i^{ésimos} metadados de interoperabilidade registrados conforme especificado pela Recomendação ITU-T T.35, cujo conteúdo não é especificado no padrão VVC.

[0213] general_level_idc indica um nível ao qual OlsInScope está em conformidade, conforme especificado no Anexo A. Os fluxos de bits não devem conter valores de general_level_idc diferentes daqueles especificados no Anexo A . Outros valores de general_level_idc são reservados para uso futuro pelo ITU-T | ISO/IEC.

[0214] NOTA 1 – Um valor maior de general_level_idc indica um nível mais alto. O nível máximo sinalizado no DPS para OlsInScope pode ser superior ao nível sinalizado no SPS para um CVS contido no OlsInScope.

[0215] NOTA 2 – Quando OlsInScope estiver em conformidade com vários perfis, general_profile_idc deve indicar o perfil que fornece o resultado decodificado preferencial ou a identificação de fluxo de bits preferencial, conforme determinado pelo codificador (de maneira não especificada neste Relatório Descritivo).

[0216] NOTA 3 – Quando a estrutura de sintaxe profile_tier_level() incluída em um DPS e CVSs de OlsInScope estiver em conformidade com perfis

diferentes, general_profile_idc e level_idc devem indicar o perfil e o nível de um decodificador capaz de decodificar OlsInScope.

[0217] sub_layer_level_present_flag[i] igual a 1 especifica que as informações de nível estão presentes na estrutura de sintaxe profile_tier_level() para a representação da subcamada com TemporalId igual a i. sub_layer_level_present_flag[i] igual a 0 especifica que as informações de nível não estão presentes na estrutura de sintaxe profile_tier_level() para a representação de subcamada com TemporalId igual a i.

[0218] ptl_alignment_zero_bits deve ser igual a 0.

[0219] A semântica do elemento de sintaxe sub_layer_level_idc[i] é, além da especificação da inferência de valores não presentes, a mesma do elemento de sintaxe general_level_idc, mas aplica-se à representação da subcamada com TemporalId igual a i.

[0220] A semântica de DPB é a seguinte.

[0221] A estrutura de sintaxe dpb_parameters(maxSubLayersMinus1, subLayerInfoFlag) fornece informações sobre o tamanho de DPB, número de reordenação de imagem máximo e latência máxima para cada CLVS do CVS.

[0222] Quando uma estrutura de sintaxe dpb_parameters() é incluída em um VPS, os OLSs aos quais a estrutura de sintaxe dpb_parameters() se aplica são especificados pelo VPS. Quando uma estrutura de sintaxe dpb_parameters() é incluída em um SPS, ela se aplica ao OLS que inclui apenas a camada que é a camada mais baixa entre as camadas que se referem ao SPS, que deve ser uma camada independente.

[0223] max_dec_pic_buffering_minus1[i] mais 1 especifica, para cada CLVS do CVS, o tamanho requerido máximo do buffer de imagem decodificada em unidades de buffers de armazenamento de imagem quando Htid é igual a i. O valor de max_dec_pic_buffering_minus1[i] deve estar na faixa de 0 a MaxDpbSize − 1, inclusive, em que MaxDpbSize é conforme especificado em outro lugar. Quando i for maior que 0, max_dec_pic_buffering_minus1[i] deve ser maior que ou igual a max_dec_pic_buffering_minus1[i−1]. Quando max_dec_pic_buffering_minus1[i] não está presente para i na faixa de 0 a maxSubLayersMinus1 − 1, inclusive, devido a subLayerInfoFlag ser igual a 0, infere-se que seja igual a max_dec_pic_buffering_minus1[maxSubLayersMinus1].

[0224] max_num_reorder_pics[i] especifica, para cada CLVS do CVS, o número permitido máximo de imagens do CLVS que podem preceder qualquer imagem no CLVS na ordem de decodificação e seguir essa imagem na ordem de emissão quando Htid for igual a i. O valor de max_num_reorder_pics[i] deve estar na faixa de 0 a max_dec_pic_buffering_minus1[i], inclusive. Quando i for maior que 0, max_num_reorder_pics[i] deve ser maior que ou igual a max_num_reorder_pics[i−1]. Quando max_num_reorder_pics[i] não está presente para i na faixa de 0 a maxSubLayersMinus1 − 1, inclusive, devido a subLayerInfoFlag ser igual a 0, infere-se que seja igual a max_num_reorder_pics[maxSubLayersMinus1].

[0225] max_latency_increase_plus1[i] diferente de 0 é usado para calcular o valor de MaxLatencyPictures[i], que especifica, para cada CLVS do CVS, o número máximo de imagens no CLVS que podem preceder qualquer imagem no CLVS na ordem de emissão e seguir essa imagem na ordem de decodificação quando Htid for igual a i.

[0226] Quando max_latency_increase_plus1[i] não é igual a 0, o valor de MaxLatencyPictures[i] é especificado da seguinte forma:

MaxLatencyPictures[i] = max_num_reorder_pics[i] + max_latency_increase_ plus1[i] − 1

[0227] Quando max_latency_increase_plus1[i] é igual a 0, nenhum limite correspondente é expresso.

[0228] O valor de max_latency_increase_plus1[i] deve estar na faixa de 0 a 232 − 2, inclusive. Quando max_latency_increase_plus1[i] não está presente para i na faixa de 0 a maxSubLayersMinus1 − 1, inclusive, devido a subLayerInfoFlag ser igual a 0, infere-se que seja igual a max_latency_increase_plus1[maxSubLayersMinus1].

[0229] A semântica de parâmetros de HRD inclui a semântica de parâmetros de HRD geral.

[0230] A semântica de parâmetros de HRD geral é a seguinte.

[0231] A estrutura de sintaxe general_hrd_parameters() fornece parâmetros de HRD usados nas operações de HRD.

[0232] num_ols_hrd_params_minus1 mais 1 especifica o número de estruturas de sintaxe ols_hrd_parameters() presentes na estrutura de sintaxe general_hrd_parameters(). O valor de num_ols_hrd_params_minus1 deve estar

na faixa de 0 a 63, inclusive. Quando TotalNumOlss é maior que 1, o valor de num_ols_hrd_params_minus1 é inferido como sendo igual a 0.

[0233] hrd_cpb_cnt_minus1 mais 1 especifica o número de especificações alternativas de CPB no fluxo de bits do CVS. O valor de hrd_cpb_cnt_minus1 deve estar na faixa de 0 a 31, inclusive.

[0234] hrd_max_temporal_id[i] especifica o TemporalId da representação de subcamada mais alta para a qual os parâmetros de HRD estão contidos na i^{ésima} estrutura de sintaxe layer_level_hrd_parameters(). O valor de hrd_max_temporal_id[i] deve estar na faixa de 0 a vps_max_sub_layers_minus1, inclusive. Quando vps_max_sub_layers_minus1 é igual a 0, o valor de hrd_max_temporal_id[i] é inferido como sendo igual a 0.

[0235] ols_hrd_idx[i] especifica o índice da estrutura de sintaxe ols_hrd_parameters() que se aplica ao i^{ésimo} OLS. O valor de ols_hrd_idx[[i] deve estar na faixa de 0 a num_ols_hrd_params_minus1, inclusive. Quando não estiver presente, o valor de ols_hrd_idx[[i] é inferido como sendo igual a 0.

[0236] A semântica de estrutura de lista de imagens de referência é a seguinte.

[0237] A estrutura de sintaxe ref_pic_list_struct(listIdx, rplsIdx) pode estar presente em um SPS ou em um cabeçalho de fatia. Dependendo se a estrutura de sintaxe está incluída em um cabeçalho de fatia ou em um SPS, o seguinte se aplica:

[0238] – Se presente em um cabeçalho de fatia, a estrutura de sintaxe ref_pic_list_struct(listIdx, rplsIdx) especifica a lista de imagens de referência listIdx da imagem atual (a imagem que contém a fatia).

[0239] – De outro modo (presente em um SPS), a estrutura de sintaxe ref_pic_list_struct(listIdx, rplsIdx) especifica um candidato para lista de imagens de referência listIdx, e o termo "a imagem atual" na semântica especificada no restante desta cláusula se refere a cada imagem que 1) tem uma ou mais fatias contendo ref_pic_list_idx[listIdx] igual a um índice na lista das estruturas de sintaxe ref_pic_list_struct(listIdx, rplsIdx) incluídas no SPS, e 2) está em um CVS que se refere ao SPS.

[0240] num_ref_entries[listIdx][rplsIdx] especifica o número de entradas na estrutura de sintaxe ref_pic_list_struct(listIdx, rplsIdx). O valor de num_ref_entries[listIdx][rplsIdx] deve estar na faixa de 0 a MaxDecPicBuffMinus1

+ 14, inclusive.

[0241] O VPS 704, o SPS 706 e o PPS 708 estão contidos em diferentes tipos de unidades de Camada de Abstração de Rede (NAL). Uma unidade NAL é uma estrutura de sintaxe que contém uma indicação do tipo de dados a seguir (por exemplo, dados de vídeo codificados). As unidades NAL são classificadas em camada de codificação de vídeo (VCL) e unidades não VCL NAL. As unidades VCL NAL contêm os dados que representam os valores das amostras nas imagens de vídeo, e as unidades não VCL NAL contêm quaisquer informações adicionais associadas, como conjuntos de parâmetros (dados importantes que podem ser aplicados a várias unidades VCL NAL) e informações suplementares de enriquecimento (informações de temporização e outros dados suplementares que podem melhorar a usabilidade do sinal de vídeo decodificado, mas não são necessários para decodificar os valores das amostras nas imagens de vídeo).

[0242] Em uma modalidade, o VPS 704 está contido em uma unidade não VCL NAL designada como uma unidade VPS NAL. Portanto, a unidade VPS NAL possui um VPS NUT. Em uma modalidade, o SPS 706 é uma unidade não VCL NAL designada como uma unidade SPS NAL. Portanto, a unidade SPS NAL possui um SPS NUT. Em uma modalidade, o PPS 708 está contido em uma unidade não VCL NAL designada como uma unidade PPS NAL. Portanto, a unidade PPS NAL possui um PPS NUT.

[0243] O cabeçalho de fatia 712 é uma estrutura de sintaxe contendo elementos de sintaxe que se aplicam a todas as fatias de uma imagem codificada (por exemplo, imagem 725). Em uma modalidade, o cabeçalho de fatia 712 está em uma VCL NAL.

[0244] Os dados de imagem 720 contêm dados de vídeo codificados de acordo com predição inter, predição intra ou predição de camada inter, bem como dados residuais transformados e quantizados correspondentes. Como mostrado na Figura 7, os dados de imagem 720 incluem um ou mais OLSs 721. Os um ou mais OLSs 721 são semelhantes a OLS 1, OLS 2 e OLS 3 da Figura 6. Cada OLS 721 inclui uma ou mais camadas 723. As camadas 723 são semelhantes às camadas 631-635 na Figura 6. Cada uma das camadas 723 inclui uma ou mais imagens 725. As imagens 725 são semelhantes às imagens 615-618, 611-614, 641-644, 651-654 e 661-664 na Figura 6.

[0245] Cada imagem 725 é um arranjo de amostras de luma em formato monocromático ou um arranjo de amostras de luma e dois arranjos correspondentes de amostras de croma em formato de cor 4:2:0, 4:2:2 e 4:4:4. As imagens 725 podem ser um quadro ou um campo. No entanto, em um CVS (por exemplo, CVS 690), todas as imagens 725 são quadros ou todas as imagens 725 são campos. O CVS 690 é uma sequência de vídeo codificada para cada sequência de vídeo de camada codificada (CLVS) no fluxo de bits de vídeo 600. Notavelmente, o CVS 690 e o CLVS são os mesmos quando o fluxo de bits de vídeo 600 inclui uma camada única. O CVS 690 e o CLVS são diferentes apenas quando o fluxo de bits de vídeo 600 inclui múltiplas camadas (por exemplo, como mostrado nas Figuras 5 e 6).

[0246] Cada imagem 725 contém uma ou mais fatias 727. Uma fatia 727 é um número inteiro de tiles completos ou um número inteiro de linhas de CTU completas consecutivas dentro de um tile de uma imagem (por exemplo, imagem 725). Cada fatia 727 está exclusivamente contida em uma única unidade NAL (por exemplo, uma unidade VCL NAL). Um tile (não mostrado) é uma região retangular de CTUs dentro de uma coluna de tile particular e uma linha de tile particular em uma imagem (por exemplo, imagem 725). Uma CTU (não mostrada) é um CTB de amostras de luma, dois CTBs correspondentes de amostras de croma de uma imagem que possui três arranjos de amostras, ou um CTB de amostras de uma imagem monocromática ou uma imagem codificada usando três planos de cor e estruturas de sintaxe separadas usadas para codificar as amostras. Um CTB (não mostrado) é um bloco N×N de amostras para algum valor de N tal que a divisão de um componente em CTBs é um particionamento. Um bloco (não mostrado) é um arranjo MxN (M colunas por N linhas) de amostras (por exemplo, pixels), ou um arranjo MxN de coeficientes de transformada.

[0247] As imagens 725 e suas fatias 727 compreendem dados associados às imagens ou vídeo que estão sendo codificados ou decodificados. Assim, as imagens 725 e suas fatias 727 podem ser simplesmente referidas como a carga útil ou dados sendo portados no fluxo de bits 700.

[0248] Os indivíduos versados na técnica apreciarão que o fluxo de bits 700 pode conter outros parâmetros e informações em aplicações práticas.

[0249] Um subfluxo de bits 701 pode ser extraído do fluxo de bits 700

de acordo com um processo de extração de subfluxo de bits 703. Um processo de extração de subfluxo de bits 703 é um mecanismo especificado que remove unidades NAL de um fluxo de bits 700 que não faz parte de um conjunto alvo resultando em um subfluxo de bits de saída 701 que inclui as unidades NAL que estão incluídas no conjunto alvo. O processo de extração de subfluxo de bits 703 pode ser realizado por um codificador ou um fatiador associado configurado para alterar dinamicamente um fluxo de bits 700 com base no comportamento/solicitações do usuário.

[0250] Como o SPS 706 contém o dpb_parameters() para o i^{ésimo} OLS 784 quando o i^{ésimo} OLS contém apenas uma camada única (por exemplo, OLS 3 na Figura 6), o VPS 704 pode ser removido durante o processo de extração 703 Ou seja, o subfluxo de bits 701 não precisa incluir o VPS 704 no subfluxo de bits 701 porque o dpb_parameters() idêntico para o i^{ésimo} OLS 784 também está presente no SPS 706. Portanto, o decodificador pode obter o dpb_parameters() para o i^{ésimo} OLS 784 do SPS 706 quando o i^{ésimo} OLS contém apenas uma camada única.

