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

(21) BR 122022009696-4 A2

(22) Data do Depósito: 06/10/2020

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

(54) Título: PREVENÇÃO DE ERRO EM EXTRAÇÃO DE SUBFLUXO DE BITS

(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.

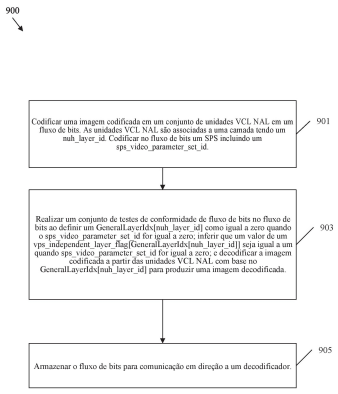
(86) Pedido PCT: PCT US2020054452 de 06/10/2020

(87) Publicação PCT: WO 2021/022271 de 04/02/2021

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

(62) Pedido original do dividido: BR112022006421-7 - 06/10/2020

(57) Resumo: PREVENÇÃO DE ERRO EM EXTRAÇÃO DE SUBFLUXO DE BITS. Um mecanismo de codificação de vídeo é revelado. O mecanismo inclui receber um fluxo de bits que compreende um conjunto de parâmetros de sequência (SPS) que inclui um identificador de conjunto de parâmetros de vídeo de SPS (sps_video_parameter_set_id). O fluxo de bits também compreende uma imagem codificada em um conjunto de unidades de camada de abstração de rede (NAL) de camada de codificação de vídeo (VCL) associadas com uma camada que tem um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id). Um índice de camada geral correspondente ao nuh_layer_id (GeneralLayerIdx[nuh_layer_id]) é definido igual a zero quando o sps_video_parameter_set_id for igual a zero. A imagem codificada é decodificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada. A imagem decodificada é encaminhada para exibição como parte de uma sequência de vídeo decodificada.



“PREVENÇÃO DE ERRO EM EXTRAÇÃO DE SUBFLUXO DE BITS”

Dividido do BR 112022006421-7, de 06/10/2020.

REFERÊNCIA CRUZADA A PEDIDOS RELACIONADOS

[0001] Este pedido de patente reivindica o benefício ao Pedido de Patente Provisório U.S. nº 62/911,808, depositado em 7 de outubro de 2019 por Ye-Kui Wang e intitulado “Scalability In Video Coding”, o qual é incorporado ao presente documento a título de referência.

CAMPO DA TÉCNICA

[0002] A presente revelação se refere, em geral, à codificação de vídeo, e se refere, de maneira específica, a mecanismos para prevenir erros quando a extração de subfluxo de bits for realizada em um fluxo de bits de múltiplas camadas.

FUNDAMENTOS

[0003] A quantidade de dados de vídeo necessária para representar até mesmo um vídeo relativamente curto pode ser substancial, que pode resultar em dificuldades quando os dados precisam ser transmitidos por streaming ou comunicados de outro modo por meio de uma rede de comunicações com capacidade limitada de largura de banda. Desse modo, os dados de vídeo são geralmente comprimidos antes de serem comunicados por meio de redes de telecomunicações atuais. O tamanho de um vídeo também pode ser uma questão quando o vídeo é armazenado em um dispositivo de armazenamento, visto que os recursos de memória podem ser limitados. Os dispositivos de compressão de vídeo normalmente utilizam software e/ou hardware na origem para codificar os dados de vídeo antes de transmissão ou armazenamento, diminuindo, desse modo, a quantidade de dados necessária para representar imagens de vídeo digitais. 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 crescentes por maior qualidade de vídeo, as técnicas de compressão e descompressão aperfeiçoadas que aperfeiçoam a razão de compressão com pouco ou nenhum comprometimento na qualidade de imagem são desejáveis.

SUMÁRIO

[0004] Em uma modalidade, a revelação inclui um método implementado por um decodificador, o método compreendendo: receber, pelo

decodificador, um fluxo de bits que compreende um conjunto de parâmetros de sequência (SPS) e uma imagem codificada, em que o SPS inclui um identificador de conjunto de parâmetros de vídeo de SPS (sps_video_parameter_set_id), e em que a imagem codificada está em um conjunto de unidades de camada de abstração de rede (NAL) de camada de codificação de vídeo (VCL) associadas a uma camada tendo um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id); definir, pelo decodificador, um índice de camada geral que corresponde ao nuh_layer_id (GeneralLayerIdx[nuh_layer_id]) igual a zero quando o sps_video_parameter_set_id for igual a zero; e decodificar, pelo decodificador, a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada.

[0005] Alguns sistemas de codificação de vídeo codificam sequências de vídeo em camadas de imagens. Imagens em camadas diferentes têm características diferentes. Então, um codificador pode transmitir camadas diferentes a um decodificador dependendo de restrições de lado de decodificador. A fim de realizar essa função, um codificador pode codificar todas as camadas em um único fluxo de bits. Mediante solicitação, o codificador pode realizar um processo de extração de subfluxo de bits para remover informações estranhas do fluxo de bits. Esse resultado é um fluxo de bits extraído que contém apenas os dados na(s) camada(s) solicitada(s) pelo decodificador. Uma descrição de como as camadas são relacionadas pode ser incluída em um conjunto de parâmetros de vídeo (VPS). Uma camada simulcast é uma camada que é configurada para exibição sem referência a outras camadas. Quando uma camada simulcast é transmitida a um decodificador, o processo de extração de subfluxo de bits pode remover o VPS, visto que as relações de camada não são necessárias para decodificar uma camada simulcast. Infelizmente, certas variáveis em outros conjuntos de parâmetros podem referenciar o VPS. Como tal, a remoção do VPS quando camadas simulcast forem transmitidas pode aumentar a eficiência de codificação, mas também pode resultar em erros. O presente exemplo inclui um mecanismo para codificar um SPS de um modo que evite erros quando um VPS é removido de um fluxo de bits codificado como parte de um processo de extração de subfluxo de bits. O SPS contém um sps_video_parameter_set_id. O sps_video_parameter_set_id indica um identificador do VPS que contém relações de camada para a sequência de vídeo. Em um exemplo, o

sps_video_parameter_set_id é definido como zero quando o VPS for removido antes da transmissão de um fluxo de bits contendo apenas uma camada simulcast. Em outro exemplo, SPSs usados por camadas simulcast podem conter um sps_video_parameter_set_id que é definido como zero no momento de codificação. Em qualquer um dos casos, quando o sps_video_parameter_set_id for definido como zero, variáveis relacionadas a SPS que referenciam o VPS são definidas para valores padrão para evitar erros. Por exemplo, um GeneralLayerIdx[nuh_layer_id] indica um índice de camada atual para uma camada correspondente (por exemplo, a camada simulcast). O GeneralLayerIdx[nuh_layer_id] é definido como/inferido como sendo zero quando o sps_video_parameter_set_id for zero. Conforme outro exemplo, uma flag de camada independente de VPS para o GeneralLayerIdx[nuh_layer_id] (vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]) é armazenada no VPS e especifica se uma camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. A predição de camada inter não é usada para camadas simulcast. Portanto, a vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é definida como/inferida como sendo um para indicar nenhuma predição de camada inter quando o sps_video_parameter_set_id for definido como zero. Desse modo, erros são evitados quando um VPS é removido de um fluxo de bits antes da transmissão de uma camada simulcast. Como resultado, a funcionalidade do codificador e do decodificador é aumentada. Além disso, a eficiência de codificação é aumentada removendo-se, de maneira bem-sucedida, um VPS desnecessário de um fluxo de bits incluindo apenas uma camada simulcast, que reduz uso de recurso de processador, memória e/ou sinalização de rede tanto no codificador quanto no decodificador.

[0006] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que GeneralLayerIdx[nuh_layer_id] é igual a um índice de camada atual.

[0007] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, que compreende, adicionalmente, inferir, pelo decodificador, que um valor de uma flag de camada independente de VPS para o GeneralLayerIdx[nuh_layer_id] (vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]) seja igual a um

quando sps_video_parameter_set_id for igual a zero.

[0008] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que a vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se a camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter.

[0009] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o sps_video_parameter_set_id especifica um valor de um identificador de conjunto de parâmetros de VPS (vps_video_parameter_set_id) quando maior que zero.

[0010] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o SPS não se refere a um VPS e nenhum VPS é referido ao decodificar cada sequência de vídeo de camada codificada que se refere ao SPS quando o sps_video_parameter_set_id for igual a zero.

[0011] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o conjunto de unidades VCL NAL é parte da camada quando o conjunto de unidades VCL NAL tiverem, todas, um valor particular de nuh_layer_id.

[0012] Em uma modalidade, a revelação inclui um método implementado por um codificador, o método compreendendo: codificar, pelo codificador, uma imagem codificada e um SPS em um fluxo de bits, em que a imagem codificada é codificada em um conjunto de unidades VCL NAL, em que as unidades VCL NAL são associadas a uma camada tendo um nuh_layer_id, e em que o SPS inclui um sps_video_parameter_set_id; realizar, pelo um decodificador de referência hipotético (HRD) no codificador, um conjunto de testes de conformidade de fluxo de bits no fluxo de bits ao: definir, por um decodificador de referência hipotético operando no codificador, um GeneralLayerIdx[nuh_layer_id] igual a zero quando o sps_video_parameter_set_id for igual a zero; e decodificar, pelo HRD operando no codificador, a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada; e armazenar, no codificador, o fluxo de bits para comunicação em direção a um decodificador.

[0013] Alguns sistemas de codificação de vídeo codificam sequências de vídeo em camadas de imagens. Imagens em camadas diferentes têm características diferentes. Então, um codificador pode transmitir camadas diferentes a um decodificador dependendo de restrições de lado de decodificador. A fim de realizar essa função, um codificador pode codificar todas as camadas em um único fluxo de bits. Mediante solicitação, o codificador pode realizar um processo de extração de subfluxo de bits para remover informações estranhas do fluxo de bits. Esse resultado é um fluxo de bits extraído que contém apenas os dados na(s) camada(s) solicitada(s) pelo decodificador. Uma descrição de como as camadas são relacionadas pode ser incluída em um conjunto de parâmetros de vídeo (VPS). Uma camada simulcast é uma camada que é configurada para exibição sem referência a outras camadas. Quando uma camada simulcast é transmitida a um decodificador, o processo de extração de subfluxo de bits pode remover o VPS, visto que as relações de camada não são necessárias para decodificar uma camada simulcast. Infelizmente, certas variáveis em outros conjuntos de parâmetros podem referenciar o VPS. Como tal, a remoção do VPS quando camadas simulcast forem transmitidas pode aumentar a eficiência de codificação, mas também pode resultar em erros. O presente exemplo inclui um mecanismo para codificar um SPS de um modo que evite erros quando um VPS é removido de um fluxo de bits codificado como parte de um processo de extração de subfluxo de bits. O SPS contém um sps_video_parameter_set_id. O sps_video_parameter_set_id indica um identificador do VPS que contém relações de camada para a sequência de vídeo. Em um exemplo, o sps_video_parameter_set_id é definido como zero quando o VPS for removido antes da transmissão de um fluxo de bits contendo apenas uma camada simulcast. Em outro exemplo, SPSs usados por camadas simulcast podem conter um sps_video_parameter_set_id que é definido como zero no momento de codificação. Em qualquer um dos casos, quando o sps_video_parameter_set_id for definido como zero, variáveis relacionadas a SPS que referenciam o VPS são definidas para valores padrão para evitar erros. Por exemplo, um GeneralLayerIdx[nuh_layer_id] indica um índice de camada atual para uma camada correspondente (por exemplo, a camada simulcast). O GeneralLayerIdx[nuh_layer_id] é definido como/inferido como sendo zero quando o sps_video_parameter_set_id for zero. Conforme outro exemplo, uma

vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é armazenada no VPS e especifica se uma camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. A predição de camada inter não é usada para camadas simulcast. Portanto, a vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é definida como/inferida como sendo um para indicar nenhuma predição de camada inter quando o sps_video_parameter_set_id for definido como zero. Desse modo, erros são evitados quando um VPS é removido de um fluxo de bits antes da transmissão de uma camada simulcast. Como resultado, a funcionalidade do codificador e do decodificador é aumentada. Além disso, a eficiência de codificação é aumentada removendo-se, de maneira bem-sucedida, um VPS desnecessário de um fluxo de bits incluindo apenas uma camada simulcast, que reduz uso de recurso de processador, memória e/ou sinalização de rede tanto no codificador quanto no decodificador.

[0014] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que GeneralLayerIdx[nuh_layer_id] é igual a um índice de camada atual.

[0015] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, que compreende, adicionalmente, inferir, pelo HRD operando no codificador, que um valor de uma vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] seja igual a um quando sps_video_parameter_set_id for igual a zero.

[0016] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se a camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter.

[0017] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o sps_video_parameter_set_id especifica um valor de um vps_video_parameter_set_id quando maior que zero.

[0018] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o SPS não se refere a um VPS e nenhum VPS é referido ao decodificar cada sequência de vídeo

de camada codificada que se refere ao SPS quando o sps_video_parameter_set_id for igual a zero.

[0019] De maneira opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o conjunto de unidades VCL NAL é parte da camada quando o conjunto de unidades VCL NAL tiverem, todas, um valor particular de nuh_layer_id.

[0020] Em uma modalidade, a revelação inclui um dispositivo de codificação de vídeo compreendendo: um processador, um receptor acoplado ao processador, uma memória acoplada ao processador, e um transmissor acoplado ao processador, em que o processador, receptor, memória e transmissor são configurados para realizar o método de qualquer um dos aspectos antecedentes.

[0021] Em uma modalidade, a revelação inclui uma mídia legível por computador não transitória que compreende um produto de programa de computador para uso por um dispositivo de codificação de vídeo, o produto de programa de computador compreende instruções executáveis em computador armazenadas na mídia legível por computador não transitória de modo que, quando executadas por um processador, fazem com que o dispositivo de codificação de vídeo realize o método de qualquer um dos aspectos antecedentes.

[0022] Em uma modalidade, a revelação inclui um decodificador compreendendo: um meio de recebimento para receber um fluxo de bits que compreende um SPS e uma imagem codificada, em que o SPS inclui um sps_video_parameter_set_id, e em que a imagem codificada está em um conjunto de unidades VCL NAL associadas a uma camada tendo um nuh_layer_id; um meio de definição para definir um GeneralLayerIdx[nuh_layer_id] igual a zero quando o sps_video_parameter_set_id for igual a zero; um meio de decodificação para decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada; e um meio de encaminhamento para encaminhar a imagem decodificada para exibição como parte de uma sequência de vídeo decodificada.

[0023] Alguns sistemas de codificação de vídeo codificam sequências de vídeo em camadas de imagens. Imagens em camadas diferentes têm características diferentes. Então, um codificador pode transmitir camadas diferentes a um decodificador dependendo de restrições de lado de decodificador. A fim de realizar essa função, um codificador pode codificar todas as camadas em

um único fluxo de bits. Mediante solicitação, o codificador pode realizar um processo de extração de subfluxo de bits para remover informações estranhas do fluxo de bits. Esse resultado é um fluxo de bits extraído que contém apenas os dados na(s) camada(s) solicitada(s) pelo decodificador. Uma descrição de como as camadas são relacionadas pode ser incluída em um conjunto de parâmetros de vídeo (VPS). Uma camada simulcast é uma camada que é configurada para exibição sem referência a outras camadas. Quando uma camada simulcast é transmitida a um decodificador, o processo de extração de subfluxo de bits pode remover o VPS, visto que as relações de camada não são necessárias para decodificar uma camada simulcast. Infelizmente, certas variáveis em outros conjuntos de parâmetros podem referenciar o VPS. Como tal, a remoção do VPS quando camadas simulcast forem transmitidas pode aumentar a eficiência de codificação, mas também pode resultar em erros. O presente exemplo inclui um mecanismo para codificar um SPS de um modo que evite erros quando um VPS é removido de um fluxo de bits codificado como parte de um processo de extração de subfluxo de bits. O SPS contém um sps_video_parameter_set_id. O sps_video_parameter_set_id indica um identificador do VPS que contém relações de camada para a sequência de vídeo. Em um exemplo, o sps_video_parameter_set_id é definido como zero quando o VPS for removido antes da transmissão de um fluxo de bits contendo apenas uma camada simulcast. Em outro exemplo, SPSs usados por camadas simulcast podem conter um sps_video_parameter_set_id que é definido como zero no momento de codificação. Em qualquer um dos casos, quando o sps_video_parameter_set_id for definido como zero, variáveis relacionadas a SPS que referenciam o VPS são definidas para valores padrão para evitar erros. Por exemplo, um GeneralLayerIdx[nuh_layer_id] indica um índice de camada atual para uma camada correspondente (por exemplo, a camada simulcast). O GeneralLayerIdx[nuh_layer_id] é definido como/inferido como sendo zero quando o sps_video_parameter_set_id for zero. Conforme outro exemplo, uma vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é armazenada no VPS e especifica se uma camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. A predição de camada inter não é usada para camadas simulcast. Portanto, a vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é definida

como/inferida como sendo um para indicar nenhuma predição de camada inter quando o sps_video_parameter_set_id for definido como zero. Desse modo, erros são evitados quando um VPS é removido de um fluxo de bits antes da transmissão de uma camada simulcast. Como resultado, a funcionalidade do codificador e do decodificador é aumentada. Além disso, a eficiência de codificação é aumentada removendo-se, de maneira bem-sucedida, um VPS desnecessário de um fluxo de bits incluindo apenas uma camada simulcast, que reduz uso de recurso de processador, memória e/ou sinalização de rede tanto no codificador quanto no decodificador.

[0024] De modo opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o decodificador é configurado, adicionalmente, para realizar o método de qualquer um dos aspectos antecedentes.

[0025] Em uma modalidade, a revelação inclui um codificador compreendendo: um meio de codificação para codificar uma imagem codificada e um SPS em um fluxo de bits, em que a imagem codificada é codificada em um conjunto de unidades VCL NAL, em que as unidades VCL NAL são associadas a uma camada tendo um nuh_layer_id, e em que o SPS inclui um sps_video_parameter_set_id; um meio de HRD para realizar um conjunto de testes de conformidade de fluxo de bits no fluxo de bits ao: definir um GeneralLayerIdx[nuh_layer_id] igual a zero quando o sps_video_parameter_set_id for igual a zero; e decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada; e um meio de armazenamento para armazenar o fluxo de bits para comunicação em direção a um decodificador.

[0026] Alguns sistemas de codificação de vídeo codificam sequências de vídeo em camadas de imagens. Imagens em camadas diferentes têm características diferentes. Então, um codificador pode transmitir camadas diferentes a um decodificador dependendo de restrições de lado de decodificador. A fim de realizar essa função, um codificador pode codificar todas as camadas em um único fluxo de bits. Mediante solicitação, o codificador pode realizar um processo de extração de subfluxo de bits para remover informações estranhas do fluxo de bits. Esse resultado é um fluxo de bits extraído que contém apenas os dados na(s) camada(s) solicitada(s) pelo decodificador. Uma descrição de como

as camadas são relacionadas pode ser incluída em um conjunto de parâmetros de vídeo (VPS). Uma camada simulcast é uma camada que é configurada para exibição sem referência a outras camadas. Quando uma camada simulcast é transmitida a um decodificador, o processo de extração de subfluxo de bits pode remover o VPS, visto que as relações de camada não são necessárias para decodificar uma camada simulcast. Infelizmente, certas variáveis em outros conjuntos de parâmetros podem referenciar o VPS. Como tal, a remoção do VPS quando camadas simulcast forem transmitidas pode aumentar a eficiência de codificação, mas também pode resultar em erros. O presente exemplo inclui um mecanismo para codificar um SPS de um modo que evite erros quando um VPS é removido de um fluxo de bits codificado como parte de um processo de extração de subfluxo de bits. O SPS contém um sps_video_parameter_set_id. O sps_video_parameter_set_id indica um identificador do VPS que contém relações de camada para a sequência de vídeo. Em um exemplo, o sps_video_parameter_set_id é definido como zero quando o VPS for removido antes da transmissão de um fluxo de bits contendo apenas uma camada simulcast. Em outro exemplo, SPSs usados por camadas simulcast podem conter um sps_video_parameter_set_id que é definido como zero no momento de codificação. Em qualquer um dos casos, quando o sps_video_parameter_set_id for definido como zero, variáveis relacionadas a SPS que referenciam o VPS são definidas para valores padrão para evitar erros. Por exemplo, um GeneralLayerIdx[nuh_layer_id] indica um índice de camada atual para uma camada correspondente (por exemplo, a camada simulcast). O GeneralLayerIdx[nuh_layer_id] é definido como/inferido como sendo zero quando o sps_video_parameter_set_id for zero. Conforme outro exemplo, uma vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é armazenada no VPS e especifica se uma camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. A predição de camada inter não é usada para camadas simulcast. Portanto, a vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é definida como/inferida como sendo um para indicar nenhuma predição de camada inter quando o sps_video_parameter_set_id for definido como zero. Desse modo, erros são evitados quando um VPS é removido de um fluxo de bits antes da transmissão de uma camada simulcast. Como resultado, a funcionalidade do codificador e do

decodificador é aumentada. Além disso, a eficiência de codificação é aumentada removendo-se, de maneira bem-sucedida, um VPS desnecessário de um fluxo de bits incluindo apenas uma camada simulcast, que reduz uso de recurso de processador, memória e/ou sinalização de rede tanto no codificador quanto no decodificador.

[0027] De modo opcional, em qualquer um dos aspectos antecedentes, outra implementação do aspecto fornece, em que o codificador é configurado, adicionalmente, para realizar o método de qualquer um dos aspectos antecedentes.

[0028] Para fins de clareza, qualquer uma dentre as modalidades antecedentes pode ser combinada com qualquer uma ou mais dentre as outras modalidades antecedentes para criar uma nova modalidade dentro do escopo da presente revelação.

[0029] Esses e outros atributos serão mais claramente entendidos a partir da descrição detalhada a seguir obtida em conjunto com os desenhos anexos e reivindicações.

BREVE DESCRIÇÃO DOS DESENHOS

[0030] Para um entendimento mais completo desta revelação, referência é feita à breve descrição a seguir, obtida em conjunto com os desenhos anexos e descrição detalhada, em que referências numéricas similares representam partes similares.

[0031] A Figura 1 é um fluxograma de um método exemplificativo de codificação de um sinal de vídeo.

[0032] A Figura 2 é um diagrama esquemático de um sistema de codificação e decodificação (codec) exemplificativo para codificação de vídeo.

[0033] A Figura 3 é um diagrama esquemático ilustrando um codificador de vídeo exemplificativo.

[0034] A Figura 4 é um diagrama esquemático que ilustra um decodificador de vídeo exemplificativo.

[0035] A Figura 5 é um diagrama esquemático ilustrando um decodificador de referência hipotético (HRD) exemplificativo.

[0036] A Figura 6 é um diagrama esquemático ilustrando uma sequência de vídeo de múltiplas camadas exemplificativa configurada para predição de camada inter.

[0037] A Figura 7 é um diagrama esquemático ilustrando um fluxo de bits exemplificativo.

[0038] A Figura 8 é um diagrama esquemático de um dispositivo de codificação de vídeo exemplificativo.

[0039] A Figura 9 é um fluxograma de um método exemplificativo de codificação de uma sequência de vídeo de múltiplas camadas em um fluxo de bits para suportar remoção de conjunto de parâmetros de vídeo (VPS) durante extração de subfluxo de bits para camadas simulcast.

[0040] A Figura 10 é um fluxograma de um método exemplificativo de decodificação de uma sequência de vídeo a partir de um fluxo de bits que inclui uma camada simulcast extraída de um fluxo de bits de múltiplas camadas em que um VPS foi removido durante extração de subfluxo de bits.

[0041] A Figura 11 é um diagrama esquemático de um sistema exemplificativo para codificar uma sequência de vídeo de múltiplas camadas em um fluxo de bits para suportar remoção de VPS durante extração de subfluxo de bits para camadas simulcast.

DESCRIÇÃO DETALHADA

[0042] Deve-se entender inicialmente que, embora uma implementação ilustrativa de uma ou mais modalidades seja fornecida abaixo, os sistemas e/ou métodos revelados podem ser implementados com o uso de qualquer número de técnicas, sejam atualmente conhecidas ou em existência. A revelação não deve, de modo algum, ser limitada às implementações ilustrativas, desenhos e técnicas ilustrados abaixo, incluindo os projetos e implementações exemplificativos ilustrados e descritos no presente documento, mas pode ser modificada dentro do escopo das reivindicações anexas em conjunto com seu escopo inteiro de equivalentes.

[0043] Os termos abaixo são definidos da maneira a seguir, a menos que utilizado em um contexto contrário na presente invenção. Especificamente, as definições a seguir se destinam a fornecer clareza adicional à presente revelação. Entretanto, termos podem ser descritos de maneira diferente em diferentes contextos. Assim, as definições a seguir devem ser consideradas como um complemento e não devem ser consideradas limitar quaisquer outras definições de descrições fornecidas para tais termos na presente invenção.

[0044] 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 das mesmas. Uma imagem que está sendo codificada ou decodificada pode ser denominada uma imagem atual para fins de clareza de discussão. Uma imagem codificada é uma representação codificada de uma imagem que compreende unidades de camada de abstração de rede (NAL) de camada de codificação de vídeo (VCL) com um valor particular de 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 aplicando-se um processo de decodificação a uma imagem codificada. Uma unidade NAL é uma estrutura de sintaxe contendo dados na forma de uma Carga Útil de Sequência de Byte Bruto (RBSP), uma indicação do tipo de dados, e intercalados conforme desejado com bytes de prevenção de emulação. Uma unidade de VCL NAL é uma unidade de NAL codificada para conter dados de vídeo, tais como uma fatia codificada de uma imagem. Uma unidade não VCL NAL é uma unidade NAL que contém dados não vídeo, tais como sintaxe e/ou parâmetros que suportam 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 quadro, tamanho de imagem, etc.), conforme indicado por Id (identificador) de camada e unidades não VCL NAL associadas. Um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id) é um elemento de sintaxe que especifica um identificador de uma camada que inclui uma unidade NAL.

[0045] Um decodificador de referência hipotético (HRD) é um modelo de decodificador que opera em um codificador que verifica a variabilidade de fluxos de bits produzidos por um processo de codificação para verificar conformidade com 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, tal como Codificação de Vídeo Versátil (VVC). Um conjunto de parâmetros de vídeo (VPS) é uma estrutura de sintaxe que contém parâmetros relacionados a todo um vídeo. Um conjunto de parâmetros de sequência (SPS) é uma estrutura de sintaxe contendo elementos de sintaxe que se aplicam a zero ou mais sequências de vídeo de camada codificada (CLVSs) inteiras. Um identificador de conjunto de parâmetros de vídeo de SPS (sps_video_parameter_set_id) é um elemento de sintaxe que especifica um identificador (ID) de uma referência de VPS por um SPS. Um índice de camada geral (GeneralLayerIdx[i]) é uma variável derivada que especifica um índice de uma camada correspondente i. Como tal, uma camada atual com um ID de camada de nuh_layer_id tem um índice especificado por GeneralLayerIdx[nuh_layer_id]. Um índice de camada atual é um índice de camada que corresponde a uma camada que está sendo codificada ou decodificada. Uma flag de camada independente de VPS (vps_independent_layer_flag[i]) é um elemento de sintaxe que especifica se uma camada correspondente i usa predição de camada inter. Como tal, vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se uma camada atual usa predição de camada inter. A predição de camada inter é um mecanismo de codificação de blocos de valores de amostras em uma imagem atual em uma camada atual com base em imagem(ns) de referência de uma camada diferente (por exemplo, e na mesma unidade de acesso). Uma unidade de acesso (AU) é um conjunto de imagens codificadas em diferentes camadas que são todas associadas ao mesmo tempo de emissão. Um identificador de conjunto de parâmetros de VPS (vps_video_parameter_set_id) é um elemento de sintaxe que fornece um ID para um VPS para referência por outras estruturas/elementos de sintaxe. Uma sequência de vídeo codificada é um conjunto de uma ou mais imagens codificadas. Uma sequência de vídeo decodificada é um conjunto de uma ou mais imagens decodificadas.

