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

(11) PI 0809510-8 B1

(22) Data do Depósito: 11/04/2008

(45) Data de Concessão: 08/10/2019

(54) Título: LADRILHAMENTO EM CODIFICAÇÃO E DECODIFICAÇÃO DE VÍDEO

(51) Int.Cl.: H04N 19/597; H04N 19/70; H04N 13/00; H04N 19/172; H04N 19/46; (...).

(52) CPC: H04N 19/597; H04N 19/70; H04N 13/0048; H04N 13/0059; H04N 19/46; (...).

(30) Prioridade Unionista: 12/04/2007 US 60/923,014; 20/04/2007 US 60/925,400.

(73) Titular(es): DOLBY INTERNATIONAL AB.

(72) Inventor(es): PURVIN BIBHAS PANDIT; PENG YIN; DONG TIAN.

(86) Pedido PCT: PCT US2008004747 de 11/04/2008

(87) Publicação PCT: WO 2008/127676 de 23/10/2008

(85) Data do Início da Fase Nacional: 07/10/2009

(57) Resumo: LADRILHAMENTO EM CODIFICAÇÃO E DECODIFICAÇÃO DE VÍDEO. São providas implementações que se referem, por exemplo, á visualização de ladrilhamento em codificação e decodificação de vídeo. Um método específico inclui acessar uma imagem de vídeo que inclui múltiplas imagens na imagem de vídeo acessada de ao menos uma das múltiplas imagens (824,826) e prover a informação acessada e a imagem de vídeo decodificada como saída (824,826). Algumas ouras implementações formatam ou processam a informação que indica como múltiplas imagens incluídas em uma única imagem de vídeo são combinadas na imagem de vídeo únca, e formatam ou processam uma representação codificada das múltiplas imagens combinadas.

"LADRILHAMENTO EM CODIFICAÇÃO E DECODIFICAÇÃO DE VÍDEO"

REFERÊNCIA REMISSIVA A PEDIDOS CORRELATOS

Esse pedido reivindica o benefício de cada um de: (1) Pedido Provisório dos Estados Unidos 60/923.014, depositado em 12 de abril de 2007, e intitulado "Multiview Information" (Nº do Dossiê do Advogado PU070078), e (2) Pedido Provisório dos Estados Unidos 60/925.400, depositado em 20 de abril de 2007 e intitulado "View Tiling in MVC Coding" (Nº do Dossiê do Advogado PU070103). Cada um desses dois pedidos é aqui incorporado integralmente mediante referência.

CAMPO TÉCNICO

Os presentes princípios se referem geralmente à codificação e/ou decodificação de vídeo.

ANTECEDENTES

Os fabricantes de vídeo podem utilizar uma arquitetura de disposição ou ladrilhamento de diferentes listas em um único quadro. As vistas podem ser então extraídas de seus locais respectivos e renderizadas.

SUMÁRIO

De acordo com um aspecto geral, é acessada uma imagem de vídeo que inclui múltiplas imagens combinadas em uma única imagem. Informação é acessada indicando como as múltiplas imagens, na imagem de vídeo acessada, são combinadas. A imagem de vídeo é decodificada para prover uma representação decodificada das múltiplas imagens combinadas. A informação acessada e a imagem de vídeo decodificada são providas como saída.

De acordo com outro aspecto geral, é gerada informação indicando como múltiplas imagens incluídas em uma imagem de vídeo são combinadas em uma única imagem. A imagem de vídeo é codificada para prover uma representação codificada das múltiplas imagens combinadas. A informação gerada e a imagem de vídeo codificada são providas como saída.

De acordo com outro aspecto geral, um sinal ou uma estrutura de sinal inclui informação indicando como múltiplas imagens incluídas em uma única imagem de vídeo são combinadas na única imagem de vídeo. O sinal ou a estrutura de sinal inclui também uma representação codificada das múltiplas imagens combinadas.

De acordo com outro aspecto geral, uma imagem de vídeo é acessada a qual inclui múltiplas imagens combinadas em uma única imagem. Informação é acessada a qual indica como as múltiplas imagens na imagem de vídeo acessada são combinadas. A imagem de vídeo é decodificada para prover uma representação decodificada de ao menos uma das múltiplas imagens. A informação acessada e a representação decodificada são providas como saída.

De acordo com outro aspecto geral, é acessada uma imagem de vídeo a qual inclui

múltiplas imagens combinadas em uma única imagem. É acessada informação a qual indica como as múltiplas imagens na imagem de vídeo acessada são combinadas. A imagem de vídeo é decodificada para prover uma representação decodificada das múltiplas imagens combinadas. Entrada de usuário é recebida a qual seleciona ao menos uma das múltiplas imagens para exibição. Uma saída decodificada da ao menos uma imagem selecionada é provida, a saída decodificada sendo provida com base na informação acessada, na representação decodificada, e na entrada de usuário.

Os detalhes de uma ou mais implementações são apresentados nos desenhos anexos e na descrição abaixo. Mesmo se descritas de uma forma específica, deve ser evidente que as implementações podem ser configuradas ou incorporadas de diversas maneiras. Por exemplo, uma implementação pode ser realizada como um método, ou incorporada como um aparelho configurado para realizar um conjunto de operações, ou incorporada como um aparelho armazenando instruções para realizar um conjunto de operações, ou incorporada em um sinal. Outros aspectos e características se tornarão evidentes a partir da descrição detalhada seguinte, considerada em conjunto com os desenhos anexos e com as reivindicações.

DESCRIÇÃO RESUMIDA DOS DESENHOS

A Figura 1 é um diagrama mostrando um exemplo de quatro vistas ladrilhadas em um único quadro;

A Figura 2 é um diagrama mostrando um exemplo de quatro vistas viradas e ladrilhadas em um único quadro;

A Figura 3 mostra um diagrama de blocos para um codificador de vídeo ao qual podem ser aplicados os presentes princípios, de acordo com uma modalidade dos presentes princípios;

A Figura 4 mostra um diagrama de blocos para um decodificador de vídeo ao qual podem ser aplicados os presentes princípios, de acordo com uma modalidade dos presentes princípios;

A Figura 5 é um fluxograma para um método para codificar imagens para uma pluralidade de vistas utilizando o Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 6 é um fluxograma para um método para codificar imagens para uma pluralidade de vistas utilizando o Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 7 é um fluxograma para um método para codificar imagens para uma pluralidade de vistas e profundidades utilizando o Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 8 é um fluxograma para um método para decodificar uma pluralidade de

vistas e profundidades utilizando o Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 9 é um diagrama mostrando um exemplo de um sinal de profundidade, de acordo com uma modalidade dos presentes princípios;

A Figura 10 é um diagrama mostrando um exemplo de um sinal de profundidade adicionado com um ladrilho, de acordo com uma modalidade dos presentes princípios;

A Figura 11 é um diagrama mostrando um exemplo de cinco vistas ladrilhadas em um único quadro, de acordo com uma modalidade dos presentes princípios.

A Figura 12 é um diagrama de blocos para um codificador de Codificação de Vídeo de Múltiplas Vistas (MVC) exemplar ao qual podem ser aplicados os presentes princípios, de acordo com uma modalidade dos presentes princípios;

A Figura 13 é um diagrama de blocos para um decodificador de Codificação de Vídeo de Múltiplas Vistas (MVC) exemplar ao qual podem ser aplicados os presentes princípios, de acordo com uma modalidade dos presentes princípios;

A Figura 14 é um fluxograma para um método para processar imagens para uma pluralidade de vistas em preparação para codificação das imagens utilizando a extensão de codificação de vídeo de múltiplas vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 15 é um fluxograma para um método para codificar imagens para uma pluralidade de vistas utilizando a extensão de Codificação de Vídeo de Múltiplas Vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 16 é um fluxograma para um método para processar imagens para uma pluralidade de vistas em preparação para a decodificação das imagens utilizando a extensão de codificação de vídeo de múltiplas vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 17 é um fluxograma para um método para decodificar imagens para uma pluralidade de vistas utilizando a extensão de Codificação de Vídeo de Múltiplas Vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 18 é um fluxograma para um método para processar imagens para uma pluralidade de vistas e profundidades em preparação para codificação das imagens utilizando a extensão de Codificação de Vídeo de Múltiplas Vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 19 é um fluxograma para um método para codificar imagens para uma pluralidade de vistas e profundidades utilizando a extensão de Codificação de Vídeo de Múltiplas Vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 20 é um fluxograma para um método para processar imagens para uma

pluralidade de vistas e profundidades em preparação para decodificação das imagens utilizando a extensão de Codificação de Vídeo de Múltiplas Vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 21 é um fluxograma para um método para decodificar imagens para uma pluralidade de vistas e profundidades utilizando a extensão de Codificação de Vídeo de Múltiplas Vistas (MVC) do Padrão MPEG-4 AVC, de acordo com uma modalidade dos presentes princípios;

A Figura 22 é um diagrama mostrando exemplos de ladrilhamento no nível de pixel, de acordo com uma modalidade dos presentes princípios; e

A Figura 23 mostra um diagrama de blocos para um dispositivo de processamento de vídeo, ao qual podem ser aplicados os presentes princípios, de acordo com uma modalidade dos presentes princípios.

DESCRIÇÃO DETALHADA

Várias implementações são dirigidas aos métodos e aparelho para visualizar ladrilhamento em codificação e decodificação de vídeo. Assim será considerado que aqueles versados na técnica poderão conceber vários arranjos que, embora não sejam aqui descritos ou mostrados explicitamente, incorporam os princípios e são incluídos dentro de seu espírito e escopo.

Todos os exemplos e linguagem condicional, aqui citados, pretendem ter propósitos pedagógicos para auxiliar o leitor no entendimento dos presentes princípios e conceitos contribuídos pelo inventor(es) para favorecer a técnica, e devem ser considerados como sendo sem limitação a tais exemplos e condições especificamente citados.

Além disso, todas as declarações feitas aqui citando princípios, aspectos e modalidades dos presentes princípios, assim como seus exemplos específicos, pretendem abranger seus equivalentes não apenas estruturais como também funcionais. Adicionalmente, pretende-se que tais equivalentes incluam os equivalentes atualmente conhecidos assim como os equivalentes desenvolvidos no futuro, isto é, quaisquer elementos desenvolvidos que realizem a mesma função, independente da estrutura.

Assim, por exemplo, será considerado por aqueles versados na técnica que os diagramas de blocos apresentados aqui representam vistas conceptuais de conjuntos de circuitos ilustrativos incorporando os presentes princípios. Similarmente, será considerado que quaisquer fluxogramas, diagramas, diagramas de transição de estado, pseudocódigo, e semelhante, representam vários processos que podem ser substancialmente representados em meios legíveis por computador e assim executados por um computador ou processador, seja o tal computador ou processador mostrado ou não explicitamente.

As funções dos vários elementos mostrados nas figuras podem ser providas através do uso de hardware dedicado assim como hardware capaz de executar software em

associação com software apropriado. Quando providas por um processador, as funções podem ser providas por um único processador dedicado, por um único processador compartilhado, ou por uma pluralidade de processadores individuais, alguns dos quais podem ser compartilhados. Além disso, o uso explícito do termo "processador" ou "controlador" não deve ser considerado como se referindo exclusivamente a hardware capaz de executar software, e pode incluir implicitamente, sem limitação, hardware de processador de sinal digital ("DSP"), memória de leitura ("ROM") para armazenar software, memória de acesso aleatório ("RAM"), e meio de armazenamento não-volátil.

Outro hardware, convencional e/ou especial, também pode ser incluído. Similarmente, quaisquer comutadores mostrados nas figuras são apenas conceptuais. A função dos mesmos pode ser realizada através da operação de lógica de programa, através de lógica dedicada, através da interação de controle de programa e lógica dedicada, ou até mesmo manualmente, a técnica específica podendo ser selecionável pelo implementador conforme entendido mais especificamente a partir do contexto.

Nas reivindicações quaisquer elementos expressos como um meio para realizar uma função especificada pretende abranger qualquer forma de realizar essa função incluindo, por exemplo, a) uma combinação de elementos de circuito que realizam essa função ou b) software em qualquer forma, incluindo, portanto, firmware, microcódigo ou semelhante, combinado com conjunto de circuito apropriado para executar esses softwares para realizar a função. Os presentes princípios conforme definidos por tais reivindicações residem no fato de que as funcionalidades providas pelos vários meios citados são combinadas e unidas na forma na qual demandada pelas reivindicações. Considera-se assim que quaisquer meios que possam prover essas funcionalidades são equivalentes àqueles aqui mostrados.

Referência no relatório descritivo a "uma modalidade" ou ("uma implementação") ou "uma modalidade" (ou "uma implementação") dos presentes princípios significa que um aspecto, estrutura, característica, específica, e assim por diante descrito em conexão com a modalidade é incluído em ao menos uma modalidade dos presentes princípios. Assim, o surgimento da frase "em uma modalidade" ou "em alguma modalidade" aparecendo em vários locais por todo o relatório descritivo não se referem todos à mesma modalidade.

Deve ser considerado que o uso dos termos "e/ou" e "ao menos um de", por exemplo, nos casos de "A e/ou B" e "ao menos um de A e B", pretende abranger a seleção da primeira opção relacionada (A) apenas, ou a seleção da segunda opção relacionada (B) apenas, ou a seleção de ambas as opções (A e B). Como um exemplo adicional, nos casos de "A, B, e/ou C" e "ao menos um de A, B e C", tal frase pretende abranger a seleção apenas da primeira opção relacionada (A), ou a seleção apenas da segunda opção relacionada (B), ou a seleção apenas da terceira opção relacionada (C), ou a seleção apenas da primeira e da segunda opção relacionada (A e B), ou a seleção apenas da primeira e terceira op-

ção relacionada (A e C), ou a seleção apenas da segunda e terceira opção relacionada (B e C), ou a seleção de todas as três opções (A e B e C). Isso pode ser estendido, como prontamente evidente para aqueles de conhecimento comum nessas técnicas e nas técnicas relacionadas, portanto, aos muitos itens relacionados.

Além disso, deve ser considerado que embora uma ou mais modalidades dos presentes princípios sejam descritas aqui com relação ao padrão MPEG-4 AVC, os presentes princípios não são limitados apenas a esse padrão e, assim, podem ser utilizados com relação a outros padrões, recomendações, e extensões dos mesmos, particularmente padrões de codificação de vídeo, recomendações, e extensões dos mesmos, incluindo extensões do padrão MPEG-4 AVC, enquanto mantendo o espírito dos presentes princípios.