[0251] A Figura 8 é uma modalidade de um método 800 de decodificação implementado por um decodificador de vídeo (por exemplo, decodificador de vídeo 400). O método 800 pode ser realizado após um fluxo de bits ter sido recebido direta ou indiretamente a partir de um codificador de vídeo (por exemplo, codificador de vídeo 300). O método 800 melhora o processo de decodificação garantindo que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0252] No bloco 802, o decodificador de vídeo recebe um subfluxo de bits (por exemplo, subfluxo de bits 701) tendo um conjunto de parâmetros de

sequência (por exemplo, SPS 706) com uma estrutura de sintaxe de buffer de imagem decodificada (DPB) (por exemplo, dpb_parameters() para i^{ésimo} OLS 784) para um conjunto de camadas de saída (OLS) com apenas uma camada (por exemplo, OLS 3).

[0253] Em uma modalidade, a estrutura de sintaxe de DPB é designada dpb_parameters(). Em uma modalidade, a apenas uma camada é a i^{ésima} camada, e em que o OLS é o i^{ésimo} OLS. Em uma modalidade, a estrutura de sintaxe de DPB está presente em uma i^{ésima} estrutura de sintaxe profile_tier_level() no SPS. Em uma modalidade, o subfluxo de bits não inclui um conjunto de parâmetros de vídeo (VPS 704). Ou seja, o VPS foi removido como parte do processo de extração 703.

[0254] No bloco 804, o decodificador de vídeo obtém a estrutura de sintaxe de DPB para o OLS com a única camada a partir do SPS. Em uma modalidade, o SPS é referido pela apenas uma camada no OLS.

[0255] No bloco 806, o decodificador de vídeo decodifica uma imagem (por exemplo, imagem 725) a partir da apenas uma camada usando a estrutura de sintaxe de DPB para obter uma imagem decodificada. Em uma modalidade, a imagem decodificada é armazenada em um buffer de imagem decodificada (DPB) até ou antes de ser exibida.

[0256] Uma vez que a imagem tenha sido decodificada, a imagem pode ser usada para gerar ou produzir uma sequência de imagem ou vídeo para exibição a um usuário no display ou tela de um dispositivo eletrônico (por exemplo, um smartphone, tablet, laptop, computador pessoal, etc...)).

[0257] A Figura 9 é uma modalidade de um método 900 de codificação de um fluxo de bits de vídeo implementado por um codificador de vídeo (por exemplo, codificador de vídeo 300). O método 900 pode ser realizado quando uma imagem (por exemplo, de um vídeo) deve ser codificada em um fluxo de bits de vídeo e então transmitida em direção a um decodificador de vídeo (por exemplo, decodificador de vídeo 400). O método 900 melhora o processo de codificação, garantindo que os parâmetros de buffer de imagem decodificada (DPB) sejam incluídos em um conjunto de parâmetros de sequência (SPS) quando um conjunto de camadas de saída (OLS) contém apenas uma camada única. Como o SPS inclui os parâmetros de DPB para o OLS com a camada única, um conjunto de parâmetros de vídeo (VPS) pode ser removido a partir do

fluxo de bits, o que reduz redundância e aumenta eficiência de codificação. Assim, o codificador/decodificador (também conhecido como “codec”) em codificação de vídeo é aprimorado em relação aos codecs atuais. Na prática, o processo de codificação de vídeo aprimorado oferece ao usuário uma melhor experiência de usuário quando vídeos são enviados, recebidos e/ou visualizados.

[0258] No bloco 902, o codificador de vídeo gera um conjunto de parâmetros de vídeo (por exemplo, VPS 704) e um conjunto de parâmetros de sequência (por exemplo, SPS 706). O VPS inclui uma lista de estruturas de sintaxe de buffer de imagem decodificada (DPB) para todos os conjuntos de camadas de saída (por exemplo, estruturas de sintaxe de PTL 780). O SPS inclui a estrutura de sintaxe de DPB para cada OLS tendo apenas uma camada (por exemplo, dpb_parameters() para $i^{ésimo}$ OLS 784). Por exemplo, o SPS inclui a estrutura de sintaxe de DPB para OLS 3 na Figura 6, que é designado dpb_parameters() para OLS3 na Figura 7. O SPS não inclui a estrutura de sintaxe de DPB para qualquer OLS com mais de uma camada. Por exemplo, o SPS não inclui a estrutura de sintaxe de DPB para OLS 1 e OLS 2 na Figura 6, que são designados dpb_parameters() para OLS2 e dpb_parameters() para OLS3 na Figura 7.

[0259] Em uma modalidade, o VPS inclui um índice para a lista de estruturas de sintaxe de PTL, e em que o índice é designado vps_ols_dpb_params_idx[i]. Em uma modalidade, um valor de vps_ols_dpb_params_idx[i] está em uma faixa de zero a um número total de estruturas de sintaxe de DPB no VPS. Em uma modalidade, o VPS inclui vps_num_dpb_params_minus1, e em que vps_num_dpb_params_minus1 mais 1 especifica o número total das estruturas de sintaxe de DPB.

[0260] No bloco 904, o codificador de vídeo codifica o VPS e o SPS em um fluxo de bits de vídeo. No bloco 906, o codificador de vídeo armazena o fluxo de bits de vídeo para comunicação em direção a um decodificador de vídeo. Em uma modalidade, o método 900 inclui adicionalmente extrair um subfluxo de bits (por exemplo, subfluxo de bits 701) do fluxo de bits (por exemplo, fluxo de bits 700). Conforme descrito acima, o processo de extração 703 retém o SPS, mas remove o VPS. Ou seja, o subfluxo de bits inclui o SPS do fluxo de bits, mas não inclui o VPS do fluxo de bits.

[0261] como o fluxo de bits de vídeo, o fluxo de bits secundário pode ser armazenado na memória para comunicação em direção a um decodificador de vídeo. Em uma modalidade, o fluxo de bits de vídeo e/ou o fluxo de bits secundário é transmitido em direção ao decodificador de vídeo. Uma vez recebido pelo decodificador de vídeo, o fluxo de bits de vídeo codificado e/ou o fluxo de bits secundário codificado pode ser decodificado (por exemplo, conforme descrito acima) para gerar ou produzir uma imagem ou sequência de vídeo para exibição a um usuário no display ou tela de um dispositivo eletrônico (por exemplo, um smartphone, tablet, laptop, computador pessoal, etc.).

[0262] Detalhes adicionais e/ou suplementares são fornecidos abaixo.

[0263] O processo geral de decodificação é o seguinte.

[0264] A entrada para esse processo é um fluxo de bits BitstreamToDecode. A saída desse processo é uma lista de imagens decodificadas.

[0265] O processo de decodificação é especificado de tal forma que todos os decodificadores que estão em conformidade com um perfil e nível especificados produzirão imagens de saída decodificadas recortadas (cropped) numericamente idênticas ao invocar o processo de decodificação associado a esse perfil para um fluxo de bits em conformidade com esse perfil e nível. Qualquer processo de decodificação que produz imagens de saída decodificadas recortadas idênticas àquelas produzidas pelo processo descrito neste documento (com a ordem de emissão ou temporização de emissão correta, conforme especificado) está em conformidade com os requisitos do processo de decodificação do padrão VVC.

[0266] Para cada IRAP AU no fluxo de bits, o seguinte se aplica.

[0267] – Se a AU for a primeira AU no fluxo de bits em ordem de decodificação, cada imagem é uma imagem de atualização de decodificação instantânea (IDR), ou cada imagem é a primeira imagem da camada que segue uma unidade NAL de fim de sequência na ordem de decodificação, a variável NoIncorrectPicOutputFlag é definida como 1.

[0268] – De outro modo, se algum meio externo não especificado neste Relatório Descritivo estiver disponível para definir a variável HandleCraAsCvsStartFlag como um valor para a AU, HandleCraAsCvsStartFlag é definida como igual ao valor fornecido pelo meio externo e

NoIncorrectPicOutputFlag é definida como igual a HandleCraAsCvsStartFlag.

[0269] – De outro modo, HandleCraAsCvsStartFlag e NoIncorrectPicOutputFlag são definidos como 0.

[0270] Para cada GDR AU no fluxo de bits, o seguinte se aplica.

[0271] – Se a AU for a primeira AU no fluxo de bits em ordem de decodificação ou cada imagem for a primeira imagem da camada que segue uma unidade NAL de fim de sequência na ordem de decodificação, a variável NoIncorrectPicOutputFlag é definida como 1.

[0272] – De outro modo, se algum meio externo não especificado neste Relatório Descritivo estiver disponível para definir a variável HandleGdrAsCvsStartFlag como um valor para a AU, HandleGdrAsCvsStartFlag é definida como igual ao valor fornecido pelo meio externo e NoIncorrectPicOutputFlag é definida como igual a HandleGdrAsCvsStartFlag.

[0273] – De outro modo, HandleGdrAsCvsStartFlag e NoIncorrectPicOutputFlag ambos são iguais a 0.

[0274] NOTA – As operações acima, tanto para imagens IRAP quanto para imagens GDR, são necessárias para a identificação dos CVSs no fluxo de bits.

[0275] A cláusula 8.1.2 é invocada repetidamente para cada imagem codificada em BitstreamToDecode na ordem de decodificação.

[0276] O processo de decodificação para a construção de listas de imagens de referência é o seguinte.

[0277] Este processo é invocado no início do processo de decodificação para cada fatia de uma imagem não IDR.

[0278] As imagens de referência são abordadas através de índices de referência. Um índice de referência é um índice em uma lista de imagens de referência. Ao decodificar uma fatia I, nenhuma lista de imagens de referência é usada na decodificação dos dados de fatia. Ao decodificar uma fatia P, apenas a lista de imagens de referência 0 (ou seja, RefPicList[0]) é usada na decodificação dos dados de fatia. Ao decodificar uma fatia B, tanto a lista de imagens de referência 0 quanto a lista de imagens de referência 1 (ou seja, RefPicList[1]) são usadas na decodificação dos dados de fatia.

[0279] É um requisito de conformidade de fluxo de bits que as seguintes restrições se apliquem.

[0280] – Para cada i igual a 0 ou 1, num_ref_entries[i][RplsIdx[i]] não deve ser menor que NumRefIdxActive[i].

[0281] – A imagem referida por cada entrada ativa em RefPicList[0] ou RefPicList[1] deve estar presente no DPB e deve ter TemporalId menor que ou igual ao da imagem atual.

[0282] – A imagem referida por cada entrada em RefPicList[0] ou RefPicList[1] não deve ser a imagem atual e deve ter non_reference_picture_flag igual a 0.

[0283] – Uma entrada de imagem de referência de curto prazo (STRP) em RefPicList[0] ou RefPicList[1] de uma fatia de uma imagem e uma entrada de LTRP em RefPicList[0] ou RefPicList[1] da mesma fatia ou de uma fatia diferente da mesma imagem não devem se referir à mesma imagem.

[0284] – Não deve haver nenhuma entrada de LTRP em RefPicList[0] ou RefPicList[1] para a qual a diferença entre PicOrderCntVal da imagem atual e PicOrderCntVal da imagem referida pela entrada seja maior que ou igual a 224.

[0285] – Deixe setOfRefPics ser o conjunto de imagens únicas referidas por todas as entradas em RefPicList[0] que têm o mesmo nuh_layer_id que a imagem atual e todas as entradas em RefPicList[1] que têm o mesmo nuh_layer_id que a imagem atual. O número de imagens em setOfRefPics deve ser menor que ou igual a MaxDecPicBuffMinus1 e setOfRefPics deve ser o mesmo para todas as fatias de uma imagem.

[0286] – Quando a imagem atual é uma imagem de Acesso de Subcamada Temporal Passo-a-Passo (step-wise) (STSA), não deve haver nenhuma entrada ativa em RefPicList[0] ou RefPicList[1] que tem TemporalId igual ao da imagem atual.

[0287] Quando a imagem atual for uma imagem que segue, em ordem de decodificação, uma imagem de STSA que tem TemporalId igual ao da imagem atual, não deve haver nenhuma imagem que tem TemporalId igual à da imagem atual incluída como entrada ativa em RefPicList [0] ou RefPicList[1] que precede a imagem de STSA na ordem de decodificação.

[0288] – A imagem referida por cada entrada de ILRP em RefPicList[0] ou RefPicList[1] de uma fatia da imagem atual deve estar na mesma unidade de acesso que a imagem atual.

[0289] – A imagem referida por cada entrada de ILRP em RefPicList[0]

ou RefPicList[1] de uma fatia da imagem atual deve estar presente no DPB e deve ter nuh_layer_id menor que o da imagem atual.

[0290] – Cada entrada de ILRP em RefPicList[0] ou RefPicList[1] de uma fatia deve ser uma entrada ativa.

[0291] Os aspectos gerais de DRH na cláusula C.1 do padrão VVC incluem o seguinte.

[0292] Este anexo especifica o decodificador de referência hipotético (HRD) e seu uso para verificar a conformidade do fluxo de bits e do decodificador.

[0293] Um conjunto de testes de conformidade de fluxo de bits é necessário para verificar a conformidade de um fluxo de bits, que é chamado de fluxo de bits inteiro, denotado como fluxo de bits inteiro. O conjunto de testes de conformidade de fluxo de bits são para testar a conformidade de cada OP de cada OLS especificado pelo VPS.

[0294] Para cada teste, as etapas ordenadas a seguir se aplicam na ordem listada, seguidas pelos processos descritos após essas etapas nesta cláusula:

[0295] 1. Um ponto de operação em teste, denotado como targetOp, é selecionado por selecionar um OLS alvo com índice de OLS opOlsIdx e um valor de TemporalId mais alto opTid. O valor de opOlsIdx está na faixa de 0 a TotalNumOlss − 1, inclusive. O valor de opTid está na faixa de 0 a vps_max_sub_layers_minus1, inclusive. Cada par de valores selecionados de opOlsIdx e opTid deve ser tal que o subfluxo de bits que é emitido por invocar o processo de extração de subfluxo de bits conforme especificado na cláusula C.6 com integerBitstream, opOlsIdx e opTid como entradas satisfaça as seguintes condições:

[0296] – Há pelo menos uma unidade VCL NAL com nuh_layer_id igual a cada um dos valores de nuh_layer_id em LayerIdInOls[opOlsIdx] em BitstreamToDecode.

[0297] – Há pelo menos uma unidade VCL NAL com TemporalId igual a opTid em BitstreamToDecode.

[0298] 2. Se as camadas em targetOp incluem todas as camadas em integerBitstream e opTid for igual a maior que o valor de TemporalId mais alto entre todas as unidades NAL em integerBitstream, BitstreamToDecode será

definido como idêntico a integerBitstream. De outro modo, BitstreamToDecode é definido para ser emitido por invocar o processo de extração de subfluxo de bits conforme especificado na cláusula C.6 com integerBitstream, opOlsIdx e opTid como entradas.