[0046] Os acrônimos a seguir são usados na presente invenção, Unidade de Acesso (AU), Bloco de Árvore de Codificação (CTB), Unidade de Árvore de Codificação (CTU), Unidade de Codificação (CU), Sequência de Vídeo de Camada Codificada (CLVS), Início de Sequência de Vídeo de Camada Codificada (CLVSS), Sequência de Vídeo Codificada (CVS), Início de Sequência de Vídeo Codificada (CVSS), Equipe Conjunta de Especialistas em Vídeo (JVET),

Decodificador de Referência Hipotético (HRD), Conjunto de Tiles de Movimento Restringido (MCTS), Unidade de Transferência Máxima (MTU), Camada de Abstração de Rede (NAL), Conjunto de Camadas de Saída (OLS), Ponto de Operação (OP), Contagem de Ordem de Imagem (POC), Ponto de Acesso Aleatório (RAP), Carga Útil de Sequência de Byte Bruto (RBSP), Conjunto de Parâmetros de Sequência (SPS), Conjunto de Parâmetros de Vídeo (VPS), Codificação de Vídeo Versátil (VVC).

[0047] Muitas técnicas de compressão de vídeo podem ser empregadas para reduzir o tamanho de arquivos de vídeo com perda mínima de dados. Por exemplo, as técnicas de compressão de vídeo podem incluir realizar predição espacial (por exemplo, imagem intra) e/ou predição temporal (por exemplo, imagem inter) para reduzir ou remover redundância de dados em sequências de vídeo. Para codificação de vídeo com base em bloco, uma fatia de vídeo (por exemplo, uma imagem de vídeo ou uma porção de uma imagem de vídeo) pode ser particionada em blocos de vídeo, que também podem ser denominados 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 com o uso de predição espacial em relação a amostras de referência em blocos vizinhos na mesma imagem. Os blocos de vídeo em uma fatia de predição unidirecional (P) ou predição bidirecional (B) codificada de modo inter de uma imagem podem ser codificados empregando-se predição espacial em relação a amostras de referência em blocos vizinhos na mesma imagem ou predição temporal em relação a amostras de referência em outras imagens de referência. As imagens podem ser denominadas quadros e/ou figuras, e imagens de referência podem ser denominadas quadros de referência e/ou imagens de referência. A predição espacial ou temporal resulta em um bloco preditivo que representa um bloco de imagem. Os dados residuais representam diferenças de pixel entre o bloco de imagem original e o bloco preditivo. Assim, um bloco codificado de modo inter é codificado de acordo com um vetor de movimento que aponta para um bloco de amostras de referência que formam o bloco preditivo, e os dados residuais indicam 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 do domínio de pixel para um domínio de transformada. Isso resulta em coeficientes de transformada residuais, que podem ser quantizados. Os coeficientes de transformada quantizados podem ser inicialmente dispostos em um arranjo bidimensional. Os coeficientes de transformada quantizados podem ser submetidos à varredura a fim de produzir um vetor unidimensional de coeficientes de transformada. A codificação de entropia pode ser aplicada para alcançar ainda mais compressão. Tais técnicas de compressão de vídeo são discutidas e mais detalhes abaixo.

[0048] Para garantir que um vídeo codificado possa ser decodificado de maneira acurada, o vídeo é codificado e decodificado de acordo com padrões de codificação de vídeo correspondentes. Os padrões de codificação de vídeo incluem Setor de Padronização de União Internacional de Telecomunicação (ITU) (ITU-T) H.261, Organização Internacional para Padronização/Comissão Eletrotécnica Internacional (ISO/IEC) Grupo de Especialistas de Imagem em Movimento (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, tais como Codificação de Vídeo Escalável (SVC), Codificação de Vídeo Multivistas (MVC) e Codificação de Vídeo Multivistas mais Profundidade (MVC+D), e AVC tridimensional (3D) (3D-AVC). HEVC inclui extensões, tais como HEVC Escalável (SHVC), HEVC Multivistas (MV-HEVC) e 3D HEVC (3D-HEVC). A equipe conjunta de especialistas em vídeo (JVET) de ITU-T e ISO/IEC deu início ao desenvolvimento de um padrão de codificação de vídeo denominado Codificação de Vídeo Versátil (VVC). A VVC está incluída em um Projeto de Trabalho (WD), que inclui JVET-O2001-v14.

[0049] Alguns sistemas de codificação de vídeo codificam sequências de vídeo em camadas de imagens. Imagens em camadas diferentes têm características diferentes. Então, um codificador pode transmitir camadas diferentes a um decodificador dependendo de restrições de lado de decodificador. A fim de realizar essa função, um codificador pode codificar todas as camadas em um único fluxo de bits. Mediante solicitação, o codificador pode realizar um processo de extração de subfluxo de bits para remover informações estranhas do fluxo de bits. Esse resultado é um fluxo de bits extraído que contém apenas os

dados na(s) camada(s) solicitada(s) pelo decodificador. Uma descrição de como as camadas são relacionadas pode ser incluída em um conjunto de parâmetros de vídeo (VPS). Uma camada simulcast é uma camada que é configurada para exibição sem referência a outras camadas. Quando uma camada simulcast é transmitida a um decodificador, o processo de extração de subfluxo de bits pode remover o VPS, visto que as relações de camada não são necessárias para decodificar uma camada simulcast. Infelizmente, certas variáveis em outros conjuntos de parâmetros podem referenciar o VPS. Como tal, a remoção do VPS quando camadas simulcast forem transmitidas pode aumentar a eficiência de codificação, mas também pode resultar em erros.

[0050] É revelado, no presente documento, um mecanismo para codificar um conjunto de parâmetros de sequência (SPS) de um modo que evite erros quando um VPS é removido de um fluxo de bits codificado como parte de um processo de extração de subfluxo de bits. O SPS contém um identificador de SPS VPS (sps_video_parameter_set_id). O sps_video_parameter_set_id indica um identificador do VPS que contém relações de camada para a sequência de vídeo. Em um exemplo, o sps_video_parameter_set_id é definido como zero quando o VPS for removido antes da transmissão de um fluxo de bits contendo apenas uma camada simulcast. Em outro exemplo, SPSs usados por camadas simulcast podem conter um sps_video_parameter_set_id que é definido como zero no momento de codificação. Em qualquer um dos casos, quando o sps_video_parameter_set_id for definido como zero, variáveis relacionadas a SPS que referenciam o VPS são definidas para valores padrão para evitar erros. Por exemplo, um índice de camada geral que corresponde a um identificador de camada de cabeçalho de unidade (nuh_layer_id) de camada de abstração de rede (NAL) (GeneralLayerIdx[nuh_layer_id]) indica um índice de camada atual para uma camada correspondente (por exemplo, a camada simulcast). O GeneralLayerIdx[nuh_layer_id] é definido como/inferido como sendo zero quando o sps_video_parameter_set_id for zero. Conforme outro exemplo, uma flag de camada independente de VPS para o GeneralLayerIdx[nuh_layer_id] (vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]) é armazenada no VPS e especifica se uma camada com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. A predição de camada inter não é usada para camadas simulcast. Portanto, a

vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é definida como/inferida como sendo um para indicar nenhuma predição de camada inter quando o sps_video_parameter_set_id for definido como zero. Desse modo, erros são evitados quando um VPS é removido de um fluxo de bits antes da transmissão de uma camada simulcast. Como resultado, a funcionalidade do codificador e do decodificador é aumentada. Além disso, a eficiência de codificação é aumentada removendo-se, de maneira bem-sucedida, um VPS desnecessário de um fluxo de bits incluindo apenas uma camada simulcast, que reduz uso de recurso de processador, memória e/ou sinalização de rede tanto no codificador quanto no decodificador.

[0051] A Figura 1 é um fluxograma de um método operacional exemplificativo 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-se 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, enquanto reduz sobrecarga de largura de banda associada. O decodificador decodifica, então, o arquivo de vídeo comprimido para reconstruir o sinal de vídeo original para exibição a um usuário final. O processo de decodificação geralmente reflete o processo de codificação para permitir que o decodificador reconstrua, de maneira consistente, o sinal de vídeo.

[0052] 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. Conforme outro exemplo, o arquivo de vídeo pode ser capturado por um dispositivo de captura de vídeo, tal como uma câmera de vídeo, e codificado para suportar transmissão por streaming em tempo real do vídeo. O arquivo de vídeo pode incluir tanto um componente de áudio quanto um componente de vídeo. O componente de vídeo contém uma série de quadros de imagem que, quando vista em uma sequência, proporciona a impressão visual de movimento. Os quadros contêm pixels que são expressos em termos de luz, denominados, na presente invenção, componentes de luma (ou amostras de luma), e cor, denominados componentes de croma (ou amostras de cor). Em alguns exemplos, os quadros também podem conter valores de profundidade para suportar visualização tridimensional.

[0053] Na etapa 103, o vídeo é particionado em blocos. O particionamento inclui subdividir os pixels em cada quadro em blocos quadrangulares e/ou retangulares para compressão. Por exemplo, em Codificação de Vídeo de Alta Eficiência (HEVC) (também conhecida como H.265 e MPEG-H Parte 2) o quadro pode ser primeiro dividido em unidades de árvore de codificação (CTUs), que são blocos de um tamanho predefinido (por exemplo, sessenta e quatro pixels por sessenta e quatro pixels). As CTUs contêm amostras de luma e de croma. As árvores de codificação podem ser empregadas para dividir as CTUs em blocos e, então, subdividir, de modo recursivo, os blocos até que configurações sejam alcançadas que suportam 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, 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.

[0054] 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 obter vantagem do fato de que objetos em uma cena comum tendem a aparecer em quadros sucessivos. Assim, um bloco que retrata um objeto em um quadro de referência não precisa ser descrito de maneira repetida em quadros adjacentes. Especificamente, um objeto, tal como uma mesa, pode permanecer em uma posição constante através de múltiplos quadros. Portanto, a mesa é descrita uma vez e quadros adjacentes podem se referir novamente ao quadro de referência. Os mecanismos de compatibilidade padrão podem ser empregados para compatibilizar objetos através de múltiplos quadros. Além disso, objetos em movimento podem ser representados através de múltiplos quadros, por exemplo, devido ao movimento de objeto ou movimento de câmera. Conforme um exemplo particular, um vídeo pode mostrar um automóvel que se move através da tela através de múltiplos quadros. Os 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.

[0055] A predição intra codifica blocos em um quadro comum. A predição intra obtém vantagem do fato de que componentes de luma e de croma tendem a se aglomerar em um quadro. Por exemplo, um patch de verde em uma porção de uma árvore tende a ser posicionado de modo adjacente a patches similares de verde. A predição intra emprega múltiplos modos de predição direcionais (por exemplo, trinta e três em HEVC), um modo plano e um modo de corrente contínua (DC). Os modos direcionais indicam que um bloco atual é similar/igual a amostras de um bloco vizinho em uma direção correspondente. O modo plano 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 plano, de fato, indica uma transição suave de luz/cor através de uma linha/coluna empregando-se uma inclinação relativamente constante na mudança de valores. O modo DC é empregado para suavização de delimitação e indica que um bloco é similar/igual a um valor médio associado a amostras de todos os blocos vizinhos associados às direções angulares dos modos de predição direcionais. Assim, blocos de predição intra podem representar blocos de imagem como vários valores de modo de predição relacional em vez dos valores reais. Além disso, blocos de predição inter podem representar blocos de imagem como valores de vetor de movimento em vez dos valores reais. Em qualquer um dos casos, os blocos de predição podem não representar exatamente os blocos de imagem em alguns casos. Quaisquer diferenças são armazenadas em blocos residuais. As transformadas podem ser aplicadas aos blocos residuais para comprimir adicionalmente o arquivo.

[0056] 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 com base em bloco discutida acima pode resultar na criação de imagens em bloco no decodificador. Além disso, o esquema de predição com base em bloco pode codificar um bloco e, então, reconstruir o bloco codificado para uso posterior como um bloco de referência. O esquema de filtragem in-loop aplica, de modo iterativo, filtros de supressão de ruído, filtros de desblocagem, filtros loop adaptativo e filtros de deslocamento adaptativo de amostra (SAO) aos

blocos/quadros. Esses filtros mitigam tais artefatos de bloqueio de modo que o arquivo codificado possa ser reconstruído de maneira acurada. Além disso, esses filtros mitigam artefatos nos blocos de referência reconstruídos de modo que seja menos provável que artefatos criem artefatos adicionais em blocos subsequentes que são codificados com base nos blocos de referência reconstruídos.

[0057] Uma vez que o sinal de vídeo tiver 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 de sinal de vídeo apropriada no decodificador. Por exemplo, tais 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 em memória para transmissão em direção a 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. Assim, as etapas 101, 103, 105, 107 e 109 podem ocorrer de maneira contínua e/ou simultânea através de muitos quadros e blocos. A ordem mostrada na Figura 1 é apresentada por questão de clareza e para facilitar a discussão, e não se destina a limitar o processo de codificação de vídeo a uma ordem particular.

[0058] 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 em dados de vídeo e sintaxe correspondentes. O decodificador emprega os dados de sintaxe a partir do fluxo de bits para determinar as partições para os quadros na etapa 111. O particionamento deve ser compatível com os resultados de particionamento de bloco na etapa 103. A codificação/decodificação de entropia, conforme empregado na etapa 111, é descrita. O codificador faz muitas escolhas durante o processo de compressão, tais como seleção de esquemas de particionamento de bloco a partir de várias escolhas possíveis com base no posicionamento espacial de valores na(s) imagem(ns) de entrada. A sinalização das escolhas exatas pode empregar um grande número de bins. Conforme usado na presente invenção, 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 sejam claramente inviáveis para um caso particular, deixando um conjunto de opções permitidas. Atribui-se, então, a cada opção permitida, uma palavra-código. O comprimento das palavras-código se baseia no número de opções permitidas (por exemplo, um bin para duas opções, dois bins para três a quatro opções, etc.) O codificador codifica, então, a palavra-código para a opção selecionada. Esse esquema reduz o tamanho das palavras-código, visto que as palavras-código são tão grandes quanto desejado para indicar exclusivamente uma seleção a partir de um subconjunto pequeno de opções permitidas em vez de indicar exclusivamente a seleção a partir de um conjunto potencialmente grande de todas as opções possíveis. O decodificador decodifica, então, a seleção determinando-se o conjunto de opções permitidas de um modo similar ao codificador. Ao determinar o conjunto de opções permitidas, o decodificador pode ler a palavra-código e determinar a seleção feita pelo codificador.

[0059] Na etapa 113, o decodificador realiza decodificação de bloco. Especificamente, o decodificador emprega transformadas reversas para gerar blocos residuais. Então, o decodificador emprega os blocos residuais e 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, conforme gerado 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. A sintaxe para a etapa 113 também pode ser sinalizada no fluxo de bits através de codificação de entropia, conforme discutido acima.

[0060] Na etapa 115, a filtragem é realizada nos quadros do sinal de vídeo reconstruído de um modo similar à etapa 107 no codificador. Por exemplo, filtros de supressão de ruído, filtros de desblocagem, filtros loop adaptativo e filtros SAO podem ser aplicados aos quadros para remover artefatos de bloqueio. Uma vez que os quadros são filtrados, o sinal de vídeo pode ser emitido a um display na etapa 117 para visualização por um usuário final.

[0061] A Figura 2 é um diagrama esquemático de um sistema de codificação e decodificação (codec) exemplificativo 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 retratar componentes empregados tanto em um codificador quanto em um decodificador. O sistema codec 200 recebe e particiona um sinal de vídeo conforme discutido em relação às etapas 101 e 103 no método operacional 100, que resulta em um sinal de vídeo particionado 201. O sistema codec 200 comprime, então, 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 operacional 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 codificação aritmética binária adaptativa ao contexto (CABAC) e formatação de cabeçalho 231. Tais componentes são acoplados conforme mostrado. Na Figura 2, linhas pretas indicam movimento de dados a serem codificados/decodificados enquanto linhas tracejadas indicam movimento de 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 serão descritos a seguir.

[0062] 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 repartição para subdividir um bloco de pixels em blocos menores de pixels. Esses blocos podem ser, então, subdivididos adicionalmente em blocos menores. Os blocos podem ser denominados nós na árvore de codificação. Nós-pai maiores são repartidos em

nós-filho menores. O número de vezes que um nó é subdividido é denominado 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 (Cr) de diferença de vermelho e um bloco(s) de croma (Cb) de diferença de azul em conjunto com instruções de sintaxe correspondentes para a CU. Os modos de repartição podem incluir uma árvore binária (BT), árvore tripla (TT) e uma quadtree (QT) empregadas para particionar um nó em dois, três ou quatro nós-filho, respectivamente, de formatos variados dependendo dos modos de repartição 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.

[0063] 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 restrições de aplicação. Por exemplo, o componente de controle de codificador geral 211 gerencia a otimização de tamanho de taxa de bit/fluxo de bits versus qualidade de reconstrução. Tais decisões podem ser tomadas com base em disponibilidade de largura de banda/espaço de armazenamento e solicitações de resolução de imagem. O componente de controle de codificador geral 211 também gerencia a utilização de buffer à luz de velocidade de transmissão para mitigar questões de insuficiência (underrun) e de saturação (overrun) de buffer. Para gerenciar essas questões, o componente de controle de codificador geral 211 gerencia 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. Portanto, o componente de controle de codificador geral 211 controla os outros componentes do sistema codec 200 para equilibrar a qualidade de reconstrução de 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.

[0064] O sinal de vídeo particionado 201 também é enviado ao componente de estimação de movimento 221 e ao 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 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 passas de codificação, por exemplo, para selecionar um modo de codificação apropriado para cada bloco de dados de vídeo.

[0065] O componente de estimação de movimento 221 e o componente de compensação de movimento 219 podem ser altamente integrados, mas são ilustrados de forma separada 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 estimam 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. Constata-se que um bloco preditivo é um bloco que é estritamente compatível com o bloco a ser codificado, em termos de diferença de pixel. Um bloco preditivo também pode ser denominado um bloco de referência. Tal diferença de pixel pode ser determinada pela soma da diferença absoluta (SAD), soma da diferença quadrática (SSD) ou outra métrica 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 divididos 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-se 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 múltiplos blocos de referência, múltiplos vetores de movimento, etc. para um bloco/quadro atual, e

pode selecionar os blocos de referência, vetores de movimento, etc. tendo as melhores características de taxa-distorção. As melhores características de taxa-distorção equilibram qualidade de reconstrução de vídeo (por exemplo, quantidade de perda de dados por compressão) com eficiência de codificação (por exemplo, tamanho da codificação final).

[0066] Em alguns exemplos, o sistema codec 200 pode calcular valores para posições subinteiras de pixel 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 fracionadas da imagem de referência. Portanto, o componente de estimação de movimento 221 pode realizar uma busca de movimento em relação às posições de pixel inteiras e posições de pixel fracionadas e emitir um vetor de movimento com precisão de pixel fracionada. 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-se 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 o componente de formatação de cabeçalho e CABAC 231 para codificação e movimento para o componente de compensação de movimento 219.

[0067] 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. Mediante o recebimento do vetor de movimento para a PU do bloco de vídeo atual, o componente de compensação de movimento 219 pode localizar o bloco preditivo para onde o vetor de movimento aponta. Um bloco de vídeo residual é, então, formado subtraindo-se valores de pixel do bloco preditivo dos valores de pixel do bloco de vídeo atual sendo codificado, formando valores de diferença de pixel. Em geral, o componente de estimação de movimento 221 realiza estimação de movimento em relação a componentes de luma, e o componente de compensação de movimento 219 usa vetores de movimento calculados com base nos componentes de luma para componentes de

croma e componentes de luma. O bloco preditivo e o bloco residual são encaminhados ao componente de escalonamento de transformada e quantização 213.

[0068] O sinal de vídeo particionado 201 também é enviado ao componente de estimação de imagem intra 215 e ao componente de predição de imagem intra 217. Assim como com o componente de estimação de movimento 221 e com o componente de compensação de movimento 219, o componente de estimação de imagem intra 215 e o componente de predição de imagem intra 217 podem ser altamente integrados, mas são ilustrados de maneira separada 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 a blocos em um quadro atual, como uma alternativa à predição inter realizada pelo componente de estimação de movimento 221 e pelo componente de compensação de movimento 219 entre quadros, conforme 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 ao componente de CABAC 231 e formatação de cabeçalho para codificação.

[0069] Por exemplo, o componente de estimação de imagem intra 215 calcula 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 tendo 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 razões 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 em otimização de taxa-distorção (RDO).

[0070] 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 em valores entre o bloco preditivo e o bloco original, representada 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 componentes de luma e de croma.

[0071] 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 discreta de cosseno (DCT), uma transformada discreta de seno (DST) ou uma transformada conceitualmente similar, ao bloco residual, produzindo um bloco de vídeo que compreende valores de coeficiente de transformada residual. As transformadas wavelet, transformadas de número inteiro, transformadas de sub-banda ou outros tipos de transformadas também podem ser usados. A transformada pode converter as informações residuais de um domínio de valor de pixel para um domínio de transformada, tal 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 em frequência. Tal escalonamento envolve aplicar um fator de escala às informações residuais de modo que diferentes informações de frequência 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 taxa de bits. O processo de quantização pode reduzir a profundidade de bit associada a alguns ou todos os coeficientes. O grau de quantização pode ser modificado ajustando-se um parâmetro de quantização. Em alguns exemplos, o componente de escalonamento de transformada e

quantização 213 pode realizar, então, 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.

[0072] O componente de escalonamento e transformada inversa 229 aplica uma operação reversa do componente de escalonamento de transformada e quantização 213 para suportar estimação de movimento. O componente de escalonamento e transformada inversa 229 aplica escalonamento inverso, transformação e/ou quantização 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 o componente de compensação de movimento 219 pode calcular um bloco de referência adicionando-se o bloco residual novamente a um bloco preditivo correspondente para uso em estimação de movimento de um bloco/quadro posterior. Os filtros são aplicados aos blocos de referência reconstruídos para mitigar artefatos criados durante escalonamento, quantização e transformada. Tais artefatos podem causar, de outro modo, predição não acurada (e criar artefatos adicionais) quando blocos subsequentes são preditos.

[0073] 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 a 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 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 ser aplicados, então, ao bloco de imagem reconstruído. Em alguns exemplos, em vez disso, os filtros podem ser aplicados aos blocos residuais. Assim como 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 em conjunto, mas são retratados de maneira separada para fins conceituais. Os filtros aplicados aos blocos de referência reconstruídos são aplicados a regiões espaciais particulares e incluem múltiplos parâmetros para ajustar o modo que tais filtros são aplicados. O componente de análise de controle de filtro 227 analisa os blocos de referência reconstruídos para determinar onde tais filtros devem ser

aplicados e define parâmetros correspondentes. Tais 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 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 de frequência, dependendo do exemplo.

[0074] Ao operar como um codificador, o bloco de predição, bloco residual e/ou bloco de imagem reconstruído filtrado são armazenados no componente de buffer de imagem decodificada 223 para uso posterior em 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.

[0075] O componente de formatação de cabeçalho e CABAC 231 recebe os dados dos vários componentes do sistema codec 200 e codifica tais 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, tais 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 quantizado 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. Tais informações também podem incluir tabelas de índice de modo de predição intra (também denominadas 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-se codificação de entropia. Por exemplo, as informações podem ser codificadas empregando-se codificação de comprimento variável adaptativa ao contexto (CAVLC), CABAC, codificação

aritmética binária com base em sintaxe adaptativa ao contexto (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.

[0076] A Figura 3 é um diagrama de blocos ilustrando um codificador de vídeo exemplificativo 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. O 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.

[0077] 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 ao componente de predição de imagem intra 217. O sinal de vídeo particionado 301 também é encaminhado a 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 ao componente de compensação de movimento 219. Os blocos de predição e os blocos residuais do componente de predição de imagem intra 317 e do componente de compensação de movimento 321 são encaminhados a 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 (em conjunto com dados de controle associados) são encaminhados a 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.

[0078] 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. Os 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.

[0079] A Figura 4 é um diagrama de blocos ilustrando um decodificador de vídeo exemplificativo 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 operacional 100. O decodificador 400 recebe um fluxo de bits, por exemplo, a partir de um codificador 300, e gera um sinal de vídeo de saída reconstruído com base no fluxo de bits para exibição para um usuário final.

[0080] 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, tal como codificação CAVLC, CABAC, SBAC, 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, tais como dados de controle gerais, dados de controle de filtro, informações de partição, dados de movimento, dados de predição e coeficientes

de transformada quantizados a partir 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.

[0081] Os blocos de predição e/ou blocos residuais reconstruídos 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 similar ao componente de estimação de imagem intra 215 e a um 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 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 ao componente de filtros in-loop 225, respectivamente. O componente de filtros in-loop 425 filtra os blocos residuais, blocos de predição e/ou blocos de imagem reconstruídos, e tais informações são armazenadas no componente de buffer de imagem decodificada 423. Os blocos de imagem reconstruídos 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 ao 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 ao 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. Tais quadros

também podem ser colocados em uma sequência. A sequência é emitida em direção a um display como um sinal de vídeo de saída reconstruído.

[0082] A Figura 5 é um diagrama esquemático ilustrando um HRD exemplificativo 500. Um HRD 500 pode ser empregado em um codificador, tal como o sistema codec 200 e/ou o codificador 300. O HRD 500 pode verificar o fluxo de bits criado na etapa 109 do método 100 antes de o fluxo de bits ser encaminhado para um decodificador, tal como decodificador 400. Em alguns exemplos, o fluxo de bits pode ser continuamente encaminhado através do HRD 500 à medida que o fluxo de bits é codificado. No caso em que uma porção do fluxo de bits não entra em conformidade com restrições associadas, o HRD 500 pode indicar tal falha a um codificador para fazer com que o codificador codifique novamente a seção correspondente do fluxo de bits com diferentes mecanismos.

[0083] O HRD 500 inclui um agendador de fluxo hipotético (HSS) 541. Um HSS 541 é um componente configurado para realizar um mecanismo de entrega hipotético. O mecanismo de entrega hipotético é usado para verificar a conformidade de um fluxo de bits ou um decodificador a respeito da temporização e fluxo de dados de um fluxo de bits 551 inserido no HRD 500. Por exemplo, o HSS 541 pode receber um fluxo de bits 551 emitido a partir de um codificador e gerenciar o processo de teste de conformidade no fluxo de bits 551. Em um exemplo particular, o HSS 541 pode controlar a taxa que imagens codificadas se movem através do HRD 500 e verificar que o fluxo de bits 551 não contém dados que não estão em conformidade.

[0084] O HSS 541 pode encaminhar o fluxo de bits 551 a um CPB 543 a uma taxa predefinida. O HRD 500 pode gerenciar dados em unidades de decodificação (DU) 553. Uma DU 553 é uma Unidade de Acesso (AU) ou um subconjunto de uma AU e unidades de camada de abstração de rede (NAL) de camada de codificação não vídeo (VCL) associadas. Especificamente, uma AU contém uma ou mais imagens associadas a um tempo de emissão. Por exemplo, uma AU pode conter uma única imagem em um fluxo de bits de camada única, e pode conter uma imagem para cada camada em um fluxo de bits de múltiplas camadas. Cada imagem de uma AU pode ser dividida em fatias que são incluídas, cada uma, em uma unidade de VCL NAL correspondente. Portanto, uma DU 553 pode conter uma ou mais imagens, uma ou mais fatias de uma imagem, ou combinações das mesmas. Além disso, parâmetros usados para decodificar a AU/

DU, imagens e/ou fatias podem ser incluídos em unidades não VCL NAL. Como tal, a DU 553 contém unidades não VCL NAL que contêm dados necessários para suportar decodificação das unidades de VCL NAL na DU 553. O CPB 543 é um buffer primeiro a entrar primeiro a sair no HRD 500. O CPB 543 contém DUs 553 incluindo dados de vídeo em ordem de decodificação. O CPB 543 armazena os dados de vídeo para uso durante verificação de conformidade de fluxo de bits.