Além disso, deve ser considerado que embora uma ou mais diferentes modalidades dos presentes princípios sejam descritas aqui com relação à extensão de codificação de vídeo de múltiplas vistas do padrão MPEG-4 AVC, os presentes princípios não são limitados apenas a essa extensão e/ou esse padrão e, assim, podem ser utilizados com relação a outros padrões de codificação de vídeo, recomendações e extensões dos mesmos relacionados à codificação de vídeo de múltiplas vistas, enquanto mantendo o espírito dos presentes princípios. Codificação de vídeo de múltiplas vistas (MVC) é a estrutura de compactação para a codificação de sequências de múltiplas vistas. Uma sequência de Codificação de Vídeo de Múltiplas Vistas (MVC) é um conjunto de duas ou mais sequências de vídeo que capturam a mesma cena de um ponto de vista diferente.

Além disso, deve ser considerado que embora uma ou mais diferentes modalidades dos presentes princípios sejam descritas aqui as quais utilizam informação de profundidade com relação ao conteúdo de vídeo, os presentes princípios não são limitados a tais modalidades e, assim, outras modalidades podem ser implementadas as quais não utilizam informação de profundidade, enquanto mantendo o espírito dos presentes princípios.

Adicionalmente, conforme aqui usado, "sintaxe de alto nível" se refere à sintaxe presente no fluxo de bits que reside hierarquicamente acima da camada de macrobloco. Por exemplo, sintaxe de alto nível, conforme aqui usada, pode se referir, mas não é limitada à sintaxe no nível de cabeçalho de seção, nível de Informação de Otimização Suplementar (SEI), nível de Conjunto de Parâmetros de Imagem (PPS), nível de Conjunto de Parâmetros de Sequência (SPS), Conjunto de Parâmetros de Vista (VPS), e nível de cabeçalho de unidade de Camada de Abstração de Rede (NAL).

Na presente implementação de codificação de multivídeo (MVC) com base na recomendação H.264 da Organização Internacional para Padronização/Comissão Eletrotécnica Internacional (ISO/IEC) da União de Telecomunicação Internacional/Padrão de Codificação de Vídeo Avançado (AVC) Parte 10 do Grupo 4 de Especialistas Cinematográficos (MPEG-4) (em seguida o "Padrão MPEG-4 AVC"), o software de referência obtém predição

de múltiplas vistas mediante codificação de cada vista com um único codificador considerando as referências de vistas cruzadas. Cada vista é codificada como um fluxo de bits, separado pelo codificador em sua resolução original e posteriormente todos os fluxos de bits são combinados para formar um único fluxo de bits o qual é então decodificado. Cada vista produz uma saída decodificada YUV separada.

Outra abordagem para predição de múltiplas vistas envolve agrupar um conjunto de vistas em pseudovistas. Em um exemplo dessa abordagem, podemos ladrilhar as imagens a partir de cada N vistas de um total de M vistas (amostradas ao mesmo tempo) em um quadro maior ou em um superquadro com possível amostragem descendente ou outras operações. De acordo agora com a Figura 1, um exemplo das quatro vistas ladrilhadas em um único quadro é indicado geralmente pelo numeral de referência 100. Todas as quatro vistas estão em sua orientação normal.

De acordo com a Figura 2, um exemplo de quatro vistas viradas e ladrilhadas em um único quadro é indicado geralmente pelo numeral de referência 200. A vista superior esquerda está em sua orientação normal. A vista superior direita está virada horizontalmente. A vista inferior esquerda está virada verticalmente. A vista inferior direita está virada horizontalmente assim como verticalmente. Assim, se houver quatro vistas, então uma imagem a partir de cada vista é disposta em um superquadro como um ladrilho. Isso resulta em uma única sequência de entrada não-codificada com uma grande resolução.

Alternativamente, podemos amostrar descendentemente a imagem para produzir uma resolução menor. Assim, criamos múltiplas sequências cada uma das quais inclui diferentes vistas que são ladrilhadas juntas. Cada tal sequência forma então uma pseudovista, onde cada pseudovista inclui N diferentes vistas ladrilhadas. A Figura 1 mostra uma pseudovista, e a Figura 2 mostra outra pseudovista. Essas pseudovistas podem ser então codificadas utilizando-se os padrões de codificação de vídeo existentes tal como o Padrão ISO/IEC MPEG-2 e o Padrão MPEG-4 AVC.

Ainda outra abordagem para predição de múltiplas vistas envolve simplesmente a codificação das diferentes vistas usando independentemente um novo padrão e, após a decodificação, ladrilhamento das vistas conforme exigido pelo aparelho de reprodução.

Adicionalmente, em outra abordagem, as vistas também podem ser ladrilhadas na forma de pixel. Por exemplo, em uma super vista que é composta de quatro vistas, pixel (x, y) pode ser a partir da vista 0, enquanto que pixel (x+1, y) pode ser a partir da vista 1, pixel (x, y+1) pode ser a partir da vista 2, e pixel (x+1, y+1) pode ser a partir da vista 3.

Muitos fabricantes de vídeo utilizam tal estrutura de arranjo ou ladrilhamento de diferentes vistas em um único quadro e então a extração das vistas a partir de seus locais respectivos e renderização das mesmas. Em tais casos, não há forma padrão para determinar se o fluxo de bits tem tal propriedade. Assim, se um sistema utiliza o método de ladrilhar

imagens de vistas diferentes em um quadro grande, então o método de extrair as diferentes vistas é registrado.

Contudo, não há uma forma padrão para determinar se o fluxo de bits tem tal propriedade. Propomos sintaxe de alto nível para facilitar o aparelho de renderização ou reprodução a extrair tal informação para auxiliar na exibição ou outro processamento posterior. Também é possível que as subimagens tenham diferentes resoluções e certa amostragem ascendente pode ser necessária para eventualmente renderizar a vista. O usuário pode pretender ter o método de amostragem ascendente também indicado na sintaxe de alto nível. Adicionalmente, os parâmetros para mudar o foco de profundidade também podem ser transmitidos.

Em uma modalidade, propomos uma mensagem de Informação de Otimização Suplementar (SEI) nova para sinalizar informação de múltiplas vistas em um Padrão MPEG-4 AVC compatível com o fluxo de bits onde cada imagem inclui subimagens que pertencem a uma vista diferente. A modalidade se destina, por exemplo, à exibição fácil e conveniente de fluxos de vídeo de múltiplas vistas em monitores tridimensionais (3D) os quais podem usar tal estrutura. O conceito pode ser estendido a outros padrões de codificação de vídeo e recomendações sinalizando tal informação usando sintaxe de alto nível.

Além disso, em uma modalidade, propomos um método de sinalização de como dispor as vistas antes de serem enviadas para o codificador e/ou decodificador de vídeo de múltiplas vistas. Vantajosamente, a modalidade pode levar a uma implementação simplificada da codificação de múltiplas vistas, e pode se beneficiar da eficiência de codificação. Certas vistas podem ser colocadas juntas e formar uma pseudovista ou supervista e então a supervista ladrilhada é tratada como uma vista normal por um codificador e/ou decodificador de vídeo de múltiplas vistas, comum, por exemplo, de acordo com a implementação baseada no Padrão MPEG-4 AVC atual de codificação de vídeo de múltiplas vistas. Um novo sinalizador é proposto na extensão Conjunto de Parâmetros de Sequência (SPS) de codificação de vídeo de múltiplas vistas para sinalizar o uso da técnica de pseudovistas. A modalidade é pretendida para a exibição fácil e conveniente dos fluxos de vídeo de múltiplas vistas em monitores 3D que utilizam tal estrutura.

Codificação/decodificação utilizando um padrão/recomendação de codificação/decodificação de vídeo de vista única

Na presente implementação de codificação de multivídeo (MVC), com base na recomendação H.264 da Organização Internacional para Padronização/Comissão Eletrotécnica Internacional (ISO/IEC) da União de Telecomunicação Internacional/Padrão de Codificação de Vídeo Avançado (AVC) Parte 10 do Grupo 4 de Especialistas Cinematográficos (MPEG-4), setor de telecomunicações (ITU-T) (em seguida o "Padrão MPEG-4 AVC"), o software de referência obtém predição de multivista mediante codificação de cada vista com

um único codificador e considerando as referências de vistas cruzadas. Cada vista é codificada como um fluxo de bits separado pelo codificador em sua resolução original e posteriormente todos os fluxos de bits são combinados para formar um único fluxo de bits que é então decodificado. Cada vista produz uma saída decodificada YUV separada.

Outra abordagem para predição de multivista envolve ladrilhar as imagens a partir de cada vista (amostradas ao mesmo tempo) em um quadro maior ou em um superquadro com uma possível operação de amostragem descendente. Voltando-se agora para a Figura 1, é indicado um exemplo de quatro vistas ladrilhadas em um único quadro geralmente indicado pelo numeral de referência 100. Voltando-se agora para a Figura 2, um exemplo de quatro vistas viradas e ladrilhadas em um único quadro é indicado geralmente pelo numeral de referência 200. Assim, se há quatro vistas, então uma imagem para cada vista é disposta em um superquadro como um ladrilho. Isso resulta em uma única sequência de entrada não-codificada com uma grande resolução. Esse sinal pode ser então codificado utilizando-se os padrões de codificação de vídeo existentes tal como o padrão ISO/IEC MPEG-2 e o padrão MPEG-4 AVC.

Ainda outra abordagem para predição de multivista envolve simplesmente codificar as diferentes vistas independentemente utilizando um novo padrão e, após a decodificação, ladrilhar as vistas conforme exigido pelo aparelho de reprodução.

Muitos fabricantes de vídeo utilizam tal estrutura de disposição ou ladrilhamento de diferentes vistas em um único quadro e então a extração das vistas a partir de seus locais respectivos e renderização das mesmas. Em tais casos, não há forma padrão de se determinar se o fluxo de bits tem tal propriedade. Assim, se um sistema utiliza o método de ladrilhar imagens de vistas diferentes em um quadro grande, então o método de extrair as diferentes vistas é registrado.

Voltando-se agora para a Figura 3, um codificador de vídeo capaz de realizar codificação de vídeo de acordo com o padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 300.

O codificador de vídeo 300 inclui um armazenador de ordenamento de quadro 310 tendo uma saída em comunicação de sinal com uma entrada de não-inversão de um combinador 385. Uma saída do combinador 385 é conectada em comunicação de sinal com uma primeira entrada de um transformador e quantizador 325. Uma saída do transformador e quantizador 325 é conectada em comunicação de sinal com uma primeira entrada de um codificador de entropia 345 e uma primeira entrada de um transformador de inversão e quantizador de inversão 350. Uma saída do codificador de entropia 345 é conectada em comunicação de sinal com uma primeira entrada de não-inversão de um combinador 390. Uma saída do combinador 390 é conectada em comunicação de sinal com uma primeira entrada de um armazenador de saída 335.

Uma primeira saída de um controlador de codificador 305 é conectada em comunicação de sinal com uma segunda entrada do armazenador de ordenamento de quadro 310, uma segunda entrada do transformador de inversão e quantizador de inversão 350, uma entrada de um módulo de decisão de tipo de imagem 315, uma entrada de um módulo de decisão de tipo de macrobloco (MB-tipo) 320, uma segunda entrada de um módulo intrapredição 360, uma segunda entrada de um filtro de desblocagem 365, uma primeira entrada de um compensador de movimento 370, uma primeira entrada de um estimador de movimento 375, e uma segunda entrada de um armazenador de imagem de referência 380.

Uma segunda saída do controlador de codificador 305 é conectada em comunicação de sinal com uma primeira entrada de um meio de inserção de Informação de Otimização Suplementar (SEI) 330, uma segunda entrada do transformador e quantizador 325, uma segunda entrada do codificador de entropia 345, uma segunda entrada do armazenador de saída 335, e uma entrada do meio de inserção de Conjunto de Parâmetros de Sequência (SPS) e Conjunto de Parâmetros de Imagem (PPS) 340.

Uma primeira saída do módulo de decisão de tipo de imagem 315 é conectada em comunicação de sinal com uma terceira entrada de um armazenador de ordenamento de quadro 310. Uma segunda saída do módulo de decisão de tipo de imagem 315 é conectada em comunicação de sinal com uma segunda entrada de um módulo de decisão de tipo de macrobloco 320.

Uma saída do meio de inserção de Conjunto de Parâmetros de Sequência (SPS) e Conjunto de Parâmetros de Imagem (PPS) 340 é conectada em comunicação de sinal com uma terceira entrada de não-inversão do combinador 390. Uma saída do meio de inserção SEI 330 é conectada em comunicação de sinal com uma segunda entrada de não-inversão do combinador 390.

Uma saída do quantizador de inversão e transformador de inversão 350 é conectada em comunicação de sinal com uma primeira entrada de não-inversão de um combinador 319. Uma saída do combinador 319 é conectada em comunicação de sinal com uma primeira entrada do módulo de intrapredição 360 e uma primeira entrada do filtro de desblocagem 365. Uma saída do filtro de desblocagem 365 é conectada em comunicação de sinal com uma primeira entrada de um armazenador de imagem de referência 380. Uma saída do armazenador de imagem de referência 380 é conectada em comunicação de sinal com uma segunda entrada do estimador de movimento 375 e com uma primeira entrada de um compensador de movimento 370. Uma primeira saída do estimador de movimento 375 é conectada em comunicação de sinal com uma segunda entrada do compensador de movimento 370. Uma segunda saída do estimador de movimento 375 é conectada em comunicação de sinal com uma terceira entrada do codificador de entropia 345.

Uma saída do compensador de movimento 370 é conectada em comunicação de

sinal com uma primeira entrada de um comutador 397. Uma saída do módulo de intrapredição 360 é conectada em comunicação de sinal com uma segunda entrada do comutador 397. Uma saída do módulo de decisão de tipo de macrobloco 320 é conectada em comunicação de sinal com uma terceira entrada do comutador 397 para prover uma entrada de controle para o comutador 397. A terceira entrada do comutador 397 determina se a entrada de "dados" do comutador (em comparação com a entrada de controle, isto é, a terceira entrada) deve ou não ser provida pelo compensador de movimento 370 ou pelo módulo de intrapredição 360. A saída do comutador 397 é conectada em comunicação de sinal com uma segunda entrada de não-inversão do combinador 319 e com uma entrada de inversão do combinador 385.

As entradas do armazenador de ordenamento de quadro 310 e do controlador de codificador 105 estão disponíveis como entradas do codificador 300, para receber uma imagem de entrada 301. Além disso, uma entrada do meio de inserção de Informação de Otimização Suplementar (SEI) 330 está disponível como uma entrada do codificador 300, para receber metadados. Uma saída do armazenador de saída 335 está disponível como uma saída do codificador 300, para emitir um fluxo de bits.

Voltando-se agora para a Figura 4, um decodificador de vídeo capaz de realizar decodificação de vídeo de acordo com o Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 400.