[0299] 3. Os valores de TargetOlsIdx e Htid são definidos como iguais a opOlsIdx e opTid, respectivamente, de targetOp.

[0300] 4. Um valor de ScIdx é selecionado. O ScIdx selecionado deve estar na faixa de 0 a hrd_cpb_cnt_minus1, inclusive.

[0301] 5. Uma unidade de acesso em BitstreamToDecode associada a mensagens SEI de período de armazenamento em buffer (presente em TargetLayerBitstream ou disponível por meios externos não especificados neste Relatório Descritivo) aplicáveis a TargetOlsIdx é selecionada como o ponto de inicialização de HRD e referida como unidade de acesso 0 para cada camada no OLS alvo.

[0302] 6. A estrutura de sintaxe ols_hrd_parameters() e a estrutura de sintaxe sub_layer_hrd_parameters() aplicáveis a BitstreamToDecode são selecionadas da seguinte forma:

[0303] – A estrutura de sintaxe ols_hrd_idx[TargetOlsIdx]-th ols_hrd_parameters() no VPS (ou fornecida através de meios externos não especificados neste Relatório Descritivo) é selecionada.

[0304] – Dentro da estrutura de sintaxe ols_hrd_parameters() selecionada, se BitstreamToDecode for um fluxo de bits Tipo I, a estrutura de sintaxe sub_layer_hrd_parameters(Htid) que segue imediatamente a condição "if(general_vcl_hrd_params_present_flag)" é selecionada e a variável NalHrdModeFlag é definida como igual a 0; caso contrário (BitstreamToDecode é um fluxo de bits Tipo II), a estrutura de sintaxe sub_layer_hrd_parameters(Htid) que segue imediatamente a condição "if(general_vcl_hrd_params_present_flag)" (neste caso a variável NalHrdModeFlag é definida como igual a 0) ou a condição "if(general_nal_hrd_params_present_flag)” (neste caso a variável NalHrdModeFlag é definida como igual a 1) é selecionada. Quando BitstreamToDecode é um fluxo de bits Tipo II e NalHrdModeFlag é igual a 0, todas as unidades não VCL NAL, exceto unidades NAL de dados de preenchimento, e todos os elementos de sintaxe lead_zero_8bits, zero_byte,

start_code_prefix_one_3bytes e trailing_zero_8bits que formam um fluxo de bytes do fluxo de unidade NAL (conforme especificado no Anexo B), quando presentes, são descartados de BitstreamToDecode e o fluxo de bits restante é atribuído a BitstreamToDecode.

[0305] 7. Quando decoding_unit_hrd_params_present_flag for igual a 1, o CPB é agendado para operar no nível de unidade de acesso (nesse caso a variável DecodingUnitHrdFlag é definida como igual a 0) ou no nível de unidade de decodificação (nesse caso a variável DecodingUnitHrdFlag é definida como igual a 1). De outro modo, DecodingUnitHrdFlag é definida como igual a 0 e o CPB é agendado para operar no nível de unidade de acesso.

[0306] 8. Para cada unidade de acesso em BitstreamToDecode a partir da unidade de acesso 0, é selecionada a mensagem SEI de período de armazenamento em buffer (presente em BitstreamToDecode ou disponível por meios externos não especificados neste Relatório Descritivo) que está associada à unidade de acesso e se aplica a TargetOlsIdx, a imagem mensagem SEI de temporização (presente em BitstreamToDecode ou disponível através de meios externos não especificados neste Relatório Descritivo) que está associada à unidade de acesso e se aplica a TargetOlsIdx é selecionada, e quando DecodingUnitHrdFlag for igual a 1 e decoding_unit_cpb_params_in_pic_timing_sei_flag for igual a 0, as mensagens SEI de informações de unidade de decodificação (presentes em BitstreamToDecode ou disponíveis por meios externos não especificados neste Relatório Descritivo) que estão associadas a unidades de decodificação na unidade de acesso e se aplicam a TargetOlsIdx são selecionadas.

[0307] Cada teste de conformidade consiste em uma combinação de uma opção em cada uma das etapas acima. Quando houver mais de uma opção para uma etapa, para qualquer teste de conformidade particular, apenas uma opção é escolhida. Todas as combinações possíveis de todas as etapas formam o conjunto completo de testes de conformidade. Para cada ponto de operação em teste, o número de testes de conformidade de fluxo de bits a serem realizados é igual a n0 * n1 * n2 * n3, em que os valores de n0, n1, n2 e n3 são especificados da seguinte forma:

[0308] – n1 é igual a hrd_cpb_cnt_minus1 + 1.

[0309] – n1 é o número de unidades de acesso em

BitstreamToDecode que estão associadas às mensagens SEI de período de armazenamento em buffer.

[0310] – n2 é derivado da seguinte forma:

[0311] – Se BitstreamToDecode for um fluxo de bits Tipo I, n0 será igual a 1.

[0312] – De outro modo (BitstreamToDecode é um fluxo de bits Tipo II), n0 é igual a 2.

[0313] – n3 é derivado da seguinte forma:

[0314] – Se decoding_unit_hrd_params_present_flag for igual a 0, n3 será igual a 1.

[0315] – De outro modo, n3 é igual a 2.

[0316] O HRD contém um extrator de fluxo de bits (opcionalmente presente), um buffer de imagem codificada (CPB), um processo de decodificação instantâneo, um buffer de imagem decodificada (DPB) que contém conceitualmente um sub-DPB para cada camada, e cropping de saída.

[0317] Para cada teste de conformidade de fluxo de bits, o tamanho de CPB (número de bits) é CpbSize[Htid][ScIdx] conforme especificado na cláusula 7.4.5.2, em que ScIdx e os parâmetros de HRD são especificados acima nesta cláusula e os parâmetros de DPB max_dec_pic_buffering_minus1[Htid], max_num_reorder_pics[Htid] e MaxLatencyPictures[Htid] para cada camada são encontrados ou derivados da estrutura de sintaxe dpb_parameters() que se aplica à camada dependendo se a camada é uma camada independente e se a camada é uma camada de saída do OLS alvo.

[0318] O HRD funciona da seguinte forma.

[0319] – O HRD é inicializado na unidade de decodificação 0, com o CPB e cada sub-DPB do DPB sendo definido como vazio (o preenchimento do sub-DPB para cada sub-DPB é definido como igual a 0).

[0320] NOTA – Após a inicialização, o HRD não é inicializado novamente pelas mensagens SEI de período de armazenamento em buffer subsequente.

[0321] – Os dados associados às unidades de decodificação que fluem para cada CPB de acordo com um agendamento de chegada especificado são entregues pelo agendador de fluxo hipotético (HSS).

[0322] – Os dados associados a cada unidade de decodificação são

removidos e decodificados instantaneamente pelo processo de decodificação instantâneo no momento da remoção do CPB da unidade de decodificação.

[0323] – Cada imagem decodificada é colocada no DPB.

[0324] – Uma imagem decodificada é removida do DPB quando ela não for mais necessária para referência de predição inter e não for mais necessária para emissão.

[0325] A operação do buffer de imagem decodificada é fornecida.

[0326] As especificações nesta cláusula se aplicam independentemente a cada conjunto de parâmetros de buffer de imagem decodificada (DPB) selecionados conforme especificado na cláusula C.1.

[0327] O buffer de imagem decodificada conceitualmente consiste em sub-DPBs e cada sub-DPB contém buffers de armazenamento de imagem para armazenamento de imagens decodificadas de uma camada. Cada um dos buffers de armazenamento de imagem pode conter uma imagem decodificada que é marcada como "usada para referência" ou é mantida para emissão futura. Os processos especificados nas cláusulas C3.2, C3.3 e C3.4 são aplicados sequencialmente conforme especificado abaixo e são aplicados independentemente para cada camada, começando da camada mais baixa no OLS, em ordem crescente de valores de nuh_layer_id das camadas no OLS. Quando esses processos são aplicados a uma camada particular, apenas o sub-DPB da camada particular é afetado. Nas descrições desses processos, o DPB refere-se ao sub-DPB para a camada particular, e a camada particular é referida como a camada atual.

[0328] NOTA – Na operação de DPB de temporização de emissão, as imagens decodificadas com PicOutputFlag igual a 1 na mesma unidade de acesso são emitidas consecutivamente em ordem crescente dos valores de nuh_layer_id das imagens decodificadas.

[0329] Deixe a imagem n e a imagem atual serem a imagem codificada ou a imagem decodificada da unidade de acesso n para um valor particular de nuh_layer_id, em que n é um número inteiro não negativo.

[0330] A remoção de imagens a partir do DPB antes de decodificar a imagem atual é descrita.

[0331] A remoção das imagens a partir do DPB antes da decodificação da imagem atual (mas depois de analisar o cabeçalho de fatia da

primeira fatia da imagem atual) acontece instantaneamente no momento da remoção do CPB da primeira unidade de decodificação da unidade de acesso n (contendo a imagem atual) e procede da seguinte forma:

[0332] – O processo de decodificação para construção de lista de imagens de referência conforme especificado na cláusula 8.3.2 é invocado e o processo de decodificação para marcação de imagens de referência conforme especificado na cláusula 8.3.3 é invocado.

[0333] – Quando a AU atual é uma AU CVSS que não é AU 0, as seguintes etapas ordenaaos são aplicadas:

[0334] 1. A variável NoOutputOfPriorPicsFlag é derivada para o decodificador em teste da seguinte forma:

[0335] - Se o valor de pic_width_max_in_luma_samples, pic_height_max_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 ou max_dec_pic_buffering_minus1 [htid] derivado de qualquer imagem na atual AU é diferente do valor de pic_width_in_luma_samples, pic_height_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 ou max_dec_pic_buffering_minus1 [htid], respectivamente, derivado para a imagem anterior no mesmo CLVS, NoOutputOfPriorPicsFlag pode (mas não deve) ser definido como 1 pelo decodificador em teste, independentemente do valor de no_output_of_prior_pics_flag.

[0336] NOTA – Embora definir NoOutputOfPriorPicsFlag como igual a no_output_of_prior_pics_flag seja preferível nessas condições, o decodificador em teste está permitido a definir NoOutputOfPriorPicsFlag como 1 neste caso.

[0337] – De outro modo, NoOutputOfPriorPicsFlag é definida como igual a no_output_of_prior_pics_flag.

[0338] 2. O valor de NoOutputOfPriorPicsFlag derivado para o decodificador em teste é aplicado para o HRD, de modo que quando o valor de NoOutputOfPriorPicsFlag for igual a 1, todos os buffers de armazenamento de imagem no DPB são esvaziados sem emissão das imagens que eles contêm e a totalidade de DPB é definida como igual a 0.

[0339] – Quando ambas as condições a seguir forem verdadeiras para

quaisquer imagens k no DPB, todas essas imagens k no DPB são removidas do DPB:

[0340] – a imagem k está marcada como "não usada para referência".

[0341] – a imagem k tem PictureOutputFlag igual a 0 ou seu tempo de emissão de DPB é menor que ou igual ao tempo de remoção de CPB da primeira unidade de decodificação (denotada como unidade de decodificação m) da imagem atual n; ou seja, DpbOutputTime[k] é menor que ou igual a DuCpbRemovalTime[m].

[0342] – Para cada imagem removida do DPB, a totalidade de DPB é decrementada em um.

[0343] A operação do DPB de ordem de emissão é descrita.

[0344] As especificações nesta cláusula se aplicam independentemente a cada conjunto de parâmetros de buffer de imagem decodificada (DPB) selecionados conforme especificado na cláusula C.1.

[0345] O buffer de imagem decodificada conceitualmente consiste em sub-DPBs e cada sub-DPB contém buffers de armazenamento de imagem para armazenamento de imagens decodificadas de uma camada. Cada um dos buffers de armazenamento de imagem contém uma imagem decodificada que é marcada como "usada para referência" ou é mantida para emissão futura.

[0346] O processo para emissão e remoção de imagens a partir do DPB antes da decodificação da imagem atual conforme especificado na cláusula C.5.2.2 é invocado, seguido pela invocação do processo para marcação e armazenamento da imagem decodificada atual conforme especificado na cláusula C.3.4, e finalmente seguido pela invocação do processo para bumping adicional conforme especificado na cláusula C.5.2.3. O processo de bumping é especificado na cláusula C.5.2.4 e é invocado conforme especificado nas cláusulas C.5.2.2 e C.5.2.3.

[0347] Esses processos são aplicados independentemente para cada camada, começando pela camada mais baixa no OLS, em ordem crescente dos valores de nuh_layer_id das camadas no OLS. Quando esses processos são aplicados a uma camada particular, apenas o sub-DPB da camada particular é afetado.

[0348] NOTA – Na operação de DPB de ordem de emissão, assim como na operação de DPB de temporização de emissão, as imagens

decodificadas com PicOutputFlag igual a 1 na mesma unidade de acesso também são emitidas consecutivamente em ordem crescente dos valores de nuh_layer_id das imagens decodificadas.

[0349] Deixe a imagem n e a imagem atual serem a imagem codificada ou a imagem decodificada da unidade de acesso n para um valor particular de nuh_layer_id, em que n é um número inteiro não negativo.

[0350] A emissão e remoção de imagens a partir do DPB são descritas.

[0351] A emissão e remoção de imagens a partir do DPB antes da decodificação da imagem atual (mas após analisar o cabeçalho de fatia da primeira fatia da imagem atual) acontecem instantaneamente quando a primeira unidade de decodificação da unidade de acesso contendo a imagem atual é removida do CPB e procede da seguinte forma:

[0352] – O processo de decodificação para a construção de lista de imagens de referência conforme especificado na cláusula 8.3.2 e o processo de decodificação para a marcação de imagens de referência conforme especificado na cláusula 8.3.3 são invocados.

[0353] – Se a AU atual for uma AU CVSS que não seja AU0, as seguintes etapas ordenadas serão aplicadas:

[0354] 1. A variável NoOutputOfPriorPicsFlag é derivada para o decodificador em teste da seguinte forma:

[0355] - Se o valor de pic_width_max_in_luma_samples, pic_height_max_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 ou max_dec_pic_buffering_minus1 [htid] derivado de qualquer imagem da AU atual é diferente do valor de pic_width_in_luma_samples, pic_height_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, bit_depth_chroma_minus8 ou max_dec_pic_buffering_minus1 [htid], respectivamente, derivado para a imagem anterior no mesmo CLVS, NoOutputOfPriorPicsFlag pode (mas não deve) ser definido como 1 pelo decodificador em teste, independentemente do valor de no_output_of_prior_pics_flag.

[0356] NOTA – Embora definir NoOutputOfPriorPicsFlag como igual a

no_output_of_prior_pics_flag seja preferível nessas condições, o decodificador em teste está permitido a definir NoOutputOfPriorPicsFlag como 1 neste caso.