[0085] O CPB 543 encaminha as DUs 553 para um componente de processo de decodificação 545. O componente de processo de decodificação 545 é um componente que está em conformidade com o padrão de VVC. Por exemplo, o componente de processo de decodificação 545 pode emular um decodificador 400 empregado por um usuário final. O componente de processo de decodificação 545 decodifica as DUs 553 a uma taxa que pode ser alcançada por um decodificador de usuário final exemplificativo. Se o componente de processo de decodificação 545 não puder decodificar as DUs 553 de maneira rápida o bastante para prevenir uma sobrecarga (ou prevenir uma insuficiência de buffer) do CPB 543, então, o fluxo de bits 551 não está em conformidade com o padrão e deve ser codificado novamente.

[0086] O componente de processo de decodificação 545 decodifica as DUs 553, que cria DUs decodificadas 555. Uma DU decodificada 555 contém uma imagem decodificada. As DUs decodificadas 555 são encaminhadas para um DPB 547. O DPB 547 pode ser substancialmente similar a um componente de buffer de imagem decodificada 223, 323 e/ou 423. Para suportar predição inter, imagens que são marcadas para uso como imagens de referência 556 que são obtidas a partir das DUs decodificadas 555 retornam ao componente de processo de decodificação 545 para suportar decodificação adicional. O DPB 547 emite a sequência de vídeo decodificada como uma série de imagens 557. As imagens 557 são imagens reconstruídas que geralmente espelham imagens codificadas no fluxo de bits 551 pelo codificador.

[0087] As imagens 557 são encaminhadas para um componente de cropping de saída 549. O componente de cropping de saída 549 é configurado para aplicar uma janela de cropping de conformidade às imagens 557. Isso resulta em imagens recortadas (cropped) de saída 559. Uma imagem recortada de saída 559 é uma imagem completamente reconstruída. Assim, a imagem recortada de saída 559 imita o que um usuário final veria mediante decodificação do fluxo de

bits 551. Como tal, o codificador pode rever as imagens recortadas de saída 559 para garantir que a codificação seja satisfatória.

[0088] O HRD 500 é inicializado com base em parâmetros de HRD no fluxo de bits 551. Por exemplo, o HRD 500 pode ler parâmetros de HRD a partir de um VPS, um SPS e/ou mensagens SEI. O HRD 500 pode, então, realizar operações de teste de conformidade no fluxo de bits 551 com base nas informações em tais parâmetros de HRD. Conforme um exemplo específico, o HRD 500 pode determinar um ou mais agendamentos de entrega de CPB a partir dos parâmetros de HRD. Um agendamento de entrega especifica temporização para entrega de dados de vídeo de e/ou para uma localização de memória, tal como um CPB e/ou um DPB. Portanto, um agendamento de entrega de CPB especifica temporização para entrega de AUs, DUs 553 e/ou imagens, de/para o CPB 543. Deve-se observar que o HRD 500 pode empregar agendamentos de entrega de DPB para o DPB 547 que são similares aos agendamentos de entrega de CPB.

[0089] O vídeo pode ser codificado em diferentes camadas e/ou OLSs para uso por decodificadores com níveis variados de capacidades de hardware bem como condições de rede variadas. Os agendamentos de entrega de CPB são selecionados para refletir essas questões. Assim, subfluxos de bits de camada mais alta são designados para condições de rede e hardware ideais e, portanto, camadas superiores podem receber um ou mais agendamentos de entrega de CPB que empregam uma grande quantidade de memória no CPB 543 e atrasos curtos para transferências das DUs 553 em direção ao DPB 547. Da mesma maneira, subfluxos de bits de camada mais baixa são designados para capacidades de hardware de decodificador limitadas e/ou condições de rede ruins. Portanto, camadas inferiores podem receber um ou mais agendamentos de entrega de CPB que empregam uma pequena quantidade de memória no CPB 543 e atrasos mais longos para transferências das DUs 553 em direção ao DPB 547. Os OLSs, camadas, subcamadas ou combinações dos mesmos podem, então, ser testados de acordo com o agendamento de entrega correspondente para garantir que o subfluxo de bits resultante possa ser corretamente decodificado sob as condições que são esperadas para o subfluxo de bits. Assim, os parâmetros de HRD no fluxo de bits 551 podem indicar os agendamentos de entrega de CPB bem como incluir dados suficientes para permitir que o HRD 500

determine os agendamentos de entrega de CPB e correlacione os agendamentos de entrega de CPB aos OLSs, camadas e/ou subcamadas correspondentes.

[0090] A Figura 6 é um diagrama esquemático ilustrando uma sequência de vídeo de múltiplas camadas exemplificativa 600 configurada para predição de camada inter 621. A sequência de vídeo de múltiplas camadas 600 pode ser codificada por um codificador, tal como sistema codec 200 e/ou codificador 300 e decodificada por um decodificador, tal como sistema codec 200 e/ou decodificador 400, por exemplo, de acordo com o método 100. Além disso, a sequência de vídeo de múltiplas camadas 600 pode ser verificada quanto à conformidade com padrão por um HRD, tal como HRD 500. A sequência de vídeo de múltiplas camadas 600 é incluída para retratar uma aplicação exemplificativa para camadas em uma sequência de vídeo codificada. Uma sequência de vídeo de múltiplas camadas 600 é qualquer sequência de vídeo que emprega uma pluralidade de camadas, tal como camada N 631 e camada N+1 632.

[0091] Em um exemplo, a sequência de vídeo de múltiplas camadas 600 pode empregar predição de camada inter 621. A predição de camada inter 621 é aplicada entre imagens 611, 612, 613 e 614 e imagens 615, 616, 617 e 618 em diferentes camadas. No exemplo mostrado, as imagens 611, 612, 613 e 614 são parte da camada N+1 632 e as imagens 615, 616, 617 e 618 são parte da camada N 631. Uma camada, tal como a camada N 631 e/ou a camada N+1 632, é um grupo de imagens que são todas associadas a um valor similar de uma característica, tal como um tamanho, qualidade, resolução, razão entre sinal e ruído, capacidade, etc. similar. Uma camada pode ser definida formalmente como um conjunto de unidades de VCL NAL e unidades não VCL NAL associadas. Uma unidade VCL NAL é uma unidade NAL codificada para conter dados de vídeo, tais como uma fatia codificada de uma imagem. Uma unidade não VCL NAL é uma unidade NAL que contém dados não vídeo, tais como sintaxe e/ou parâmetros que suportam decodificação dos dados de vídeo, desempenho de verificação de conformidade ou outras operações.

[0092] No exemplo mostrado, a camada N+1 632 é associada a um tamanho de imagem maior que a camada N 631. Assim, as imagens 611, 612, 613 e 614 na camada N+1 632 têm um tamanho de imagem maior (por exemplo, altura e largura maiores e, portanto, mais amostras) do que as imagens 615, 616, 617 e 618 na camada N 631 neste exemplo. Entretanto, tais imagens podem ser

separadas entre a camada N+1 632 e a camada N 631 por outras características. Embora apenas duas camadas, camada N+1 632 e camada N 631, sejam mostradas, um conjunto de imagens pode ser separado em qualquer número de camadas com base em características associadas. A camada N+1 632 e a camada N 631 também podem ser denotadas por um Id de camada. Um Id de camada é um item de dados que é associado a uma imagem e denota que a imagem é parte de uma camada indicada. Assim, cada imagem 611 a 618 pode ser associada a um Id de camada correspondente para indicar qual camada N+1 632 ou camada N 631 inclui a imagem correspondente. Por exemplo, um Id de camada pode incluir um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id), que é um elemento de sintaxe que especifica um identificador de uma camada que inclui uma unidade NAL (por exemplo, que inclui fatias e/ou parâmetros das imagens em uma camada). Atribui-se, de modo geral, a uma camada associada a um tamanho de fluxo de bits/qualidade inferior, tal como a camada N 631, um Id de camada mais baixa e é denominada uma camada mais baixa. Além disso, atribui-se, de modo geral, a uma camada associada a um tamanho de fluxo de bits/qualidade maior, tal como camada N+1 632, um Id de camada mais alta e é denominada uma camada mais alta.

[0093] As imagens 611 a 618 em diferentes camadas 631 a 632 são configuradas para ser exibidas em alternativa. Conforme um exemplo específico, um decodificador pode decodificar e exibir a imagem 615 em um tempo de exibição atual se uma imagem menor for desejada ou o decodificador pode decodificar e exibir a imagem 611 no tempo de exibição atual se uma imagem maior for desejada. Como tal, as imagens 611 a 614 na camada mais alta N+1 632 contêm substancialmente os mesmos dados de imagem que as imagens correspondentes 615 a 618 na camada mais baixa N 631 (apesar da diferença de tamanho de imagem). Especificamente, a imagem 611 contém substancialmente os mesmos dados de imagem que a imagem 615, a imagem 612 contém substancialmente os mesmos dados de imagem que a imagem 616, etc.

[0094] As imagens 611 a 618 podem ser codificadas por referência a outras imagens 611 a 618 na mesma camada N 631 ou N+1 632. A codificação de uma imagem em referência a outra imagem na mesma camada resulta em predição inter 623. A predição inter 623 é retratada por setas de linha contínua. Por exemplo, a imagem 613 pode ser codificada empregando-se predição inter

623 usando uma ou duas dentre as imagens 611, 612 e/ou 614 na camada N+1 632 como uma referência, em que uma imagem é referida para predição inter unidirecional e/ou duas imagens são referidas para predição inter bidirecional. Além disso, a imagem 617 pode ser codificada empregando-se predição inter 623 usando uma ou duas dentre as imagens 615, 616 e/ou 618 na camada N 631 como uma referência, em que uma imagem é referida para predição inter unidirecional e/ou duas imagens são referidas para predição inter bidirecional. Quando uma imagem é usada como uma referência para outra imagem na mesma camada ao realizar predição inter 623, a imagem pode ser denominada uma imagem de referência. Por exemplo, a imagem 612 pode ser uma imagem de referência usada para codificar a imagem 613 de acordo com a predição inter 623. A predição inter 623 também pode ser denominada predição de camada intra em um contexto de múltiplas camadas. Como tal, a predição inter 623 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.

[0095] As imagens 611 a 618 também podem ser codificadas por referência a outras imagens 611 a 618 em diferentes camadas. Esse processo é conhecido como predição de camada inter 621, e é retratado por setas tracejadas. A predição de camada inter 621 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 diferentes camadas e, portanto, têm diferentes IDs de camada. Por exemplo, uma imagem em uma camada mais baixa N 631 pode ser usada como uma imagem de referência para codificar uma imagem correspondente em uma camada mais alta N+1 632. Conforme um exemplo específico, a imagem 611 pode ser codificada por referência à imagem 615 de acordo com predição de camada inter 621. Em tal caso, a imagem 615 é usada como uma imagem de referência de camada inter. Uma imagem de referência de camada inter é uma imagem de referência utilizada para predição de camada inter 621. Na maioria dos casos, a predição de camada inter 621 é restringida de modo que uma imagem atual, tal como a imagem 611, só possa usar imagem(ns) de referência de camada inter que são incluídas na mesma AU e que estão em uma camada mais baixa, tal como a imagem 615. Quando múltiplas camadas (por exemplo, mais de duas) estiverem disponíveis, a

predição de camada inter 621 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.

[0096] Um codificador de vídeo pode empregar uma sequência de vídeo de múltiplas camadas 600 para codificar imagens 611 a 618 por meio de muitas combinações e/ou permutações diferentes de predição inter 623 e predição de camada inter 621. Por exemplo, a imagem 615 pode ser codificada de acordo com predição intra. As imagens 616 a 618 podem, então, ser codificadas de acordo com predição inter 623 usando-se a imagem 615 como uma imagem de referência. Além disso, a imagem 611 pode ser codificada de acordo com predição de camada inter 621 usando-se a imagem 615 como uma imagem de referência de camada inter. As imagens 612 a 614 podem ser, então, codificadas de acordo com predição inter 623 usando-se a imagem 611 como uma imagem de referência. Como tal, uma imagem de referência pode servir tanto como uma imagem de referência de camada única quanto como uma imagem de referência de camada inter para diferentes mecanismos de codificação. Codificando-se imagens de camada mais alta N+1 632 com base em imagens de camada mais baixa N 631, a camada mais alta N+1 632 pode evitar o emprego de predição intra, que tem eficiência de codificação muito mais baixa do que predição inter 623 e predição de camada inter 621. Como tal, a baixa eficiência de codificação de predição intra pode ser limitada às imagens de qualidade menor/inferior 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 em entradas de lista(s) de imagens de referência contida em uma estrutura de lista de imagens de referência.

[0097] Deve-se observar que camadas, tais como a camada N+1 632 e a camada N 631, podem ser incluídas em conjuntos de camadas de saída (OLSs). Um OLS é um conjunto de uma ou mais camadas, em que pelo menos uma camada é uma camada de saída. Por exemplo, a camada N 631 pode ser incluída em um primeiro OLS e a camada N 631 e a camada N-1 632 podem ser, ambas, incluídas em um segundo OLS. Isso permite que um OLSs diferente seja enviado para diferentes decodificadores, dependendo das condições de lado de decodificador. Por exemplo, um processo de extração de subfluxo de bits pode remover dados que não estão relacionados a um OLS alvo da sequência de vídeo

de múltiplas camadas 600 antes de o OLS alvo ser enviado a um decodificador. Como tal, uma cópia codificada da sequência de vídeo de múltiplas camadas 600 pode ser armazenada em um codificador (ou um servidor de conteúdo correspondente), e vários OLSs podem ser extraídos e enviados a diferentes decodificadores mediante solicitação.

[0098] Uma camada simulcast é uma camada que não emprega predição de camada inter 621. Por exemplo, a camada N+1 632 é codificada por referência à camada N 631 com base em predição de camada inter 621. Entretanto, a camada 631 não é codificada por referência a outra camada. Como tal, a camada 631 é uma camada simulcast. Sequências de vídeo escaláveis, tais como sequência de vídeo de múltiplas camadas 600, geralmente empregam uma camada base e uma ou mais camadas de enriquecimento que enriquecem alguma propriedade da camada base. Na Figura 6, a camada N 631 é uma camada base. Uma camada base é geralmente codificada como uma camada simulcast. Deve-se observar também que a Figura 6 é exemplificativa e não limitante, visto que sequências de vídeo com múltiplas camadas podem usar muitas combinações/permutações diferentes de dependências. Um fluxo de bits pode conter qualquer número de camadas e qualquer número de tais camadas podem ser camadas simulcast. Por exemplo, predição de camada inter 621 pode ser completamente omitida, caso em que todas as camadas são camadas simulcast. Conforme outro exemplo, aplicações multivistas exibem duas ou mais camadas de saída. Como tal, uma aplicação multivistas geralmente inclui duas ou mais camadas base, que são camadas simulcast, e pode incluir camadas de enriquecimento que correspondem a cada camada base.

[0099] As camadas simulcast podem ser manuseadas de maneira diferente de camadas que usam predição de camada inter 621. Por exemplo, ao codificar camadas que usam predição de camada inter 621, um codificador deve indicar o número de camadas bem como as dependências entre as camadas a fim de suportar decodificação. Entretanto, tais informações podem ser omitidas para camadas simulcast. Por exemplo, a configuração da camada N+1 632 e da camada N 631 pode ser indicada em um VPS conforme discutido abaixo em mais detalhes. Entretanto, a camada N 631 pode ser decodificada sem tais informações. Como tal, o VPS pode ser removido de um fluxo de bits correspondente quando apenas a camada N 631 for transmitida a um

decodificador. Entretanto, isso pode criar erros se parâmetros remanescentes no fluxo de bits referenciarem o VPS. Essas e outras questões são discutidas em mais detalhes abaixo.

[0100] A Figura 7 é um diagrama esquemático ilustrando um fluxo de bits exemplificativo 700. Por exemplo, o fluxo de bits 700 pode ser gerado por um sistema codec 200 e/ou um codificador 300 para decodificação por um sistema codec 200 e/ou um decodificador 400 de acordo com o método 100. Além disso, o fluxo de bits 700 pode incluir uma sequência de vídeo de múltiplas camadas 600. Além disso, o fluxo de bits 700 pode incluir vários parâmetros para controlar a operação de um HRD, tal como HRD 500. Com base em tais parâmetros, o HRD 500 pode verificar o fluxo de bits 700 quanto à conformidade com padrões antes de transmissão em direção a um decodificador para decodificação.

[0101] O fluxo de bits 700 inclui um VPS 711, um ou mais SPSs 713, uma pluralidade de conjuntos de parâmetros imagem (PPSs) 715, uma pluralidade de cabeçalhos de fatia 717 e dados de imagem 720. Um VPS 711 contém dados relacionados a todo o fluxo de bits 700. Por exemplo, o VPS 711 pode conter OLSs, camadas e/ou subcamadas relacionadas a dados utilizadas no fluxo de bits 700. Um SPS 713 contém dados de sequência comuns a todas as imagens em uma sequência de vídeo codificada contida no fluxo de bits 700. Por exemplo, cada camada pode conter uma ou mais sequências de vídeo codificadas, e cada sequência de vídeo codificada pode se referir a um SPS 713 para parâmetros correspondentes. Os parâmetros em um SPS 713 podem incluir dimensionamento de imagem, profundidade de bit, parâmetros de ferramenta de codificação, restrições de taxa de bit, etc. Deve-se observar que, embora cada sequência se refira a um SPS 713, um único SPS 713 pode conter dados para múltiplas sequências em alguns exemplos. O PPS 715 contém parâmetros que se aplicam a toda uma imagem. Portanto, cada imagem na sequência de vídeo pode se referir a um PPS 715. Deve-se observar que, embora cada imagem se refira a um PPS 715, um único PPS 715 pode conter dados para múltiplas imagens em alguns exemplos. Por exemplo, múltiplas imagens similares podem ser codificadas de acordo com parâmetros similares. Em tal caso, um único PPS 715 pode conter dados para tais imagens similares. O PPS 715 pode indicar ferramentas de codificação disponíveis para fatias em imagens correspondentes, parâmetros de quantização, deslocamentos, etc.

[0102] O cabeçalho de fatia 717 contém parâmetros que são específicos a cada fatia em uma imagem. Portanto, pode haver um cabeçalho de fatia 717 por fatia na sequência de vídeo. O cabeçalho de fatia 717 pode conter informações de tipo de fatia, contagens de ordem de imagem (POCs), listas de imagens de referência, pesos de predição, pontos de entrada de tile, parâmetros de desblocagem, etc. Deve-se observar que, em alguns exemplos, um fluxo de bits 700 também pode incluir um cabeçalho de imagem, que é uma estrutura de sintaxe que contém parâmetros que se aplicam a todas as fatias em uma única imagem. Por essa razão, um cabeçalho de imagem e um cabeçalho de fatia 717 podem ser usados de maneira intercambiável em alguns contextos. Por exemplo, certos parâmetros podem ser movidos entre o cabeçalho de fatia 717 e um cabeçalho de imagem dependendo de tais parâmetros serem comuns a todas as fatias em uma imagem.

[0103] Os dados de imagem 720 contêm dados de vídeo codificados de acordo com predição inter, predição de camada inter e/ou predição intra bem como dados residuais transformados e quantizados correspondentes. Por exemplo, os dados de imagem 720 podem incluir camadas 723 e 724, imagens 725 e 726 e/ou fatias 727 e 728. Uma camada 723 e 724 é um conjunto de unidades VCL NAL 741 que compartilham uma característica especificada (por exemplo, uma resolução comum, taxa de quadro, tamanho de imagem, etc.), conforme indicado por um ID de camada, tal como um nuh_layer_id 732, e unidades não VCL NAL associadas 742. Por exemplo, uma camada 723 pode incluir um conjunto de imagens 725 que compartilham o mesmo nuh_layer_id 732. Da mesma maneira, uma camada 724 pode incluir um conjunto de imagens 726 que compartilham o mesmo nuh_layer_id 732. As camadas 723 e 724 podem ser substancialmente similares, mas podem conter conteúdo diferente. Por exemplo, as camadas 723 e 724 podem conter a camada N 631 e a camada N+1 632, respectivamente, da Figura 6. Como tal, um fluxo de bits codificado 700 pode incluir múltiplas camadas 723 e 724. Embora apenas duas camadas 723 e 724 sejam mostradas por questão de clareza de discussão, qualquer número de camadas 723 e 724 pode ser incluído no fluxo de bits 700.

[0104] Um nuh_layer_id 732 é um elemento de sintaxe que especifica um identificador de uma camada 723 e/ou 724 que inclui pelo menos uma unidade NAL. Por exemplo, uma camada de qualidade mais baixa possível, conhecida

como uma camada base, pode incluir o menor valor de nuh_layer_id 732 com valores crescentes de nuh_layer_id 732 para camadas de qualidade mais alta. Portanto, uma camada mais baixa é uma camada 723 ou 724 com um valor menor de nuh_layer_id 732 e uma camada mais alta é uma camada 723 ou 724 com um valor maior de nuh_layer_id 732. Os dados das camadas 723 e 724 são correlacionados com base no nuh_layer_id 732. Por exemplo, conjuntos de parâmetros e dados de vídeo podem ser associados a um valor de nuh_layer_id 732 que corresponde à camada mais baixa possível 723 ou 724 que inclui tais conjuntos de parâmetros/dados de vídeo. Como tal, um conjunto de unidades VCL NAL 741 é parte de uma camada 723 e/ou 724 quando o conjunto de unidades VCL NAL 741 tiver, todas, um valor particular de nuh_layer_id 732.

[0105] Uma imagem 725 e 726 é um arranjo de amostras de luma e/ou um arranjo de amostras de croma que criam um quadro ou um campo das mesmas. Por exemplo, uma imagem 725/726 é uma imagem codificada que pode ser emitida para exibição ou usada para suportar codificação de outra(s) imagem(ns) para emissão. As imagens 725 e 726 são substancialmente similares, mas a imagem 725 está contida na camada 723 enquanto a imagem 726 está contida na camada 724. Uma imagem 725 e 726 contém uma ou mais fatias 727 e 728, respectivamente. Uma fatia 727/728 pode ser definida como um número inteiro de tiles completos ou um número inteiro de linhas de unidade de árvore de codificação (CTU) completas consecutivas (por exemplo, dentro de um tile) de uma imagem 725/726 que estão exclusivamente contidos em uma única unidade NAL, tal como uma unidade VCL NAL 741. As fatias 727 e as fatias 728 são substancialmente similares, exceto que as fatias 727 são incluídas nas imagens 725 e na camada 723 enquanto as fatias 728 são incluídas nas imagens 726 e na camada 724. As fatias 727/728 são divididas, de modo adicional, em CTUs e/ou blocos de árvore de codificação (CTBs). Uma CTU é um grupo de amostras de um tamanho predefinido que podem ser particionadas por uma árvore de codificação. Um CTB é um subconjunto de uma CTU e contém componentes de luma ou componentes de croma da CTU. As CTUs /CTBs são divididas, de modo adicional, em blocos de codificação com base em árvores de codificação. Os blocos de codificação podem ser, então, codificados/decodificados de acordo com mecanismos de predição.

[0106] Um fluxo de bits 700 pode ser codificado como uma sequência

de unidades NAL. Uma unidade NAL é um recipiente para dados de vídeo e/ou sintaxe de suporte. Uma unidade NAL pode ser uma unidade VCL NAL 741 ou uma unidade não VCL NAL 742. Uma unidade VCL NAL 741 é uma unidade NAL codificada para conter dados de vídeo, tais como dados de imagem 720 e um cabeçalho de fatia associado 717. Conforme um exemplo específico, cada fatia 727 e 728 e um cabeçalho de fatia associado 717 podem ser codificados em uma única unidade VCL NAL 741. Uma unidade não VCL NAL 742 é uma unidade NAL que contém dados não vídeo, tais como sintaxe e/ou parâmetros que suportam decodificação dos dados de vídeo, desempenho de verificação de conformidade ou outras operações. Por exemplo, uma unidade não VCL NAL 742 pode conter um VPS 711, um SPS 713, um PPS 715, um cabeçalho de imagem ou outra sintaxe de suporte. Como tal, um fluxo de bits 700 é uma série de unidades VCL NAL 741 e unidades não VCL NAL 742. Cada unidade NAL contém um nuh_layer_id 732, que permite que um codificador ou decodificador determine qual camada 723 ou 724 inclui a unidade NAL correspondente.

[0107] Um fluxo de bits 700, incluindo múltiplas camadas 723 e 724, pode ser codificado e armazenado até ser solicitado por um decodificador. Por exemplo, um decodificador pode solicitar uma camada 723, uma camada 724 e/ou um OLS contendo múltiplas camadas 723 e 724. Em um exemplo particular, a camada 723 é uma camada base e a camada 724 é uma camada de enriquecimento. Camadas adicionais também podem ser empregadas no fluxo de bits 700. O codificador e/ou um servidor de conteúdo devem enviar apenas as camadas 723 e/ou 724, ao decodificador, que são necessárias para decodificar camada(s) de saída solicitada(s). Por exemplo, quando camadas forem usadas para diferentes tamanhos de imagem, um decodificador que solicita o maior tamanho de imagem pode receber todo o fluxo de bits 700 com ambas as camadas 723 e 724. Um decodificador que solicita o menor tamanho de imagem pode receber as únicas camadas 723. Um decodificador que solicita um tamanho de imagem intermediário pode receber a camada 723 e outra(s) camada(s) intermediária(s), mas não a camada mais alta 724 e, portanto, não todo o fluxo de bits. A mesma abordagem também pode ser usada para outras características de camada, tais como taxa de quadro, resolução de imagem, etc.

[0108] Um processo de extração de subfluxo de bits 729 é empregado para extrair um subfluxo de bits 701 do fluxo de bits 700 para suportar a

funcionalidade descrita acima. Um subfluxo de bits 701 é um subconjunto das unidades NAL (por exemplo, unidades não VCL NAL 742 e unidades VCL NAL 741) do fluxo de bits 700. Especificamente, um subfluxo de bits 701 pode conter dados relacionados a uma ou mais camadas, mas não dados relacionados a outras camadas. No exemplo mostrado, o subfluxo de bits 701 contém dados relacionados à camada 723 mas não dados relacionados à camada 724. Portanto, o subfluxo de bits 701 contém SPSs 713, PPS 715, cabeçalhos de fatia 717 e dados de imagem 720 incluindo camada 723, imagens 725 e fatias 727. O processo de extração de subfluxo de bits 729 remove unidades NAL com base em nuh_layer_id 732. Por exemplo, unidades VCL NAL 741 e unidades não VCL NAL 742 associadas apenas à camada mais alta 724 incluem valores mais altos de nuh_layer_id 732 e, portanto, a remoção de todas as unidades NAL com os valores mais altos de nuh_layer_id 732 extrai a camada mais baixa 723 e parâmetros associados. Cada unidade NAL contém um valor de nuh_layer_id 732 que é menor ou igual ao nuh_layer_id 732 da camada mais baixa que inclui a unidade NAL para suportar o processo de extração de subfluxo de bits 729. Deve-se observar que um fluxo de bits 700 e um subfluxo de bits 701 podem ser denominados, cada um, de modo geral, um fluxo de bits.