O decodificador de vídeo 400 inclui um armazenador de entrada 410 tendo uma saída conectada em comunicação de sinal com uma primeira entrada do decodificador de entropia 445. Uma primeira saída do decodificador de entropia 445 é conectada em comunicação de sinal com uma primeira entrada de um transformador de inversão e quantizador de inversão 450. Uma saída do transformador de inversão e do quantizador de inversão 450 é conectada em comunicação de sinal com uma segunda entrada de não-inversão de um combinador 425. Uma saída do combinador 425 é conectada em comunicação de sinal com uma segunda entrada de um filtro de desblocagem 465 e uma primeira entrada de um módulo de intrapredição 460. Uma segunda saída do filtro de desblocagem 465 é conectada em comunicação de sinal com uma primeira entrada de um armazenador de imagem de referência 480. Uma saída do armazenador de imagem de referência 480 é conectada em comunicação de sinal com uma segunda entrada de um compensador de movimento 470.

Uma segunda saída do decodificador de entropia 445 é conectada em comunicação de sinal com uma terceira entrada do compensador de movimento 470 e uma primeira entrada do filtro de desblocagem 465. Uma terceira saída do decodificador de entropia 445 é conectada em comunicação de sinal com uma entrada de um controlador de decodificador 405. Uma primeira saída do controlador de decodificador 405 é conectada em comunicação de sinal com uma segunda entrada do decodificador de entropia 445. Uma segunda saída

do controlador de decodificador 405 é conectada em comunicação de sinal com uma segunda entrada do transformador de inversão e quantizador de inversão 450. Uma terceira saída do controlador de decodificador 405 é conectada em comunicação de sinal com uma terceira entrada do filtro de desblocagem 465. Uma quarta saída do controlador de decodificador 405 é conectada em comunicação de sinal com uma segunda entrada do módulo de intrapredição 460, com uma primeira entrada do compensador de movimento 470, e com uma segunda entrada do armazenador de imagem de referência 480.

Uma saída do compensador de movimento 470 é conectada em comunicação de sinal com uma primeira entrada de um comutador 497. Uma saída do módulo de intrapredição 460 é conectada em comunicação de sinal com uma segunda entrada do comutador 497. Uma saída do comutador 497 é conectada em comunicação de sinal com uma primeira entrada de não-inversão do combinador 425.

Uma entrada do armazenador de entrada 410 está disponível como uma entrada do decodificador 400, para receber um fluxo de bits de entrada. Uma primeira saída do filtro de desblocagem 465 está disponível como uma saída do decodificador 400, para emitir uma imagem de saída.

Voltando-se para a Figura 5, um método exemplar para codificar imagens para uma pluralidade de vistas utilizando o Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 500. O método 500 inclui um bloco inicial 502 que passa o controle para um bloco de função 504. O bloco de função 504 arranja cada vista em uma instância temporal específica como uma subimagem no formato de ladrilho, e passa o controle para um bloco de função 506. O bloco de função 506 estabelece um elemento de sintaxe num_codec_views_minus1, e passa o controle para um bloco de função 508. O bloco de função 508 estabelece os elementos de sintaxe org_pic_width_in_mbs_minus1 e org_pic_height_in_mbs_minus1, e passa o controle para um bloco de função 510. O bloco de função 510 estabelece uma variável i igual a zero, e passa o controle para o bloco de decisão 512. O bloco de decisão 512 determina se a variável i é ou não menor do que o número de vistas. Se for afirmativo, então o controle é passado para um bloco de função 514. Caso contrário, o controle é passado para um bloco de função 524.

O bloco de função 514 estabelece um elemento de sintaxe view_id[i], e passa o controle para um bloco de função 516. O bloco de função 516 estabelece um elemento de sintaxe num_parts[view_id[i], e passa o controle para um bloco de função 518. O bloco de função 518 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 520. O bloco de decisão 520 determina se o valor atual da variável j é ou não menor do que o valor atual do elemento de sintaxe num_parts[view_id[i]. Se for afirmativo, então o controle é passado para um bloco de função 522. Caso contrário, o controle é passado para um bloco de função 528.

O bloco de função 522 estabelece os seguintes elementos de sintaxe, incrementa a variável j, e então retorna o controle para o bloco de decisão 520:

depth_flag[view_id[i]][j];flip_dir[view_id[i]][j]; loc_left_offset[view_id[i]][j]; loc_top_offset[view_id[i]][j]; frame_crop_left_offset[view_id[i]][j]; frame_crop_right_offset[view_id[i]][j]; frame_crop_top_offset[view_id[i]][j]; e frame_crop_bottom_offset[view_id[i]][j].

O bloco de função 528 estabelece um elemento de sintaxe upsample_view_flag[view_id[i]], e passa o controle para um bloco de decisão 530. O bloco de decisão 530 determina se o valor atual do elemento de sintaxe upsample_view_flag[view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 532. Caso contrário, o controle é passado para um bloco de decisão 534.

O bloco de função 532 estabelece um elemento de sintaxe upsample_filter[view_id[i]], e passa o controle para o bloco de decisão 534.

O bloco de decisão 534 determina se o valor atual do elemento de sintaxe upsample_filter[view_id[i]] é ou não igual a três. Se for, então o controle é passado para um bloco de função 536. Caso contrário, o controle é passado para um bloco de função 540.

O bloco de função 536 estabelece os seguintes elementos de sintaxe e passa o controle para um bloco de função 538: vert_dim[view_id[i]]; hor_dim[view_id[i]]; e quantizer[view_id[i]].

O bloco de função 538 estabelece os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 540.

O bloco de função 540 incrementa a variável i, e retorna o controle para o bloco de decisão 512.

O bloco de função 524 grava esses elementos de sintaxe em ao menos um de: Conjunto de Parâmetros de Sequência (SPS), Conjunto de Parâmetros de Imagem (PPS), mensagem de Informação de Otimização Suplementar (SEI), cabeçalho de unidade de Camada de Abstração de Rede (NAL), e cabeçalho de seção, e passa o controle para um bloco de função 526. O bloco de função 526 codifica cada imagem utilizando o Padrão MPEG-4 AVC ou outro codec de vista única, e passa o controle para um bloco final 599.

Voltando-se para a Figura 6, um método exemplar para decodificar imagens para uma pluralidade de vistas utilizando o Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 600.

O método 600 inclui um bloco inicial 602 que passa o controle para um bloco de função 604. O bloco de função 604 analisa os elementos de sintaxe seguintes a partir de ao menos um de Conjunto de Parâmetros de Sequência (SPS), Conjunto de Parâmetros de Imagem (PPS), mensagem de Informação de Otimização Suplementar (SEI), cabeçalho de unidade de Camada de Abstração de Rede (NAL), e cabeçalho de seção, e passa o controle

para um bloco de função 606. O bloco de função 606 analisa um elemento de sintaxe num_codec_views_minus1, e passa o controle para um bloco de função 608. O bloco de função 608 analisa os elementos de sintaxe org_pic_width_in_mbs_minus1 e org_pic_heigth_in_mbs_minus1 e passa o controle para um bloco de função 610. O bloco de função 610 estabelece uma variável i igual a zero, e passa o controle para o bloco de decisão 612. O bloco de decisão 612 determina se a variável i é ou não menor do que o número de vistas. Se for, então o controle é passado para um bloco de função 614. Caso contrário, o controle é passado para um bloco de função 624.

O bloco de função 614 analisa um elemento de sintaxe view_id[i], e passa o controle para um bloco de função 616. O bloco de função 616 analisa um elemento de sintaxe num_parts_minus1[view_id[i]], e passa o controle para um bloco de função 618. O bloco de função 618 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 620. O bloco de decisão 629 determina se o valor atual da variável j é ou não menor do que o valor atual do elemento de sintaxe num_parts[view_id[i]]. Se for, então o controle é passado para um bloco de função 622. Caso contrário, o controle é passado para um bloco de função 628.

O bloco de função 622 analisa os seguintes elementos de sintaxe, incrementa a variável j, e então retorna o controle para o bloco de decisão 620:

depth_flag[view_id[i]]j];flip_dir[view_id[i]][j]; loc_left_offset[view_id[i]][j]; loc_top_offset[view_id[i]][j]; frame_crop_left_offset[view_id[i]][j]; frame_crop_right_offset[view_id[i]][j]; frame_crop_top_offset[view_id[i]][j]; e frame_crop_bottom_offset[view_id[i]][j].

O bloco de função 628 analisa um elemento de sintaxe upsample_view_flag[view_id[i]], e passa o controle para um bloco de decisão 630. O bloco de decisão 630 determina se o valor atual do elemento de sintaxe upsample_view_flag[view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 632. Caso contrário, o controle é passado para um bloco de decisão 634.

O bloco de função 632 analisa um elemento de sintaxe upsample_filter[view_id[i]], e passa o controle para o bloco de decisão 634.

O bloco de decisão 634 determina se o valor atual do elemento de sintaxe upsample_filter[view_id[i]] é ou não igual a três. Se for, então o controle é passado para um bloco de função 636. Caso contrário, o controle é passado para um bloco de função 640.

O bloco de função 636 analisa os seguintes elementos de sintaxe e passa o controle para um bloco de função 638: vert_dim[view_id[i]]; hor_dim[view_id[i]]; e quantizer[view_id[i]].

O bloco de função 638 analisa os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 640.

O bloco de função 640 incrementa a variável i, e retorna o controle para o bloco de decisão 612.

O bloco de função 624 decodifica cada imagem utilizando o padrão MPEG-4 AVC ou outro codec de vista única, e passa o controle para um bloco de função 626. O bloco de função 626 separa cada vista a partir da imagem utilizando a sintaxe de alto nível, e passa o controle para um bloco final 699.

Voltando-se para a Figura 7, um método exemplar para codificar imagens para uma pluralidade de vistas e profundidades utilizando o Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 700.

O método 700 inclui um bloco inicial 702 que passa o controle para um bloco de função 704. O bloco de função 704 arranja cada vista e profundidade correspondente em uma instância temporal específica como uma subimagem no formato de ladrilho, e passa o controle para um bloco de função 706. O bloco de função 706 estabelece um elemento de sintaxe num_coded_views_minus1, e passa o controle para um bloco de função 708. O bloco de função 708 estabelece os elementos de sintaxe org_pic_width_in_mbs_minus1 e org_pic_height_in_mbs_minus1, e passa o controle para um bloco de função 710. O bloco de função 710 estabelece uma variável i igual a zero, e passa o controle para um bloco de decisão 712. O bloco de decisão 712 determina se a variável i é ou não inferior ao número de vistas. Se for, então o controle é passado para um bloco de função 714. Caso contrário, o controle é passado para um bloco de função 724.

O bloco de função 714 estabelece um elemento de sintaxe view_id[i], e passa o controle para um bloco de função 716. O bloco de função 716 estabelece um elemento de sintaxe num_parts[view_id[i]], e passa o controle para um bloco de função 718. O bloco de função 718 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 720. O bloco de decisão 720 determina se o valor atual da variável j é ou não menor do que o valor atual do elemento de sintaxe num_parts[view_id[i]]. Se for, então o controle é passado para um bloco de função 722. Caso contrário, o controle é passado para um bloco de função 728.

O bloco de função 722 estabelece os seguintes elementos de sintaxe, incrementa a variável j, e então retorna o controle para o bloco de decisão 720:

depth_flag[view_id[i]][j]; flip_dir[view_id[i]][j]; loc_left_offset[view_id[i]][j]; loc_top_offset[view_id[i]][j]; frame_crop_left_offset[view_id[i]][j]; frame_crop_right_offset[view_id[i]][j]; frame_crop_top_offset[view_id[i]][j]; e frame_crop_bottom_offset[view_id[i]][j].

O bloco de função 728 estabelece um elemento de sintaxe upsample_view_flag[view_id[i]], e passa o controle para um bloco de decisão 730. O bloco de decisão 730 determina se o valor atual do elemento de sintaxe upsample_view_flag[view_id[i]] é

ou não igual a um. Se for, então o controle é passado para um bloco de função 732. Caso contrário, o controle é passado para um bloco de decisão 734.

O bloco de função 732 estabelece um elemento de sintaxe upsample_filter[view_id[i]], e passa o controle para o bloco de decisão 734.

O bloco de decisão 734 determina se o valor atual do elemento de sintaxe upsample_filter[view_id[i]] é ou não igual a três. Se for, então o controle é passado para um bloco de função 736. Caso contrário, o controle é passado para um bloco de função 740.

O bloco de função 736 estabelece os seguintes elementos de sintaxe e passa o controle para um bloco de função 738: vert_dim[view_id[i]]; hor_dim[view_id[i]]; e quantizer[view_id[i]].

O bloco de função 738 estabelece os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 740.

O bloco de função 740 incrementa a variável i, e retorna o controle para o bloco de decisão 712.

O bloco de função 724 grava esses elementos de sintaxe para ao menos um de: Conjunto de Parâmetros de Sequência (SPS), Conjunto de Parâmetros de Imagem (PPS), mensagem de Informação de Otimização Suplementar (SEI), cabeçalho de unidade de Camada de Abstração de Rede (NAL), e cabeçalho de seção, e passa o controle para um bloco de função 726. O bloco de função 726 codifica cada imagem utilizando o Padrão MPEG-4 AVC ou outro codec de vista única, e passa o controle para um bloco final 799.

Voltando-se para a Figura 8, um método exemplar para decodificar imagens para uma pluralidade de vistas e profundidades utilizando o Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 800.

O método 800 inclui um bloco inicial 802 que passa o controle para um bloco de função 804. O bloco de função 804 analisa os seguintes elementos de sintaxe a partir de ao menos um de Conjunto de Parâmetros de Sequência (SPS), Conjunto de Parâmetros de Imagem (PPS), mensagem de Informação de Otimização Suplementar (SEI), cabeçalho de unidade de Camada de Abstração de Rede (NAL), e cabeçalho de seção, e passa o controle para um bloco de função 806. O bloco de função 806 analisa um elemento de sintaxe num_codec_views_minus1 e passa o controle para um bloco de função 808. O bloco de função 808 analisa os elementos de sintaxe org_pic_width_in_mbs_minus1 e org_pic_height_in_mbs_minus1, e passa o controle para um bloco de função 810. O bloco de função 810 estabelece uma variável i igual a zero, e passa o controle para um bloco de decisão 812. O bloco de decisão 812 determina se a variável i é ou não menor do que o número de vistas. Se for, então o controle é passado para um bloco de função 814. Caso contrário, o controle é passado para um bloco de função 824.