[0357] – De outro modo, NoOutputOfPriorPicsFlag é definida como igual a no_output_of_prior_pics_flag.

[0358] 2. O valor de NoOutputOfPriorPicsFlag derivado para o decodificador em teste é aplicado para o HRD da seguinte forma:

[0359] – Se NoOutputOfPriorPicsFlag for igual a 1, todos os buffers de armazenamento de imagens no DPB serão esvaziados sem emissão das imagens que eles contêm e a totalidade de DPB será definida como igual a 0.

[0360] – De outro modo (NoOutputOfPriorPicsFlag é igual a 0), todos os buffers de armazenamento de imagem contendo uma imagem marcada como “não necessária para emissão” e “não usada para referência" são esvaziados (sem emissão) e todos os buffers de armazenamento de imagem não vazios no DPB são esvaziados por invocar repetidamente o processo de bumping especificado na cláusula C.5.2.4 e a totalidade de DPB é definida como igual a 0.

[0361] – De outro modo (a imagem atual não é uma imagem de CLVSS), todos os buffers de armazenamento de imagem contendo uma imagem marcada como "não necessária para emissão” e "não usada para referência" são esvaziados (sem emissão). Para cada buffer de armazenamento de imagem que é esvaziado, a totalidade de DPB é decrementada em um. Quando uma ou mais das seguintes condições são verdadeiras, o processo de bumping especificado na cláusula C.5.2.4 é invocado repetidamente enquanto adicionalmente decrementa a totalidade de DPB em um para cada buffer de armazenamento de imagem adicional que é esvaziado, até que nenhuma das seguintes condições seja verdadeira:

[0362] O número de imagens no DPB que são marcadas como "necessárias para emissão” é maior que max_num_reorder_pics[Htid].

[0363] max_latency_increase_plus1[Htid] não é igual a 0 e há pelo menos uma imagem no DPB marcada como "necessária para emissão” para a qual a variável associada PicLatencyCount é maior que ou igual a MaxLatencyPictures[Htid].

[0364] O número de imagens no DPB é maior que ou igual a max_dec_pic_buffering_minus1[Htid] + 1.

[0365] Bumping adicional é descrito.

[0366] Os processos especificados nesta cláusula acontecem instantaneamente quando a última unidade de decodificação da unidade de acesso n contendo a imagem atual é removida do CPB.

[0367] Quando a imagem atual tem PictureOutputFlag igual a 1, para cada imagem no DPB que está marcada como "necessária para emissão” e segue a imagem atual na ordem de emissão, a variável associada PicLatencyCount é definida como igual a PicLatencyCount + 1.

[0368] Aplica-se o seguinte:

[0369] – Se a imagem decodificada atual tiver PictureOutputFlag igual a 1, ela é marcada como "necessária para emissão” e sua variável associada PicLatencyCount é definida como igual a 0.

[0370] – De outro modo (a imagem decodificada atual tem PictureOutputFlag igual a 0), ela é marcada como "não necessária para emissão".

[0371] Quando uma ou mais das condições a seguir são verdadeiras, o processo de bumping especificado na cláusula C.5.2.4 é invocado repetidamente até que nenhuma das seguintes condições seja verdadeira:

[0372] – O número de imagens no DPB que são marcadas como "necessárias para emissão” é maior que max_num_reorder_pics[Htid].

[0373] – max_latency_increase_plus1[Htid] não é igual a 0 e há pelo menos uma imagem no DPB marcada como "necessária para emissão” para a qual a variável associada PicLatencyCount é maior que ou igual a MaxLatencyPictures[Htid].

[0374] O processo de bumping é descrito.

[0375] O processo de bumping consiste nas seguintes etapas ordenadas:

[0376] A imagem ou imagens que são as primeiras para emissão são selecionadas como aquelas tendo o menor valor de PicOrderCntVal de todas as imagens no DPB marcadas como "necessárias para emissão".

[0377] Cada uma dessas imagens, em ordem crescente de nuh_layer_id, é recortada, usando a janela de cropping de conformidade para a imagem, a imagem recortada é emitida, e a imagem é marcada como "não necessária para emissão".

[0378] Cada buffer de armazenamento de imagem que contém uma imagem marcada como "não usada para referência" e que foi uma das imagens recortadas e emitidas é esvaziada e a totalidade do sub-DPB associado é decrementada em um.

[0379] NOTA – Para quaisquer duas imagens picA e picB que pertençam ao mesmo CVS e sejam emitidas pelo "processo de bumping", quando picA é emitida antes de picB, o valor de PicOrderCntVal de picA é menor que o valor de PicOrderCntVal de picB.

[0380] O processo de extração de subfluxo de bits é descrito.

[0381] As entradas para esse processo são um fluxo de bits inBitstream, um índice de OLS alvo targetOlsIdx e um valor de TemporalId mais alto alvo tIdTarget.

[0382] A saída desse processo é um outBitstream subfluxo de bits.

[0383] É um requisito de conformidade de fluxo de bits para o fluxo de bits de entrada que qualquer subfluxo de bits de saída que seja a saída do processo especificado nesta cláusula com o fluxo de bits, targetOlsIdx igual a um índice para a lista de OLSs especificada pelo VPS e tIdTarget igual a qualquer valor na faixa de 0 a 6, inclusive, como entradas, e que satisfaça as seguintes condições deve ser um fluxo de bits em conformidade:

[0384] – O subfluxo de bits de saída contém pelo menos uma unidade VCL NAL com nuh_layer_id igual a cada um dos valores de nuh_layer_id em LayerIdInOls[targetOlsIdx].

[0385] – O subfluxo de bits de saída contém pelo menos uma unidade VCL NAL com TemporalId igual a tIdTarget.

[0386] NOTA – Um fluxo de bits em conformidade contém uma ou mais unidades NAL de fatia codificadas com TemporalId igual a 0, mas não precisa conter unidades NAL de fatia codificadas com nuh_layer_id igual a 0.

[0387] O subfluxo de bits de saída OutBitstream é derivado da seguinte forma:

[0388] – O fluxo de bits outBitstream é definido para ser idêntico ao fluxo de bits inBitstream.

[0389] – Remover a partir do outBitstream todas as unidades NAL com TemporalId maior que tIdTarget.

[0390] – Remover a partir do outBitstream todas as unidades NAL

com nuh_layer_id não incluídas na lista LayerIdInOls[targetOlsIdx].

[0391] – Remover a partir do outBitstream todas as unidades SEI NAL que contêm uma mensagem SEI de aninhamento escalável que tem nesting_ols_flag igual a 1 e não há valor de i na faixa de 0 a nesting_num_olss_minus1, inclusive, de modo que NestingOlsIdx[i] seja igual a targetOlsIdx.

[0392] – Quando targetOlsIdx for maior que 0, remover do outBitstream todas as unidades SEI NAL que contêm uma mensagem SEI aninhada não escalável com payloadType igual a 0 (período de armazenamento em buffer), 1 (temporização de imagem) ou 130 (informações de unidade de decodificação).

[0393] A sintaxe de mensagem SEI de aninhamento escalável é fornecida.

scalable_nesting(payloadSize) {	Descritor
nesting_ols_flag	u(1)
if(nesting_ols_flag) {	
nesting_num_olss_minus1	ue(v)
for(i = 0; i <= nesting_num_olss_minus1; i++)	
nesting_ols_idx_delta_minus1[i]	ue(v)
} else {	
nesting_all_layers_flag	u(1)
if(!nesting_all_layers_flag) {	
nesting_num_layers_minus1	ue(v)
for(i = 1; i <= nesting_num_layers_minus1; i++)	
nesting_layer_id[i]	u(6)
}	
}	
nesting_num_seis_minus1	ue(v)
while(!byte_aligned())	
nesting_zero_bit /* equal to 0 */	u(1)
for(i = 0; i <= nesting_num_seis_minus1; i++)	
sei_message()	

}	

[0394] A semântica de carga útil de SEI geral é fornecida.

[0395] O seguinte se aplica às camadas aplicáveis ou OLS de mensagens SEI aninhadas não escaláveis.

[0396] – Para uma mensagem SEI aninhada não escalável, quando payloadType for igual a 0 (período de armazenamento em buffer), 1 (temporização de imagem) ou 130 (informações de unidade de decodificação), a mensagem SEI aninhada não escalável se aplica apenas ao 0ésimo OLS.

[0397] – Para uma mensagem SEI aninhada não escalável, quando payloadType for igual a qualquer valor entre VclAssociatedSeiList, a mensagem SEI aninhada não escalável se aplica apenas à camada para a qual as unidades VCL NAL têm nuh_layer_id igual ao nuh_layer_id da unidade SEI NAL que contém a mensagem SEI.

[0398] É um requisito de conformidade de fluxo de bits que as seguintes restrições se apliquem ao valor de nuh_layer_id de unidades SEI NAL:

[0399] – Quando uma mensagem SEI aninhada não escalável tem payloadType igual a 0 (período de armazenamento em buffer), 1 (temporização de imagem) ou 130 (informações de unidade de decodificação), a unidade SEI NAL que contém a mensagem SEI aninhada não escalável deve ter nuh_layer_id igual a vps_layer_id[0].

[0400] – Quando uma mensagem SEI aninhada não escalável tem payloadType igual a qualquer valor entre VclAssociatedSeiList, a unidade SEI NAL que contém a mensagem SEI aninhada não escalável deve ter nuh_layer_id igual ao valor de nuh_layer_id da unidade VCL NAL associada à unidade SEI NAL.

[0401] – Uma unidade SEI NAL contendo uma mensagem SEI de aninhamento escalável deve ter nuh_layer_id igual ao valor mais baixo de nuh_layer_id de todas as camadas às quais a mensagem SEI aninhada escalável se aplica (quando nesting_ols_flag da mensagem SEI de aninhamento escalável é igual a 0) ou o menor valor de nuh_layer_id de todas as camadas nos OLSs aos quais a mensagem SEI aninhada escalável se aplica (quando nesting_ols_flag da mensagem SEI de aninhamento escalável é igual a 1).

[0402] A semântica de mensagem SEI de aninhamento escalável é fornecida.

[0403] A mensagem SEI de aninhamento escalável fornece um mecanismo para associar mensagens SEI com OLSs específicos ou com camadas específicas.

[0404] Uma mensagem SEI de aninhamento escalável contém uma ou mais mensagens SEI. As mensagens SEI contidas na mensagem SEI de aninhamento escalável também são chamadas de mensagens SEI aninhadas escaláveis.

[0405] É um requisito de conformidade de fluxo de bits que as seguintes restrições se apliquem à contenção de mensagens SEI em uma mensagem SEI de aninhamento escalável:

[0406] – Uma mensagem SEI que tem payloadType igual a 132 (hash de imagem decodificado) ou 133 (aninhamento escalável) não deve estar contida em uma mensagem SEI de aninhamento escalável.

[0407] – Quando uma mensagem SEI de aninhamento escalável contém um período de armazenamento em buffer, temporização de imagem ou mensagem SEI de informações de unidade de decodificação, a mensagem SEI de aninhamento escalável não deve conter nenhuma outra mensagem SEI com payloadType diferente de 0 (período de armazenamento em buffer), 1 (temporização de imagem), ou 130 (informações de unidade de decodificação).

[0408] É um requisito de conformidade de fluxo de bits que as seguintes restrições se apliquem ao valor do nal_unit_type da unidade SEI NAL que contém uma mensagem SEI de aninhamento escalável:

[0409] – Quando uma mensagem SEI de aninhamento escalável contém uma mensagem SEI que tem payloadType igual a 0 (período de armazenamento em buffer), 1 (temporização de imagem), 130 (informações de unidade de decodificação), 145 (indicação de RAP dependente) ou 168 (informações de campo de quadro), a unidade SEI NAL que contém a mensagem SEI de aninhamento escalável deve ter nal_unit_type igual a PREFIX_SEI_NUT.

[0410] nesting_ols_flag igual a 1 especifica que as mensagens SEI aninhadas escaláveis se aplicam a OLSs específicos. nesting_ols_flag igual a 0 especifica que as mensagens SEI aninhadas escaláveis se aplicam a camadas específicas.

[0411] É um requisito de conformidade de fluxo de bits que as seguintes restrições se apliquem ao valor de nesting_ols_flag:

[0412] – Quando a mensagem SEI de aninhamento escalável contém uma mensagem SEI que tem payloadType igual a 0 (período de armazenamento em buffer), 1 (temporização de imagem) ou 130 (informações de unidade de decodificação), o valor de nesting_ols_flag deve ser igual a 1.

[0413] – Quando a mensagem SEI de aninhamento escalável contém uma mensagem SEI que tem payloadType igual a um valor em VclAssociatedSeiList, o valor de nesting_ols_flag deve ser igual a 0.

[0414] nesting_num_olss_minus1 mais 1 especifica o número de OLSs aos quais as mensagens SEI aninhadas escaláveis se aplicam. O valor de nesting_num_olss_minus1 deve estar na faixa de 0 a TotalNumOlss − 1, inclusive.

[0415] nesting_ols_idx_delta_minus1[i] é usado para derivar a variável NestingOlsIdx[i] que especifica o índice de OLS do i^{ésimo} OLS ao qual as mensagens SEI aninhadas escaláveis se aplicam quando nesting_ols_flag é igual a 1. O valor de nesting_ols_idx_delta_minus1[i] deve estar na faixa de 0 a TotalNumOlss − 2, inclusive, inclusive.

[0416] A variável NestingOlsIdx[i] é derivada da seguinte forma.

```
if(i = = 0)
    NestingOlsIdx[i] = nesting_ols_idx_delta_minus1[i]
else
    NestingOlsIdx[i] = NestingOlsIdx[i−1] +
nesting_ols_idx_delta_minus1[i] + 1
```

[0417] nesting_all_layers_flag igual a 1 especifica que as mensagens SEI aninhadas escaláveis se aplicam a todas as camadas que têm nuh_layer_id maior que ou igual ao nuh_layer_id da unidade SEI NAL atual. nesting_all_layers_flag igual a 0 especifica que as mensagens SEI aninhadas escaláveis podem ou não se aplicar a todas as camadas que têm nuh_layer_id maior que ou igual ao nuh_layer_id da unidade SEI NAL atual.

[0418] nesting_num_layers_minus1 mais 1 especifica o número de camadas às quais as mensagens SEI aninhadas escaláveis se aplicam. O valor de nesting_num_layers_minus1 deve estar na faixa de 0 a vps_max_layers_minus1 − GeneralLayerIdx[nuh_layer_id], inclusive, em que nuh_layer_id é o nuh_layer_id da unidade SEI NAL atual.