[0109] No exemplo mostrado, o subfluxo de bits 701 inclui uma camada simulcast (por exemplo, uma camada base). Conforme observado acima, a camada simulcast é qualquer camada que não usa predição de camada inter. O VPS 711 contém dados que descrevem a configuração de camadas 723 e 724. Entretanto, esses dados não são necessários para decodificar uma camada simulcast, tal como a camada 723. Como tal, o processo de extração de subfluxo de bits 729 remove o VPS 711 para suportar eficiência de codificação aumentada ao extrair uma camada simulcast. Isso pode causar problemas em alguns sistemas de codificação de vídeo. Especificamente, certos parâmetros no SPS 713 podem se referir ao VPS 711. Quando o VPS 711 é removido, o decodificador e/ou o HRD pode não ser capaz de solucionar tais parâmetros, visto que os dados referenciados por tais parâmetros não estão mais presentes. Isso pode resultar em um erro ao realizar teste de conformidade no HRD em camadas simulcast. De maneira alternativa, isso pode resultar em erros imprevisíveis em um decodificador quando uma camada simulcast é transmitida para exibição no decodificador.

[0110] A presente revelação aborda esses erros. Especificamente, o SPS 713 inclui um sps_video_parameter_set_id 731. O sps_video_parameter_set_id 731 é um elemento de sintaxe que especifica um ID de um VPS 711 referenciado pelo SPS 713. Especificamente, o VPS 711 contém um vps_video_parameter_set_id 735, que é um elemento de sintaxe que fornece um ID para o VPS 711 para referência por outras estruturas/elementos de sintaxe. Quando o VPS 711 estiver presente, o sps_video_parameter_set_id 731 é definido como o valor de vps_video_parameter_set_id 735. Entretanto, quando o SPS 713 for usado para uma camada simulcast, o sps_video_parameter_set_id 731 é definido como zero. Estabelecido de maneira diferente, o sps_video_parameter_set_id 731, quando maior que zero, especifica o valor de vps_video_parameter_set_id 735 para o VPS 711 referido pelo SPS 713. Quando o sps_video_parameter_set_id 731 for igual a zero, o SPS 713 não se refere a um VPS 711, e nenhum VPS 711 é referido ao decodificar qualquer sequência de vídeo de camada codificada que se refere ao SPS 713. Isso pode ser alcançado usando-se um SPS separado 713 para camadas diferentes (por exemplo, um SPS para uma camada simulcast e outro SPS para uma camada não simulcast(s)) ou mudando-se o valor de sps_video_parameter_set_id 731 durante o processo de extração de subfluxo de bits 729. Desse modo, o sps_video_parameter_set_id 731 não referencia, de maneira errônea, um ID que está indisponível quando o VPS 711 é removido durante o processo de extração de subfluxo de bits 729.

[0111] Adicionalmente, várias variáveis que são derivadas pelo HRD e/ou pelo decodificador também referenciam parâmetros no VPS 711. Assim, tais variáveis são definidas para valores padrão quando sps_video_parameter_set_id 731 for definido como zero. Isso garante que tais variáveis possam ser solucionadas de maneira apropriada para um valor acionável quando o VPS 711 é extraído para camadas simulcast, enquanto ainda operam corretamente para fluxos de bits de múltiplas camadas. Por exemplo, um decodificador e/ou um HRD pode derivar um GeneralLayerIdx[i] com base em um fluxo de bits 700 e/ou um subfluxo de bits 701. Um GeneralLayerIdx[i] é uma variável derivada que especifica um índice de uma camada correspondente i. Como tal, o GeneralLayerIdx[i] pode ser empregado para determinar um índice de camada de uma camada atual incluindo-se o nuh_layer_id 732 da camada atual como camada i em GeneralLayerIdx[i]. Isso pode ser expresso como um índice de

camada geral que corresponde a um nuh_layer_id (GeneralLayerIdx[nuh_layer_id]). Portanto, o GeneralLayerIdx[nuh_layer_id] indica um índice de camada atual para uma camada correspondente. Esse processo funciona de maneira correta para camadas não simulcast, tais como a camada 724, mas pode causar erros para a camada simulcast 723. Assim, o GeneralLayerIdx[nuh_layer_id] é definido como e/ou é inferido como sendo zero quando o sps_video_parameter_set_id 731 for zero (indicando uma camada simulcast).

[0112] Conforme outro exemplo, o VPS 711 pode conter uma flag de camada independente de VPS (vps_independent_layer_flag) 733. A vps_independent_layer_flag 733 especifica se camadas correspondentes, tais como a camada 723 e/ou 724 usam predição de camada inter. Assim, vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se uma camada atual com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. Entretanto, o VPS 711 contendo a vps_independent_layer_flag 733 não é enviado ao decodificador quando a camada 723 enviada ao decodificador for uma camada simulcast. Como tal, a referência pode causar um erro. Entretanto, camadas simulcast não usam predição de camada inter. Como tal, a vps_independent_layer_flag 733 para uma camada simulcast pode ser inferida como sendo igual a um, que indica que nenhuma predição de camada inter é usada para uma camada correspondente 723. Portanto, vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] é definida como/inferida como sendo um para indicar que predição de camada inter não é usada para a camada atual quando o sps_video_parameter_set_id for definido como zero. Desse modo, erros são evitados quando um VPS é removido de um fluxo de bits antes da transmissão de uma camada simulcast, tal como a camada 723. Como resultado, a funcionalidade do codificador e do decodificador é aumentada. Além disso, a eficiência de codificação é aumentada removendo-se, de maneira bem-sucedida, um VPS desnecessário de um fluxo de bits incluindo apenas uma camada simulcast, que reduz uso de recurso de processador, memória e/ou sinalização de rede tanto no codificador quanto no decodificador.

[0113] As informações precedentes são descritas em mais detalhes abaixo na presente invenção. A codificação de vídeo em camadas também é denominada codificação de vídeo escalável ou codificação de vídeo com

escalabilidade. A escalabilidade em codificação de vídeo pode ser suportada utilizando-se técnicas de codificação de múltiplas camadas. Um fluxo de bits de múltiplas camadas compreende uma camada de base (BL) e uma ou mais camadas de aprimoramento (ELs). Exemplo de escalabilidades inclui escalabilidade espacial, escalabilidade de qualidade / razão entre sinal e ruído (SNR), escalabilidade de múltiplas vistas, escalabilidade de taxa de quadro, etc. Quando uma técnica de codificação de múltiplas camadas é utilizada, uma imagem ou uma parte da mesma pode ser codificada sem usar uma imagem de referência (predição intra), pode ser codificada referindo-se a imagens de referência que estão na mesma camada (predição inter) e/ou pode ser codificada referindo-se a imagens de referência que estão em outra(s) camada(s) (predição de camada inter). Uma imagem de referência usada para predição de camada inter da imagem atual é denominada uma imagem de referência de camada inter (ILRP). A Figura 6 ilustra um exemplo de codificação de múltiplas camadas para escalabilidade espacial em que imagens em diferentes camadas têm diferentes resoluções.

[0114] Algumas famílias de codificação de vídeo fornecem suporte para escalabilidade em perfil(is) separado(s) do(s) perfil(is) para codificação de camada única. A codificação de vídeo escalável (SVC) é uma extensão escalável da codificação de vídeo avançada (AVC) que fornece suporte para escalabilidades espacial, temporal e de qualidade. Para SVC, uma flag é sinalizada em cada macrobloco (MB) em imagens EL para indicar se o EL MB é predito utilizando o bloco colocalizado a partir de uma camada mais baixa. A predição a partir do bloco colocalizado pode incluir textura, vetores de movimento e/ou modos de codificação. As implementações de SVC podem não reutilizar diretamente implementações de AVC não modificadas em seu projeto. A sintaxe de macrobloco de SVC EL e processo de decodificação diferem da sintaxe de AVC e processo de decodificação.

[0115] HEVC Escalável (SHVC) é uma extensão de HEVC que fornece suporte para escalabilidades espacial e de qualidade. HEVC de múltiplas vistas (MV-HEVC) é uma extensão de HEVC que fornece suporte para escalabilidade de múltiplas vistas. HEVC 3D (3D-HEVC) é uma extensão de HEVC que fornece suporte para codificação de vídeo 3D que é mais avançada e mais eficiente do que MV-HEVC. A escalabilidade temporal pode ser incluída como uma parte

integral de um codec HEVC de única camada. Na extensão de múltiplas camadas de HEVC, imagens decodificadas usadas para predição de camada inter são provenientes apenas da mesma AU e são tratadas como imagens de referência de longo prazo (LTRPs). Atribui-se, a tais imagens, índices de referência na(s) lista(s) de imagens de referência em conjunto com outras imagens de referência temporais na camada atual. A predição de camada inter (ILP) é alcançada no nível de unidade de predição (PU) definindo-se o valor do índice de referência para se referir à(s) imagem(ns) de referência de camada inter na(s) lista(s) de imagens de referência. A escalabilidade espacial reamostra (resamples) uma imagem de referência ou parte da mesma quando uma ILRP tiver uma resolução espacial diferente da imagem atual sendo codificada ou decodificada. A reamostragem de imagem de referência pode ser realizada no nível de imagem ou no nível de bloco de codificação.

[0116] VVC também pode suportar codificação de vídeo em camadas. Um fluxo de bits de VVC pode incluir múltiplas camadas. As camadas podem ser todas independentes umas das outras. Por exemplo, cada camada pode ser codificada sem usar predição de camada inter. Neste caso, as camadas também são denominadas camadas simulcast. Em alguns casos, algumas dentre as camadas são codificadas usando ILP. Uma flag no VPS pode indicar se as camadas são camadas simulcast ou se algumas camadas usam ILP. Quando algumas camadas utilizarem ILP, a relação de dependência de camada entre camadas também é sinalizada no VPS. Diferentemente de SHVC e MV-HEVC, VVC pode não especificar OLSs. Um OLS inclui um conjunto especificado de camadas, em que uma ou mais camadas, no conjunto de camadas, são especificadas para ser camadas de saída. Uma camada de saída é uma camada de um OLS que é emitida. Em algumas implementações de VVC, apenas uma camada pode ser selecionada para decodificação e emissão quando as camadas forem camadas simulcast. Em algumas implementações de VVC, todo o fluxo de bits incluindo todas as camadas é especificado para ser decodificado quando qualquer camada usar ILP. Além disso, certas camadas entre as camadas são especificadas para ser camadas de saída. As camadas de saída podem ser indicadas para ser apenas a camada mais alta, todas as camadas ou a camada mais alta mais um conjunto de camadas inferiores indicadas.

[0117] Os aspectos antecedentes contêm certos problemas

relacionados à escalabilidade. O projeto de escalabilidade em tais sistemas inclui perfil, tier e nível (PTL) específicos de camada, bem como operações de buffer de imagem codificada (CPB) específico de camada. A eficiência de sinalização de PTL deve ser aperfeiçoada. A eficiência de sinalização para parâmetros de HRD de nível de sequência para subcamadas deve ser aperfeiçoada. A sinalização de parâmetros de DPB deve ser aperfeiçoada. Alguns projetos fazem com que os fluxos de bits de única camada se refiram a VPSs. A faixa de valores do num_ref_entries[ ][ ] em tais projetos é incorreta e causa erros inesperados para decodificadores. O processo de decodificação em tais projetos envolve extração de subfluxo de bits, que acrescenta uma carga a implementações de decodificador. O processo de decodificação geral para tais projetos pode não funcionar para fluxos de bits escaláveis contendo múltiplas camadas com predição de camada inter. A derivação do valor da variável NoOutputOfPriorPicsFlag em tais projetos pode ser com base em imagem e não com base em AU em tais projetos. A mensagem SEI de aninhamento escalável em tais projetos deve ser simplificada para se aplicar diretamente a OLSs, em vez de camadas de OLSs, quando nesting_ols_flag for igual a um. Uma mensagem SEI aninhada não escalável, quando payloadType for igual a zero (período de buffering), um (temporização de imagem) ou cento e trinta (informações de unidade de decodificação), pode ser especificada para se aplicar apenas ao 0^{ésimo} OLS.

[0118] Em geral, esta revelação descreve várias abordagens para escalabilidade em codificação de vídeo. As descrições das técnicas se baseiam em VVC. Entretanto, as técnicas também se aplicam à codificação de vídeo em camadas com base em outras especificações de codec de vídeo. Um ou mais dos problemas supracitados podem ser solucionados do modo a seguir. Especificamente, esta revelação inclui métodos para suporte de escalabilidade aperfeiçoado em codificação de vídeo.

[0119] O conteúdo a seguir são várias definições exemplificativas. Um OP pode ser um subconjunto temporal de um OLS, identificado por um índice de OLS e um valor mais alto possível de TemporalId. Uma camada de saída pode ser uma camada de um OLS que é emitida. Um OLS pode ser um conjunto de camadas, em que uma ou mais camadas no conjunto de camadas são especificadas para ser camadas de saída. Um índice de camada de OLS pode ser um índice, de uma camada em um OLS, na lista de camadas no OLS. Um

processo de extração de subfluxo de bits pode ser um processo especificado através do qual unidades NAL, em um fluxo de bits, que não pertencem a um conjunto alvo, determinado por um índice de OLS alvo e um TemporalId alvo mais alto possível, são removidas do fluxo de bits, com o subfluxo de bits de saída compreendendo as unidades NAL, no fluxo de bits, que pertencem ao conjunto alvo.

[0120] Uma sintaxe de RBSP de conjunto de parâmetros de vídeo exemplificativa é da maneira a seguir.

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( )	
}	

[0121] Uma sintaxe de RBSP de conjunto de parâmetros de sequência exemplificativa é da maneira a seguir.

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)	
for(i = (sps_sub_layer_ordering_info_present_flag ? 0 : sps_max_sub_layers_minus1); i <= sps_max_sub_layers_minus1; i++) {	
sps_max_dec_pic_buffering_minus1[i]	ue(v)
sps_max_num_reorder_pics[i]	ue(v)
sps_max_latency_increase_plus1[i]	ue(v)
}	
long_term_ref_pics_flag	u(1)

...	
sps_scaling_list_enabled_flag	u(1)
general_hrd_parameters_present_flag	u(1)
if(general_hrd_parameters_present_flag) {	
num_units_in_tick	u(32)
time_scale	u(32)
sub_layer_cpb_parameters_present_flag	u(1)
if(sub_layer_cpb_parameters_present_flag)	
general_hrd_parameters(0, sps_max_sub_layers_minus1)	
else	
general_hrd_parameters(sps_max_sub_layers_minus1, sps_max_sub_layers_minus1)	
}	
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( )	
}	

[0122] Uma sintaxe de parâmetros de DPB exemplificativa é da maneira a seguir.

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)

}	
}	
}	

[0123] Uma sintaxe de parâmetros de HRD geral exemplificativa é da maneira a seguir.

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)
decoding_unit_cpb_params_in_pic_timing_sei_flag	u(1)
}	
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)
}	

[0124] Uma sintaxe de parâmetros de OLS HRD exemplificativa é da maneira a seguir.

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)	
}	
}	

[0125] Uma sintaxe de parâmetros de HRD de subcamada exemplificativa é da maneira a seguir.

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)
}	
}	

[0126] Uma semântica de RBSP de conjunto de parâmetros de vídeo exemplificativa é da maneira a seguir. Um vps_max_layers_minus1 mais 1 especifica o número máximo permitido de camadas em cada CVS que se refere ao VPS. Um vps_max_sub_layers_minus1 mais 1 especifica o número máximo de subcamadas temporais que podem estar presentes em cada CVS que se refere ao VPS. O valor de vps_max_sub_layers_minus1 pode estar na faixa de zero a seis, inclusive. Uma vps_all_layers_same_num_sub_layers_flag igual a um especifica que o número de subcamadas temporais é o mesmo para todas as camadas em cada CVS que se refere ao VPS. Uma vps_all_layers_same_num_sub_layers_flag igual a zero especifica que as camadas em cada CVS que se refere ao VPS podem ou não ter o mesmo número de subcamadas temporais. Quando não presente, o valor de vps_all_layers_same_num_sub_layers_flag pode ser inferido como sendo igual a um. Uma vps_all_independent_layers_flag igual a um especifica que todas as camadas na CVS são codificadas de maneira independente sem usar predição de camada inter. Uma vps_all_independent_layers_flag igual a zero especifica que uma ou mais das camadas na CVS podem usar predição de camada inter. Quando não presente, o valor de vps_all_independent_layers_flag pode ser inferido como sendo igual a um. Quando vps_all_independent_layers_flag for igual a um, o valor de vps_independent_layer_flag[i] é inferido como sendo igual a um. Quando vps_all_independent_layers_flag for igual a zero, o valor de vps_independent_layer_flag[0] é inferido como sendo igual a um.

[0127] Uma vps_direct_dependency_flag[i][j] igual a zero especifica que a camada com índice j não é uma camada de referência direta para a camada com índice i. Uma vps_direct_dependency_flag [i][j] igual a um 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 estiver presente para i e j na faixa de zero a vps_max_layers_minus1, inclusive, a flag é inferida como sendo igual a zero. A variável DirectDependentLayerIdx[i][j], que especifica a j^{ésima} camada dependente direta da i^{ésima} camada, e a variável LayerUsedAsRefLayerFlag[j], que especifica se a camada com índice de camada j é usada como uma camada de referência por qualquer outra camada, podem ser derivadas da seguinte maneira:

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
                        }
```

[0128] A variável GeneralLayerIdx[i], que especifica o índice de camada da camada com nuh_layer_id igual a vps_layer_id[i], pode ser derivada da maneira a seguir.

```
for( i = 0; i <= vps_max_layers_minus1; i++ )
        GeneralLayerIdx[ vps_layer_id[ i ] ] = i
```

[0129] Uma each_layer_is_an_ols_flag igual a um especifica que cada conjunto de camadas de saída contém apenas uma camada e cada camada por si só no fluxo de bits é um conjunto de camadas de saída com a única camada incluída sendo a única camada de saída. Uma each_layer_is_an_ols_flag igual a zero especifica que um conjunto de camadas de saída pode conter mais de uma camada. Se vps_max_layers_minus1 for igual a zero, o valor da each_layer_is_an_ols_flag é inferido como sendo igual a um. De outro modo, quando vps_all_independent_layers_flag for igual a zero, o valor de each_layer_is_an_ols_flag é inferido como sendo igual a zero.

[0130] Um ols_mode_idc igual a zero 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 zero a i, inclusive, e, para cada OLS, apenas a camada mais alta no OLS é emitida. Um ols_mode_idc igual a um 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 zero a i, inclusive, e, para cada OLS, todas as camadas no OLS são emitidas. Um ols_mode_idc igual a dois 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 mais baixas no OLS são emitidos. O valor de ols_mode_idc pode estar em uma faixa de zero a dois, inclusive. Quando vps_all_independent_layers_flag for igual a um e

each_layer_is_an_ols_flag for igual a zero, o valor de ols_mode_idc é inferido como sendo igual a dois. Um num_output_layer_sets_minus1 mais 1 especifica o número total de OLSs especificados pelo VPS quando ols_mode_idc for igual a dois.

[0131] A variável TotalNumOlss, que especifica o número total de OLSs especificados pelo VPS, pode ser derivada da seguinte maneira:

```
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
```

[0132] Uma layer_included_flag[i][j] especifica se a j^{ésima} camada (por exemplo, a camada com nuh_layer_id igual a vps_layer_id[j]) é incluída no i^{ésimo} OLS quando ols_mode_idc for igual a dois. Uma layer_included_flag[i][j] igual a um especifica que a j^{ésima} camada é incluída no i^{ésimo} OLS. Uma layer_included_flag[i][j] igual a zero especifica que a j^{ésima} camada não é incluída no i^{ésimo} OLS. A variável NumLayersInOls[i], que especifica o número de camadas no i^{ésimo} OLS, e a variável LayerIdInOls[i][j], que especifica o valor de nuh_layer_id da j^{ésima} camada no i^{ésimo} OLS, podem ser derivadas da seguinte maneira:

```
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
        }
}
```

[0133] A variável OlsLayeIdx[i][j], que especifica o índice de camada de OLS da camada com nuh_layer_id igual a LayerIdInOls[i][j], pode ser derivada da seguinte maneira:

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

[0134] A camada mais baixa possível em cada OLS deve ser uma camada independente. Em outras palavras, para cada i na faixa de zero a TotalNumOlss − 1, inclusive, o valor de vps_independent_layer_flag[GeneralLayerIdx[LayerIdInOls[i][0]]] deve ser igual a um. 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] para k na faixa de zero 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 zero a TotalNumOlss − 1, inclusive, e j está na faixa de NumLayersInOls[i] − 1, inclusive, de modo que o valor de LayerIdInOls[i][j] seja igual a nuhLayerId. 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.

[0135] Uma vps_output_layer_flag[i][j] especifica se a j^{ésima} camada no i^{ésimo} OLS é emitida quando ols_mode_idc for igual a dois. Uma vps_output_layer_flag[i] igual a um especifica que a j^{ésima} camada no i^{ésimo} OLS é emitida. Uma vps_output_layer_flag[i] igual a zero especifica que a j^{ésima} camada no i^{ésimo} OLS não é emitida. Quando vps_all_independent_layers_flag for igual a um e each_layer_is_an_ols_flag for igual a zero, o valor de vps_output_layer_flag[i] é inferido como sendo igual a um. A variável OutputLayerFlag[i][j], para a qual o valor um especifica que a j^{ésima} camada no i^{ésimo} OLS é emitida e o valor zero especifica que a j^{ésima} camada no i^{ésimo} OLS não é emitida, pode ser derivada da maneira a seguir.