O bloco de função 814 analisa um elemento de sintaxe view_id[i], e passa o contro-

le para um bloco de função 816. O bloco de função 816 analisa um elemento de sintaxe num_parts_minus1[view_id[i]], e passa o controle para um bloco de função 818. O bloco de função 818 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 820. O bloco de decisão 820 determina se o valor atual da variável j é ou não inferior ao valor atual do elemento de sintaxe num_parts[view_id[i]]. Se for, então o controle é passado para um bloco de função 822. Caso contrário, o controle é passado para um bloco de função 828.

O bloco de função 822 analisa os seguintes elementos de sintaxe, incrementa a variável j, e então retorna o controle para o bloco de decisão 820:

depth_flag[view_id[i]][j]; flip_dir[view_id[i]][j]; loc_left_offset[view_id[i]][j]; loc_top_offset[view_id[i]][j]; frame_crop_left_offset[view_id[i]][j]; frame_crop_right_offset[view_id[i]][j]; frame_crop_top_offset[view_id[i]][j]; e frame_crop_bottom_offset[view_id[i]][j].

O bloco de função 828 analisa um elemento de sintaxe upsample_view_flag[view_id[i]], e passa o controle para um bloco de decisão 830. O bloco de decisão 830 determina se o valor atual do elemento de sintaxe upsample_view_flag[view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 832. Caso contrário, o controle é passado para um bloco de decisão 834.

O bloco de função 832 analisa um elemento de sintaxe upsample_filter[view_id[i]], e passa o controle para o bloco de decisão 834.

O bloco de decisão 834 determina se o valor atual do elemento de sintaxe upsample_filter[view_id[i]] é ou não igual a três. Se for, então o controle é passado para um bloco de função 836. Caso contrário, o controle é passado para um bloco de função 840.

O bloco de função 836 analisa os seguintes elementos de sintaxe e passa o controle para um bloco de função 838: vert_dim[view_id[i]]; hor_dim[view_id[i]]; e quantizer[view_id[i]].

O bloco de função 838 analisa os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 840.

O bloco de função 840 incrementa a variável i, e retorna o controle para o bloco de decisão 812.

O bloco de função 824 decodifica cada imagem utilizando o Padrão MPEG-4 AVC ou outro codec de vista única, e passa o controle para um bloco de função 826. O bloco de função 826 separa cada vista e a profundidade correspondente a partir da imagem utilizando a sintaxe de alto nível, e passa o controle para um bloco de função 827. O bloco de função 827 potencialmente realiza a síntese das vistas utilizando a vista extraída e os sinais de profundidade, e passa o controle para um bloco final 899.

Com relação à profundidade usada nas Figuras 7 e 8, a Figura 9 mostra um exem-

plo de um sinal de profundidade 900, onde a profundidade é provida como um valor de pixel para cada local correspondente de uma imagem (não mostrada). Adicionalmente, a Figura 10 mostra um exemplo de dois sinais de profundidade incluídos em um ladrilho 1000. A porção superior direita do ladrilho 1000 é um sinal de profundidade tendo valores de profundidade correspondendo à imagem na porção superior esquerda do ladrilho 1000. A porção inferior direita do ladrilho 100 é um sinal de profundidade tendo valores de profundidade correspondendo à imagem na porção inferior esquerda do ladrilho 1000.

Voltando-se para a Figura 11, um exemplo das cinco vistas ladrilhadas em um único quadro é indicado geralmente pelo numeral de referência 1100. As quatro vistas superiores estão em uma orientação normal. A quinta vista também está em uma orientação normal, porém é dividida em duas porções ao longo da parte inferior do ladrilho 1100. Uma porção esquerda da quinta vista mostra o "topo" da quinta vista, e uma porção direita da quinta vista mostra a "parte inferior" da quinta vista.

Codificação/decodificação utilizando um padrão/recomendação de codificação/decodificação de vídeo de multivista

Voltando-se para a Figura 12, um codificador exemplar de Codificação de Vídeo de Multivista (MVC) é indicado geralmente pelo numeral de referência 1200. O codificador 1200 inclui um combinador 1205 tendo uma saída conectada em comunicação de sinal com uma entrada de um transformador 1210. Uma saída do transformador 1210 é conectada em comunicação de sinal com uma entrada do quantizador 1215. Uma saída do quantizador 1215 é conectada em comunicação de sinal com uma entrada de um codificador de entropia 1220 e uma entrada de um quantizador inverso 1225. Uma saída do quantizador inverso 1225 é conectada em comunicação de sinal com uma entrada de um transformador de inversão 1230. Uma saída do transformador de inversão 1230 é conectada em comunicação de sinal com uma primeira entrada de não-inversão de um combinador 1235. Uma saída do combinador 1235 é conectada em comunicação de sinal com uma entrada de um intra preditor 1245 e uma entrada de um filtro de desblocagem 1250. Uma saída do filtro de desblocagem 1250 é conectada em comunicação de sinal com uma entrada de um meio de armazenamento de imagem de referência 1255 (para a vista i). Uma saída do meio de armazenamento de imagem de referência 1255 é conectada em comunicação de sinal com uma primeira entrada de um compensador de movimento 1275 e uma primeira entrada de um estimador de movimento 1280. Uma saída do estimador de movimento 1280 é conectada em comunicação de sinal com uma segunda entrada do compensador de movimento 1275.

Uma saída de um meio de armazenamento de imagem de referência 1260 (para outras vistas) é conectada em comunicação de sinal com uma primeira entrada de um estimador de disparidade 1270 e uma primeira entrada de um compensador de disparidade 1265. Uma saída do estimador de disparidade 1270 é conectada em comunicação de sinal

com uma segunda entrada do compensador de disparidade 1265.

Uma saída do decodificador de entropia 1220 está disponível como uma saída do codificador 1200. Uma entrada de não-inversão do combinador 1205 está disponível como uma entrada do codificador 1200, e é conectada em comunicação de sinal com uma segunda entrada do estimador de disparidade 1270, e uma segunda entrada do estimador de movimento 1280. Uma saída de um comutador 1285 é conectada em comunicação de sinal com uma segunda entrada de não-inversão do combinador 1235 e com uma entrada de inversão do combinador 1205. O comutador 1285 inclui uma primeira entrada conectada em comunicação de sinal com uma saída do compensador de movimento 1275, uma segunda entrada conectada em comunicação de sinal com uma saída do compensador de disparidade 1265, e uma terceira entrada conectada em comunicação de sinal com uma saída do intra preditor 1245.

Um módulo de decisão de modo 1240 tem uma saída conectada ao comutador 1285 para controlar qual entrada é selecionada pelo comutador 1285.

Voltando-se para a Figura 13, um decodificador exemplar de Codificação de Vídeo de Multivista (MVC) é indicado geralmente pelo numeral de referência 1300. O decodificador 1300 inclui um decodificador de entropia 1305 tendo uma saída conectada em comunicação de sinal com uma entrada de um quantizador de inversão 1310. Uma saída do quantizador de inversão é conectada em comunicação de sinal com uma entrada de um transformador inverso 1315. Uma saída do transformador inverso 1315 é conectada em comunicação de sinal com uma primeira entrada de não-inversão de um combinador 1320. Uma saída do combinador 1320 é conectada em comunicação de sinal com uma entrada de um filtro de desblocagem 1325 e uma entrada de um intra preditor 1330. Uma saída do filtro de desblocagem 1325 é conectada em comunicação de sinal com uma entrada de um meio de armazenamento de imagem de referência 1340 (para vista i). Uma saída do meio de armazenamento de imagem de referência 1340 é conectada em comunicação de sinal com uma primeira entrada de um compensador de movimento 1335.

Uma saída de um meio de armazenamento de imagem de referência 1345 (para outras vistas) é conectada em comunicação de sinal com uma primeira entrada de um compensador de disparidade 1350.

Uma entrada do codificador de entropia 1305 está disponível como uma entrada para o decodificador 1300, para receber um fluxo de bits de resíduo. Além disso, uma entrada de um módulo de modo 1360 também está disponível como uma entrada para o decodificador 1300, para receber sintaxe de controle para controlar qual entrada é selecionada pelo comutador 1355. Adicionalmente, uma segunda entrada do compensador de movimento 1355 está disponível como uma entrada do decodificador 1300, para receber vetores de movimento. Além disso, uma segunda entrada do compensador de disparidade 1350 está

disponível como uma entrada para o decodificador 1300, para receber vetores de disparidade.

Uma saída de um comutador 1355 é conectada em comunicação de sinal com uma segunda entrada de não-inversão do combinador 1320. Uma primeira entrada do comutador 1355 é conectada em comunicação de sinal com uma saída do compensador de disparidade 1350. Uma segunda entrada do comutador 1355 é conectada em comunicação de sinal com uma saída do compensador de movimento 1335. Uma terceira entrada do comutador 1355 é conectada em comunicação de sinal com uma saída do intra preditor 1330. Uma saída do módulo de modo 1360 é conectada em comunicação de sinal com o comutador 1355 para controlar qual entrada é selecionada pelo comutador 1355. Uma saída do filtro de desblocagem 1325 está disponível como uma saída do decodificador 1300.

Voltando-se para a Figura 14, um método exemplar para processar imagens para uma pluralidade de vistas em preparação para codificar as imagens utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 1400.

O método 1400 inclui um bloco inicial 1405 que passa o controle para um bloco de função 1410. O bloco de função 1410 arranja cada uma das n vistas, entre um total de M vistas, em uma instância temporal específica como uma superimagem no formato de ladrilho, e passa o controle para um bloco de função 1415. O bloco de função 1415 estabelece um elemento de sintaxe num_coded_views_minus1, e passa o controle para um bloco de função 1420. O bloco de função 1420 estabelece um elemento de sintaxe view_id[i] para todas as vistas (num_coded_views_minus1 + 1), e passa o controle para um bloco de função 1425. O bloco de função 1425 estabelece a informação de dependência de referência entre vistas para imagens de âncora, e passa o controle para um bloco de função 1430. O bloco de função 1430 estabelece a informação de dependência de referência entre vistas para imagens de não-âncora, e passa o controle para um bloco de função 1435. O bloco de função 1435 estabelece um elemento de sintaxe pseudo_view_present_flag, e passa o controle para um bloco de decisão 1440. O bloco de decisão 1440 determina se o valor atual do elemento de sintaxe pseudo_view_present_flag é ou não verdadeiro. Se for, então o controle é passado para um bloco de função 1445. Caso contrário, o controle é passado para um bloco final 1499.

O bloco de função 1445 estabelece os seguintes elementos de sintaxe, e passa o controle para um bloco de função 1450: tiling_mode; org_pic_width_in_mbs_minus1; e org_pic_height_in_mbs_minus1. O bloco de função 1450 demanda um elemento de sintaxe pseudo_view_info(view_id) para cada vista codificada, e passa o controle para o bloco final 1499.

Voltando-se para a Figura 15, um método exemplar para codificar imagens para

uma pluralidade de vistas usando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 1500.

O método 1500 inclui um bloco inicial 1502 que tem um parâmetro de entrada pseudo_view_id e passa o controle para um bloco de função 1504. O bloco de função 1504 estabelece um elemento de sintaxe num_sub_views_minus1, e passa o controle para um bloco de função 1506. O bloco de função 1506 estabelece uma variável i igual a zero, e passa o controle para um bloco de decisão 1508. O bloco de decisão 1508 determina se a variável i é ou não menor do que o número de subvistas. Se for, então o controle é passado para o bloco de função 1510. Caso contrário, o controle é passado para um bloco de função 1520.

O bloco de função 1520 estabelece um elemento de sintaxe sub_view_id[i], e passa o controle para um bloco de função 1512. O bloco de função 1512 estabelece um elemento de sintaxe num_parts_minus1[sub_view_[id]], e passa o controle para um bloco de função 1514. O bloco de função 1514 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 1516. O bloco de decisão 1516 determina se a variável j é ou não inferior ao elemento de sintaxe num_parts_minus1[sub_view_[id]]. Se for, então o controle é passado para um bloco de função 1518. Caso contrário, o controle é passado para um bloco de decisão 1522.

O bloco de função 1518 estabelece os seguintes elementos de sintaxe, incrementa a variável j, e retorna o controle para o bloco de decisão 1516:

loc_left_offset[sub_view_id[i]][j]; loc_top_offset[sub_view_id[i]][j]; frame_crop_left_offset[sub_view_id[i]][j]; frame_crop_right_offset[sub_view_id[i]][j]; frame_crop_top_offset[sub_view_id[i]][j]; e frame_crop_bottom_offset[sub_view_id[i][j].

O bloco de função 1520 codifica a imagem atual para a vista atual utilizando codificação de vídeo de multivista (MVC), e passa o controle para um bloco final 1599.

O bloco de decisão 1522 determina se um elemento de sintaxe tiling_mode é ou não igual a zero. Se for, então o controle é passado para um bloco de função 1524. Caso contrário, o controle é passado para um bloco de função 1538.

O bloco de função 1524 estabelece um elemento de sintaxe flip_dir[sub_view_id[i]] e um elemento de sintaxe upsample_view_flag[sub_view_id[i]], e passa o controle para um bloco de decisão 1526. O bloco de decisão 1526 determina se o valor atual do elemento de sintaxe upsample_view_flag[sub_view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 1528. Caso contrário, o controle é passado para um bloco de decisão 1530.

O bloco de função 1528 estabelece um elemento de sintaxe upsample_flag[sub_view_id[i]], e passa o controle para o bloco de decisão 1530. O bloco de decisão 1530 determina se um valor do elemento de sintaxe upsample_flag[sub_view_id[i]] é ou

não igual a três. Se for, o controle é passado para um bloco de função 1532. Caso contrário, o controle é passado para um bloco de função 1536.

O bloco de função 1532 estabelece os seguintes elementos de sintaxe, e passa o controle para um bloco de função 1534: vert_dim[sub_view_id[i]]; hor_dim[sub_view_id[i]]; e quantizer[sub_view_id[i]]. O bloco de função 1534 estabelece os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 1536.

O bloco de função 1536 incrementa a variável i, e retorna o controle para o bloco de decisão 1508. O bloco de função 1538 estabelece um elemento de sintaxe pixel_dist_x[sub_view_id[i]] e o elemento de sintaxe flip_dist_y[sub_view_id[i]], e passa o controle para um bloco de função 1540. O bloco de função 1540 estabelece a variável j igual a zero, e passa o controle para um bloco de decisão 1542. O bloco de decisão 1542 determina se o valor atual da variável j é ou não menor do que o valor atual do elemento de sintaxe num_parts[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 1544. Caso contrário, o controle é passado para o bloco de função 1536.