[0419] nesting_layer_id[i] especifica o valor de nuh_layer_id da i^{ésima}

camada à qual as mensagens SEI aninhadas escaláveis se aplicam quando nesting_all_layers_flag é igual a 0. O valor de nesting_layer_id[i] deve ser maior que nuh_layer_id, em que nuh_layer_id é o nuh_layer_id da unidade SEI NAL atual.

[0420] Quando nesting_ols_flag é igual a 0, a variável NestingNumLayers, especificando o número de camadas às quais as mensagens SEI aninhadas escaláveis se aplicam, e a lista NestingLayerId[i] para i na faixa de 0 a NestingNumLayers − 1, inclusive, especificando a lista do valor de nuh_layer_id das camadas às quais as mensagens SEI aninhadas escaláveis se aplicam, são derivadas como segue, em que nuh_layer_id é o nuh_layer_id da unidade SEI NAL atual.

```
if(nesting_all_layers_flag) {
   NestingNumLayers =
ps_max_layers_minus1 + 1 − GeneralLayerIdx[nuh_layer_id]
   for(i = 0; i < NestingNumLayers; i ++)
         NestingLayerId[i] =
vps_layer_id[GeneralLayerIdx[nuh_layer_id] + i]
} else {
   NestingNumLayers = nesting_num_layers_minus1 + 1
   for(i = 0; i < NestingNumLayers; i ++)
         NestingLayerId[i] = (i  = = 0) ? nuh_layer_id :
nesting_layer_id[i]
}
```

[0421] nesting_num_seis_minus1 mais 1 especifica o número de mensagens SEI aninhadas escaláveis. O valor de nesting_num_seis_minus1 deve estar na faixa de 0 a 63, inclusive.

[0422] nesting_zero_bit deve ser igual a 0.

[0423] A Figura 10 é um diagrama esquemático de um dispositivo de codificação de vídeo 1000 (por exemplo, um codificador de vídeo 300 ou um decodificador de vídeo 400) de acordo com uma modalidade da revelação. O dispositivo de codificação de vídeo 1000 é adequado para implementar as modalidades divulgadas conforme descrito neste documento. O dispositivo de codificação de vídeo 1000 compreende portas de ingresso 1010 e unidades receptoras (Rx) 1020 para receber dados; um processador, unidade lógica ou

unidade de processamento central (CPU) 1030 para processar os dados; unidades transmissoras (Tx) 1040 e portas de egresso 1050 para transmissão dos dados; e uma memória 1060 para armazenar os dados. O dispositivo de codificação de vídeo 1000 também pode compreender componentes óptico-elétricos (OE) e componentes elétrico-ópticos (EO) acoplados às portas de ingresso 1010, às unidades receptoras 1020, às unidades transmissoras 1040 e às portas de egresso 1050 para egresso ou ingresso de sinais ópticos ou elétricos.

[0424] O processador 1030 é implementado por hardware e software. O processador 1030 pode ser implementado como um ou mais chips de CPU, núcleos (por exemplo, como um processador de múltiplos núcleos), arranjos de portas programáveis em campo (FPGAs), circuitos integrados de aplicação específica (ASICs) e processadores de sinal digital (DSPs). O processador 1030 está em comunicação com as portas de ingresso 1010, unidades receptoras 1020, unidades transmissoras 1040, portas de egresso 1050 e memória 1060. O processador 1030 compreende um módulo de codificação 1070. O módulo de codificação 1070 implementa as modalidades divulgadas descritas acima. Por exemplo, o módulo de codificação 1070 implementa, processa, prepara ou fornece as várias funções de codec. A inclusão do módulo de codificação 1070 fornece, portanto, uma melhoria substancial na funcionalidade do dispositivo de codificação de vídeo 1000 e efetua uma transformação do dispositivo de codificação de vídeo 1000 para um estado diferente. Alternativamente, o módulo de codificação 1070 é implementado como instruções armazenadas na memória 1060 e executadas pelo processador 1030.

[0425] O dispositivo de codificação de vídeo 1000 também pode incluir dispositivos de entrada e/ou saída (I/O) 1080 para comunicação de dados de e para um usuário. Os dispositivos de I/O 1080 podem incluir dispositivos de saída, como um display para exibir dados de vídeo, alto-falantes para emitir dados de áudio, etc. Os dispositivos de I/O 1080 também podem incluir dispositivos de entrada, como teclado, mouse, trackball, etc., e/ou interfaces correspondentes para interagir com tais dispositivos de saída.

[0426] A memória 1060 compreende um ou mais discos, unidades de fita e unidades de estado sólido e pode ser usada como um dispositivo de armazenamento de dados de sobrefluxo, para armazenar programas quando tais

programas são selecionados para execução e para armazenar instruções e dados que são lidos durante a execução do programa. A memória 1060 pode ser volátil e/ou não volátil e pode ser memória somente de leitura (ROM), memória de acesso aleatório (RAM), memória endereçável por conteúdo ternário (TCAM) e/ou memória de acesso aleatório estática (SRAM).

[0427] A Figura 11 é um diagrama esquemático de uma modalidade de um meio para codificação 1100. Em uma modalidade, o meio para codificação 1100 é implementado em um dispositivo de codificação de vídeo 1102 (por exemplo, um codificador de vídeo 300 ou um decodificador de vídeo 400). O dispositivo de codificação de vídeo 1102 inclui meio de recebimento 1101. O meio de recebimento 1101 é configurado para receber uma imagem para codificação ou para receber um fluxo de bits para decodificação. O dispositivo de codificação de vídeo 1102 inclui meio de transmissão 1107 acoplado ao meio de recebimento 1101. O meio de transmissão 1107 é configurado para transmitir o fluxo de bits para um decodificador ou para transmitir uma imagem decodificada para um meio de exibição (por exemplo, um dos dispositivos de I/O 1080).

[0428] O dispositivo de codificação de vídeo 1102 inclui um meio de armazenamento 1103. O meio de armazenamento 1103 é acoplado a pelo menos um do meio de recebimento 1101 ou meio de transmissão 1107. O meio de armazenamento 1103 é configurado para armazenar instruções. O dispositivo de codificação de vídeo 1102 também inclui meio de processamento 1105. O meio de processamento 1105 é acoplado ao meio de armazenamento 1103. O meio de processamento 1105 é configurado para executar as instruções armazenadas no meio de armazenamento 1103 para realizar os métodos divulgados neste documento.

[0429] Também deve ser entendido que as etapas dos métodos exemplares estabelecidas neste documento não são necessariamente necessárias para serem realizadas na ordem descrita, e a ordem das etapas de tais métodos deve ser entendida como meramente exemplificativa. Da mesma forma, etapas adicionais podem ser incluídas em tais métodos e certas etapas podem ser omitidas ou combinadas, em métodos consistentes com várias modalidades da presente revelação.

[0430] Embora várias modalidades tenham sido fornecidas na

presente revelação, deve ser entendido que os sistemas e métodos revelados podem ser incorporados em muitas outras formas específicas sem se afastar da essência ou escopo da presente revelação. Os presentes exemplos devem ser considerados ilustrativos e não restritivos, e a intenção não se limita aos detalhes fornecidos neste documento. Por exemplo, os vários elementos ou componentes podem ser combinados ou integrados em outro sistema ou certos recursos podem ser omitidos ou não implementados.

[0431] Além disso, técnicas, sistemas, subsistemas e métodos descritos e ilustrados nas várias modalidades como discretos ou separados podem ser combinados ou integrados com outros sistemas, módulos, técnicas ou métodos sem se afastar do escopo da presente revelação. Outros itens mostrados ou discutidos como acoplados ou diretamente acoplados ou se comunicando entre si podem ser indiretamente acoplados ou se comunicar através de alguma interface, dispositivo ou componente intermediário, seja eletricamente, mecanicamente ou de outra forma. Outros exemplos de mudanças, substituições e alterações são verificáveis por um indivíduo versado na técnica e podem ser feitos sem se afastar da essência e do escopo revelados neste documento.

REIVINDICAÇÕES

1. Método de codificação implementado por um codificador de vídeo, **CARACTERIZADO** pelo fato de que o método compreende:

gerar, pelo codificador de vídeo, um conjunto de parâmetros de vídeo (VPS) e um conjunto de parâmetros de sequência (SPS), em que o VPS inclui uma lista de estruturas de sintaxe de buffer de imagem decodificada (DPB) para todos os conjuntos de camadas de saída (OLSs), e em que o SPS inclui a estrutura de sintaxe de DPB para cada OLS tendo apenas uma camada;

codificar, pelo codificador de vídeo, o VPS e o SPS em um fluxo de bits de vídeo; e

armazenar, pelo codificador de vídeo, o fluxo de bits de vídeo para comunicação em direção a um decodificador de vídeo.

2. Método, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que o VPS inclui um índice à lista de estruturas de sintaxe de DPB, e em que o índice é designado vps_ols_dpb_params_idx[i].

3. Método, de acordo com qualquer uma das reivindicações 1 a 2, **CARACTERIZADO** pelo fato de que um valor de vps_ols_dpb_params_idx[i] está em uma faixa de zero a um número total das estruturas de sintaxe de DPB no VPS.

4. Método, de acordo com qualquer uma das reivindicações 2 a 3, **CARACTERIZADO** pelo fato de que o VPS inclui vps_num_dpb_params_minus1, e em que vps_num_dpb_params_minus1 mais 1 especifica um número total das estruturas de sintaxe de DPB no VPS.

5. Método, de acordo com qualquer uma das reivindicações 2 a 4, **CARACTERIZADO** pelo fato de que compreende adicionalmente extrair um subfluxo de bits a partir do fluxo de bits, em que o subfluxo de bits inclui o SPS a partir do fluxo de bits mas não inclui o VPS a partir do fluxo de bits.

6. Método, de acordo com qualquer uma das reivindicações 2 a 5, **CARACTERIZADO** pelo fato de que compreende adicionalmente transmitir o subfluxo de bits em direção a um decodificador de vídeo.

7. Dispositivo de codificação, **CARACTERIZADO** pelo fato de que compreende:

uma memória contendo instruções;

um processador acoplado à memória, o processador configurado para

implementar as instruções para fazer o dispositivo de codificação:

gerar um conjunto de parâmetros de vídeo (VPS) e um conjunto de parâmetros de sequência (SPS), em que o VPS inclui uma lista de estruturas de sintaxe de buffer de imagem decodificada (DPB) para todos os conjuntos de camadas de saída (OLSs), e em que o SPS inclui a estrutura de sintaxe de DPB para cada OLS tendo apenas uma camada; e

codificar o VPS e o SPS em um fluxo de bits de vídeo; e

um transmissor acoplado ao processador, o transmissor configurado para transmitir o fluxo de bits de vídeo em direção a um decodificador de vídeo.

8. Dispositivo de codificação, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que o VPS inclui um índice à lista de estruturas de sintaxe de DPB, e em que o índice é designado vps_ols_dpb_params_idx[i].

9. Dispositivo de codificação, de acordo com qualquer uma das reivindicações 7 a 8, **CARACTERIZADO** pelo fato de que um valor de vps_ols_dpb_params_idx [i] está em uma faixa de zero a um número total das estruturas de sintaxe de DPB no VPS.

10. Dispositivo de codificação, de acordo com qualquer uma das reivindicações 7 a 9, **CARACTERIZADO** pelo fato de que o VPS inclui vps_num_dpb_params_minus1, e em que vps_num_dpb_params_minus1 mais 1 especifica o número total das estruturas de sintaxe de DPB no VPS.

11. Aparelho de codificação, **CARACTERIZADO** pelo fato de que compreende:

um receptor configurado para receber uma imagem para codificação ou para receber um fluxo de bits para decodificação;

um transmissor acoplado ao receptor, o transmissor configurado para transmitir o fluxo de bits para um decodificador ou para transmitir uma imagem decodificada para um display;

uma memória acoplada a pelo menos um dentre o receptor ou o transmissor, a memória configurada para armazenar instruções; e

um processador acoplado à memória, o processador configurado para executar as instruções armazenadas na memória para realizar o método conforme definido em qualquer uma das reivindicações 1 a 6.

12. Aparelho de codificação, de acordo com a reivindicação 11, **CARACTERIZADO** pelo fato de que compreende adicionalmente um display

configurado para exibir uma imagem decodificada.

13. Sistema, **CARACTERIZADO** pelo fato de que compreende:

um codificador; e

um decodificador em comunicação com o codificador, em que o codificador ou o decodificador inclui o dispositivo de decodificação, o dispositivo de codificação, ou o aparelho de codificação conforme definidos em qualquer uma das reivindicações 7 a 12.

14. Meio para codificação, **CARACTERIZADO** pelo fato de que compreende:

meio de recebimento configurado para receber uma imagem para codificação ou para receber um fluxo de bits para decodificação;

meio de transmissão acoplado ao meio de recebimento, o meio de transmissão configurado para transmitir o fluxo de bits para um meio de decodificação ou para transmitir uma imagem decodificada para um meio de exibição;

meio de armazenamento acoplado a pelo menos um dentre o meio de recebimento ou o meio de transmissão, o meio de armazenamento configurado para armazenar instruções; e

meio de processamento acoplado ao meio de armazenamento, o meio de processamento configurado para executar as instruções armazenadas no meio de armazenamento para realizar o método conforme definido em qualquer uma das reivindicações 1 a 6.