```
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 ]
}
```

[0136] Deve-se observar que um 0ésimo OLS contém apenas a camada mais baixa (por exemplo, a camada com nuh_layer_id igual a vps_layer_id[0]) e, para o 0ésimo OLS, a única camada incluída é emitida. Um vps_num_ptls especifica o número de estruturas de sintaxe profile_tier_level( ) no VPS. Uma pt_present_flag[i] igual a um especifica que perfil, tier e informações de restrições gerais estão presentes na iésima estrutura de sintaxe profile_tier_level( ) no VPS. Uma pt_present_flag[i] igual a zero especifica que perfil, tier e informações de restrições gerais não estão presentes na iésima estrutura de sintaxe profile_tier_level( ) no VPS. O valor de pt_present_flag[0] é inferido como sendo igual a zero. Quando pt_present_flag[i] for igual a zero, infere-se que o perfil, tier e informações de restrições gerais para a iésima estrutura de sintaxe profile_tier_level( ) no VPS sejam iguais àqueles para a (i − 1)ésima estrutura de sintaxe profile_tier_level( ) no VPS.

[0137] Um ptl_max_temporal_id[i] especifica o TemporalId da representação de subcamada mais alta para a qual as informações de nível estão presentes na iésima estrutura de sintaxe profile_tier_level( ) no VPS. O valor de ptl_max_temporal_id[i] deve estar na faixa de zero a vps_max_sub_layers_minus1, inclusive. Quando vps_max_sub_layers_minus1 for igual a zero, o valor de ptl_max_temporal_id[i] é inferido como sendo igual a zero. Quando vps_max_sub_layers_minus1 for maior que zero e vps_all_layers_same_num_sub_layers_flag for igual a um, o valor de ptl_max_temporal_id[i] é inferido como sendo igual a vps_max_sub_layers_minus1. Um vps_ptl_byte_alignment_zero_bit deve ser igual a zero.

[0138] Um ols_ptl_idx[i] especifica o índice, na 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 zero a vps_num_ptls − 1, inclusive. Quando NumLayersInOls[i] for igual a um, 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. Um 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 zero a dezesseis, inclusive. Quando não presente, o valor de vps_num_dpb_params pode ser inferido como sendo igual a zero. Uma same_dpb_size_output_or_nonoutput_flag igual a um especifica que não há elemento de sintaxe layer_nonoutput_dpb_params_idx[i] presente no VPS. Uma same_dpb_size_output_or_nonoutput_flag igual a zero especifica que pode haver ou não elementos de sintaxe layer_nonoutput_dpb_params_idx[i] presentes no VPS. Uma vps_sub_layer_dpb_params_present_flag é usada 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 presente, vps_sub_dpb_params_info_present_flag é inferida como sendo igual a zero.

[0139] Uma dpb_size_only_flag[i] igual a um especifica que os elementos de sintaxe max_num_reorder_pics[ ] e max_latency_increase_plus1[ ] não estão presentes nas i^{ésimas} estruturas de sintaxe dpb_parameters( ) o VPS. Uma dpb_size_only_flag[i] igual a um especifica que os elementos de sintaxe max_num_reorder_pics[ ] e max_latency_increase_plus1[ ] podem estar presentes nas i^{ésimas} estruturas de sintaxe dpb_parameters( ) o VPS. Um 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 zero a vps_max_sub_layers_minus1, inclusive. Quando vps_max_sub_layers_minus1 for igual a zero, o valor de dpb_max_temporal_id[i] pode ser inferido como sendo igual a zero. Quando vps_max_sub_layers_minus1 for maior que zero e vps_all_layers_same_num_sub_layers_flag for igual a um, o valor de dpb_max_temporal_id[i] é inferido como sendo igual a vps_max_sub_layers_minus1. Um layer_output_dpb_params_idx[i] especifica o

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

[0140] Se vps_independent_layer_flag[i] for igual a um, a estrutura de sintaxe dpb_parameters( ) que se aplica à i^{ésima} camada quando for uma camada de saída é a estrutura de sintaxe dpb_parameters( ) presente no SPS referido pela camada. De outro modo, (vps_independent_layer_flag[i] for igual a um), o conteúdo a seguir se aplica. Quando vps_num_dpb_params for igual a um, o valor de layer_output_dpb_params_idx[i] é inferido como sendo igual a zero. A conformidade de fluxo de bits pode exigir que o valor de layer_output_dpb_params_idx[i] seja de modo que dpb_size_only_flag[layer_output_dpb_params_idx[i]] seja igual a zero.

[0141] Um layer_nonoutput_dpb_params_idx[i] especifica o índice, na lista de estruturas de sintaxe dpb_parameters( ) no VPS, da estrutura de sintaxe dpb_parameters( ) que se aplica à i^{ésima} camada quando a i^{ésima} camada for uma camada de não saída em um OLS. Quando presente, o valor de layer_nonoutput_dpb_params_idx[i] deve estar na faixa de zero a vps_num_dpb_params − 1, inclusive. Se same_dpb_size_output_or_nonoutput_flag for igual a 1, o conteúdo a seguir se aplica. Se vps_independent_layer_flag[i] for igual a um, a estrutura de sintaxe dpb_parameters( ) que se aplica à i^{ésima} camada quando a i^{ésima} camada for uma camada de não saída é a estrutura de sintaxe dpb_parameters( ) presente no SPS referido pela camada. De outro modo (vps_independent_layer_flag[i] é igual a um), o valor de layer_nonoutput_dpb_params_idx[i] é inferido como sendo igual a layer_output_dpb_params_idx[i]. De outro modo (same_dpb_size_output_or_nonoutput_flag é igual a zero), quando vps_num_dpb_params for igual a um, o valor de layer_output_dpb_params_idx[i] é inferido como sendo igual a zero.

[0142] Uma general_hrd_params_present_flag igual a um especifica que os elementos de sintaxe num_units_in_tick e time_scale e a estrutura de sintaxe general_hrd_parameters( ) estão presentes na estrutura de sintaxe SPS RBSP. Uma general_hrd_params_present_flag igual a zero especifica que os

elementos de sintaxe num_units_in_tick e time_scale e a estrutura de sintaxe general_hrd_parameters( ) não estão presentes na estrutura de sintaxe SPS RBSP. Um num_units_in_tick é o número de unidades de tempo de um relógio operando na frequência time_scale hertz (Hz) que corresponde a um incremento (denominado uma marcação de relógio) de um contador de marcação de relógio. Um num_units_in_tick deve ser maior que zero. Uma marcação de relógio, em unidades de segundos, é igual ao quociente de num_units_in_tick dividido por time_scale. Por exemplo, quando a taxa de imagem de um sinal de vídeo for vinte e cinco Hz, time_scale pode ser igual a 27.000.000 e num_units_in_tick pode ser igual a 1.080.000 e, consequentemente, uma marcação de relógio pode ser igual a 0,04 segundos. Um time_scale é o número de unidades de tempo que passam em um segundo. Por exemplo, um sistema de coordenadas de tempo que mede o tempo usando um relógio de 27 MHz tem um time_scale de 27.000.000. O valor de time_scale deve ser maior que zero.

[0143] Uma vps_extension_flag igual a zero especifica que nenhum elemento de sintaxe vps_extension_data_flag está presente na estrutura de sintaxe VPS RBSP. Uma vps_extension_flag igual a um especifica que há elementos de sintaxe vps_extension_data_flag presentes na estrutura de sintaxe VPS RBSP. Uma vps_extension_data_flag pode ter qualquer valor. A presença e valor de vps_extension_data_flag podem não afetar a conformidade de decodificador a perfis. Os decodificadores em conformidade podem ignorar todos os elementos de sintaxe vps_extension_data_flag.

[0144] Uma semântica de RBSP de conjunto de parâmetros de sequência exemplificativa é da maneira a seguir. Uma SPS RBSP deve estar disponível ao processo de decodificação antes de ser referenciada, incluída em pelo menos uma unidade de acesso com TemporalId igual a zero, ou fornecida através de meios externos, e a unidade SPS NAL contendo a SPS RBSP deve ter nuh_layer_id igual ao valor mais baixo de nuh_layer_id de unidades PPS NAL que se referem à unidade SPS NAL. Todas as unidades SPS NAL com um valor particular de sps_seq_parameter_set_id em uma CVS devem ter o mesmo conteúdo. Um sps_decoding_parameter_set_id, quando maior que zero, especifica o valor de dps_decoding_parameter_set_id para o DPS referido pelo SPS. Quando sps_decoding_parameter_set_id for igual a zero, o SPS não se refere a um DPS e nenhum DPS é referido ao decodificar cada CLVS que se refere

ao SPS. O valor de sps_decoding_parameter_set_id deve ser o mesmo em todos os SPSs que são referidos por imagens codificadas em um fluxo de bits.

[0145] Um sps_video_parameter_set_id, quando maior que zero, especifica o valor de vps_video_parameter_set_id para o VPS referido pelo SPS. Quando sps_video_parameter_set_id for igual a zero, o SPS pode não se referir a um VPS e nenhum VPS é referido ao decodificar cada CLVS que se refere ao SPS, e o valor de GeneralLayerIdx[nuh_layer_id] deve ser inferido como sendo igual a zero, e o valor de vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] pode ser inferido como sendo igual a um. Quando vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] for igual a um, o SPS referido por uma CLVS com um valor de nuh_layer_id particular nuhLayerId deve ter nuh_layer_id igual a nuhLayerId.

[0146] Um sps_max_sub_layers_minus1 mais 1 especifica o número máximo de subcamadas temporais que podem estar presentes em cada CLVS que se refere ao SPS. O valor de sps_max_sub_layers_minus1 deve estar na faixa de zero a vps_max_sub_layers_minus1, inclusive. Um sps_reserved_zero_4bits deve ser igual a zero em fluxos de bits em conformidade. Outros valores para sps_reserved_zero_4bits podem ser reservados.

[0147] Uma sps_ptl_dpb_present_flag igual a um especifica que uma estrutura de sintaxe profile_tier_level( ) e uma estrutura de sintaxe dpb_parameters( ) estão presentes no SPS. Uma sps_ptl_dpb_present_flag igual a zero especifica que nenhuma estrutura de sintaxe profile_tier_level( ) e nenhuma estrutura de sintaxe dpb_parameters( ) está presente no SPS. O valor de sps_ptl_dpb_present_flag deve ser igual a vps_independent_layer_flag[nuh_layer_id]. Se vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] for igual a um, a variável MaxDecPicBuffMinus1 é definida 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 layer_nonoutput_dpb_params_idx[GeneralLayerIdx[nuh_layer_id]]ésima estrutura de sintaxe dpb_parameters( ) no VPS. Uma gdr_enabled_flag igual a um

especifica que imagens GDR podem estar presentes em CLVSs que se refere ao SPS. Uma gdr_enabled_flag igual a zero especifica que imagens GDR não estão presentes em CLVSs que se referem ao SPS.

[0148] Uma sps_sub_layer_dpb_params_flag é usada 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 sintaxe dpb_parameters( ) no SPS. Quando não presente, sps_sub_dpb_params_info_present_flag é inferida como sendo igual a zero. Uma long_term_ref_pics_flag igual a zero especifica que nenhuma LTRP é usada para predição inter de qualquer imagem codificada na CLVS. Uma long_term_ref_pics_flag igual a um especifica que LTRPs podem ser usadas para predição inter de uma ou mais imagens codificadas na CLVS.

[0149] Uma semântica de perfil, tier e nível geral exemplificativa é da maneira a seguir. Uma estrutura de sintaxe profile_tier_level( ) fornece informações de nível e, de maneira opcional, perfil, tier, subperfil e informações de restrições gerais (denotadas como informações PT). Quando a estrutura de sintaxe profile_tier_level( ) for 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( ) for incluída em um VPS, o OlsInScope é um ou mais OLSs especificados pelo VPS. Quando a estrutura de sintaxe profile_tier_level( ) for incluída em um SPS, o OlsInScope é o 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.

[0150] Um general_profile_idc indica um perfil ao qual OlsInScope está em conformidade. Uma general_tier_flag especifica o contexto de tier para a interpretação de general_level_idc. Um num_sub_profiles especifica o número dos elementos de sintaxe general_sub_profile_idc[i]. Um general_sub_profile_idc[i] indica os $i^{ésimos}$ metadados de interoperabilidade registrados. Um general_level_idc indica um nível ao qual OlsInScope está em conformidade. Deve-se observar que um valor maior de general_level_idc indica um nível mais alto. O nível máximo sinalizado no DPS para OlsInScope pode ser maior que o nível sinalizado no SPS para uma CVS contida dentro de OlsInScope. Deve-se observar também que, quando OlsInScope está em conformidade com múltiplos 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. Deve-se observar também que, quando a estrutura de sintaxe profile_tier_level( ) for incluída em um DPS e CVSs de OlsInScope estiverem em conformidade com diferentes perfis, general_profile_idc e level_idc deve indicar o perfil e nível para um decodificador que é capaz de decodificar OlsInScope.

[0151] Uma sub_layer_level_present_flag[i] igual a um especifica que informações de nível estão presentes na estrutura de sintaxe profile_tier_level( ) para a representação de subcamada com TemporalId igual a i. Uma sub_layer_level_present_flag[i] igual a zero especifica que informações de nível não estão presentes na estrutura de sintaxe profile_tier_level( ) para a representação de subcamada com TemporalId igual a i. Um ptl_alignment_zero_bits deve ser igual a zero. A semântica do elemento de sintaxe sub_layer_level_idc[i] é, além da especificação da inferência de valores não presentes, igual ao elemento de sintaxe general_level_idc, mas se aplica à representação de subcamada com TemporalId igual a i.

[0152] Uma semântica de parâmetros de DPB exemplificativa é da maneira a seguir. A estrutura de sintaxe dpb_parameters(maxSubLayersMinus1, subLayerInfoFlag) fornece informações de tamanho de DPB, número máximo de reordem de imagem e latência máxima para cada CLVS da CVS. Quando uma estrutura de sintaxe dpb_parameters( ) for 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( ) for incluída em um SPS, a estrutura de sintaxe dpb_parameters( ) 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.

[0153] Um max_dec_pic_buffering_minus1[i] mais 1 especifica, para cada para cada CLVS da CVS, o tamanho máximo exigido do buffer de imagem decodificada em unidades de buffers de armazenamento de imagem quando Htid for igual a i. O valor de max_dec_pic_buffering_minus1[i] deve estar na faixa de 0 a MaxDpbSize − 1, inclusive. Quando i for maior que zero, 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 estiver presente para i na faixa de zero a

maxSubLayersMinus1 − 1, inclusive, devido à subLayerInfoFlag ser igual a zero, max_dec_pic_buffering_minus1[i] é inferido como sendo igual a max_dec_pic_buffering_minus1[maxSubLayersMinus1].

[0154] Um max_num_reorder_pics[i] especifica, para cada CLVS da CVS, o número máximo permitido de imagens da CLVS que podem preceder qualquer imagem na CLVS em ordem de decodificação e suceder essa imagem em ordem de emissão quando Htid for igual a i. O valor de max_num_reorder_pics[i] deve estar na faixa de zero a max_dec_pic_buffering_minus1[i], inclusive. Quando i for maior que zero, 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 estiver presente para i na faixa de zero a maxSubLayersMinus1 − 1, inclusive, devido à subLayerInfoFlag ser igual a zero, max_num_reorder_pics[i] é inferido como sendo igual a max_num_reorder_pics[maxSubLayersMinus1].

[0155] Um max_latency_increase_plus1[i] diferente de zero é usado para computar o valor de MaxLatencyPictures[i], que especifica, para cada CLVS da CVS, o número máximo de imagens na CLVS que pode preceder qualquer imagem na CLVS em ordem de emissão e suceder essa imagem em ordem de decodificação quando Htid for igual a i. Quando max_latency_increase_plus1[i] não for igual a zero, o valor de MaxLatencyPictures[i] pode ser especificado da maneira a seguir.

MaxLatencyPictures[i] =

max_num_reorder_pics[i] + max_latency_increase_plus1[i] − 1

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

[0156] O valor de max_latency_increase_plus1[i] deve estar na faixa de zero a duzentos e trinta e dois menos dois, inclusive. Quando max_latency_increase_plus1[i] não estiver presente para i na faixa de zero a maxSubLayersMinus1 − 1, inclusive, devido à subLayerInfoFlag ser igual a zero, max_latency_increase_plus1[i] é inferido como sendo igual a max_latency_increase_plus1[maxSubLayersMinus1].

[0157] Uma semântica de parâmetros de HRD geral exemplificativa é da maneira a seguir. A estrutura de sintaxe general_hrd_parameters( ) fornece parâmetros de HRD usados nas operações de HRD. Um

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 zero a sessenta e três, inclusive. Quando TotalNumOlss for maior que um, o valor de num_ols_hrd_params_minus1 é inferido como sendo igual a zero. O hrd_cpb_cnt_minus1 mais 1 especifica o número de especificações de CPB alternativas no fluxo de bits da CVS. O valor de hrd_cpb_cnt_minus1 deve estar na faixa de zero a trinta e um, inclusive. Um 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 zero a vps_max_sub_layers_minus1, inclusive. Quando vps_max_sub_layers_minus1 for igual a zero, o valor de hrd_max_temporal_id[i] é inferido como sendo igual a zero. Um 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 zero a num_ols_hrd_params_minus1, inclusive. Quando não presente, o valor de ols_hrd_idx[[i] é inferido como sendo igual a zero.

[0158] Uma semântica de estrutura de lista de imagens de referência exemplificativa é da maneira a seguir. A estrutura de sintaxe ref_pic_list_struct(listIdx, rplsIdx) pode estar presente em um SPS ou em um cabeçalho de fatia. Dependendo de a estrutura de sintaxe ser incluída em um cabeçalho de fatia ou em um SPS, o conteúdo a seguir se aplica. Se presente em um cabeçalho de fatia, a estrutura de sintaxe ref_pic_list_struct(listIdx, rplsIdx) especifica lista de imagens de referência listIdx da imagem atual (a imagem contendo a fatia). 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 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 está em uma CVS que se refere ao SPS. Um 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 zero a

MaxDecPicBuffMinus1 + catorze, inclusive.

[0159] Um processo de decodificação geral exemplificativo é da maneira a seguir. Entrada para esse processo é um fluxo de bits BitstreamToDecode. Saída desse processo é uma lista de imagens decodificadas. O processo de decodificação é especificado de modo que todos os decodificadores que estão em conformidade com um perfil e nível especificados produzam imagens de saída decodificadas recortadas numericamente idênticas ao invocar o processo de decodificação associado àquele perfil para um fluxo de bits em conformidade com aquele perfil e nível. Qualquer processo de decodificação que produz imagens de saída decodificadas recortadas idênticas àquelas produzidas pelo processo descrito na presente invenção (com a ordem de emissão ou temporização de emissão corretas, conforme especificado) está em conformidade com as exigências de processo de decodificação.

[0160] Para cada IRAP AU no fluxo de bits, o conteúdo a seguir se aplica. Se a AU for a primeira AU no fluxo de bits em ordem de decodificação, cada imagem for uma imagem de atualização de decodificação instantânea (IDR) ou cada imagem for a primeira imagem da camada que sucede uma unidade NAL de fim de sequência em ordem de decodificação, a variável NoIncorrectPicOutputFlag é definida como igual a um. De outro modo, se a variável HandleCraAsCvsStartFlag for definida como um valor para a AU, HandleCraAsCvsStartFlag é definida como igual a um valor fornecido por um mecanismo externo e NoIncorrectPicOutputFlag é definida como igual a HandleCraAsCvsStartFlag. De outro modo, HandleCraAsCvsStartFlag e NoIncorrectPicOutputFlag são, ambas, definidas como iguais a zero.

[0161] Para cada AU de atualização de decodificação gradual (GDR) no fluxo de bits, o conteúdo a seguir se aplica. 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 sucede uma unidade NAL de fim de sequência em ordem de decodificação, a variável NoIncorrectPicOutputFlag é definida como igual a um. De outro modo, se algum mecanismo externo estiver disponível para definir a variável HandleGdrAsCvsStartFlag para um valor para a AU, HandleGdrAsCvsStartFlag é definida como igual ao valor fornecido pelo mecanismo externo e NoIncorrectPicOutputFlag é definida como igual a HandleGdrAsCvsStartFlag. De outro modo, HandleGdrAsCvsStartFlag e

NoIncorrectPicOutputFlag são, ambas, definidas como iguais a zero. As operações acima, tanto para imagens IRAP quanto para imagens GDR, são usadas para identificação das CVSs no fluxo de bits. A decodificação é repetidamente invocada para cada imagem codificada em BitstreamToDecode em ordem de decodificação.

[0162] Um processo de decodificação exemplificativo para construção de listas de imagem de referência é da maneira a seguir. Esse processo é invocado no início do processo de decodificação para cada fatia de uma imagem não IDR. As imagens de referência são endereçadas 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 (por exemplo, RefPicList[0]), é usada na decodificação dos dados de fatia. Ao decodificar uma fatia B, a lista de imagens de referência 0 e a lista de imagens de referência 1 (por exemplo, RefPicList[1]) são usadas na decodificação dos dados de fatia.

[0163] As restrições a seguir se aplicam para conformidade de fluxo de bits. Para cada i igual a zero ou um, num_ref_entries[i][RplsIdx[i]] não deve ser menor que NumRefIdxActive[i]. 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 àquele da imagem atual. 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 zero. Uma entrada de imagem de referência de curto prazo (STRP) na RefPicList[0] ou RefPicList[1] de uma fatia de uma imagem e uma entrada de imagem de referência de longo prazo (LTRP) na RefPicList[0] ou RefPicList[1] da mesma fatia ou uma fatia diferente da mesma imagem não devem se referir à mesma imagem. Não deve haver entrada de LTRP em RefPicList[0] ou RefPicList[1] para a qual a diferença entre o PicOrderCntVal da imagem atual e o PicOrderCntVal da imagem referida pela entrada é maior que ou igual a duzentos e vinte e quatro.

[0164] Supõe-se que setOfRefPics seja o conjunto de imagens exclusivas 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 igual para todas as fatias de uma imagem. Quando a imagem atual for uma imagem de acesso de subcamada temporal passo-a-passo (step-wise) (STSA), não deve haver entrada ativa na RefPicList[0] ou RefPicList[1] que tem TemporalId igual àquele da imagem atual. Quando a imagem atual for uma imagem que sucede, em ordem de decodificação, uma imagem STSA que tem TemporalId igual àquele da imagem atual, não deve haver imagem que tem TemporalId igual àquele da imagem atual incluída como uma entrada ativa na RefPicList[0] ou RefPicList[1] que precede a imagem STSA em ordem de decodificação.

[0165] A imagem referida por cada entrada de imagem de referência de camada inter (ILRP) na RefPicList[0] ou RefPicList[1] de uma fatia da imagem atual deve estar na mesma unidade de acesso que a imagem atual. A imagem referida por cada entrada de ILRP na RefPicList[0] ou RefPicList[1] de uma fatia da imagem atual deve estar presente no DPB e deve ter nuh_layer_id menor que aquele da imagem atual. Cada entrada de ILRP em RefPicList[0] ou RefPicList[1] de uma fatia deve ser uma entrada ativa.

[0166] Uma especificação de HRD exemplificativa é da maneira a seguir. O HRD é usado para verificar conformidade de decodificador e fluxo de bits. Um conjunto de testes de conformidade de fluxo de bits é usado para verificar a conformidade de um fluxo de bits, que é denominado todo o fluxo de bits, denotado como entireBitstream. O conjunto de testes de conformidade de fluxo de bits é para testar a conformidade de cada OP de cada OLS especificado pelo VPS.

[0167] Para cada teste, as etapas ordenadas a seguir se aplicam na ordem listada, seguido pelos processos descritos após essas etapas nesta cláusula. Um ponto de operação em teste, denotado como targetOp, é selecionado selecionando-se um OLS alvo com índice de OLS opOlsIdx e um valor mais alto possível de TemporalId opTid. O valor de opOlsIdx está na faixa de zero a TotalNumOlss − 1, inclusive. O valor de opTid está na faixa de zero a vps_max_sub_layers_minus1, inclusive. Cada par de valores selecionados de opOlsIdx e opTid deve ser de modo que o subfluxo de bits que é a saída invocando-se o processo de extração de subfluxo de bits com entireBitstream, opOlsIdx e opTid como entradas satisfaça as condições a seguir. 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. Há pelo menos uma unidade VCL NAL com TemporalId igual a opTid em BitstreamToDecode.