O bloco de função 1544 estabelece um elemento de sintaxe num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]], e passa o controle para um bloco de função 1546. O bloco de função 1546 estabelece os coeficientes para todos os filtros de ladrilhamento de pixel, e passa o controle para o bloco de função 1536.

Voltando-se para a Figura 16, um método exemplar para processar imagens para uma pluralidade de vistas em preparação para decodificação das imagens utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 1600.

O método 1600 inclui um bloco inicial 1605 que passa o controle para um bloco de função 1615. O bloco de função 1615 analisa um elemento de sintaxe num_coded_views_minus1, e passa o controle para um bloco de função 1620. O bloco de função 1620 analisa um elemento de sintaxe view_id[i] para todas as vistas (num_coded_views_minus1 + 1), e passa o controle para um bloco de função 1625. O bloco de função 1625 analisa a informação de dependência de referência entre vistas para imagens de âncora, e passa o controle para um bloco de função 1630. O bloco de função 1630 analisa a informação de dependência de referência entre vistas para imagens de não-âncora, e passa o controle para um bloco de função 1635. O bloco de função 1635 analisa um elemento de sintaxe pseudo_view_present_flag, e passa o controle para um bloco de decisão 1640. O bloco de decisão 1640 determina se o valor atual do elemento de sintaxe pseudo_view_present_flag é ou não igual a verdadeiro. Se for, então o controle é passado para um bloco de função 1645. Caso contrário, o controle é passado para um bloco final 1699.

O bloco de função 1645 analisa os seguintes elementos de sintaxe, e passa o con-

trole para um bloco de função 1650: tiling_mode: org_pic_width_in_mbs_minus1; e org_pic_height_in_mbs_minus1. O bloco de função 1650 demanda um elemento de sintaxe pseudo_view_info(view_id) para cada vista codificada, e passa o controle para o bloco final 1699.

Voltando-se agora para a Figura 17, um método exemplar para decodificar imagem para uma pluralidade de vistas utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 1700.

O método 1700 inclui um bloco inicial 1702 que começa com o parâmetro de entrada pseudo_view_id e passa o controle para um bloco de função 1704. O bloco de função 1704 analisa um elemento de sintaxe num_sub_views_minus1, e passa o controle para um bloco de função 1706. O bloco de função 1706 estabelece uma variável i igual a zero, e passa o controle para um bloco de decisão 1708. O bloco de decisão 1708 determina se a variável i é ou não menor do que o número de subvistas. Se for, então o controle é passado para um bloco de função 1710. Caso contrário, o controle é passado para um bloco de função 1720.

O bloco de função 1710 analisa um elemento de sintaxe sub_view_id[i], e passa o controle para um bloco de função 1712. O bloco de função 1712 analisa um elemento de sintaxe num_parts_minus1[sub_view_id[i]], e passa o controle para um bloco de função 1714. O bloco de função 1714 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 1716. O bloco de decisão 1716 determina se a variável j é ou não menor do que o elemento de sintaxe num_parts_minus1[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 1718. Caso contrário, o controle é passado para um bloco de decisão 1722.

O bloco de função 1718 estabelece os seguintes elementos de sintaxe, incrementa a variável j, e retorna o controle para o bloco de decisão 1716:

loc_left_offset[sub_view_id[i]][j]; loc_top_offset[sub_view_id[i]][j]; frame_crop_left_offset[sub_view_id[i]]0]; frame_crop_right_offset[sub_view_id[i]]j]; frame_crop_top_offset[sub_view_id[i]][j]; e frame_crop_bottom_offset[sub_view_id[i][j].

O bloco de função 1720 decodifica a imagem atual para a vista atual utilizando codificação de vídeo de multivista (MVC), e passa o controle para um bloco de função 1721. O bloco de função 1721 separa cada vista da imagem utilizando a sintaxe de alto nível, e passa o controle para um bloco final 1799.

A separação de cada vista a partir da imagem decodificada é feita utilizando a sintaxe de alto nível indicada no fluxo de bits. Essa sintaxe de alto nível pode indicar o exato local e a possível orientação das vistas (e possível profundidade correspondente) presente na imagem.

O bloco de decisão 1722 determina se um elemento de sintaxe tiling_mode é igual

a zero. Se for, então o controle é passado para um bloco de função 1724. Caso contrário, o controle é passado para um bloco de função 1738.

O bloco de função 1724 analisa um elemento de sintaxe flip_dir[sub_view_id[i]]; e um elemento de sintaxe upsample_view_flag[sub_view_id[i]], e passa o controle para um bloco de decisão 1726. O bloco de decisão 1726 determina se o valor atual do elemento de sintaxe upsample_view_flag[sub_view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 1728. Caso contrário, o controle é passado para um bloco de decisão 1730.

O bloco de função 1728 analisa um elemento de sintaxe upsample_filter[sub_view_id[i]], e passa o controle para o bloco de decisão 1730. O bloco de decisão 1730 determina se o valor do elemento de sintaxe upsample_filter[sub_view_id[i]] é ou não igual a três. Se for, o controle é passado para um bloco de função 1732. Caso contrário, o controle é passado para um bloco de função 1736.

O bloco de função 1732 analisa os seguintes elementos de sintaxe, e passa o controle para um bloco de função 1734: vert_dim[sub_view_id[i]]; hor_dim[sub_view_id[i]]; e quantizer[sub_view_id[i]]. O bloco de função 1734 analisa os coeficientes de filtro para cada componente YUV e passa o controle para um bloco de função 1736.

O bloco de função 1736 incrementa a variável i, e retorna o controle para o bloco de decisão 1708.

O bloco de função 1738 analisa um elemento de sintaxe pixel_dist_x[sub_view_id[i]] e o elemento de sintaxe flip_dist_y[sub_view_id[i]], e passa o controle para um bloco de função 1740. O bloco de função 1740 estabelece a variável i igual a zero, e passa o controle para um bloco de função 1742. O bloco de função 1742 determina se o valor atual da variável j é ou não inferior ao valor atual do elemento de sintaxe num_parts[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 1744. Caso contrário, o controle é passado para um bloco de função 1736.

O bloco de função 1744 analisa um elemento de sintaxe num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]], e passa o controle para um bloco de função 1746. O bloco de função 1776 analisa os coeficientes para todos os filtros de ladrilhamento de pixel, e passa o controle para o bloco de função 1736.

Voltando-se agora para a Figura 18, um método exemplar para processar imagens para uma pluralidade de vistas e profundidades em preparação para codificar as imagens utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 1800.

O método 1800 inclui um bloco inicial 1805 que passa o controle para um bloco de função 1810. O bloco de função 1810 arranja todas as N vistas e mapas de profundidade, entre um total de M vistas e mapas de profundidade, em uma instância temporal e específi-

ca como uma superimagem no formato de ladrilho, e passa o controle para um bloco de função 1815. O bloco de função 1815 estabelece um elemento de sintaxe num_coded_views_minus1, e passa o controle para um bloco de função 1820. O bloco de função 1820 estabelece um elemento de sintaxe view_id[i] para todas as profundidades (num_coded_views_minus1 + 1) correspondendo a view_id[i], e passa o controle para um bloco de função 1825. O bloco de função 1825 estabelece a informação de dependência de referência entre vistas para imagens de profundidade de âncora, e passa o controle para um bloco de função 1830. O bloco de função 1830 estabelece a informação de dependência de referência entre vistas para imagens de profundidade de não-âncora, e passa o controle para um bloco de função 1835. O bloco de função 1835 estabelece um elemento de sintaxe pseudo_view_present_flag, e passa o controle para um bloco de decisão 1840. O bloco de decisão 1840 determina se o valor atual do elemento de sintaxe pseudo_view_present_flag é ou não igual a verdadeiro. Se for, então o controle é passado para um bloco de função 1845. Caso contrário, o controle é passado para um bloco final 1899.

O bloco de função 1845 estabelece os seguintes elementos de sintaxe, e passa o controle para um bloco de função 1850: tiling_mode; org_pic_width_in_mbs_minus1; e org_pic_height_in_mbs_minus1. O bloco de função 1850 demanda um elemento de sintaxe pseudo_view_info(view_id) para cada vista codificada, e passa o controle para o bloco final 1899.

Voltando-se para a Figura 19, um método exemplar para codificar imagens para uma pluralidade de vistas e profundidades utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 1900.

O método 1900 inclui um bloco inicial 1902 que passa o controle para um bloco de função 1904. O bloco de função 1904 estabelece um elemento de sintaxe num_sub_views_minus1, e passa o controle para um bloco de função 1906. O bloco de função 1906 estabelece uma variável i igual a zero, e passa o controle para um bloco de decisão 1908. O bloco de decisão 1908 determina se a variável i é ou não menor do que o número de subvistas. Se for, então o controle é passado para um bloco de função 1910. Caso contrário, o controle é passado para um bloco de função 1920.

O bloco de função 1910 estabelece um elemento de sintaxe sub_view_id[i], e passa o controle para um bloco de função 1912. O bloco de função 1912 estabelece um elemento de sintaxe num_parts_minus1[sub_view_id[i]], e passa o controle para um bloco de função 1914. O bloco de função 1914 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 1916. O bloco de decisão 1916 determina se a variável j é ou não menor do que o elemento de sintaxe num_parts_minus1[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 1918. Caso contrário, o controle é passado

para um bloco de função 1922.

O bloco de função 1918 estabelece os seguintes elementos de sintaxe, incrementa a variável j, e retorna o controle para o bloco de decisão 1916:

loc_left_offset[sub_view_id[i]][j]; loc_top_offset[sub_view_id[i]][j]; frame_crop_left_offset[sub_view_id[i]][j]; frame_crop_right_offset[sub_view_id[i]][j]; frame_crop_top_offset[sub_view_id[i]][j]; e frame_crop_bottom_offset[sub_view_id[i][j].

O bloco de função 1920 codifica a profundidade atual da vista atual utilizando codificação de vídeo de multivista (MVC), e passa o controle para um bloco final 1999. O sinal de profundidade pode ser codificado de forma semelhante à forma em que seu sinal de vídeo correspondente é codificado. Por exemplo, o sinal de profundidade para uma vista pode ser incluído em um ladrilho que inclui apenas outros sinais de profundidade, ou apenas sinais de vídeo, ou ambos, sinais de profundidade e de vídeo. O ladrilho (pseudo-vista) é então tratado como uma única vista para MVC, e há também presumivelmente outros ladrilhos que são tratados como outras vistas para MVC.

O bloco de decisão 1922 determina se um elemento de sintaxe tiling_mode é ou não igual a zero. Se for, então o controle é passado para um bloco de função 1924. Caso contrário, o controle é passado para um bloco de função 1938.

O bloco de função 1924 estabelece um elemento de sintaxe flip_dir[sub_view_id[i]] e um elemento de sintaxe upsample_view_flag[sub_view_id[i]], e passa o controle para um bloco de decisão 1926. O bloco de decisão 1926 determina se o valor atual do elemento de sintaxe upsample_view_flag[sub_view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 1928. Caso contrário, o controle é passado para um bloco de decisão 1930.

O bloco de função 1928 estabelece um elemento de sintaxe upsample_filter[sub_view_id[i]], e passa o controle para o bloco de decisão 1930. O bloco de decisão 1930 determina se um valor do elemento de sintaxe upsample_filter[sub_view_id[i]] é ou não igual a três. Se for, o controle é passado para um bloco de função 1932. Caso contrário, o controle é passado para um bloco de função 1936.

O bloco de função 1932 estabelece os seguintes elementos de sintaxe, e passa o controle para um bloco de função 1934: vert_dim[sub_view_id[i]]; hor_dim[sub_view_id[i]]; e quantizer[sub_view_id[i]]. O bloco de função 1934 estabelece os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 1936.

O bloco de função 1936 incrementa a variável i, e retorna o controle para o bloco de decisão 1908.

O bloco de função 1938 estabelece um elemento de sintaxe pixel_dist_x[sub_view_id[i]] e o elemento de sintaxe flip_dist_y[sub_view_id[i]], e passa o controle para um bloco de função 1940. O bloco de função 1940 estabelece uma variável j igual

a zero, e passa o controle para um bloco de decisão 1942. O bloco de decisão 1942 determina se o valor atual da variável j é ou não inferior ao valor atual do elemento de sintaxe num_parts[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 1944. Caso contrário, o controle é passado para o bloco de função 1936.

O bloco de função 1944 estabelece um elemento de sintaxe num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]], e passa o controle para um bloco de função 1946. O bloco de função 1946 estabelece os coeficientes para todos os filtros de ladrilhamento e pixel, e passa o controle para o bloco de função 1936.

Voltando-se para a Figura 20, um método exemplar para processar imagens para uma pluralidade de vistas e profundidades em preparação para decodificação das imagens utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 2000.

O método 2000 inclui um bloco inicial 2005 que passa o controle para um bloco de função 2015. O bloco de função 2015 analisa um elemento de sintaxe num_coded_views_minus1, e passa o controle para um bloco de função 2020. O bloco de função 2020 analisa um elemento de sintaxe view_id[i] para todas as profundidades (num_coded_views_minus1 + 1) correspondendo a view_id[i], e passa o controle para um bloco de função 2025. O bloco de função 2025 analisa a informação de dependência de referência entre vistas para imagens de profundidade de âncora, e passa o controle para um bloco de função 2030. O bloco de função 2030 analisa a informação de dependência de referência entre vistas para imagens de profundidade de não-âncora, e passa o controle para um bloco de função 2035. O bloco de função 2035 analisa um elemento de sintaxe pseudo_view_present_flag, e passa o controle para um bloco de decisão 2040. O bloco de decisão 2040 determina se o valor atual do elemento de sintaxe pseudo_view_present_flag é ou não igual a verdadeiro. Se for, então o controle é passado para um bloco de função 2045. Caso contrário, o controle é passado para um bloco final 2099.

O bloco de função 2045 analisa os seguintes elementos de sintaxe, e passa o controle para um bloco de função 2050 tiling_mode; org_pic_width_in_mbs_minus1; e org_pic_height_in_mbs_minus1. O bloco de função 2050 demanda um elemento de sintaxe pseudo_view_info(view_id) para cada vista codificada, e passa o controle para o bloco final 2099.

Voltando-se para a Figura 21, um método exemplar para decodificar imagem para uma pluralidade de vistas e profundidades utilizando a extensão de codificação de vídeo de multivista (MVC) do Padrão MPEG-4 AVC é indicado geralmente pelo numeral de referência 2100.

O método 2100 inclui um bloco inicial 2102 que começa com o parâmetro de entrada pseudo_view_id, e passa o controle para um bloco de função 2104. O bloco de função

2104 analisa um elemento de sintaxe num_sub_views_minus1, e passa o controle para um bloco de função 2106. O bloco de função 2106 estabelece uma variável i igual a zero, e passa o controle para um bloco de decisão 2108. O bloco de decisão 2108 determina se a variável i é ou não inferior ao número de subvistas. Se for, então o controle é passado para um bloco de função 2110. Caso contrário, o controle é passado para um bloco de função 2120.