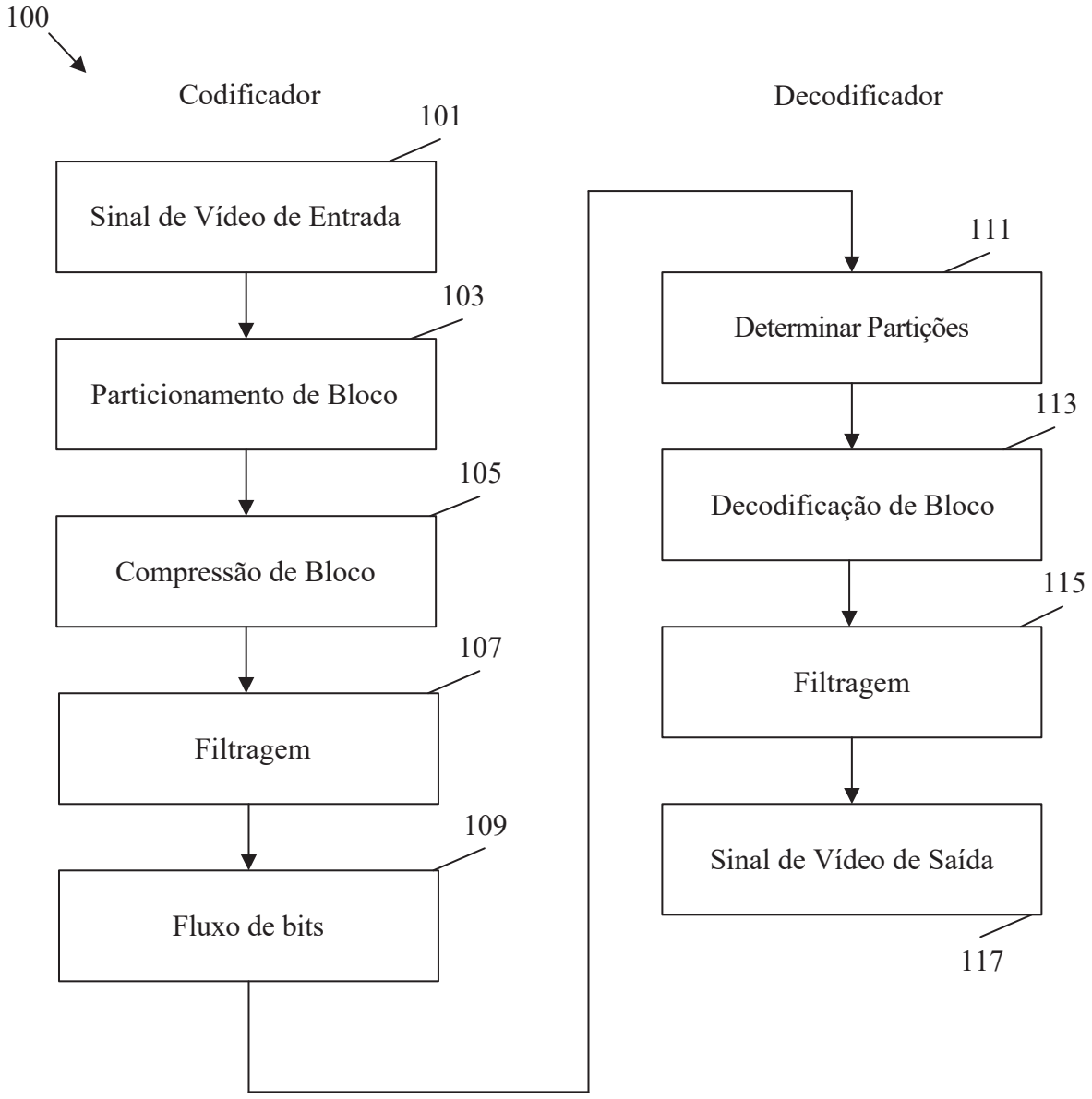

100
Codificador
Decodificador
101
Sinal de Vídeo de Entrada
103
Particionamento de Bloco
105
Compressão de Bloco
107
Filtragem
109
Fluxo de bits
111
Determinar Partições
113
Decodificação de Bloco
115
Filtragem
Sinal de Vídeo de Saída
117

FIG. 1

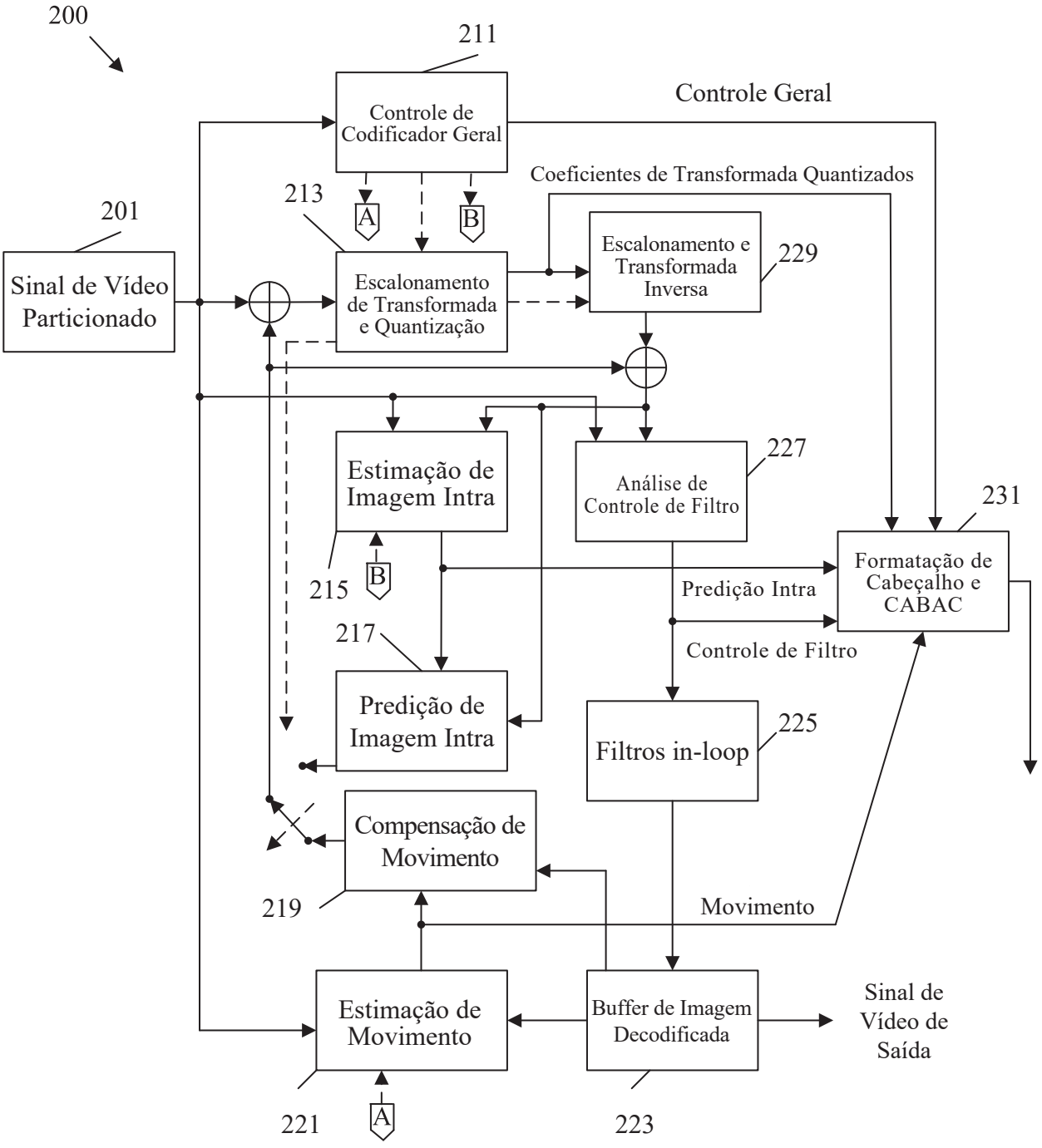

200
211
Controle de Codificador Geral
Controle Geral
Coeficientes de Transformada Quantizados
213
A
B
201
Sinal de Vídeo Particionado
Escalonamento de Transformada e Quantização
Escalonamento e Transformada Inversa
229
227
Estimação de Imagem Intra
Análise de Controle de Filtro
231
215
B
Formatação de Cabeçalho e CABAC
Predição Intra
217
Controle de Filtro
Predição de Imagem Intra
Filtros in-loop
225
Compensação de Movimento
219
Movimento
Estimação de Movimento
Buffer de Imagem Decodificada
Sinal de Vídeo de Saída
221
A
223

FIG. 2

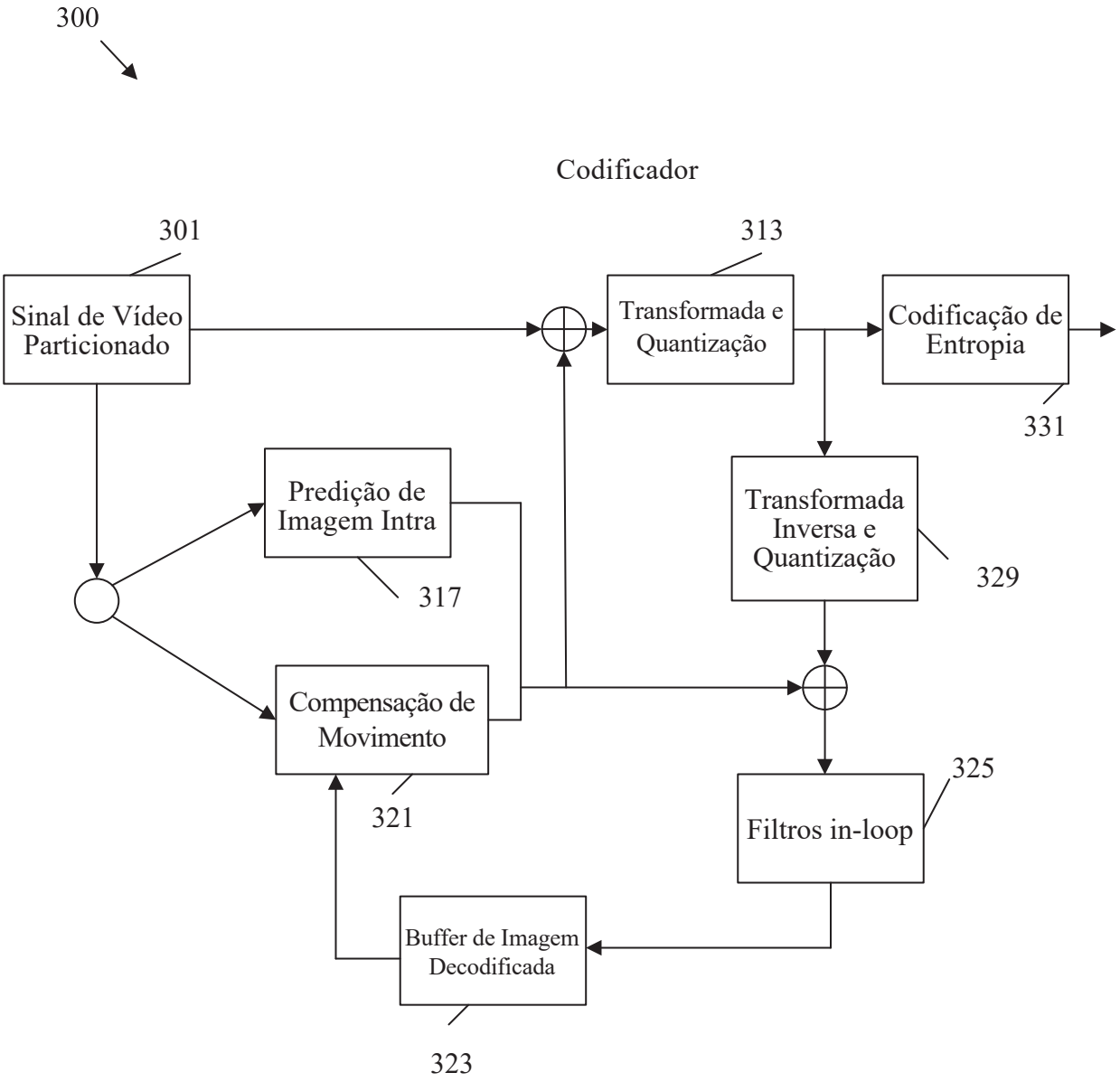

300
Codificador
301
Sinal de Vídeo Particionado
313
Transformada e Quantização
Codificação de Entropia
331
Predição de Imagem Intra
317
Transformada Inversa e Quantização
329
Compensação de Movimento
321
325
Filtros in-loop
Buffer de Imagem Decodificada
323

FIG. 3

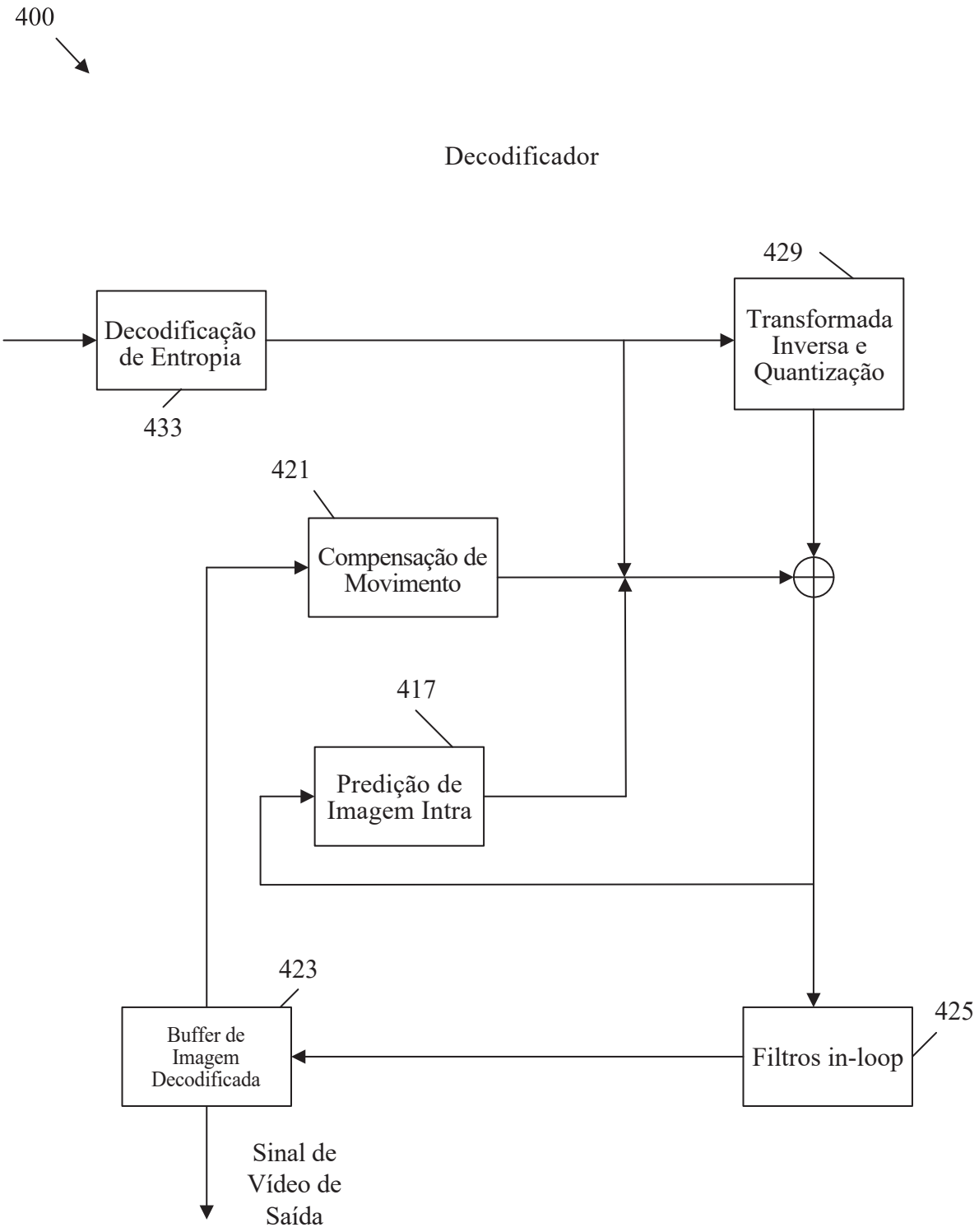

400
Decodificador
429
Decodificação de Entropia
Transformada Inversa e Quantização
433
421
Compensação de Movimento
417
Predição de Imagem Intra
423
Buffer de Imagem Decodificada
425
Filtros in-loop
Sinal de Vídeo de Saída