[0168] Se as camadas em targetOp incluírem todas as camadas no entireBitstream e opTid for igual a maior que o valor mais alto de TemporalId entre todas as unidades NAL em entireBitstream, BitstreamToDecode é definido como ser idêntico a entireBitstream. De outro modo, BitstreamToDecode é definido como ser a saída invocando-se o processo de extração de subfluxo de bits com entireBitstream, opOlsIdx e opTid como entradas. Os valores de TargetOlsIdx e Htid são definidos como iguais a opOlsIdx e opTid, respectivamente, de targetOp. Um valor de ScIdx é selecionado. O ScIdx selecionado deve estar na faixa de zero a hrd_cpb_cnt_minus1, inclusive. Uma unidade de acesso em BitstreamToDecode associada a mensagens SEI de período de buffering (presentes em TargetLayerBitstream ou disponíveis através de mecanismos externos) aplicáveis a TargetOlsIdx é selecionada como o ponto de inicialização de HRD e denominada unidade de acesso zero para cada camada no OLS alvo.

[0169] As etapas subsequentes se aplicam a cada camada com índice de camada de OLS TargetOlsLayerIdx no OLS alvo. A estrutura de sintaxe ols_hrd_parameters( ) e a estrutura de sintaxe sub_layer_hrd_parameters( ) aplicáveis a BitstreamToDecode são selecionadas da maneira a seguir. A ols_hrd_idx[TargetOlsIdx]^{ésima} estrutura de sintaxe ols_hrd_parameters( ) no VPS (ou fornecido através de um mecanismo externo) é selecionada. 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 sucede imediatamente a condição if(general_vcl_hrd_params_present_flag) é selecionada e a variável NalHrdModeFlag é definida como igual a zero. De outro modo (BitstreamToDecode é um fluxo de bits Tipo II), a estrutura de sintaxe sub_layer_hrd_parameters(Htid) que sucede imediatamente a condição if(general_vcl_hrd_params_present_flag) (neste caso, a variável NalHrdModeFlag é definida como igual a zero) 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 for um fluxo de bits Tipo II e NalHrdModeFlag for igual a zero, todas as unidades não VCL NAL exceto unidades NAL de dados de preenchimento, e todos os elementos de sintaxe leading_zero_8bits, zero_byte,

start_code_prefix_one_3bytes e trailing_zero_8bits que formam um fluxo de byte a partir do fluxo de unidade NAL, quando presente, são descartados de BitstreamToDecode e o fluxo de bits remanescente é atribuído a BitstreamToDecode.

[0170] Quando decoding_unit_hrd_params_present_flag for igual a um, o CPB é agendado para operar no nível de unidade de acesso (caso em que a variável DecodingUnitHrdFlag é definida como igual a zero) ou no nível de unidade de decodificação (caso em que a variável DecodingUnitHrdFlag é definida como igual a um). De outro modo, DecodingUnitHrdFlag é definida como igual a zero e o CPB é agendado para operar no nível de unidade de acesso.

[0171] Para cada unidade de acesso em BitstreamToDecode iniciando a partir de unidade de acesso zero, a mensagem SEI de período de buffering (presente em BitstreamToDecode ou disponível através de mecanismos externos) que é associada à unidade de acesso e se aplica a TargetOlsIdx é selecionada, a mensagem SEI de temporização de imagem (presente em BitstreamToDecode ou disponível através de mecanismos externos) que é associada à unidade de acesso e se aplica a TargetOlsIdx é selecionada e, quando DecodingUnitHrdFlag for igual a um e decoding_unit_cpb_params_in_pic_timing_sei_flag for igual a zero, as mensagens SEI de informações de unidade de decodificação (presentes em BitstreamToDecode ou disponíveis através de mecanismos externos) que são associadas às unidades de decodificação na unidade de acesso e se aplicam a TargetOlsIdx são selecionadas.

[0172] Cada teste de conformidade inclui 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 todo o conjunto de testes de conformidade. Para cada ponto de operação em teste, o número de testes de conformidade de fluxo de bits a ser realizado é igual a n0 * n1 * n2 * n3, em que os valores de n0, n1, n2 e n3 são especificados da maneira a seguir. n1 é igual a hrd_cpb_cnt_minus1 + 1. n1 é o número de unidades de acesso em BitstreamToDecode que são associadas a mensagens SEI de período de buffering. n2 é derivado da maneira a seguir. Se BitstreamToDecode for um fluxo de bits Tipo I, n0 é igual a um. De outro modo (BitstreamToDecode é um fluxo de bits Tipo II), n0 é igual a dois. n3 é derivado da

maneira a seguir. Se decoding_unit_hrd_params_present_flag for igual a zero, n3 é igual a um. De outro modo, n3 é igual a dois.

[0173] O HRD contém um extrator de fluxo de bits (presente, de maneira opcional), um buffer de imagem codificada (CPB), um processo de decodificação instantânea, um buffer de imagem decodificada (DPB) que contém, conceitualmente, um sub-DPB para cada camada, e cropping de saída. Para cada teste de conformidade de fluxo de bits, o tamanho de CPB (número de bits) é CpbSize[Htid][ScIdx], e parâmetros de DPB max_dec_pic_buffering_minus1[Htid], max_num_reorder_pics[Htid] e MaxLatencyPictures[Htid] para cada camada são encontrados em ou derivados da estrutura de sintaxe dpb_parameters( ) que se aplica à camada dependendo de a camada ser uma camada independente e de a camada ser uma camada de saída do OLS alvo.

[0174] O HRD pode operar da maneira a seguir. O HRD é inicializado na unidade de decodificação zero, com o CPB e cada sub-DPB do DPB sendo definidos para estarem vazios (a totalidade de sub-DPB para cada sub-DPB é definida como igual a zero). Após inicialização, o HRD pode não ser inicializado novamente por mensagens SEI de período de buffering subsequentes. Os dados associados a unidades de decodificação que fluem em cada CPB de acordo com um agendamento de chegada especificado são entregues pelo agendador de fluxo hipotético (HSS). Os dados associados a cada unidade de decodificação são removidos e decodificados de modo instantâneo pelo processo de decodificação instantânea no tempo de remoção de CPB da unidade de decodificação. Cada imagem decodificada é colocada no DPB. Uma imagem decodificada é removida do DPB quando a imagem decodificada não for mais necessária para referência de predição inter e não mais necessária para emissão.

[0175] Uma operação exemplificativa do buffer de imagem decodificada é da maneira a seguir. Essas especificações podem se aplicar de maneira independente para cada conjunto de parâmetros de buffer de imagem decodificada (DPB) selecionado. O buffer de imagem decodificada inclui, conceitualmente, 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 posterior. Os processos descritos na presente invenção são aplicados de maneira sequencial, e são aplicados de maneira independente para cada camada, iniciando a partir da camada mais baixa no OLS, em ordem crescente de valores de nuh_layer_id das camadas no OLS. Quando esses processos são aplicados para uma camada particular, apenas o sub-DPB para a camada particular é afetado. Nas descrições desses processos, o DPB se refere ao sub-DPB para a camada particular, e a camada particular é denominada a camada atual.

[0176] Na operação do DPB de temporização de emissão, imagens decodificadas com PicOutputFlag igual a um na mesma unidade de acesso são emitidas consecutivamente em ordem crescente dos valores de nuh_layer_id das imagens decodificadas. Supõe-se que a imagem n e a imagem atual sejam a imagem codificada ou imagem decodificada da unidade de acesso n para um valor particular de nuh_layer_id, em que n é um número inteiro não negativo. A remoção de imagens do DPB antes da decodificação da imagem atual ocorre da maneira a seguir. A remoção de imagens do DPB antes da decodificação da imagem atual (mas após analisar o cabeçalho de fatia da primeira fatia da imagem atual) ocorre de modo substancialmente instantâneo no tempo de remoção de CPB da primeira unidade de decodificação de unidade de acesso n (contendo a imagem atual) e prossegue da maneira a seguir.

[0177] O processo de decodificação para construção de lista de imagens de referência é invocado e o processo de decodificação para marcação de imagem de referência é invocado. Quando a AU atual for uma AU de início de sequência de vídeo codificada (CVSS) que não é AU zero, as etapas ordenadas a seguir são aplicadas. A variável NoOutputOfPriorPicsFlag é derivada para o decodificador em teste da maneira a seguir. 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] derivados para qualquer imagem na AU atual for 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, derivados para a imagem antecedente na mesma CLVS, NoOutputOfPriorPicsFlag pode ser definida como um pelo decodificador em teste, independentemente do valor de

no_output_of_prior_pics_flag. Embora a definição de NoOutputOfPriorPicsFlag como igual a no_output_of_prior_pics_flag possa ser preferencial sob essas condições, o decodificador em teste está permitido a definir NoOutputOfPriorPicsFlag para um neste caso. De outro modo, NoOutputOfPriorPicsFlag é definida como igual a no_output_of_prior_pics_flag.

[0178] 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 um, todos os buffers de armazenamento de imagem no DPB sejam esvaziados sem emissão das imagens contidas nos mesmos, e a totalidade de DPB é definida como igual a zero. Quando ambas as condições a seguir forem verdadeiras para quaisquer imagens k no DPB, todas as tais imagens k no DPB são removidas do DPB. A imagem k pode ser marcada como não usada para referência, ou a imagem k pode ter uma PictureOutputFlag igual a zero ou um tempo de emissão de DPB é menor ou igual ao tempo de remoção de CPB da primeira unidade de decodificação (denotada como unidade de decodificação m) da imagem atual n, em que DpbOutputTime[k] é menor ou igual a DuCpbRemovalTime[m]. Para cada imagem que é removida do DPB, a totalidade de DPB é decrementada em um.

[0179] A operação do DPB de ordem de emissão pode ser da maneira a seguir. Esses processos podem ser aplicados de maneira independente para cada conjunto de parâmetros de buffer de imagem decodificada (DPB) selecionado. O buffer de imagem decodificada contém, conceitualmente, 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. O processo para emissão e remoção de imagens do DPB antes da decodificação da imagem atual é invocado, seguido pela invocação do processo para marcação e armazenamento de imagem decodificada atual e, por fim, seguido pela invocação do processo para bumping adicional. Esses processos são aplicados de maneira independente para cada camada, iniciando da camada mais baixa no OLS, em ordem crescente dos valores de nuh_layer_id das camadas no OLS. Quando esses processos são aplicados para uma camada particular, apenas o sub-DPB para a camada particular é afetado.

[0180] Na operação de DPB de ordem de emissão, igual à operação de DPB de temporização de emissão, imagens decodificadas com PicOutputFlag igual a um na mesma unidade de acesso também são emitidas consecutivamente em ordem crescente dos valores de nuh_layer_id das imagens decodificadas. Supõe-se que a imagem n e a imagem atual sejam a imagem codificada ou imagem decodificada da unidade de acesso n para um valor particular de nuh_layer_id, em que n é um número inteiro não negativo. A emissão e remoção de imagens do DPB são descritas da maneira a seguir.

[0181] A emissão e remoção de imagens do DPB antes da decodificação da imagem atual (mas após analisar o cabeçalho de fatia da primeira fatia da imagem atual) ocorrem de modo substancialmente instantâneo quando a primeira unidade de decodificação da unidade de acesso contendo a imagem atual é removida do CPB e prossegue da maneira a seguir. O processo de decodificação para construção de lista de imagens de referência e o processo de decodificação para marcação de imagem de referência são invocados. Se a AU atual for uma CVSS AU que não é AU zero, as etapas ordenadas a seguir são aplicadas. A variável NoOutputOfPriorPicsFlag é derivada para o decodificador em teste da maneira a seguir. 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] derivados para qualquer imagem da AU atual for 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, derivados para a imagem antecedente na mesma CLVS, NoOutputOfPriorPicsFlag pode ser definida como um pelo decodificador em teste, independentemente do valor de no_output_of_prior_pics_flag.

[0182] Embora a definição de NoOutputOfPriorPicsFlag como igual a no_output_of_prior_pics_flag possa ser preferencial sob essas condições, o decodificador em teste está permitido a definir NoOutputOfPriorPicsFlag para um neste caso. De outro modo, NoOutputOfPriorPicsFlag é definida como igual a no_output_of_prior_pics_flag. O valor de NoOutputOfPriorPicsFlag derivado para o decodificador em teste é aplicado para o HRD da maneira a seguir. Se NoOutputOfPriorPicsFlag for igual a um, todos os buffers de armazenamento de

imagem no DPB são esvaziados sem emissão das imagens contidas nos mesmos e a totalidade de DPB é definida como igual a zero. De outro modo (NoOutputOfPriorPicsFlag é igual a zero), todos os buffers de armazenamento de imagem contendo uma imagem que é marcada como não necessária para emissão e não utilizada para referência são esvaziados (sem emissão) e todos os buffers de armazenamento de imagem não vazios no DPB são esvaziados invocando-se repetidamente um bumping e a totalidade de DPB é definida como igual a zero.

[0183] De outro modo (a imagem atual não é uma imagem CLVSS), todos os buffers de armazenamento de imagem contendo uma imagem que são marcados como não necessários para emissão e não usados 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 condições a seguir forem verdadeiras, o processo de bumping é invocado repetidamente enquanto decrementa, adicionalmente, a totalidade de DPB em um para cada buffer de armazenamento de imagem adicional que é esvaziado até que nenhuma das condições a seguir seja verdadeira. O número de imagens no DPB que são marcadas como necessárias para emissão é maior que max_num_reorder_pics[Htid]. O max_latency_increase_plus1[Htid] não é igual a zero e há pelo menos uma imagem no DPB que é marcada como necessária para emissão para a qual a variável associada PicLatencyCount é maior que ou igual a MaxLatencyPictures[Htid]. O número de imagens no DPB é maior que ou igual a max_dec_pic_buffering_minus1[Htid] + 1.

[0184] Em um exemplo, bumping adicional pode ocorrer da maneira a seguir. Os processos especificados podem ocorrer de modo substancialmente instantâneo quando a última unidade de decodificação de unidade de acesso n contendo a imagem atual for removida do CPB. Quando a imagem atual tiver PictureOutputFlag igual a um, para cada imagem no DPB que é marcada como necessária para emissão e sucede a imagem atual em ordem de emissão, a variável associada PicLatencyCount é definida como igual a PicLatencyCount + 1. O conteúdo a seguir também se aplica. Se a imagem decodificada atual tiver PictureOutputFlag igual a um, a imagem decodificada atual é marcada como necessária para emissão e uma variável associada PicLatencyCount é definida como igual a zero. De outro modo (a imagem decodificada atual tem

PictureOutputFlag igual a zero), a imagem decodificada atual é marcada como não necessária para emissão.

[0185] Quando uma ou mais das condições a seguir forem verdadeiras, o processo bumping é invocado repetidamente até que nenhuma das condições a seguir seja verdadeira. O número de imagens no DPB que são marcadas como necessárias para emissão é maior que max_num_reorder_pics[Htid]. O max_latency_increase_plus1[Htid] não é igual a zero e há pelo menos uma imagem no DPB que é marcada como necessária para emissão para a qual a variável associada PicLatencyCount que é maior que ou igual a MaxLatencyPictures[Htid].

[0186] O processo de bumping inclui as etapas ordenadas a seguir. A imagem ou imagens que são as primeiras para emissão são selecionadas como tendo o menor valor de PicOrderCntVal de todas as imagens no DPB marcadas como necessárias para emissão. 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. Cada buffer de armazenamento de imagem que contém uma imagem marcada como não usada para referência e que era uma das imagens recortadas e emitidas é esvaziado e a totalidade do sub-DPB associado é decrementada em um. Para quaisquer duas imagens picA e picB que pertencem à mesma CVS e são emitidas pelo processo bumping, quando picA for emitida antes da picB, o valor de PicOrderCntVal de picA é menor que o valor de PicOrderCntVal de picB.

[0187] Um processo de extração de subfluxo de bits exemplificativo é da maneira a seguir. Entradas para esse processo são um fluxo de bits inBitstream, um índice de OLS alvo targetOlsIdx, e um valor alvo mais alto de TemporalId tIdTarget. Saída desse processo é um subfluxo de bits outBitstream. A conformidade de fluxo de bits pode exigir que, para qualquer fluxo de bits de entrada, um subfluxo de bits de saída que é a saída desse processo com o fluxo de bits, targetOlsIdx igual a um índice na lista de OLSs especificados pelo VPS, e tIdTarget igual a qualquer valor na faixa de zero a seis, inclusive, como entradas, e que satisfaz as condições a seguir deva ser um fluxo de bits em conformidade. 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]. O subfluxo de bits de saída contém pelo menos uma unidade VCL NAL com TemporalId igual a tIdTarget. Um fluxo de bits em conformidade contém uma ou mais unidades NAL de fatia codificada com TemporalId igual a zero, mas não tem que conter unidades NAL de fatia codificada com nuh_layer_id igual a zero.

[0188] O subfluxo de bits de saída OutBitstream é derivado da maneira a seguir. O fluxo de bits outBitstream é definido como ser idêntico ao fluxo de bits inBitstream. Todas as unidades NAL com TemporalId maior que tIdTarget são removidas de outBitstream. Todas as unidades NAL com nuh_layer_id não incluído na lista LayerIdInOls[targetOlsIdx] são removidas de outBitstream. Todas as unidades SEI NAL, que contêm uma mensagem SEI de aninhamento escalável que tem nesting_ols_flag igual a um e não há valor de i na faixa de zero a nesting_num_olss_minus1, inclusive, de modo que NestingOlsIdx[i] seja igual a targetOlsIdx, são removidas do outBitstream. Quando targetOlsIdx for maior que zero, todas as unidades SEI NAL que contêm uma mensagem SEI aninhada não escalável com payloadType igual a zero (período de buffering), um (temporização de imagem) ou cento e trinta (informações de unidade de decodificação) são removidas do outBitstream.

[0189] Uma sintaxe de mensagem SEI de aninhamento escalável exemplificativa é da maneira a seguir.

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( )	
}	

[0190] Uma semântica de carga útil de SEI geral exemplificativa é da maneira a seguir. O conteúdo a seguir se aplica nas camadas aplicáveis ou OLS de mensagens SEI aninhadas não escaláveis. Para uma mensagem SEI aninhada não escalável, quando payloadType for igual a zero (período de buffering), um (temporização de imagem) ou cento e trinta (informações de unidade de decodificação), a mensagem SEI aninhada não escalável se aplica apenas ao $0^{ésimo}$ OLS. 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 contendo a mensagem SEI.

[0191] A conformidade de fluxo de bits pode exigir que as restrições a seguir se apliquem no valor de nuh_layer_id de unidades SEI NAL. Quando uma mensagem SEI aninhada não escalável tiver payloadType igual a zero (período de buffering), um (temporização de imagem) ou cento e trinta (informações de unidade de decodificação), a unidade SEI NAL contendo a mensagem SEI aninhada não escalável deve ter nuh_layer_id igual a vps_layer_id[0]. Quando uma mensagem SEI aninhada não escalável tiver payloadType igual a qualquer valor entre VclAssociatedSeiList, a unidade SEI NAL contendo 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. 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 for igual a zero) ou o valor mais baixo 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 for igual a um).

[0192] Uma semântica de mensagem SEI de aninhamento escalável exemplificativa é da maneira a seguir. A mensagem SEI de aninhamento escalável fornece um mecanismo para associar mensagens SEI com OLSs específicos ou com camadas específicas. Uma mensagem SEI de aninhamento escalável contém uma ou mais mensagens SEI. As mensagens SEI contidas na mensagem SEI de aninhamento escalável também são denominadas as mensagens SEI aninhadas escaláveis. A conformidade de fluxo de bits pode exigir que as restrições a seguir se apliquem em contenção de mensagens SEI em uma mensagem SEI de aninhamento escalável.

[0193] Uma mensagem SEI que tem payloadType igual a cento e trinta e dois (hash de imagem decodificada) ou cento e trinta e três (aninhamento escalável) pode não estar contida em uma mensagem SEI de aninhamento escalável. Quando uma mensagem SEI de aninhamento escalável contiver um período de buffering, temporização de imagem ou mensagem SEI de informações de unidade de decodificação, a mensagem SEI de aninhamento escalável não deve conter qualquer outra mensagem SEI com payloadType diferente de zero (período de buffering), um (temporização de imagem) ou cento e trinta (informações de unidade de decodificação).

[0194] A conformidade de fluxo de bits pode exigir que as restrições a seguir se apliquem no valor do nal_unit_type da unidade SEI NAL contendo uma mensagem SEI de aninhamento escalável. Quando uma mensagem SEI de aninhamento escalável contiver uma mensagem SEI que tem payloadType igual a zero (período de buffering), um (temporização de imagem), cento e trinta (informações de unidade de decodificação), cento e quarenta e cinco (indicação de RAP dependente) ou cento e sessenta e oito (informações de campo de quadro), a unidade SEI NAL contendo a mensagem SEI de aninhamento escalável deve ter um nal_unit_type igual a PREFIX_SEI_NUT.

[0195] Uma nesting_ols_flag igual a um especifica que as mensagens SEI aninhadas escaláveis se aplicam a OLSs específicos. Uma nesting_ols_flag igual a zero especifica que as mensagens SEI aninhadas escaláveis se aplicam a camadas específicas. A conformidade de fluxo de bits pode exigir que as restrições a seguir se apliquem no valor de nesting_ols_flag. Quando a mensagem

SEI de aninhamento escalável contiver uma mensagem SEI que tem payloadType igual a zero (período de buffering), um (temporização de imagem) ou cento e trinta (informações de unidade de decodificação), o valor de nesting_ols_flag deve ser igual a um. Quando a mensagem SEI de aninhamento escalável contiver uma mensagem SEI que tem payloadType igual a um valor em VclAssociatedSeiList, o valor de nesting_ols_flag deve ser igual a zero. O 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.

[0196] Um 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 for igual a um. O valor de nesting_ols_idx_delta_minus1[i] deve estar na faixa de zero a TotalNumOlss menos dois, inclusive, inclusive. A variável NestingOlsIdx[i] pode ser derivada da maneira a seguir.

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

[0197] Uma nesting_all_layers_flag igual a um 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. Uma nesting_all_layers_flag igual a zero 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. Um 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 zero 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. Um 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 for igual a zero. 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.

[0198] Quando nesting_ols_flag for igual a zero, a variável NestingNumLayers, que especifica o número de camadas às quais as mensagens SEI aninhadas escaláveis se aplicam, e a lista NestingLayerId[i] para i na faixa de zero a NestingNumLayers − 1, inclusive, que especifica a lista de valor de nuh_layer_id das camadas às quais as mensagens SEI aninhadas escaláveis se aplicam, podem ser derivadas da maneira a seguir, em que nuh_layer_id é o nuh_layer_id da unidade SEI NAL atual.