O bloco de função 2110 analisa um elemento de sintaxe sub_view_id[i], e passa o controle para um bloco de função 2112. O bloco de função 2112 analisa um elemento de sintaxe num_parts_minus1[sub_view_id[i]], e passa o controle para um bloco de função 2114. O bloco de função 2114 estabelece uma variável j igual a zero, e passa o controle para um bloco de decisão 2116. O bloco de decisão 2116 determina se a variável j é ou não menor do que o elemento de sintaxe num_parts_minus1[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 2118. Caso contrário, o controle é passado para um bloco de decisão 2122.

O bloco de função 2118 estabelece os seguintes elementos de sintaxe, incrementa a variável j, e retorna o controle para o bloco de decisão 2116:

loc_left_offset[sub_view_id[i]][j]; loc_top_offset[sub_view_id[i]][j]; frame_crop_left_offset[sub_view_id[i]][j]; frame_crop_right_offset[sub_view_id[i]][j]; frame_crop_top_offset[sub_view_id[i]][j]; e frame_crop_bottom_offset[sub_view_id[i][j].

O bloco de função 2120 decodifica a imagem atual utilizando codificação de vídeo de multivista (MVC), e passa o controle para um bloco de função 2121. O bloco de função 2121 separa cada vista da imagem utilizando a sintaxe de alto nível, e passa o controle para um bloco final 2199. A separação de cada vista utilizando sintaxe de alto nível é conforme descrito anteriormente.

O bloco de decisão 2122 determina se um elemento de sintaxe tiling_mode é igual a zero. Se for, então o controle é passado para um bloco de função 2124. Caso contrário, o controle é passado para um bloco de função 2138.

O bloco de função 2124 analisa um elemento de sintaxe flip_dir[sub_view_id[i]] e

e um elemento de sintaxe upsample_view_flag[sub_view_id[i]], e passa o controle para um bloco de decisão 2126. O bloco de decisão 2126 determina se o valor atual do elemento de sintaxe upsample_view_flag[sub_view_id[i]] é ou não igual a um. Se for, então o controle é passado para um bloco de função 2128. Caso contrário, o controle é passado para um bloco de decisão 2130.

O bloco de função 2128 analisa um elemento de sintaxe upsample_filter[sub_view_id[i]], e passa o controle para o bloco de decisão 2130. O bloco de decisão 2130 determina se um valor do elemento de sintaxe upsample_filter[sub_view_id[i]] é ou não igual a três. Se for, o controle é passado para um bloco de função 2132. Caso contrário,

o controle é passado para um bloco de função 2136.

O bloco de função 2132 analisa os seguintes elementos de sintaxe, e passa o controle para um bloco de função 2134: vert_dim[sub_view_id[i]]; hor_dim[sub_view_id[i]]; e quantizer[sub_view_id[i]]. O bloco de função 2134 analisa os coeficientes de filtro para cada componente YUV, e passa o controle para o bloco de função 2136.

O bloco de função 2136 incrementa a variável i, e retorna o controle para o bloco de decisão 2108.

O bloco de função 2138 analisa um elemento de sintaxe pixel_dist_x[sub_view_id[i]] e o elemento de sintaxe flip_dist_y[sub_view_id[i]], e passa o controle para um bloco de função 2140. O bloco de função 2140 estabelece a variável j igual a zero, e passa o controle para um bloco de decisão 2142. O bloco de decisão 2142 determina se o valor atual da variável j é ou não menor do que o valor atual do elemento de sintaxe num_parts[sub_view_id[i]]. Se for, então o controle é passado para um bloco de função 2144. Caso contrário, o controle é passado para o bloco de função 2136.

O bloco de função 2144 analisa um elemento de sintaxe num_pixel_tiling_filter_coeffs_minus1 [sub_view_id[i]], e passa o controle para um bloco de função 2146. O bloco de função 2146 analisa os coeficientes para todos os filtros de ladrilhamento de pixel, e passa o controle para o bloco de função 2136.

Voltando-se para a Figura 22, exemplos de ladrilhamento no nível de pixel são indicados geralmente pelo numeral de referência 2200. A Figura 22 é descrita adicionalmente abaixo.

LADRILHAMENTO DE VISTA UTILIZANDO MPEG-4 AVC OU MVC

Uma aplicação de codificação de vídeo de multivista é TV de ponto de vista livre (ou FTV). Essa aplicação requer que o usuário possa se mover livremente entre duas ou mais vistas. Para realizar isso, as vistas "virtuais" entre duas vistas precisam ser interpoladas ou sintetizadas. Há vários métodos para realizar interpolação de vista. Um dos métodos utiliza profundidade para interpolação/síntese de vista.

Cada vista pode ter um sinal de profundidade associado. Assim, a profundidade pode ser considerada como sendo outra forma de sinal de vídeo. A Figura 9 mostra um exemplo de um sinal de profundidade 900. Para habilitar aplicações tais como FTV, o sinal de profundidade é transmitido junto com o sinal de vídeo. Na estrutura proposta de ladrilhamento, o sinal de profundidade também pode ser adicionado como um dos ladrilhos. A Figura 10 mostra um exemplo de sinais de profundidade adicionados como ladrilhos. Os sinais/ladrilhos de profundidade são mostrados no lado direito da Figura 10.

Quando a profundidade é codificada como um ladrilho do quadro inteiro, a sintaxe de alto nível deve indicar qual ladrilho é o sinal de profundidade de modo que o renderizador possa utilizar apropriadamente o sinal de profundidade.

No caso quando a sequência de entrada (tal como aquela mostrada na Figura 1) é codificada utilizando um codificador de Padrão MPEG-4 AVC (ou um codificador correspondendo a um padrão e/ou recomendação de codificação de vídeo diferente), a sintaxe de alto nível proposta pode estar presente, por exemplo, no Conjunto de Parâmetros de Sequência (SPS), no Conjunto de Parâmetros de Imagem (PPS), em um cabeçalho de seção, e/ou em uma mensagem de Informação de Otimização Suplementar (SEI). Uma modalidade do método proposto é mostrada na Tabela 1 onde a sintaxe está presente em uma mensagem de Informação de Otimização Suplementar (SEI).

No caso quando as sequências de entrada das pseudovistas (tais como aquelas mostradas na Figura 1) são codificadas utilizando a extensão de codificação de vídeo de multivista (MVC) do codificador de Padrão MPEG-4 AVC (ou um codificador correspondendo ao padrão de codificação de vídeo de multivista com relação a um padrão e/ou recomendação de codificação de vídeo diferente), a sintaxe de alto nível proposta pode estar presente no SPS, no PPS, no cabeçalho de seção, em uma mensagem SEI, ou em um perfil especificado. Uma modalidade do método proposto é mostrada na Tabela 1. A Tabela 1 mostra elementos de sintaxe presentes na estrutura de Conjunto de Parâmetros de Sequência (SPS), incluindo elementos de sintaxe propostos de acordo com uma modalidade dos presentes princípios

TABLE 1

seq_parameter_set_mvc_extension() {	c	Descritor
num_views_minus_1		ue(v)
para(i = 0; i <= num_views_minus_1; i++)		
view_id[i]		ue(v)
para(i = 0; i <= num_views_minus_1; i++) {		
num_anchor_refs_l0[i]		ue(v)
para(j = 0; j < num_anchor_refs_l0[i]; j++)		
anchor_ref_l0[i][j]		ue(v)
num_anchor_refs_H [i]		ue(v)
para(j = 0; j < num_anchor_refs_H[i]; j++)		
anchor_ref_H[i][j]		ue(v)
}		
para(i = 0; i <= num_views_minus_1; i++) {		
num_non_anchor_refs_l0[i]		ue(v)
para(j = 0; j < num_non_anchor_refs_l0[i]; j++)		
non_anchor_ref_l0[i][j]		ue(v)
num_non_anchor_refs_M [i]		ue(v)
para(j = 0; j < num_non_anchor_refs_H[i]; j++)		
non_anchor_refJ1 [i][j]		ue(v)
}		
pseudo_view_present_flag		u(l)
se (pseudo_view_present_flag) {		
tiling_mode		

org_pic_width_in_mbs_minus1		
org_pic_height_in_mbs_minus1		
para(i = 0; i < num_views_minus_1; i++)		
pseudo_view_info(i);		
}		
}		

A Tabela 2 mostra os elementos de sintaxe para o elemento de sintaxe pseudo_view_info da TABELA 1, de acordo com uma modalidade dos presentes princípios.

TABELA 2

pseudo_view_info (pseudo_view_id) {	C	Descritor
num_sub_views_minus_1[pseudo_view_id]	5	ue(v)
se (num_sub_views_minus_1 != 0) {		
para (i = 0; i < num_sub_views_minus_1[pseudo_view_id]; i++) {		
sub_view_id[i]	5	ue(v)
num_parts_minus1[sub_view_id[i]]	5	ue(v)
para(j = 0; j <= num_parts_minus1[sub_view_id[i]]; j++) {		
loc_left_offset[sub_view_id[i]] [j]	5	ue(v)
loc_top_offset[sub_view_id[i]] [j]	5	ue(v)
frame_cropJeft_offset[sub_view_id[i]] [j]	5	ue(v)
frame_crop_right_offset[sub_view_id[i]] [j]	5	ue(v)
frame_crop_top_offset[sub_view_id[i]] [j]	5	ue(v)
frame_crop_bottom_offset[sub_view_id[i]] [j]	5	ue(v)
}		
se (tiling_mode == 0) {		
flip_dir[sub_view_id[i][j]	5	u(2)
upsample_view_flag[sub_view_id[i]]	5	u(l)
se(upsample_view_flag[sub_view_id[i]])		
upsample_filter[sub_view_id[i]]	5	u(2)
se(upsample_fiter[sub_view_id[i]] == 3) {		
vert_dim[sub_view_id[i]]	5	ue(v)
hor_dim[sub_view_id[i]]	5	ue(v)
quantizer[sub_view_id[i]]	5	ue(v)
para (yuv= 0; yuv< 3; yuv++) {		
para (y = 0; y < vert_dim[sub_view_id[i]] -1; y ++) {		
para (x = 0; x < hor_dim[sub_view_id[i]] -1; x ++)		
filter_coeffs[sub_view_id[i]] [yuv][y][x]	5	se(v)
}		
}		
}		
} // se(tiling_mode == 0)		
ou se (tiling_mode == 1) {		
pixel_dist_x[sub_view_id[i]]		
pixel_dist_y[sub_view_id[i]]		
para(j = 0; j <= num_parts[sub_view_id[i]]; j++) {		
num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]][j]		

para (coeff_dx = 0; coeff_dx <=		
pixel_tiling_filter_coeffs[sub_view_id[i]][j]		
} // para (j = 0; j <= num_parts[sub_view_id[i]]; j++)		
} // ou se (tiling_mode == 1)		
} // para (i = 0; i < num_sub_views_minus_1; i++)		
} // se (num_sub_views_minus_1 != 0)		
}		

Semânticas dos elementos de sintaxe apresentados na TABELA 1 e TABELA 2

pseudo_view_present_flag igual a verdadeiro indica que certa vista é uma supervista de múltiplas subvistas.

tiling_mode igual a zero indica que as subvistas são ladrilhadas no nível de imagem. Um valor de 1 indica que o ladrilhamento é feito no nível de pixel.

A mensagem SEI nova poderia usar um valor para o tipo de carga útil SEI que não foi usado no Padrão MPEG-4 AVC ou uma extensão do Padrão MPEG-4 AVC. A mensagem SEI nova inclui vários elementos de sintaxe com as seguintes semânticas.

num_coded_views_minus1 plus 1 indica o número de vistas codificadas suportadas pelo fluxo de bits. O valor de num_coded_views_minus1 está no escopo de 0 a 1023, inclusive.

org_pic_width_in_mbs_minus1 plus 1 especifica a largura de uma imagem em cada vista em unidades de macroblocos.

A variável para a largura de imagem em unidades de macroblocos é derivada como a seguir:

PicWidthInMbs = org_pic_width_in_mbs_minus1 + 1

A variável para largura de imagem para o componente luma é derivada como a seguir:

PicWidthInSamplesL = PicWidthInMbs * 16

A variável para largura de imagem para os componentes croma é derivada como a seguir:

PicWidthInSamplesC = PicWidthInMbs * MbWidthC

org_pic_height_in__mbs_minus1 plus 1 especifica a altura de uma imagem em cada vista em unidades de macroblocos.

A variável para a altura de imagem em unidades de macroblocos é derivada como a seguir:

PicHeightInMbs = org_pic_height_in_mbs_minus1 + 1

A variável para altura de imagem para o componente luma é derivada como a seguir:

PicHeightInSamplesL = PicHeightInMbs * 16

A variável para altura de imagem para os componentes croma é derivada como a seguir:

PicHeightInSamplesC = PicHeightInMbs * MbHeightC

num_sub_views_minus1 plus 1 indica o número de subvistas codificadas incluídas na vista atual. O valor de num_coded_views_minus1 está no escopo de 0 a 1023, inclusive.

sub_view_id[i] especifica o sub_view_id da subvista com ordem de decodificação indicada por i.

num_parts[sub_view_id[i]] especifica o número de partes nas quais a imagem de sub_view_id[i] é dividida.

loc_left_offset[sub_view_id[i]][j] e loc_top_offset[sub_view_id[i]][j] especificam os locais em deslocamentos de pixels esquerda e superior, respectivamente, onde a parte atual j está localizada na imagem reconstruída final da vista com sub_view_id igual a sub_view_id[i].

view_id[i] especifica a view_id da vista com a ordem de codificação indicada por i.

frame_crop_left_offset[view_id[i]][j], frame_crop_right_offset[view_id[i]][j], frame_crop_top_offset[view_id[i]]j], e frame_crop_bottom_offset[view_id[i]][j] especifica as amostras da imagens na sequência de vídeo codificada que constituem parte de num_part j e view_id i, em termos de uma região retangular especificada nas coordenadas de quadro para emissão.