FIG. 4

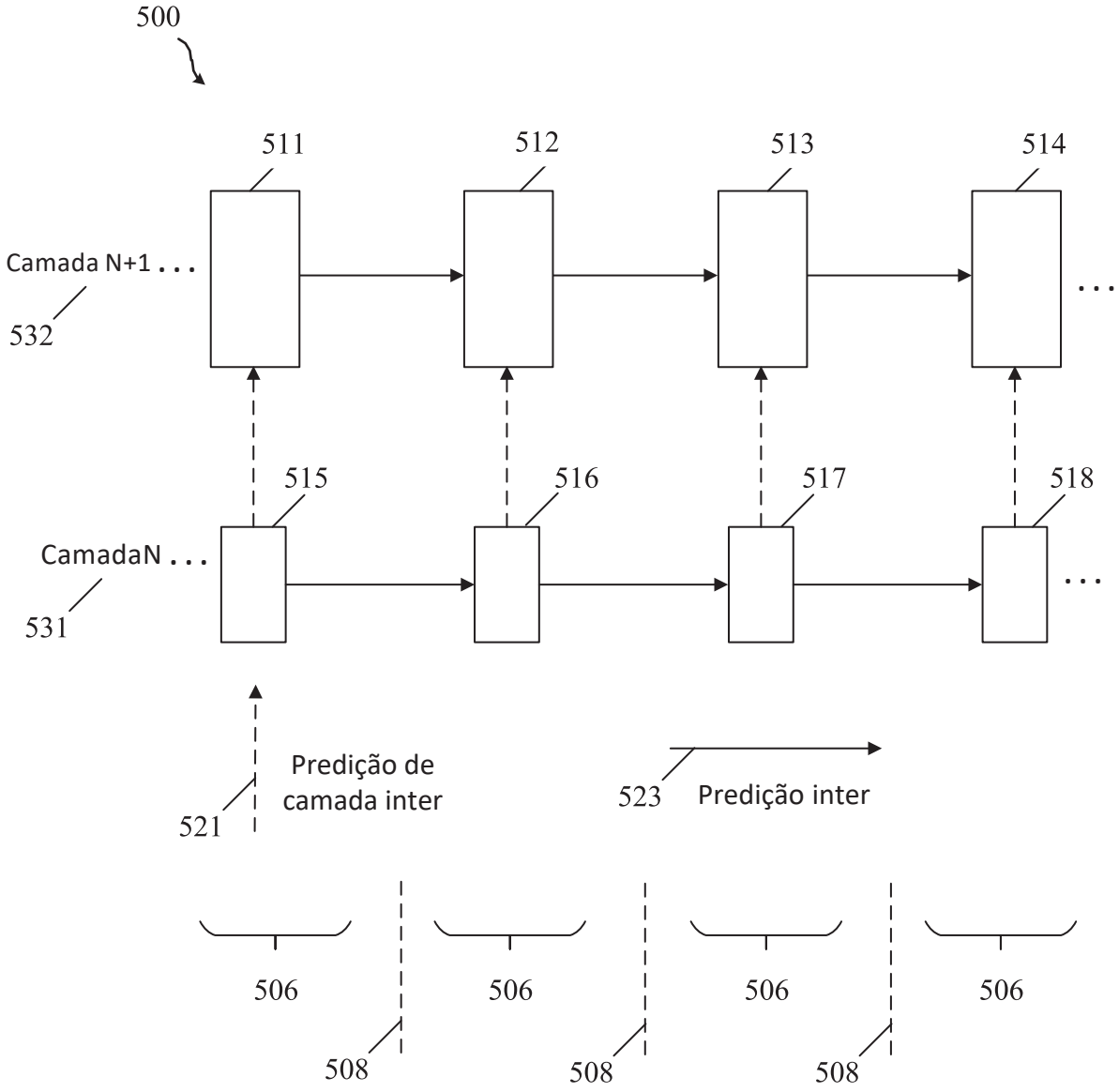

500
511
512
513
514
Camada N+1 . . .
532
. . .
515
516
517
518
CamadaN . . .
531
. . .
Predição de
camada inter
521
523
Predição inter
506
506
506
506
508
508
508

FIG. 5

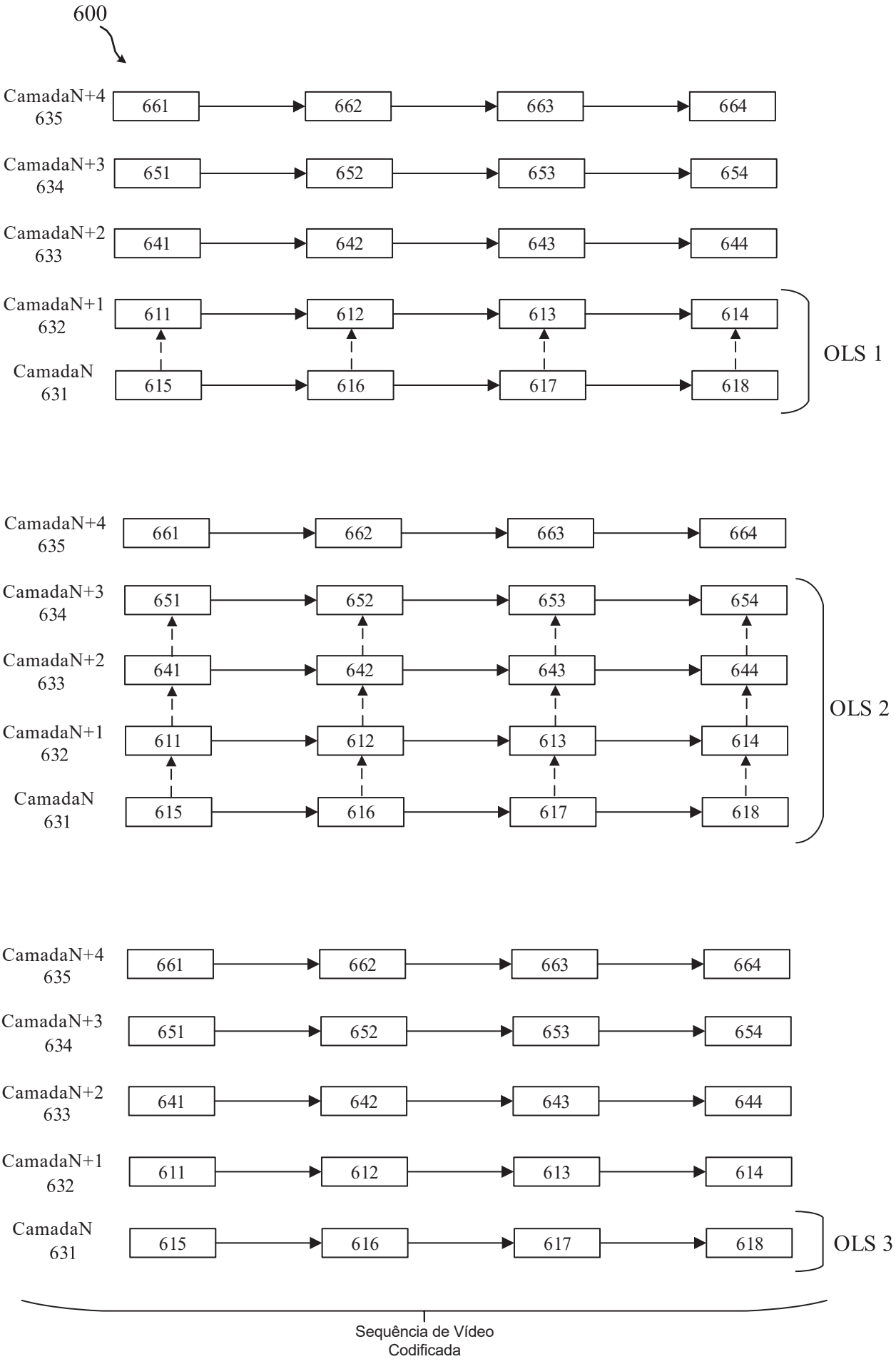

600
CamadaN+4
635
661
662
663
664
CamadaN+3
634
651
652
653
654
CamadaN+2
633
641
642
643
644
CamadaN+1
632
611
612
613
614
CamadaN
631
615
616
617
618
OLS 1
CamadaN+4
635
661
662
663
664
CamadaN+3
634
651
652
653
654
CamadaN+2
633
641
642
643
644
CamadaN+1
632
611
612
613
614
CamadaN
631
615
616
617
618
OLS 2
CamadaN+4
635
661
662
663
664
CamadaN+3
634
651
652
653
654
CamadaN+2
633
641
642
643
644
CamadaN+1
632
611
612
613
614
CamadaN
631
615
616
617
618
OLS 3
Sequência de Vídeo
Codificada
690

FIG. 6

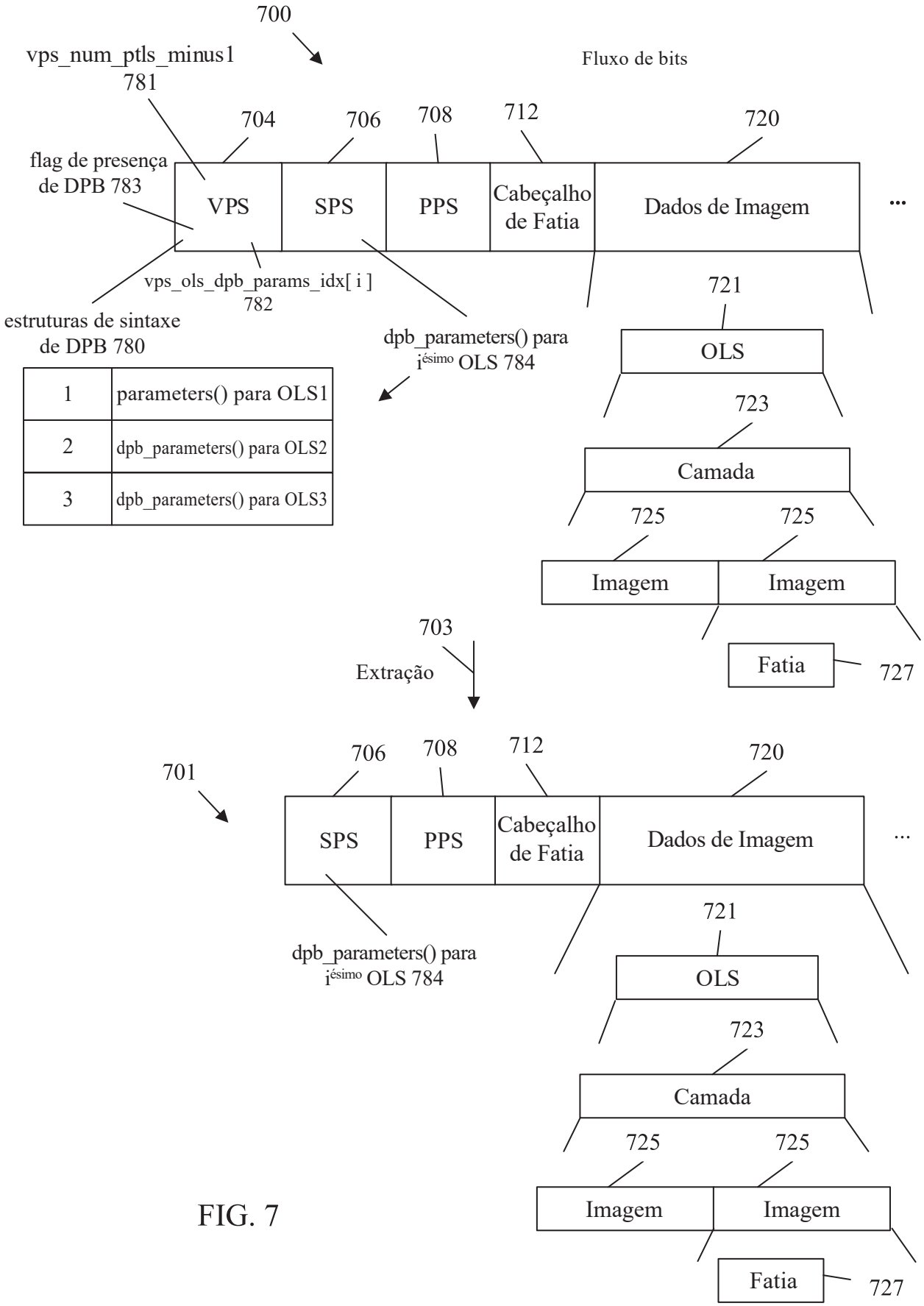

700
vps_num_ptls_minus1
781
Fluxo de bits
704
706
708
712
720
flag de presença
de DPB 783
VPS
SPS
PPS
Cabeçalho
de Fatia
Dados de Imagem
...
vps_ols_dpb_params_idx[i]
782
721
estruturas de sintaxe
de DPB 780
dpb_parameters() para
iésimo OLS 784
OLS
723
Camada
725
725
Imagem
Imagem
1
parameters() para OLS1
2
dpb_parameters() para OLS2
3
dpb_parameters() para OLS3
703
Extração
Fatia
727
701
706
708
712
720
SPS
PPS
Cabeçalho
de Fatia
Dados de Imagem
...
721
dpb_parameters() para
iésimo OLS 784
OLS
723
Camada
725
725
Imagem
Imagem
Fatia
727