```
if( nesting_all_layers_flag ) {
        NestingNumLayers =
vps_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 ]
}
```

[0199] Um nesting_num_seis_minus1 mais um especifica o número de mensagens SEI aninhadas escaláveis. O valor de nesting_num_seis_minus1 deve estar na faixa de zero a sessenta e três inclusive. O nesting_zero_bit deve ser igual a zero.

[0200] A Figura 8 é um diagrama esquemático de um dispositivo de codificação de vídeo exemplificativo 800. O dispositivo de codificação de vídeo 800 é adequado para implementar os exemplos/modalidades revelados, conforme descrito na presente invenção. O dispositivo de codificação de vídeo 800 compreende portas a jusante 820, portas a montante 850 e/ou unidades transceptoras (Tx/Rx) 810, incluindo transmissores e/ou receptores para comunicar dados a montante e/ou a jusante através de uma rede. O dispositivo de codificação de vídeo 800 também inclui um processador 830 que inclui uma unidade lógica e/ou unidade central de processamento (CPU) para processar os dados e uma memória 832 para armazenar os dados. O dispositivo de codificação de vídeo 800 também pode compreender componentes elétricos, ópticos para

elétricos (OE), componentes elétricos para ópticos (EO) e/ou componentes de comunicação sem fio acoplados às portas a montante 850 e/ou portas a jusante 820 para comunicação de dados através de redes de comunicação elétricas, ópticas ou sem fio. O dispositivo de codificação de vídeo 800 também pode incluir dispositivos de entrada e/ou saída (I/O) 860 para comunicar dados de/para um usuário. Os dispositivos de I/O 860 podem incluir dispositivos de saída, tais como um display para exibir dados de vídeo, alto-falantes para emitir dados de áudio, etc. Os dispositivos de I/O 860 também podem incluir dispositivos de entrada, tais como um teclado, mouse, trackball, etc., e/ou interfaces correspondentes para interagir com tais dispositivos de saída.

[0201] O processador 830 é implementado por hardware e software. O processador 830 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 830 está em comunicação com as portas a jusante 820, Tx/Rx 810, portas a montante 850 e a memória 832. O processador 830 compreende um módulo de codificação 814. O módulo de codificação 814 implementa as modalidades reveladas descritas na presente invenção, tais como os métodos 100, 900 e 1000, que podem empregar uma sequência de vídeo de múltiplas camadas 600 um fluxo de bits 700 e/ou um subfluxo de bits 701. O módulo de codificação 814 também pode implementar qualquer outro método/mecanismo descrito na presente invenção. Além disso, o módulo de codificação 814 pode implementar um sistema codec 200, um codificador 300, um decodificador 400 e/ou um HRD 500. Por exemplo, o módulo de codificação 814 pode ser empregado para codificar, extrair e/ou decodificar um fluxo de bits incluindo uma camada simulcast e nenhum VPS. Além disso, o módulo de codificação 814 pode ser empregado para definir e/ou inferir vários elementos de sintaxe e/ou variáveis para evitar erros com base em referências a um VPS que é extraído como parte de uma extração de subfluxo de bits. Assim, o módulo de codificação 814 pode ser configurado para realizar mecanismos para abordar um ou mais dos problemas discutidos acima. Portanto, o módulo de codificação 814 faz com que o dispositivo de codificação de vídeo 800 forneça funcionalidade adicional e/ou eficiência de codificação ao codificar dados de vídeo. Como tal, o módulo de codificação 814 aperfeiçoa a funcionalidade do dispositivo

de codificação de vídeo 800 bem como aborda problemas que são específicos às técnicas de codificação de vídeo. Além disso, o módulo de codificação 814 realiza uma transformação do dispositivo de codificação de vídeo 800 para um estado diferente. De modo alternativo, o módulo de codificação 814 pode ser implementado como instruções armazenadas na memória 832 e executadas pelo processador 830 (por exemplo, como um produto de programa de computador armazenado em uma mídia não transitória).

[0202] A memória 832 compreende um ou mais tipos de memória, tais como discos, unidades de fita, unidades de estado sólido, memória somente leitura (ROM), memória de acesso aleatório (RAM), memória flash, memória ternária de conteúdo endereçável (TCAM), memória de acesso aleatório estática (SRAM), etc. A memória 832 pode ser utilizada 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 de programa.

[0203] A Figura 9 é um fluxograma de um método exemplificativo 900 de codificação de uma sequência de vídeo de múltiplas camadas em um fluxo de bits, tal como fluxo de bits 700, para suportar remoção de VPS 711 durante um processo de extração de subfluxo de bits 729 para camadas simulcast. O método 900 pode ser empregado por um codificador, tal como um sistema codec 200, um codificador 300 e/ou um dispositivo de codificação de vídeo 800 ao realizar o método 100. Além disso, o método 900 pode operar em um HRD 500 e, portanto, pode realizar testes de conformidade em uma sequência de vídeo de múltiplas camadas 600 e/ou uma camada extraída da mesma.

[0204] O método 900 pode iniciar quando um codificador recebe uma sequência de vídeo e determina codificar essa sequência de vídeo em um fluxo de bits de múltiplas camadas, por exemplo, com base em entrada de usuário. Na etapa 901, o codificador codifica imagens codificadas em um conjunto de unidades VCL NAL no fluxo de bits. Por exemplo, o codificador pode codificar as imagens na sequência de vídeo em uma ou mais camadas e codificar as camadas em um fluxo de bits de múltiplas camadas. Portanto, o fluxo de bits compreende uma ou mais camadas. Uma camada pode incluir um conjunto de unidades VCL NAL com o mesmo Id de camada e unidades não VCL NAL associadas. Conforme um exemplo específico, as unidades VCL NAL podem ser associadas a uma camada

identificada por/tendo um nuh_layer_id. Especificamente, um conjunto de unidades VCL NAL é parte de uma camada quando o conjunto de unidades VCL NAL tiverem, todas, um valor particular de nuh_layer_id. Uma camada pode incluir um conjunto de unidades VCL NAL que contêm dados de vídeo de imagens codificadas bem como quaisquer conjuntos de parâmetros usados para codificar tais imagens. Tais parâmetros podem ser incluídos em um VPS, SPS, PPS, cabeçalho de imagem, cabeçalho de fatia ou outro conjunto de parâmetros ou estrutura de sintaxe. Conforme um exemplo específico, o codificador pode codificar, no fluxo de bits, um SPS incluindo um sps_video_parameter_set_id. Uma ou mais dentre as camadas podem ser camadas de saída. As camadas que não são uma camada de saída são denominadas camadas de referência e são codificadas para suportar reconstrução da(s) camada(s) de saída, mas tais camadas de suporte não são destinadas para emissão em um decodificador. Desse modo, o codificador pode codificar várias combinações de camadas para transmissão a um decodificador mediante solicitação. A camada pode ser transmitida conforme desejado para permitir que o decodificador obtenha diferentes representações da sequência de vídeo dependendo de condições de rede, capacidades de hardware e/ou definições de usuário. No presente exemplo, pelo menos uma das camadas é uma camada simulcast que não usa predição de camada inter.

[0205] Na etapa 903, um HRD operando no codificador pode realizar um conjunto de testes de conformidade de fluxo de bits nas camadas para garantir conformidade com VVC ou outros padrões. Por exemplo, o HRD pode obter um sps_video_parameter_set_id de um SPS. O sps_video_parameter_set_id especifica um valor de um vps_video_parameter_set_id de um VPS referenciado pelo SPS quando o sps_video_parameter_set_id for maior que zero. Adicionalmente, o SPS não se refere a um VPS e nenhum VPS é referido ao decodificar cada sequência de vídeo de camada codificada que se refere ao SPS quando o sps_video_parameter_set_id for igual a zero. Assim, o sps_video_parameter_set_id é definido comocomo e/ou inferido como sendo zero quando o sps_video_parameter_set_id for obtido a partir de um SPS referenciado por uma sequência de vídeo de camada codificada contida em uma camada simulcast.

[0206] O HRD pode definir e/ou inferir que um

GeneralLayerIdx[nuh_layer_id] seja igual a zero quando o sps_video_parameter_set_id for igual a zero. O GeneralLayerIdx[nuh_layer_id] é igual a e, portanto, indica, um índice de camada atual para uma camada correspondente. Como tal, o índice de camada atual para uma camada simulcast é definido como/inferido como sendo zero. Adicionalmente, o HRD pode inferir que um valor de uma vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] seja igual a um quando sps_video_parameter_set_id for igual a zero. Especificamente, uma vps_independent_layer_flag [i] está contida em um VPS e pode ser definida como zero para indicar que uma i^{ésima} camada usa predição de camada inter ou definida como um para indicar que a i^{ésima} camada não usa predição de camada inter. Como tal, vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se uma camada atual com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. Quando a camada atual for uma camada simulcast, o VPS é omitido e a predição de camada inter não é empregada. Assim, a inferência de um valor de um quando o sps_video_parameter_set_id for igual a zero garante que a camada simulcast opere de maneira apropriada no HRD e durante decodificação enquanto evita uma referência ao VPS, que é extraído durante extração de fluxo de bits para camadas simulcast. Como tal, a inferência evita erros de extração de subfluxo de bits que ocorreriam, de outro modo, quando o VPS é removido para uma camada simulcast. O HRD pode, então, decodificar a imagem codificada a partir das unidades VCL NAL na camada simulcast com base no SPS, no sps_video_parameter_set_id, no GeneralLayerIdx[nuh_layer_id] e/ou na vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] para produzir uma imagem decodificada. Como tal, o HRD pode verificar se a camada simulcast do fluxo de bits de múltiplas camadas está em conformidade com fluxo de bits sem erros inesperados causados por omissão do VPS para camadas simulcast.

[0207] Na etapa 905, o codificador pode armazenar o fluxo de bits para comunicação em direção a um decodificador mediante solicitação. O codificador também pode realizar extração de subfluxo de bits para obter a camada simulcast e transmitir o fluxo de bits/subfluxo de bits em direção ao codificador conforme desejado.

[0208] A Figura 10 é um fluxograma de um método exemplificativo 1000 de decodificação de uma sequência de vídeo a partir de um fluxo de bits, tal

como subfluxo de bits 701, que inclui uma camada simulcast extraída de um fluxo de bits de múltiplas camadas, tal como fluxo de bits 700, em que um VPS 711 foi removido durante um processo de extração de subfluxo de bits 729. O método 1000 pode ser empregado por um decodificador, tal como um sistema codec 200, um decodificador 400 e/ou um dispositivo de codificação de vídeo 800 ao realizar o método 100. Além disso, o método 1000 pode ser empregado em uma sequência de vídeo de múltiplas camadas 600, ou uma camada extraída da mesma, que foi verificada quanto à conformidade por um HRD, tal como HRD 500.

[0209] O método 1000 pode iniciar quando um decodificador começar a receber um fluxo de bits contendo uma sequência de vídeo codificada em uma camada simulcast extraída de um fluxo de bits de múltiplas camadas, por exemplo, como resultado do método 900. Na etapa 1001, o decodificador recebe um fluxo de bits que compreende uma camada simulcast extraída de um fluxo de bits de múltiplas camadas por um codificador ou outro servidor de conteúdo intermediário. A camada simulcast contém uma sequência de vídeo de camada codificada incluindo um conjunto de imagens codificadas. Por exemplo, o fluxo de bits compreende imagens codificadas em que cada imagem codificada é incluída em um conjunto de uma ou mais unidades VCL NAL associadas à camada simulcast conforme identificado por/tendo um nuh_layer_id. Uma camada pode incluir um conjunto de unidades VCL NAL com o mesmo Id de camada e unidades não VCL NAL associadas. Por exemplo, uma camada pode incluir um conjunto de unidades VCL NAL que contêm dados de vídeo de imagens codificadas bem como quaisquer conjuntos de parâmetros usados para codificar tais imagens. Como tal, o conjunto de unidades VCL NAL é parte da camada quando o conjunto de unidades VCL NAL tiverem, todas, um valor particular de nuh_layer_id. A camada simulcast também é uma camada de saída e não emprega predição de camada inter. O fluxo de bits também compreende um SPS incluindo um sps_video_parameter_set_id. O sps_video_parameter_set_id especifica um valor de um vps_video_parameter_set_id de um VPS referenciado pelo SPS quando o sps_video_parameter_set_id for maior que zero. Adicionalmente, o SPS não se refere a um VPS e nenhum VPS é referido ao decodificar cada sequência de vídeo de camada codificada que se refere ao SPS quando o sps_video_parameter_set_id for igual a zero. Assim, o sps_video_parameter_set_id é definido como e/ou inferido como sendo zero

quando o sps_video_parameter_set_id for obtido a partir de um SPS referenciado por uma sequência de vídeo de camada codificada contida em uma camada simulcast. Além disso, o fluxo de bits não contém um VPS quando o fluxo de bits contiver apenas uma camada simulcast.

[0210] Na etapa 1003, o decodificador pode definir e/ou inferir que um GeneralLayerIdx[nuh_layer_id] seja igual a zero quando o sps_video_parameter_set_id for igual a zero. O GeneralLayerIdx[nuh_layer_id] é igual a e, portanto, indica, um índice de camada atual para uma camada correspondente. Como tal, o índice de camada atual para a camada simulcast recebida é definido como/inferido como sendo zero.

[0211] Na etapa 1005, o decodificador pode definir/inferir que um valor de uma vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] seja igual a um quando sps_video_parameter_set_id for igual a zero. Especificamente, uma vps_independent_layer_flag [i] está contida em um VPS e pode ser definida como zero para indicar que uma i^{ésima} camada usa predição de camada inter ou definida como um para indicar que a i^{ésima} camada não usa predição de camada inter. Como tal, vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se uma camada atual com índice GeneralLayerIdx[nuh_layer_id] usa predição de camada inter. Quando a camada atual for uma camada simulcast, o VPS é omitido do fluxo de bits e predição de camada inter não é empregada. Assim, a inferência de um valor de um quando o sps_video_parameter_set_id for igual a zero garante que a camada simulcast opere de maneira apropriada durante decodificação enquanto evita uma referência ao VPS, que é extraído durante extração de fluxo de bits para camadas simulcast e, portanto, não recebido no decodificador. Como tal, a inferência evita erros de extração de subfluxo de bits que ocorreriam, de outro modo, quando o VPS é removido para uma camada simulcast.

[0212] Na etapa 1007, o decodificador pode decodificar a imagem codificada a partir das unidades VCL NAL na camada simulcast com base no SPS, no sps_video_parameter_set_id, no GeneralLayerIdx[nuh_layer_id] e/ou na vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] para produzir uma imagem decodificada. O decodificador pode encaminhar, então, a imagem decodificada para exibição como parte de uma sequência de vídeo decodificada na etapa 1009.