As variáveis CropUnitX e CropUnitY são derivadas como a seguir:

— Se chroma_format_idc for igual a 0, CropUnitX e CropUnitY são derivadas como a seguir:

CropUnitX = 1

CropUnitY = 2 -frame_mbs_only_flag

— Caso contrário (chroma_format_idc é igual a 1, 2, ou 3), CropUnitX e CropUnitY são derivadas como a seguir:

CropUnitX = SubWidthC

CropUnitY = SubHeightC * (2 - frame_mbs_only_flag)

O retângulo de recorte de quadro inclui as amostras luma com coordenadas de quadros horizontais a partir do seguinte:

CropUnitX * frame_crop_left_offset para PicWidthInSamplesL - (CropUnitX * frame_crop_right_offset + 1) e coordenadas de quadros verticais a partir de CropUnitY * frame_crop_top_offset para (16 * FrameHeightInMbs) - (CropUnitY * frame_crop_bottom_offset +1), inclusive. O valor de frame_crop_left_offset deve estar na faixa de 0 a (PicWidthInSamplesL / CropUnitX) - (frame_crop_right_offset + 1), inclusive; e o valor de frame_crop_top_offset deve estar na faixa de 0 a (16 * FrameHeightInMbs / CropUnitY) - (frame_crop_bottom_offset +1), inclusive.

Quando chroma_format_idc não é igual a 0, as amostras especificadas correspondentes dos dois sistemas croma são as amostras tendo coordenadas de quadro SubWidthC, y / SubHeightC), onde (x, y) são as coordenadas de quadro das amostras luma especificadas.

Para campos decodificados, as amostras especificadas do campo decodificado são as amostras que estão compreendidas dentro do retângulo especificado nas coordenadas de quadro.

num_parts[view_id[i]] especifica o número de partes nas quais a imagem de view_id[i] é dividida.

depth_flag[view_id[i]] especifica se a parte atual é ou não um sinal de profundidade. Se depth_flag for igual a 0, então a parte atual não é um sinal de profundidade. Se depth_flag for igual a 1, então a parte atual é um sinal de profundidade associado com a vista identificada por view_id[i].

flip_dir[sub_view_id[i]][j] especifica a direção de viragem da parte atual. flip_dir igual a 0 indica nenhuma viragem, flip_dir igual a 1 indica viragem em uma direção horizontal, flip_dir igual a 2 indica viragem em uma direção vertical, e flip_dir igual a 3 indica viragem nas direções horizontal e vertical.

flip_dir[view_id[i]][j] especifica a direção de viragem para a parte atual. flip_dir igual a 0 indica nenhuma viragem, flip_dir igual a 1 indica viragem em uma direção horizontal, flip_dir igual a 2 indica viragem na direção vertical, e flip_dir igual a 3 indica viragem nas direções horizontal e vertical.

loc_left_offset[view_id[i]][j], loc_top_offset[view_id[i]][j] especifica o local nos deslocamentos de pixels, onde a parte atual j está localizada na imagem reconstruída final da vista com view_id igual a view_id[i].

upsample_view_flag[view_id[i]] indica se a imagem pertencendo à vista especificada por view_id[i] precisa ser amostrada ascendentemente. upsample_view_flag[view_id[i]] igual a 0 especifica que a imagem com view_id igual a view_id[i] não será amostrada ascendentemente. upsample_view_flag[view_id[i]] igual a 1 especifica que a imagem com view_id igual a view_id[i] será amostrada ascendentemente.

upsample_filter[view_id[i]] indica o tipo de filtro que deve ser usado para amostragem ascendente. upsample_filter[view_id[i]] igual a 0 indica que o filtro AVC de 6 derivações deve ser usado, upsample_filter[view_id[i]] igual a 1 indica que o filtro SVC de 4 derivações deve ser usado, upsample_filter[view_id[i]] igual a 2 indica que o filtro bilinear deve ser usado, upsample_filter[view_id[i]] igual a 3 indica que os coeficientes de filtro especial são transmitidos. Quando upsample_filter[view_id[i]] não estiver presente ele é ajustado para 0. Nessa modalidade, utilizamos o filtro 2D customizado. O mesmo pode ser facilmente estendido para filtro 1D, e algum outro filtro não-linear.

vert_dim[view_id[i]] especifica a dimensão vertical do filtro 2D especial.

hor_dim[view_id[i]] especifica a dimensão horizontal do filtro 2D especial.

quantizer[view_id[i]] especifica o fator de quantização para cada coeficiente de do filtro.

filter_coeffs[view_id[i]] [yuv][y][x] especifica os coeficientes de filtro quantizados. yuv sinaliza o componente para o qual se aplicam os coeficientes de filtro. yuv igual a 0 especifica o componente Y, yuv igual a 1 especifica o componente U, e yuv igual a 2 especifica o componente V.

pixel_dist_x[sub_view_id[i]] e pixel_dist_y[sub_view_id[i]] especificam respectivamente a distância na direção horizontal e a direção vertical na pseudovista reconstruída final entre pixels adjacentes na vista com sub_view_id igual a sub_view_id[i].

num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i][j] mais um indica o número de coeficientes de filtro quando o modo de ladrilhamento é estabelecido igual a 1.

pixel_tiling_filter_coeffs[sub_view_id[i][j] sinaliza os coeficientes de filtro que são exigidos para representar um filtro que pode ser usado para filtrar a imagem ladrilhada.

Exemplos de ladrilhamento em nível de pixel

Voltando-se para a Figura 22, dois exemplos mostrando a composição de uma pseudovista mediante ladrilhamento de pixels a partir de quatro vistas são indicados respectivamente pelos numerais de referência 2210 e 2220, respectivamente. As quatro vistas são indicadas coletivamente pelo numeral de referência 2250. Os valores de sintaxe para o primeiro exemplo na Figura 22 são providos na TABELA 3 abaixo.

TABLE 3

pseudo_view_info (pseudo_view_id) {	Valor
num_sub_views_minus_1 [pseudo_view_id]	3
sub_view_id[0]	0
num_parts_minus1 [0]	0
loc_left_offset[0][0]	0
loc_top_offset[0][0]	0
pixel_dist_x[0][0]	0
pixel_dist_y[0][0]	0
sub_view_id[1]	0
num_parts_minus1[1]	0
loc_left_offset[1][0]	1
loc_top_offset[1][0]	0
pixel_dist_x[1][0]	0
pixel_dist_y[1][0]	0
sub_view_id[2]	0
num_parts_minus1 [2]	0
loc_left_offset[2][0]	0
loc_top_offset[2][0]	1
pixel_dist_x[2][0]	0
pixel_dist_y[2][0]	0

sub_view_id[3]	0
num_parts_minus1[3]	0
loc_left_offset[3][0]	1
loc_top_offset[3][0]	1
pixel_dist_x[3][0]	0
pixel_dist_y[3][0]	0

Os valores de sintaxe para o segundo exemplo em 22 são todos idênticos exceto os seguintes dois elementos de sintaxe: loc_left_offset[3][0] igual a 5 e loc_top_offset[3][0] igual a 3.

O deslocamento indica que os pixels correspondendo a uma vista devem começar em um certo local deslocado. Isso é mostrado na Figura 22 (2220). Isso pode ser feito, por exemplo, quando duas vistas produzem imagens nas quais objetos comuns parecem estar deslocados a partir de uma vista para a outra. Por exemplo, se primeira e segunda câmeras (representando primeira e segunda vistas) captam imagens de um objeto, o objeto pode parecer estar deslocado em cinco pixels para a direita na segunda vista em comparação com a primeira vista. Isso significa que o pixel (i-5, j) na primeira vista corresponde ao pixel (i, j) na segunda vista. Se os pixels das duas vistas estão simplesmente ladrilhados pixel por pixel, então pode não haver muita correlação entre os pixels adjacentes no ladrilho, e os ganhos de codificação espacial podem ser pequenos. Inversamente, mediante deslocamento do ladrilhamento de modo que o pixel (i-5, j) a partir de uma vista seja colocado próximo ao pixel (i, j) a partir da vista dois, correlação espacial pode ser aumentada e o ganho de codificação espacial também pode ser aumentado. Isso ocorre porque, por exemplo, os pixels correspondentes para o objeto na primeira e segunda vista estão sendo ladrilhados próximos um do outro.

Assim, a presença de loc_left_offset e loc_top_offset pode se beneficiar da eficiência de codificação. A informação de deslocamento pode ser obtida mediante meios externos. Por exemplo, a informação de posição das câmeras ou os vetores de disparidade global entre as vistas podem ser usados para determinar tal informação de deslocamento.

Como resultado do deslocamento, a alguns pixels na pseudovista não são atribuídos valores de pixel a partir de qualquer vista. Continuando o exemplo acima, ao ladrilhar o pixel (i-5, j) a partir da vista um ao longo do pixel (i, j) a partir da vista dois, para valores de i=0...4 não há pixel (i-5, j) a partir da vista um para ladrilhar, de modo que esses pixels estão vazios no ladrilho. Para aqueles pixels na pseudovista (ladrilho) aos quais não são atribuídos valores de pixel a partir de qualquer vista, ao menos uma implementação utiliza um procedimento de interpolação similar ao procedimento de interpolação de subpixel em compensação de movimento em AVC. Isto é, os pixels de ladrilho vazio podem ser interpolados a partir de pixels adjacentes. Tal interpolação pode resultar em maior correlação espacial no ladrilho e maios ganho de codificação para o ladrilho.

Em codificação de vídeo, podemos escolher um tipo de codificação diferente para cada imagem, tal como imagens I, P e B. Para codificação de vídeo de multivista, em adição, definimos imagens de âncora e imagens de não-âncora. Em uma modalidade, propomos que a decisão de agrupamento possa ser feita com base no tipo de imagem. Essa informação de agrupamento é sinalizada na sintaxe de alto nível.

Voltando-se para a Figura 11, um exemplo de cinco vistas ladrilhadas em um único quadro é indicado geralmente pelo numeral de referência 1100. Particularmente, a sequência "ballroom" é mostrada com 5 vistas ladrilhadas em um único quadro. Adicionalmente, pode ser visto que a quinta vista é dividida em duas partes de modo que ela pode ser arranjada em um quadro retangular. Aqui, cada vista é de tamanho QVGA de modo que a dimensão de quadro total é de 640x600. Como 600 não é um múltiplo de 16 ele deve ser estendido para 608.

Para esse exemplo, a mensagem SEI possível poderia ser conforme mostrado na TABELA 4.

TABLE 4

multiview_display_info(payloadSize) {	Valor
num_coded_views_minus 1	5
org_pic_width_in_mbs_minus1	40
org_pic_height_in_mbs_minus1	30
view_id[0]	0
num_parts[view_id[0]]	1
depth_flag[view_id[0]][0]	0
flip_dir[view_id[0]][0]	0
loc_left_offset[view_id[0]] [0]	0
loc_top_offset[view_id[0]] [0]	0
frame_crop_left_offset[view_id[0]] [0]	0
frame_crop_right_offset[view_id[0]] [0]	320
frame_crop_top_offset[view_id[0]] [0]	0
frame_crop_bottom_offset[view_id[0]] [0]	240
upsample_view_flag[view_id[0]]	1
se(upsample_view_flag[view_id[0]]) {	
vert_dim[view_id[0]]	6
hor_dim[view_id[0]]	6
quantizer[view_id[0]]	32
para (yuv= 0; yuv< 3; yuv++) {	
para (y = 0; y < vert_dim[view_id[i]] -1; y ++) {	
para (x = 0; x < hor_dim[view_id[i]] -1; x ++)	
filter_coeffs[view_id[i]] [yuv][y][x]	XX

view_id[1]	1
num_parts[view_id[1]]	1
depth_flag[view_id[0]][0]	0
flip_dir[view_id[1]][0]	0
loc_left_offset[view_id[1]] [0]	0
loc_top_offset[view_id[1]] [0]	0
frame_crop_left_offset[view_id[1]] [0]	320
frame_crop_right_offset[view_id[1]] [0]	640
frame_crop_top_offset[view_id[1]] [0]	0
frame_crop_bottom_offset[view_id[1]] [0]	320
upsample_view_flag[view_id[1]]	1
se(upsample_view_flag[viewjd[1]]) {	
vert_dim[view_id[1]]	6
hor_dim[view_id[1]]	6
quantizer[view_id[1]]	32
para (yuv= 0; yuv< 3; yuv++) {	
para (y = 0; y < vert_dim[view_id[i]] -1; y ++) {	
para (x = 0; x < hor_dim[view_id[i]] -1; x ++)	
filter_coeffs[view_id[i]] [yuv][y][x]	XX
..... (similarmente para vista 2,3)	
view_id[4]	4
num_parts[view_id[4]]	2
depth_flag[view_id[0]][0]	0
flip_dir[view_id[4]][0]	0
loc_left_offset[view_id[4]] [0]	0
loc_top_offset[view_id[4]] [0]	0
frame_crop_left_offset[view_id[4]] [0]	0
frame_crop_right_offset[view_id[4]] [0]	320
frame_crop_top_offset[view_id[4]] [0]	480
frame_crop_bottom_offset[view_id[4]] [0]	600
flip_dir[view_id[4]][1]	0
loc_left_offset[view_id[4]] [1]	0
loc_top_offset[view_id[4]] [1]	120
frame_crop_left_offset[view_id[4]] [1]	320
frame_crop_right_offset[view_id[4]] [1]	640
frame_crop_top_offset[view_id[4]] [1]	480

frame_crop_bottom_offset[view_id[4]] [1]	600
upsample_view_flag[view_id[4]]	1
se(upsample_view_flag[view_id[4]]) {	
vert_dim[view_id[4]]	6
hor_dim[view_id[4]]	6
quantizer[view_id[4]]	32
para (yuv= 0; yuv< 3; yuv++) {	
para (y = 0; y < vert_dim[view_id[i]] -1; y ++) {	
para(x = 0; x < hor_dim[view_id[i]] -1; x ++)	
filter_coeffs[view_id[i]] [yuv][y][x]	XX

A TABELA 5 mostra a estrutura de sintaxe geral para transmitir informação de multivista para o exemplo mostrado na Tabela 4.