FIG. 7

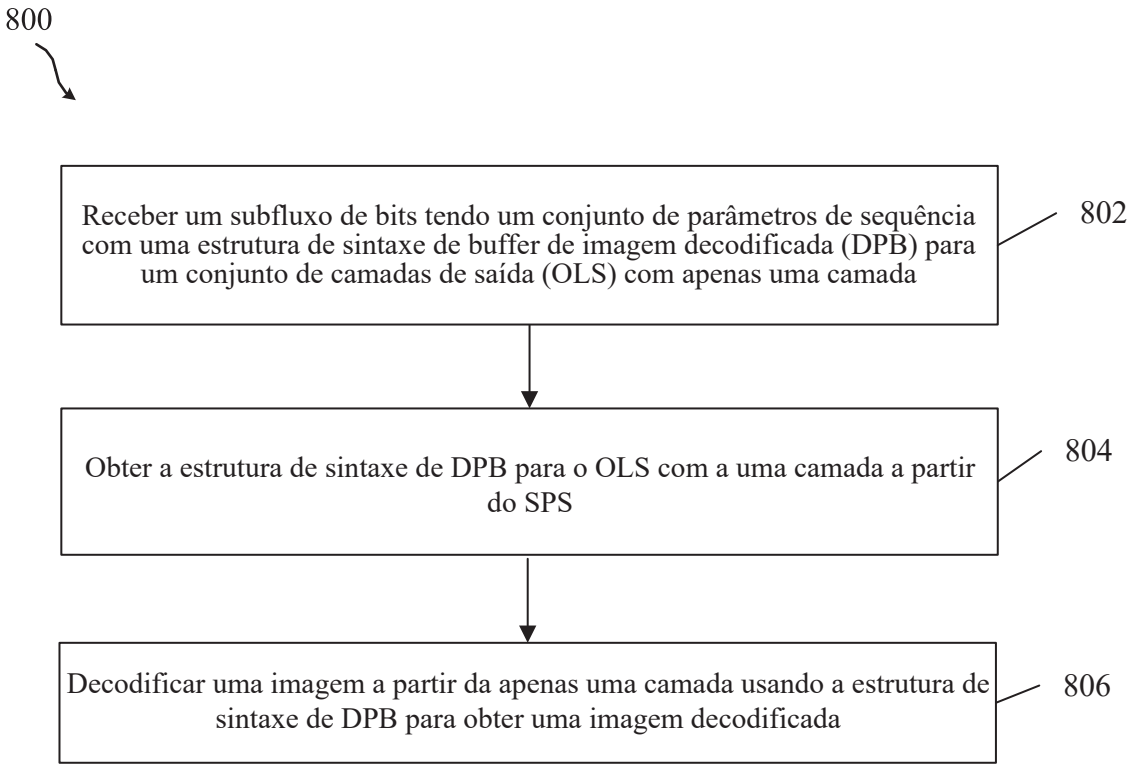

800
Receber um subfluxo de bits tendo um conjunto de parâmetros de sequência com uma estrutura de sintaxe de buffer de imagem decodificada (DPB) para um conjunto de camadas de saída (OLS) com apenas uma camada
802
Obter a estrutura de sintaxe de DPB para o OLS com a uma camada a partir do SPS
804
Decodificar uma imagem a partir da apenas uma camada usando a estrutura de sintaxe de DPB para obter uma imagem decodificada
806

FIG. 8

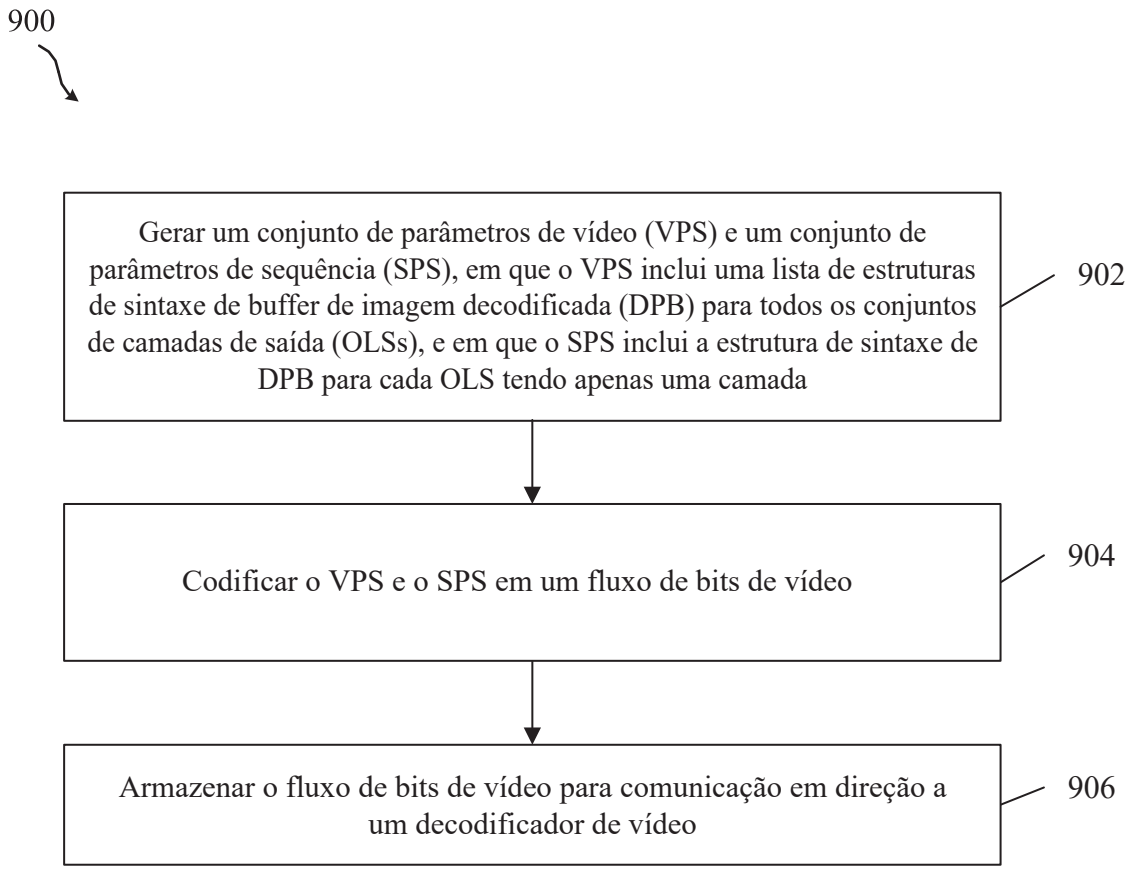

900
Gerar um conjunto de parâmetros de vídeo (VPS) e um conjunto de parâmetros de sequência (SPS), em que o VPS inclui uma lista de estruturas de sintaxe de buffer de imagem decodificada (DPB) para todos os conjuntos de camadas de saída (OLSs), e em que o SPS inclui a estrutura de sintaxe de DPB para cada OLS tendo apenas uma camada
902
Codificar o VPS e o SPS em um fluxo de bits de vídeo
904
Armazenar o fluxo de bits de vídeo para comunicação em direção a um decodificador de vídeo
906

FIG. 9

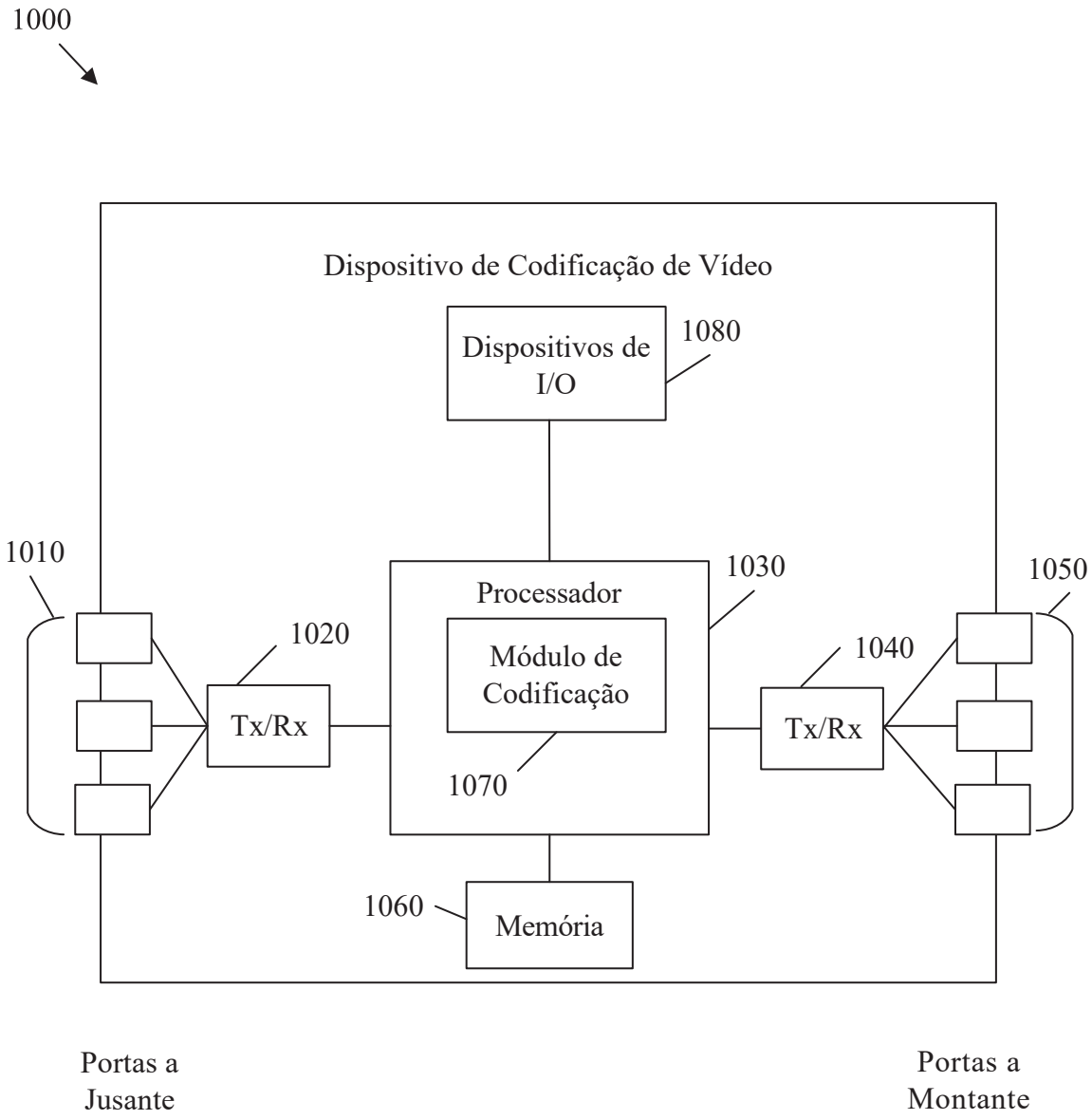

1000
Dispositivo de Codificação de Vídeo
Dispositivos de I/O
1080
1010
1020
Tx/Rx
Processador
Módulo de Codificação
1070
1030
1040
Tx/Rx
1050
1060
Memória
Portas a Jusante
Portas a Montante

FIG. 10

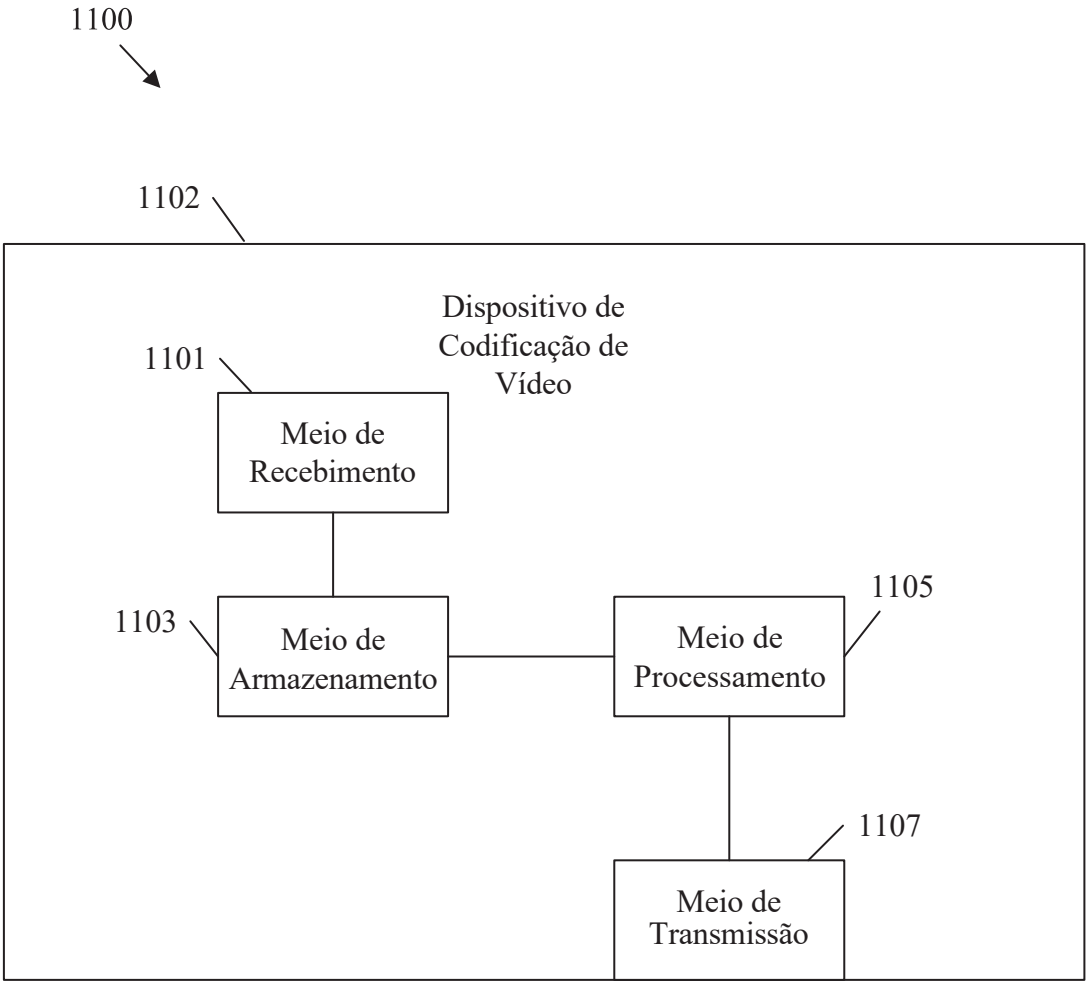

1100
1102
Dispositivo de Codificação de Vídeo
1101
Meio de Recebimento
1103
Meio de Armazenamento
1105
Meio de Processamento
1107
Meio de Transmissão

FIG. 11

RESUMO

“PREVENÇÃO DE SINALIZAÇÃO REDUNDANTE EM FLUXOS DE BITS DE VÍDEO DE MÚLTIPLAS CAMADAS”

Um método de decodificação é fornecido. O método inclui receber um subfluxo de bits tendo um conjunto de parâmetros de sequência (SPS) com uma estrutura de sintaxe de buffer de imagem decodificada (DPB) para um conjunto de camadas de saída (OLS) com apenas uma camada; definir um número de camadas no OLS como igual a um com base no OLS tendo a apenas uma camada; obter a estrutura de sintaxe de DPB para o OLS com a apenas uma camada a partir do SPS quando o número de camadas no OLS tiver sido definido como igual a um; e decodificar uma imagem a partir da apenas uma camada usando a estrutura de sintaxe de DPB para obter uma imagem decodificada. Um método correspondente de codificação também é fornecido.