[0213] A Figura 11 é um diagrama esquemático de um sistema

exemplificativo 1100 para codificar uma sequência de vídeo de múltiplas camadas em um fluxo de bits 700 para suportar remoção de VPS 711 durante processo de extração de subfluxo de bits 729 para camadas simulcast. O sistema 1100 pode ser implementado por um codificador e um decodificador, tal como um sistema codec 200, um codificador 300, um decodificador 400 e/ou um dispositivo de codificação de vídeo 800. Além disso, o sistema 1100 pode empregar um HRD 500 para realizar testes de conformidade em uma sequência de vídeo de múltiplas camadas 600, um fluxo de bits 700 e/ou um subfluxo de bits 701. Além disso, o sistema 1100 pode ser empregado ao implementar o método 100, 900 e/ou 1000.

[0214] O sistema 1100 inclui um codificador de vídeo 1102. O codificador de vídeo 1102 compreende um módulo de codificação 1103 para codificar uma imagem codificada e um SPS em um fluxo de bits, em que a imagem codificada é codificada em um conjunto de unidades VCL NAL, em que as unidades VCL NAL são associadas a uma camada tendo um nuh_layer_id, e em que o SPS inclui um sps_video_parameter_set_id. O codificador de vídeo 1102 compreende, adicionalmente, um módulo de HRD 1105 para realizar um conjunto de testes de conformidade de fluxo de bits no fluxo de bits definindo-se um GeneralLayerIdx[nuh_layer_id] igual a zero quando o sps_video_parameter_set_id for igual a zero e decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada. O codificador de vídeo 1102 compreende, adicionalmente, um módulo de armazenamento 1106 para armazenar o fluxo de bits para comunicação em direção a um decodificador. O codificador de vídeo 1102 compreende, adicionalmente, um módulo de transmissão 1107 para transmitir o fluxo de bits em direção a um decodificador de vídeo 1110. O codificador de vídeo 1102 pode ser configurado, adicionalmente, para realizar qualquer uma das etapas do método 900.

[0215] O sistema 1100 também inclui um decodificador de vídeo 1110. O decodificador de vídeo 1110 compreende um módulo de recebimento 1111 para receber um fluxo de bits que compreende um SPS e uma imagem codificada, em que o SPS inclui um sps_video_parameter_set_id, e em que a imagem codificada está em um conjunto de unidades VCL NAL associadas a uma camada tendo um nuh_layer_id. O decodificador de vídeo 1110 compreende, adicionalmente, um módulo de definição 1113 para definir um GeneralLayerIdx[nuh_layer_id] igual a

zero quando o sps_video_parameter_set_id for igual a zero. O decodificador de vídeo 1110 compreende, adicionalmente, um módulo de decodificação 1115 para decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada. O decodificador de vídeo 1110 compreende, adicionalmente, um módulo de encaminhamento 1117 para encaminhar a imagem decodificada para exibição como parte de uma sequência de vídeo decodificada. O decodificador de vídeo 1110 pode ser configurado, adicionalmente, para realizar qualquer uma das etapas do método 1000.

[0216] Um primeiro componente é acoplado diretamente a um segundo componente quando não houver componentes intermediários, exceto uma linha, um traço ou outro meio entre o primeiro componente e o segundo componente. O primeiro componente é acoplado indiretamente ao segundo componente quando houver componentes intermediários diferentes de uma linha, um traço ou outro meio entre o primeiro componente e o segundo componente. O termo “acoplado” e suas variantes incluem tanto acoplado diretamente quanto acoplado indiretamente. O uso do termo “cerca de” significa uma faixa que inclui ±10% do número subsequente, a menos que estabelecido de outra maneira.

[0217] Deve-se entender também que não é exigido que as etapas dos métodos exemplificativos estabelecidos na presente invenção sejam necessariamente desempenhadas na ordem descrita, e a ordem das etapas de tais métodos deve ser entendida apenas como exemplificativa. Da mesma maneira, as etapas adicionais podem ser incluídas em tais métodos, e determinadas etapas podem ser omitidas ou combinadas, em métodos consistentes com várias modalidades da presente revelação.

[0218] Embora diversas modalidades tenham sido fornecidas na presente revelação, pode-se entender que os sistemas e métodos revelados podem ser incorporados a muitas outras formas específicas sem se afastar da essência ou escopo da presente revelação. Os presentes exemplos devem ser considerados como ilustrativos e não restritivos, e a intenção não deve ser limitada aos detalhes fornecidos no presente documento. Por exemplo, os vários elementos ou componentes podem ser combinados ou integrados em outro sistema ou determinados atributos podem ser omitidos, ou não implementados.

[0219] Além disso, técnicas, sistemas, subsistemas e métodos

descritos e ilustrados nas várias modalidades como distintos ou separados podem ser combinados ou integrados a outros sistemas, componentes, técnicas ou métodos sem se afastar do escopo da presente revelação. Outros exemplos de mudanças, substituições e alterações são verificados por um técnico no assunto e podem ser realizados sem se afastar da essência e escopo revelados no presente documento.

REIVINDICAÇÕES

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

codificar, pelo codificador, uma imagem codificada e um conjunto de parâmetros de sequência (SPS) em um fluxo de bits, em que a imagem codificada é codificada em um conjunto de unidades de camada de abstração de rede (NAL) de camada de codificação de vídeo (VCL), em que as unidades VCL NAL são associadas a uma camada que tem um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id), e em que o SPS inclui um identificador de conjunto de parâmetros de vídeo de SPS (sps_video_parameter_set_id);

realizar, por um decodificador de referência hipotético (HRD) no codificador, um conjunto de testes de conformidade de fluxo de bits no fluxo de bits por:

definir, por um decodificador de referência hipotético que opera no codificador, um índice de camada geral que corresponde ao nuh_layer_id (GeneralLayerIdx[nuh_layer_id]) igual a zero quando o sps_video_parameter_set_id for igual a zero; e

decodificar, pelo HRD que opera no codificador, a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada; e

armazenar, no codificador, o fluxo de bits para comunicação em direção a um decodificador.

2. Método, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que GeneralLayerIdx[nuh_layer_id] é igual a um índice de camada atual.

3. Método, de acordo com qualquer uma das reivindicações 1 a 2, **CARACTERIZADO** pelo fato de que compreende adicionalmente inferir, pelo HRD que opera no codificador, que um valor de uma flag de camada independente de conjunto de parâmetros de vídeo (VPS) para o GeneralLayerIdx[nuh_layer_id] (vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]) seja igual a um quando sps_video_parameter_set_id for igual a zero.

4. Método, de acordo com qualquer uma das reivindicações 1 a 3, **CARACTERIZADO** pelo fato de que vps_independent_layer_flag [GeneralLayerIdx[nuh_layer_id]] especifica se a camada com índice

GeneralLayerIdx[nuh_layer_id] usa predição de camada inter.

5. Método, de acordo com qualquer uma das reivindicações 1 a 4, **CARACTERIZADO** pelo fato de que o sps_video_parameter_set_id especifica um valor de um identificador de conjunto de parâmetros de VPS (vps_video_parameter_set_id) quando maior que zero.

6. Método, de acordo com qualquer uma das reivindicações 1 a 5, **CARACTERIZADO** pelo fato de que o SPS não se refere a um VPS e nenhum VPS é referido ao decodificar cada sequência de vídeo de camada codificada que se refere ao SPS quando o sps_video_parameter_set_id for igual a zero.

7. Método, de acordo com qualquer uma das reivindicações 1 a 6, **CARACTERIZADO** pelo fato de que o conjunto de unidades VCL NAL é parte da camada quando o conjunto de unidades VCL NAL tiverem, todas, um valor particular de nuh_layer_id.

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

um processador, um receptor acoplado ao processador, uma memória acoplada ao processador, e um transmissor acoplado ao processador, em que o processador, receptor, memória, e transmissor são configurados para realizar o método conforme definido em qualquer uma das reivindicações 1 a 7.

9. Mídia legível por computador não transitória, **CARACTERIZADA** pelo fato de que compreende um produto de programa de computador para uso por um dispositivo de codificação de vídeo, o produto de programa de computador compreendendo instruções executáveis em computador armazenadas na mídia legível por computador não transitória tais que quando executadas por um processador fazem com que o dispositivo de codificação de vídeo realize o método conforme definido em qualquer uma das reivindicações 1 a 7.

10. Codificador, **CARACTERIZADO** pelo fato de que compreende:

um meio de codificação para codificar uma imagem codificada e um conjunto de parâmetros de sequência (SPS) em um fluxo de bits, em que a imagem codificada é codificada em um conjunto de unidades de camada de abstração de rede (NAL) de camada de codificação de vídeo (VCL), em que as unidades VCL NAL estão associadas com uma camada que tem um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id), e em que o SPS inclui um identificador de conjunto de parâmetros de vídeo de SPS

(sps_video_parameter_set_id);

um meio de decodificador de referência hipotético (HRD) para realizar um conjunto de testes de conformidade de fluxo de bits no fluxo de bits por:

definir um índice de camada geral que corresponde ao nuh_layer_id (GeneralLayerIdx[nuh_layer_id]) igual a zero quando o sps_video_parameter_set_id for igual a zero; e

decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada; e

um meio de armazenamento para armazenar o fluxo de bits para comunicação em direção a um decodificador.

11. Codificador, de acordo com a reivindicação 10, **CARACTERIZADO** pelo fato de que o codificador é adicionalmente configurado para realizar o método conforme definido em qualquer uma das reivindicações 1 a 7.

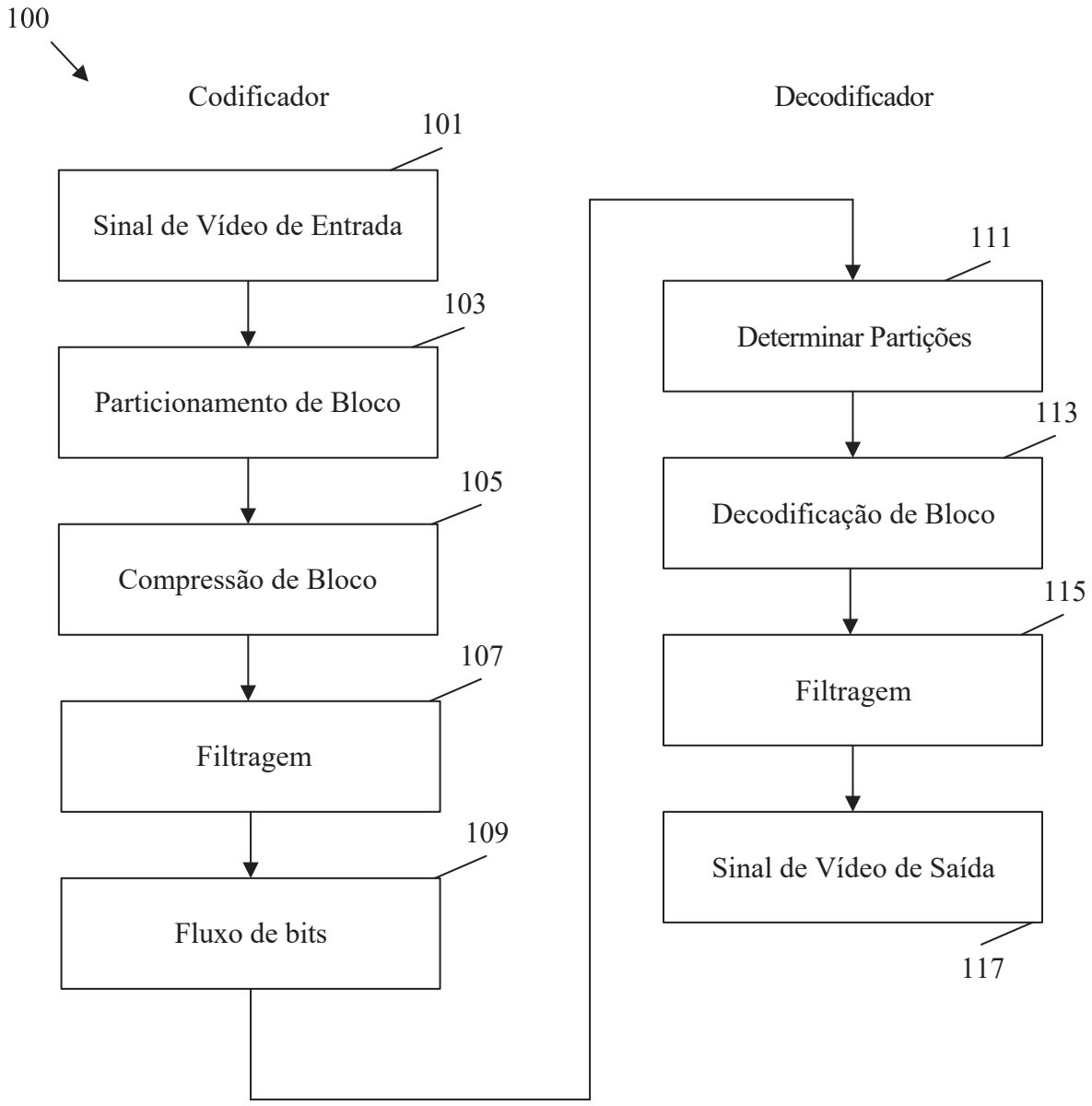

100
Codificador
Decodificador
101
Sinal de Vídeo de Entrada
103
Particionamento de Bloco
105
Compressão de Bloco
107
Filtragem
109
Fluxo de bits
111
Determinar Partições
113
Decodificação de Bloco
115
Filtragem
Sinal de Vídeo de Saída
117

FIG. 1

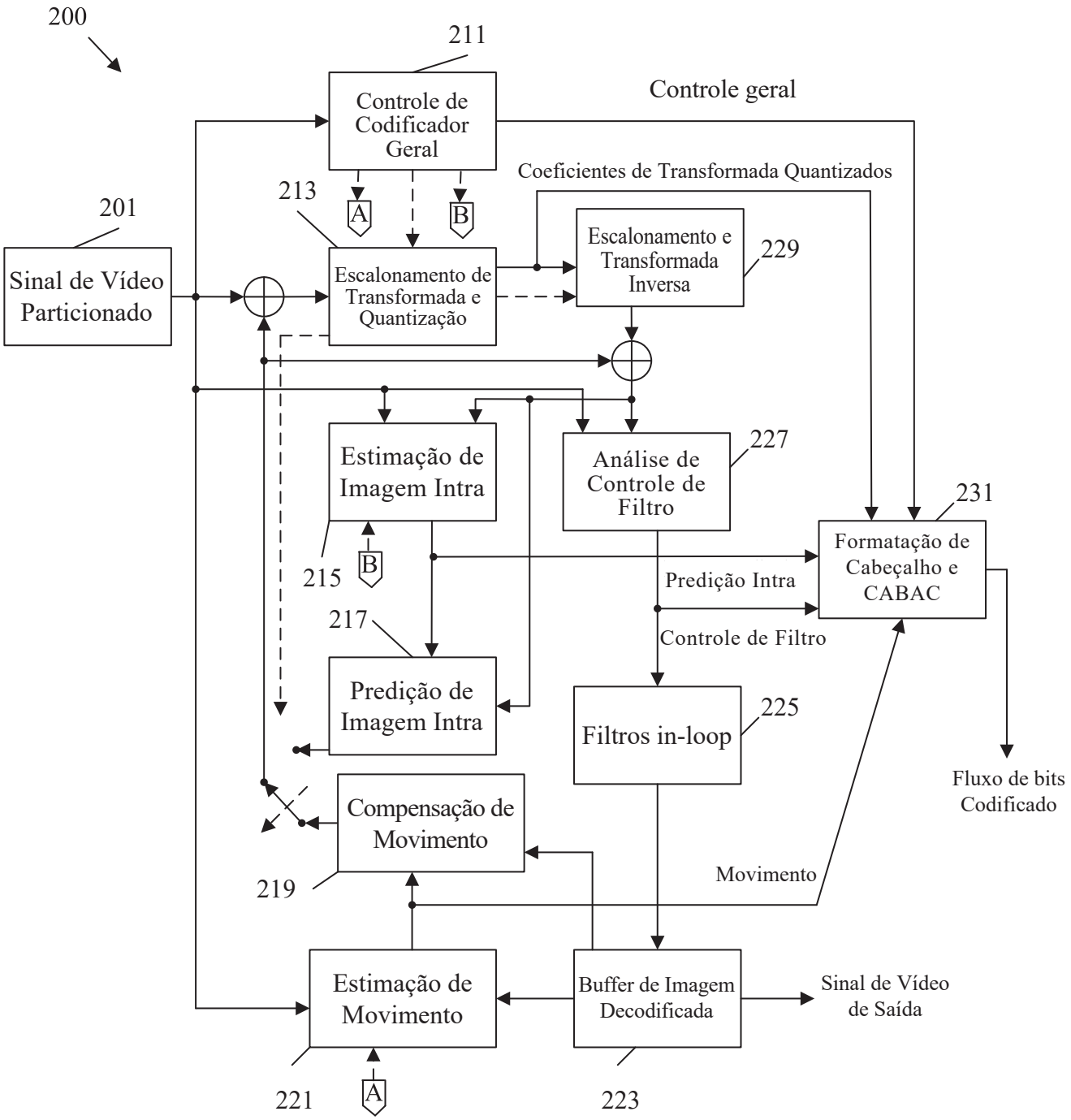

200
211
Controle de Codificador Geral
Controle geral
Coeficientes de Transformada Quantizados
213
A
B
201
Sinal de Vídeo Particionado
Escalonamento de Transformada e Quantização
Escalonamento e Transformada Inversa
229
Estimação de Imagem Intra
Análise de Controle de Filtro
227
231
Formatação de Cabeçalho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

FIG. 2

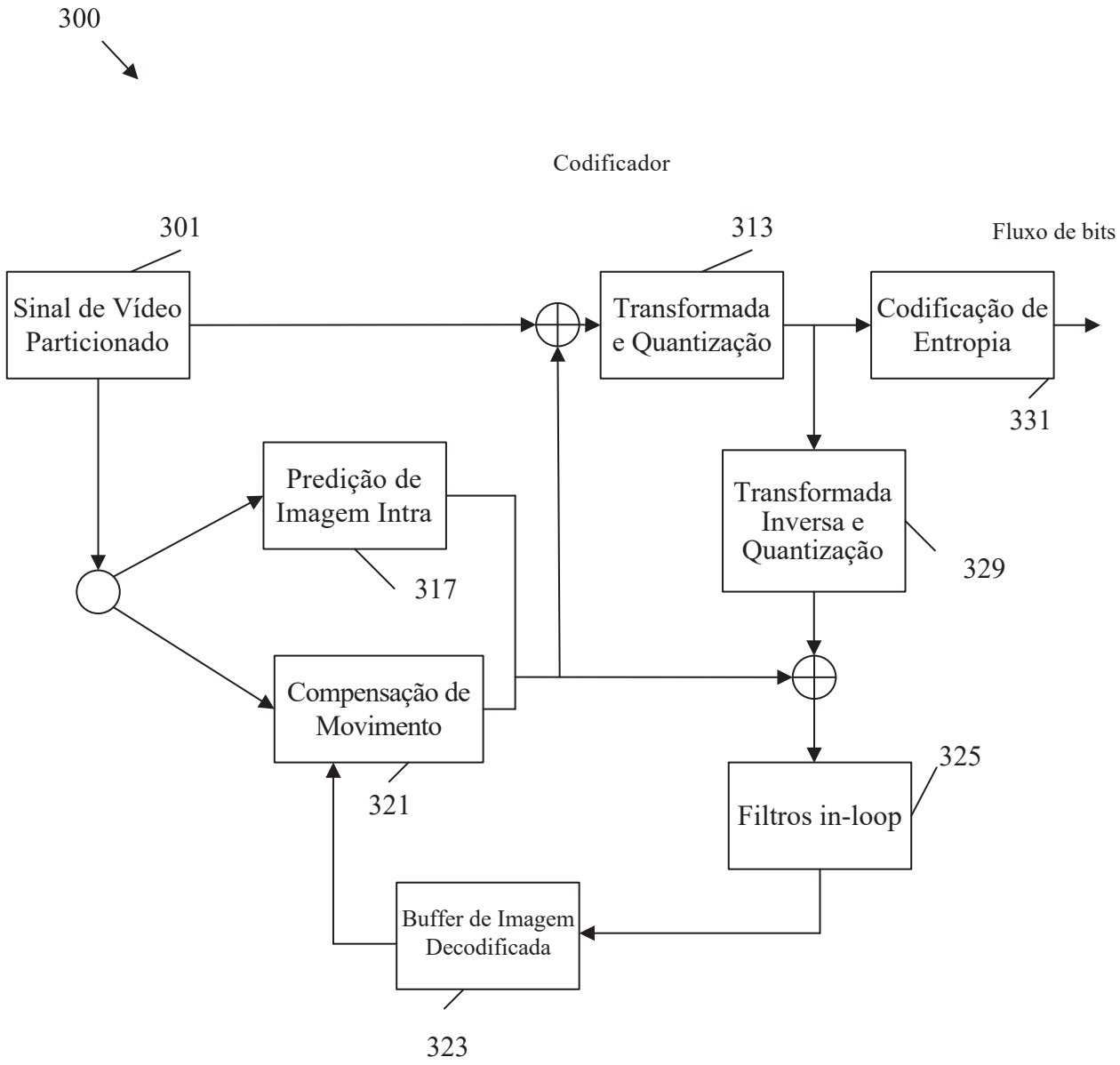

300
Codificador
301
Sinal de Vídeo Particionado
313
Transformada e Quantização
Codificação de Entropia
Fluxo de bits
331
Predição de Imagem Intra
317
Transformada Inversa e Quantização
329
Compensação de Movimento
321
325
Filtros in-loop
Buffer de Imagem Decodificada
323

FIG. 3

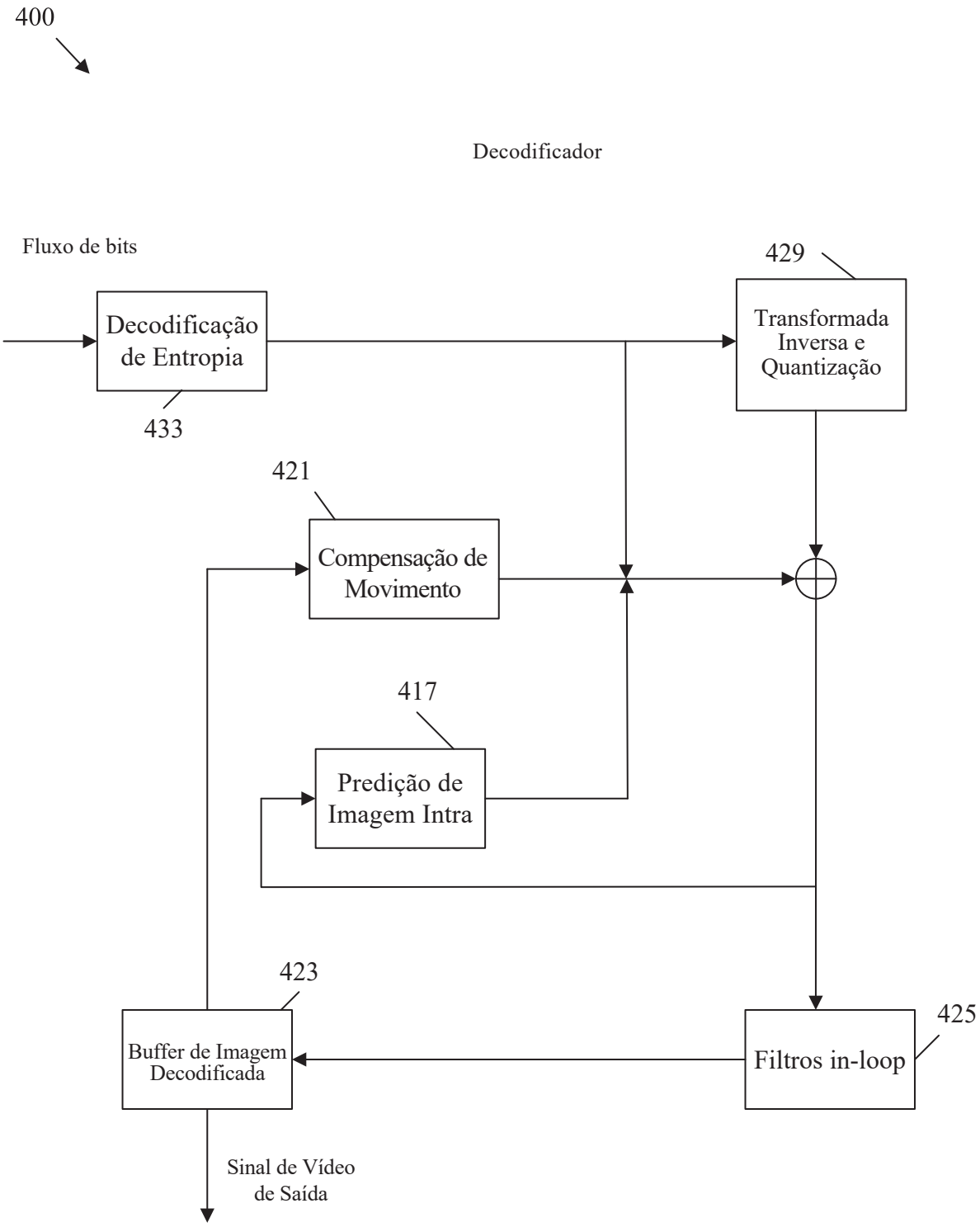

400
Decodificador
Fluxo de bits
Decodificação de Entropia
433
429
Transformada Inversa e Quantização
421
Compensação de Movimento
417
Predição de Imagem Intra
423
Buffer de Imagem Decodificada
425
Filtros in-loop
Sinal de Vídeo de Saída

FIG. 4

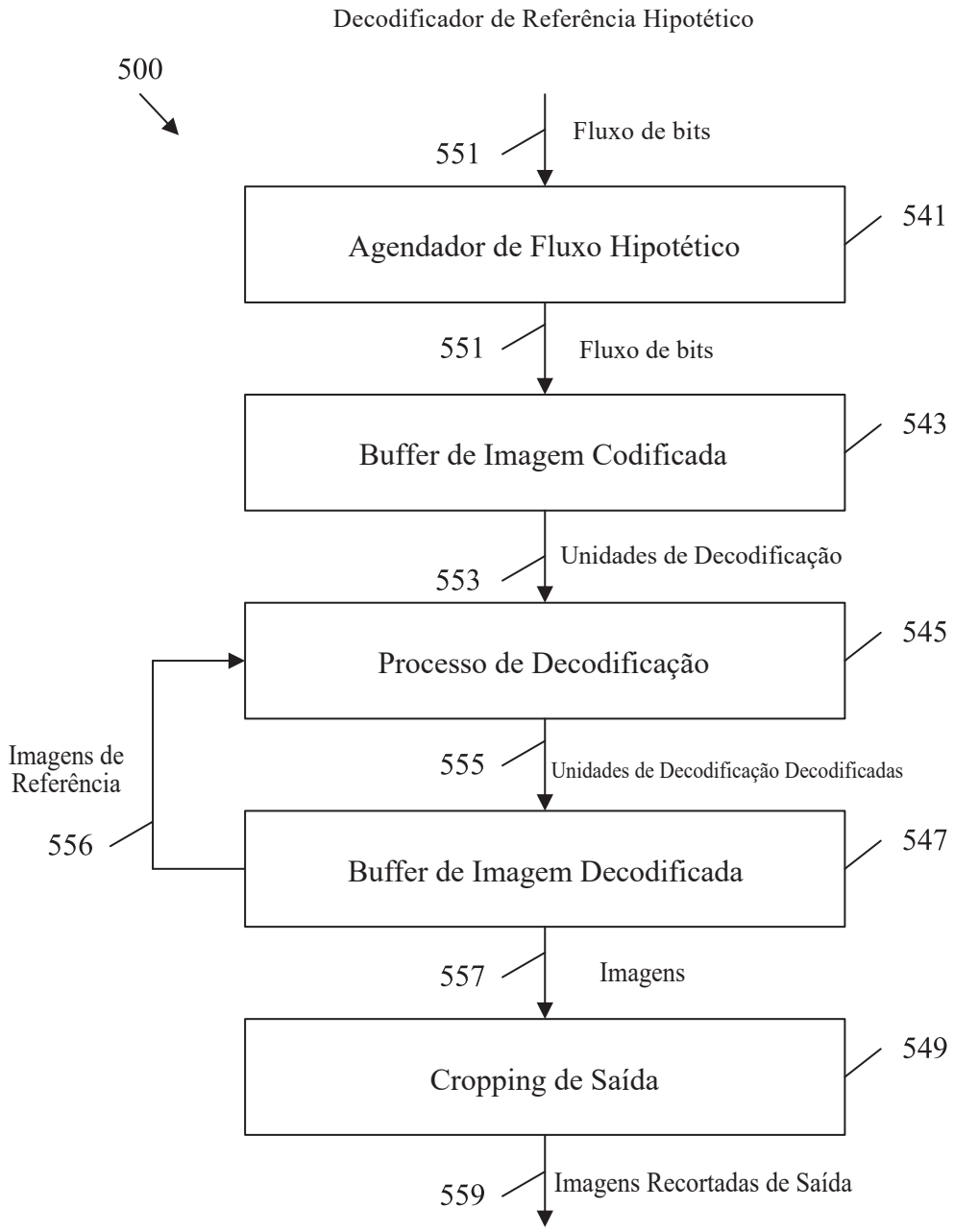

Decodificador de Referência Hipotético
500
551
Fluxo de bits
Agendador de Fluxo Hipotético
541
551
Fluxo de bits
Buffer de Imagem Codificada
543
553
Unidades de Decodificação
Processo de Decodificação
545
Imagens de Referência
555
Unidades de Decodificação Decodificadas
556
Buffer de Imagem Decodificada
547
557
Imagens
Cropping de Saída
549
559
Imagens Recortadas de Saída

FIG. 5

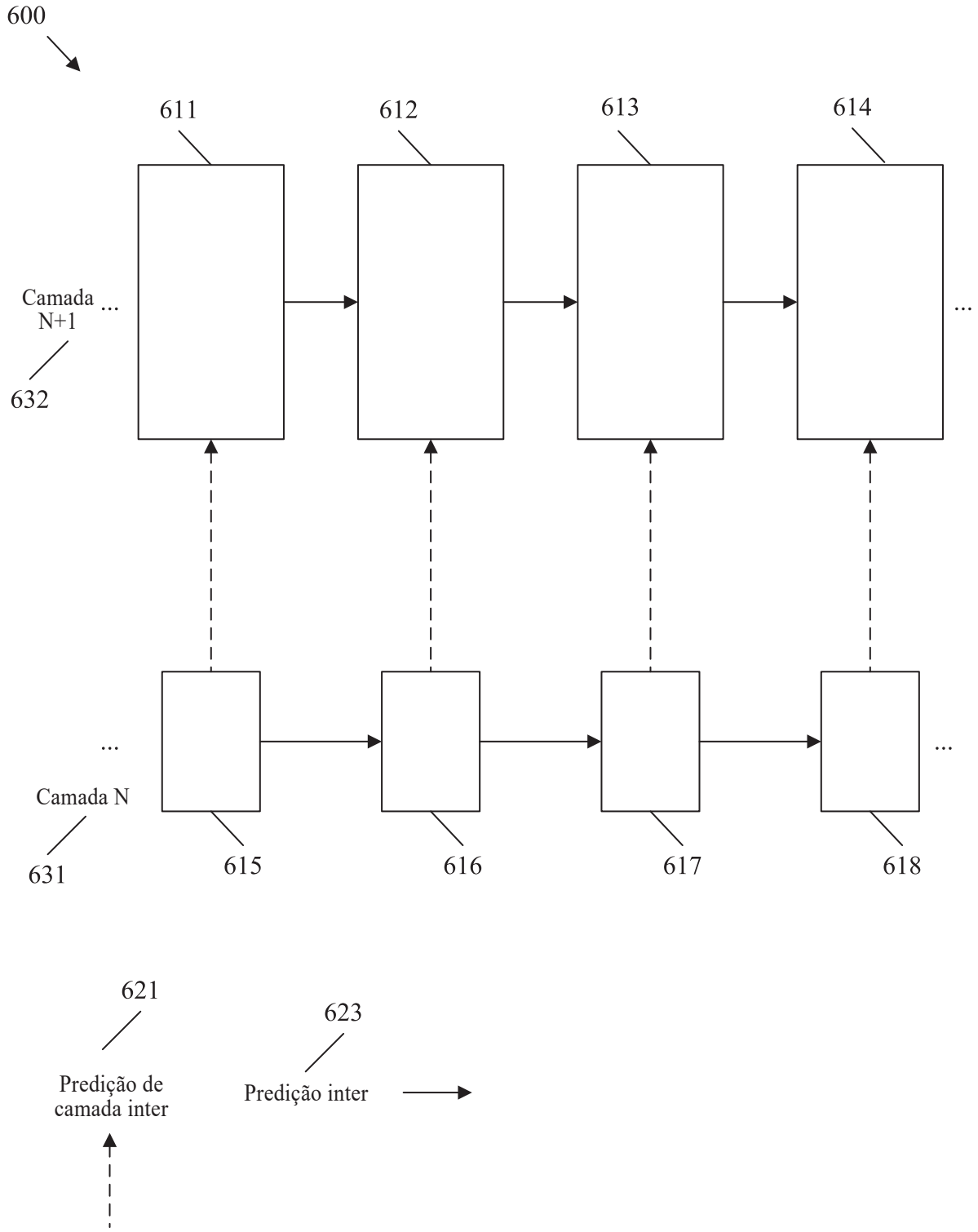

600
611
612
613
614
Camada N+1
632
631
Camada N
615
616
617
618
621
Predição de camada inter
623
Predição inter

FIG. 6

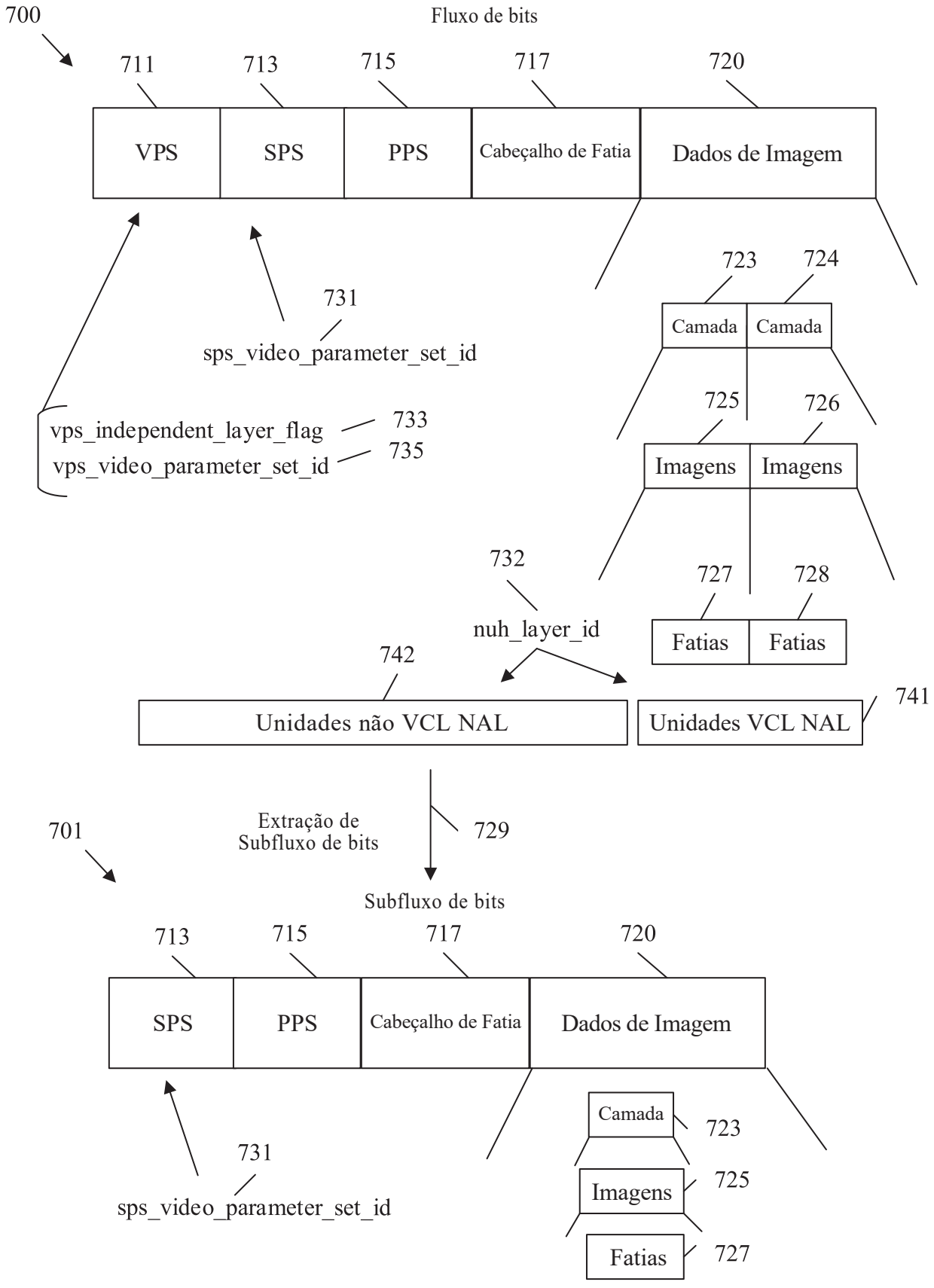

700
Fluxo de bits
711
713
715
717
720
VPS
SPS
PPS
Cabeçalho de Fatia
Dados de Imagem
723
724
Camada
Camada
731
sps_video_parameter_set_id
725
726
vps_independent_layer_flag
733
vps_video_parameter_set_id
735
Imagens
Imagens
732
727
728
nuh_layer_id
Fatias
Fatias
742
741
Unidades não VCL NAL
Unidades VCL NAL
701
Extração de
Subfluxo de bits
729
Subfluxo de bits
713
715
717
720
SPS
PPS
Cabeçalho de Fatia
Dados de Imagem
731
sps_video_parameter_set_id
Camada
723
Imagens
725
Fatias
727

FIG. 7

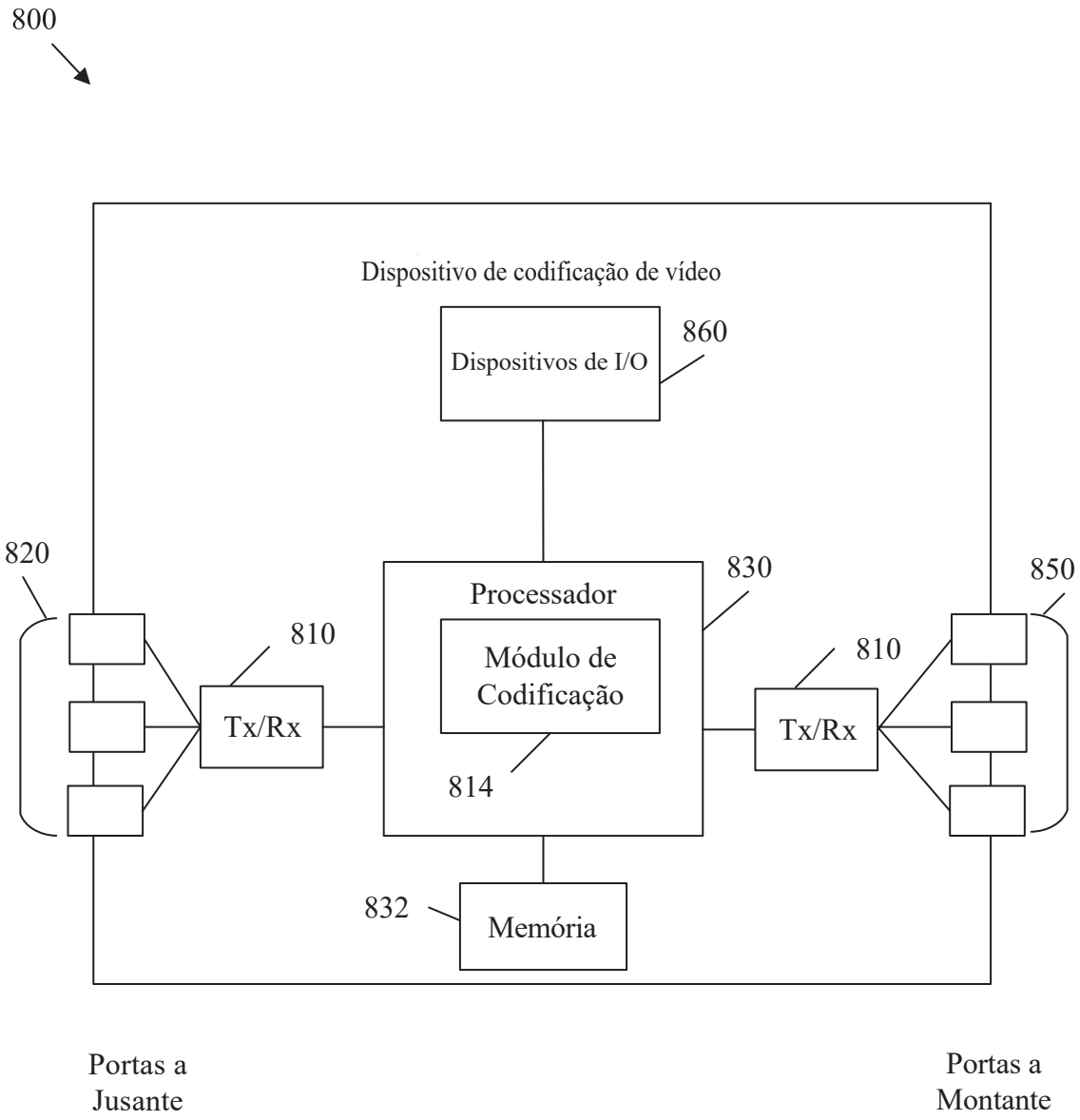

800
Dispositivo de codificação de vídeo
Dispositivos de I/O
860
820
830
850
Processador
810
Módulo de Codificação
810
Tx/Rx
Tx/Rx
814
832
Memória
Portas a Jusante
Portas a Montante

FIG. 8

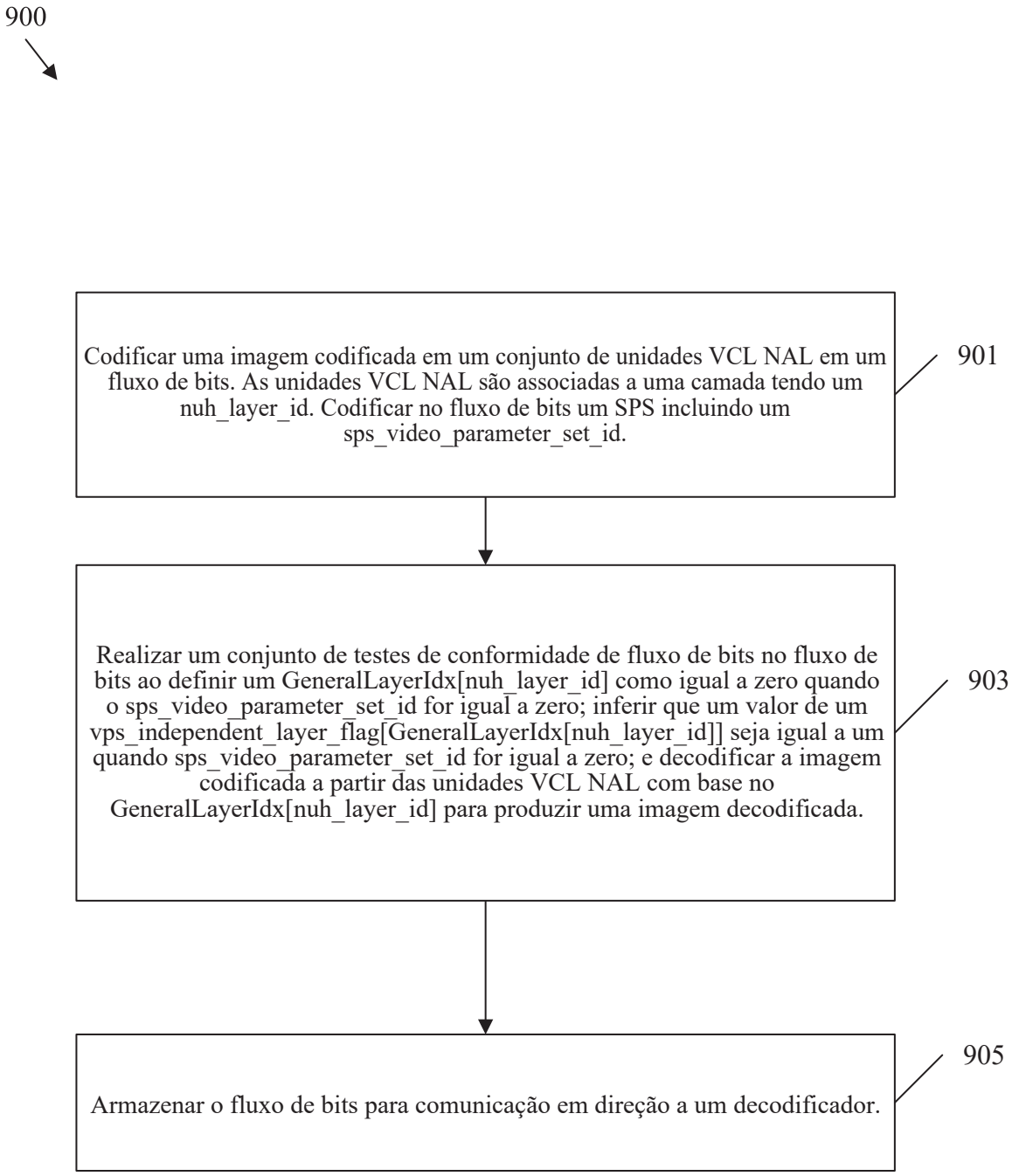

900
Codificar uma imagem codificada em um conjunto de unidades VCL NAL em um fluxo de bits. As unidades VCL NAL são associadas a uma camada tendo um nuh_layer_id. Codificar no fluxo de bits um SPS incluindo um sps_video_parameter_set_id.
901
Realizar um conjunto de testes de conformidade de fluxo de bits no fluxo de bits ao definir um GeneralLayerIdx[nuh_layer_id] como igual a zero quando o sps_video_parameter_set_id for igual a zero; inferir que um valor de um vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] seja igual a um quando sps_video_parameter_set_id for igual a zero; e decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada.
903
Armazenar o fluxo de bits para comunicação em direção a um decodificador.
905

FIG. 9

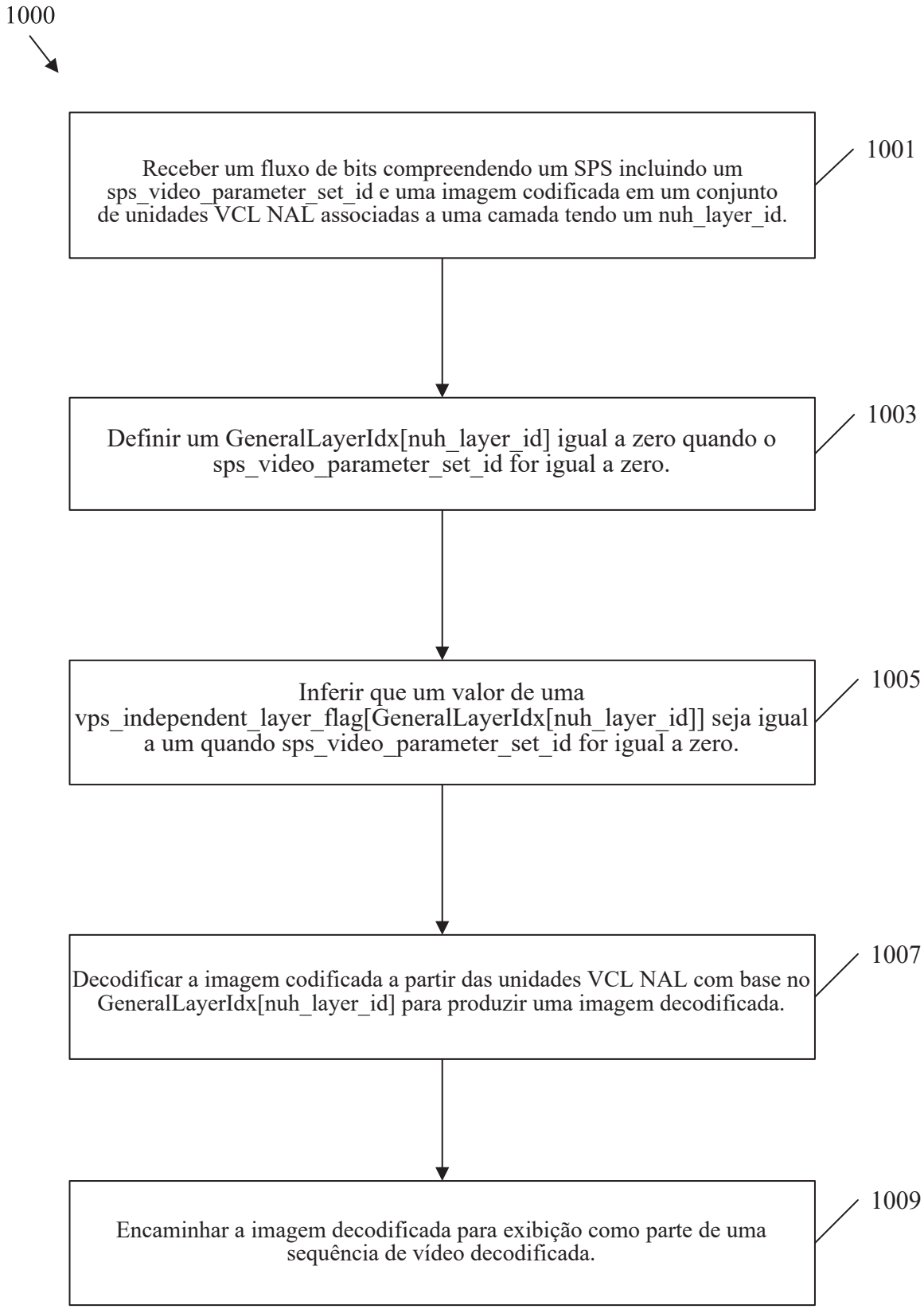

1000
Receber um fluxo de bits compreendendo um SPS incluindo um sps_video_parameter_set_id e uma imagem codificada em um conjunto de unidades VCL NAL associadas a uma camada tendo um nuh_layer_id.
1001
Definir um GeneralLayerIdx[nuh_layer_id] igual a zero quando o sps_video_parameter_set_id for igual a zero.
1003
Inferir que um valor de uma vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]] seja igual a um quando sps_video_parameter_set_id for igual a zero.
1005
Decodificar a imagem codificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada.
1007
Encaminhar a imagem decodificada para exibição como parte de uma sequência de vídeo decodificada.
1009

FIG. 10

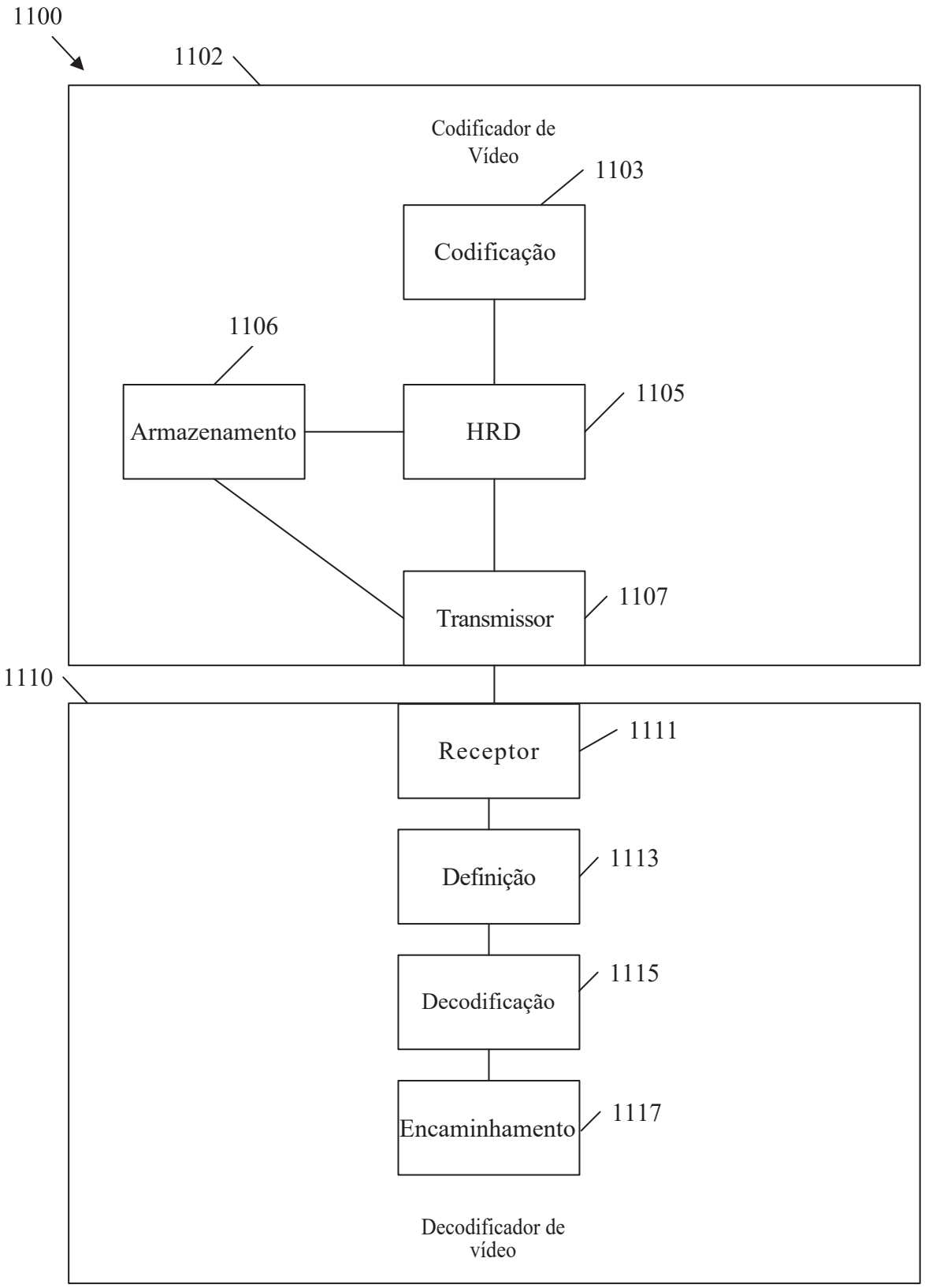

1100
1102
Codificador de Vídeo
1103
Codificação
1106
Armazenamento
HRD
1105
Transmissor
1107
1110
Receptor
1111
Definição
1113
Decodificação
1115
Encaminhamento
1117
Decodificador de vídeo

FIG. 11

RESUMO

“PREVENÇÃO DE ERRO EM EXTRAÇÃO DE SUBFLUXO DE BITS”

Um mecanismo de codificação de vídeo é revelado. O mecanismo inclui receber um fluxo de bits que compreende um conjunto de parâmetros de sequência (SPS) que inclui um identificador de conjunto de parâmetros de vídeo de SPS (sps_video_parameter_set_id). O fluxo de bits também compreende uma imagem codificada em um conjunto de unidades de camada de abstração de rede (NAL) de camada de codificação de vídeo (VCL) associadas com uma camada que tem um identificador de camada de cabeçalho de unidade NAL (nuh_layer_id). Um índice de camada geral correspondente ao nuh_layer_id (GeneralLayerIdx[nuh_layer_id]) é definido igual a zero quando o sps_video_parameter_set_id for igual a zero. A imagem codificada é decodificada a partir das unidades VCL NAL com base no GeneralLayerIdx[nuh_layer_id] para produzir uma imagem decodificada. A imagem decodificada é encaminhada para exibição como parte de uma sequência de vídeo decodificada.