TABLE 5

multiview_display_info(payloadSize) {	C	Descritor
num_coded_views_minus1	5	ue(v)
org_pic_width_in_mbs_minus1	5	ue(v)
org_pic_height_in_mbs_minus1	5	ue(v)
para(i = 0; i <= num_coded_views_minus1; i++) {		
view_id[i]	5	ue(v)
num_parts[view_id[i]]	5	ue(v)
para(j = 0; j <= num_parts[i]; j++) {		
depth_flag[view_id[i]][j]		
flip_dir[view_id[i]][j]	5	u(2)
loc_left_offset[view_id[i]] [j]	5	ue(v)
loc_top_offset[view_id[i]] [j]	5	ue(v)
frame_crop_left_offset[view_id[i]] [j]	5	ue(v)
frame_crop_right_offset[view_id[i]] [j]	5	ue(v)
frame_crop_top_offset[view_id[i]] [j]	5	ue(v)
frame_crop_bottom_offset[view_id[i]] [j]	5	ue(v)
}		
upsample_view_flag[view_id[i]]	5	u(1)
se(upsample_view_flag[view_id[i]])		
upsample_filter[view_id[i]]	5	u(2)
se(upsample_fiter[view_id[i]] == 3) {		
vert_dim[view_id[i]]	5	ue(v)
hor_dim[view_id[i]]	5	ue(v)
quantizer[view_id[i]]	5	ue(v)
para (yuv= 0; yuv< 3; yuv++) {		
para (y = 0; y < vert_dim[view_id[i]] -1; y ++) {		
para (x = 0; x < hor_dim[view_id[i]] -1; x ++)		
filter_coeffs[view_id[i]] [yuv][y][x]	5	se(v)
}		

}		
}		
}		
}		

Com referência à Figura 23, é mostrado um dispositivo de processamento de vídeo 2300. O dispositivo de processamento de vídeo 2300 pode ser, por exemplo, um conversor de sinais ou outro dispositivo que recebe vídeo codificado e provê, por exemplo, vídeo decodificado para exibição para um usuário ou para armazenamento. Assim, o dispositivo 2300 pode prover sua saída para uma televisão, monitor de computador, ou um computador ou outro dispositivo de processamento.

O dispositivo 2300 inclui um decodificador 2310 que recebe um sinal de dados 2320. O sinal de dados 2320 pode incluir, por exemplo, um fluxo compatível com AVC ou um fluxo compatível com MVC. O decodificador 2310 decodifica todo ou parte do sinal recebido 2320 e provê como saída um sinal de vídeo decodificado 2330 e informação de ladrilhamento 2340. O vídeo decodificado 2330 e a informação de ladrilhamento 2340 são providos a um seletor 2350. O dispositivo 2300 inclui também uma interface de usuário 2360 que recebe uma entrada de usuário 2370. A interface de usuário 2360 provê um sinal de seleção de imagem 2380, com base na entrada de usuário 2370, para o seletor 2350. O sinal de seleção de imagem 2380 e a entrada de usuário 2370 indicam qual das múltiplas imagens um usuário pretende ter exibida. O seletor 2350 provê a imagem(ns) selecionada como uma saída 2390. O seletor 2350 usa a informação de seleção de imagem 2380 para selecionar qual das imagens no vídeo codificado 2330 prover como a saída 2390. O seletor 2350 utiliza a informação de ladrilhamento 2340 para localizar a imagem(ns) selecionada no vídeo decodificado 2330.

Em várias implementações, o seletor 2350 inclui a interface de usuário 2360, e em outras implementações nenhuma interface de usuário 2360 é necessária porque o seletor 2350 recebe a entrada de usuário 2370 diretamente sem uma função de interface separada sendo realizada. O seletor 2350 pode ser implementado em software ou como um circuito integrado, por exemplo. O seletor 2350 também pode incorporar o decodificador 2310.

Mais geralmente, os decodificadores de várias implementações descritas nesse pedido podem prover uma saída decodificada que inclui um ladrilho inteiro. Adicionalmente ou alternativamente, os decodificadores podem prover uma saída decodificada que inclui apenas uma ou mais imagens selecionadas (imagens ou sinais de profundidade, por exemplo) a partir do ladrilho.

Conforme observado acima, sintaxe de alto nível pode ser usada para realizar sinalização de acordo com uma ou mais modalidades dos presentes princípios. A sintaxe de alto nível pode ser usada, por exemplo, mas não é limitada à sinalização de qualquer um dos

seguintes: o número de vistas codificadas presentes no quadro maior, a largura e altura originais de todas as vistas; para cada vista codificada, o identificador de vista correspondendo à vista; para cada vista codificada, o número de partes em que o quadro de uma vista é dividido; para cada parte da vista, a direção de viragem (a qual pode ser, por exemplo, nenhuma viragem, viragem apenas horizontal, viragem apenas vertical ou viragem horizontal e vertical); para cada parte da vista, a posição esquerda em pixels ou o número de macroblocos onde a parte atual pertence no quadro final para a vista; para cada parte da vista, a posição superior da parte em pixels ou número de macroblocos onde a parte atual pertence no quadro final para a vista; para cada parte da vista, a posição esquerda, no quadro decodificado/codificado grande atual, da janela de recorte em pixels ou número de macroblocos; para cada parte da vista, a posição à direita, no quadro decodificado/codificado grande atual, da janela de recorte em pixels ou número de macroblocos; para cada parte da vista, a posição superior, no quadro decodificado/codificado grande atual, da janela de recorte em pixels ou número de macroblocos; e, para cada parte da vista, a posição inferior, no quadro decodificado/codificado grande atual, da janela de recorte em pixels ou o número de macroblocos; para cada vista codificada se a vista precisa ser amostrada ascendentemente antes da saída (onde se a amostragem ascendente precisa ser realizada, uma sintaxe de alto nível pode ser usada para indicar o método para amostragem ascendente (incluindo, mas não limitado a, filtro de 6 derivações AVC, filtro de 4 derivações SVC, filtro bilinear ou um filtro 1D especial, 2D linear ou não-linear).

Deve ser observado que os termos, "codificador" e "decodificador" conotam estruturas gerais e não são limitados a quaisquer funções ou características específicas. Por exemplo, um decodificador pode receber um portador modulado que transporta um fluxo de bits codificado, e demodula o fluxo de bits codificado, assim como decodifica o fluxo de bits.

Vários métodos foram descritos. Muitos desses métodos são detalhados para prover ampla revelação. Contudo, observa-se que variações são consideradas que podem variar uma ou muitas das características específicas descritas para esses métodos. Adicionalmente, muitas das características que são citadas são conhecidas na técnica e, consequentemente, não são descritas em grande detalhe.

Adicionalmente, foi feita referência ao uso de sintaxe de alto nível para enviar certa informação em várias implementações. Contudo, deve ser entendido que outras implementações utilizam sintaxe de nível inferior, ou na realidade outros mecanismos de modo geral (tal como, por exemplo, enviando informação como parte dos dados codificados) para prover a mesma informação (ou variações dessa informação).

Várias implementações proporcionam ladrilhamento e sinalização apropriada para permitir que múltiplas vistas (imagens, mais geralmente) sejam ladrilhadas em uma única imagem, codificadas como uma única imagem, e enviadas como uma única imagem. A in-

formação de sinalização pode permitir que um pós-processador separe as vistas/imagens. Além disso, as múltiplas imagens que são ladrilhadas poderiam ser vistas, mas ao menos uma das imagens poderia ser informação de profundidade. Essas implementações podem prover uma ou mais vantagens. Por exemplo, os usuários podem pretender exibir múltiplas vistas de uma maneira ladrilhada, e essas várias implementações proporcionam uma forma eficiente de codificar e transmitir ou armazenar tais vistas mediante ladrilhamento das mesmas antes de codificar e transmitir/armazenar as mesmas de uma maneira ladrilhada.

Implementações que ladrilham múltiplas vistas no contexto de AVC e/ou MVC também proporcionam vantagens adicionais. AVC é usado apenas ostensivamente para uma única vista, de modo que nenhuma vista adicional é esperada. Contudo, tais implementações baseadas em AVC podem prover múltiplas vistas em um ambiente AVC porque as vistas ladrilhadas podem ser arranjadas de modo que, por exemplo, um decodificador sabe que as imagens ladrilhadas pertencem às diferentes vistas (por exemplo, imagem esquerda superior na pseudovista é vista 1, imagem superior direita é vista 2, etc.).

Adicionalmente, MVC já inclui múltiplas vistas, de modo que múltiplas vistas não devem ser incluídas em uma única pseudovista. Adicionalmente, MVC tem um limite no número de vistas que podem ser suportadas, e tais implementações baseadas em MVC efetivamente aumentam o número de vistas que podem ser suportadas pelo fato de permitir (como nas implementações baseadas em AVC) que vistas adicionais sejam ladrilhadas. Por exemplo, cada pseudovista pode corresponder a uma das vistas suportadas de MVC, e o decodificador pode ter conhecimento de que cada "vista suportada" efetivamente inclui quatro vistas em uma ordem ladrilhada pré-arranjada. Desse modo, em tal implementação, o número de possíveis vistas é de quatro vezes o número de "vistas suportadas".

As implementações aqui descritas podem ser implementadas, por exemplo, em um método ou processo, em um aparelho, ou em um programa de software. Mesmo se discutido apenas no contexto de uma única forma de implementação (por exemplo, discutida apenas como um método), a implementação de características discutidas pode ser implementada em outras formas (por exemplo, um aparelho ou programa). Um aparelho pode ser implementado, por exemplo, em hardware, software e firmware apropriado. Os métodos podem ser implementados, por exemplo, em um aparelho tal como, por exemplo, um processador, o qual se refere aos dispositivos de processamento em geral, incluindo, por exemplo, um computador, um microprocessador, um circuito integrado, ou um dispositivo lógico programável. Os dispositivos de processamento incluem também dispositivo de comunicação, tal como, por exemplo, computadores, telefones celulares, assistentes digitais pessoais/portáteis ('PDAs"), e outros dispositivos que facilitam a comunicação de informação entre usuários finais.

Implementações dos vários processos e características aqui descritos podem ser

incorporadas em uma variedade de diferentes equipamentos ou aplicações, particularmente, por exemplo, equipamentos ou aplicações associados à codificação e decodificação de dados. Exemplos de equipamento incluem codificadores de vídeo, decodificadores de vídeo, codecs de vídeo, servidores de Rede, conversores de sinais, laptops, computadores pessoais, telefones celulares, PDAs, e outros dispositivos de comunicação. Como deve ser evidente, o equipamento pode ser móvel e até mesmo instalado em um veículo móvel.

Adicionalmente, os métodos podem ser implementados mediante instruções sendo realizadas por um processador, e tais instruções podem ser armazenadas em um meio legível por processador tal como, por exemplo, um circuito integrado, um portador de software ou outro dispositivo de armazenamento tal como, por exemplo, um disco rígido, um disquete, uma memória de acesso aleatório ("RAM"), ou uma memória de leitura ("ROM"). As instruções podem formar um programa de aplicação incorporado de forma tangível em um meio legível por processador. Como deve ser evidente, um processador pode incluir um meio legível por processador tendo, por exemplo, instruções para realizar um processo. Tais programas de aplicação podem ser transferidos para, e executados por uma máquina compreendendo qualquer arquitetura adequada. Preferivelmente, a máquina é implementada em uma plataforma de computador tendo hardware tal como uma ou mais unidades centrais de processamento ("CPU"), uma memória de acesso aleatório ("RAM"), e interfaces de entrada/saída ("I/O"). A plataforma de computador também pode incluir um sistema operacional e código de microinstrução. Os vários processos e funções aqui descritos podem ser parte do código de microinstrução ou parte do programa de aplicação, ou qualquer combinação dos mesmos, os quais podem ser executados por uma CPU. Além disso, várias outras unidades periféricas podem ser conectadas à plataforma de computador tal como uma unidade de armazenamento de dados, adicional e uma unidade de impressão.

Como deve ser evidente para aqueles versados na técnica, implementações também podem produzir um sinal formatado para portar informação que pode ser, por exemplo, armazenada ou transmitida. A informação pode incluir, por exemplo, instruções para realizar um método, ou dados produzidos por uma das implementações descritas. Tal sinal pode ser formatado, por exemplo, como uma onda eletromagnética (por exemplo, utilizando uma porção de espectro de radiofrequência) ou como um sinal de banda base. A formatação pode incluir, por exemplo, codificar um fluxo de dados, produzir sintaxe, e modular um portador com o fluxo de dados codificado e a sintaxe. A informação que o sinal transporta pode ser, por exemplo, informação analógica ou digital. O sinal pode ser transmitido através de uma variedade de diferentes links cabeados ou sem fio, conforme é sabido.

Deve ser adicionalmente entendido que, devido a alguns dos componentes de sistema, constituintes e métodos ilustrados nos desenhos anexos são preferivelmente implementados em software, as conexões efetivas entre os componentes de sistema ou os blocos

de função de processo podem diferir dependendo da forma na qual os presentes princípios são programados. Dados os presentes ensinamentos, aqueles de conhecimento comum na técnica pertinente serão capazes de considerar essas e similares implementações ou configurações dos presentes princípios.

Algumas implementações foram descritas. Não obstante, será entendido que podem ser feitas várias modificações. Por exemplo, elementos de diferentes implementações podem ser combinados, suplementados, modificados, ou removidos para produzir outras implementações. Adicionalmente, aqueles de conhecimento comum na técnica entenderão que outras estruturas ou processos podem ser substitutos daqueles revelados e as implementações resultantes realizarão ao menos substancialmente a mesma função(ões) em ao menos substancialmente a mesma forma(s) para obter ao menos substancialmente o mesmo resultado(s) que as implementações reveladas. Particularmente, embora modalidades ilustrativas sejam descritas aqui com referência aos desenhos anexos, deve-se entender que os presentes princípios não são limitados àquelas precisas modalidades, e que diversas alterações e modificações podem ser realizadas nas mesmas por aqueles versados na técnica pertinente sem se afastar do espírito ou escopo dos presentes princípios. Consequentemente, essas e outras implementações são consideradas por esse pedido e estão dentro do escopo das reivindicações a seguir.

REIVINDICAÇÕES

1. Método, **CARACTERIZADO** por compreender:

acessar uma imagem de vídeo que inclui múltiplas imagens combinadas em uma única imagem, as múltiplas imagens incluindo uma imagem de uma primeira visualização e uma imagem de uma segunda visualização nas quais um objeto ou região comum aparece deslocado a partir de uma visualização para a outra, a imagem de vídeo sendo parte de um fluxo de vídeo recebido;

acessar informação indicando como as imagens múltiplas na imagem de vídeo acessada são combinadas, em que a informação acessada indica se pelo menos uma das múltiplas imagens é invertida individualmente em uma ou mais de uma direção horizontal ou uma direção vertical de modo que o objeto ou região comum da imagem de primeira visualização é justaposto ao objeto ou região comum da imagem de segunda visualização, e em que a informação acessada é parte do fluxo de vídeo recebido;

decodificar a imagem de vídeo para fornecer uma representação decodificada de pelo menos uma das múltiplas imagens; e

processar posteriormente a representação decodificada utilizando a informação acessada.

2. Método, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que as múltiplas imagens incluem uma imagem da primeira visualização, e uma segunda imagem incluindo informação de profundidade para a imagem da primeira visualização.

3. Método, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que o acesso da imagem de vídeo, o acesso da informação, e a decodificação da imagem de vídeo são realizados por um decodificador.

4. Método, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que pelo menos uma dentre as imagens múltiplas representa informação de profundidade.

100

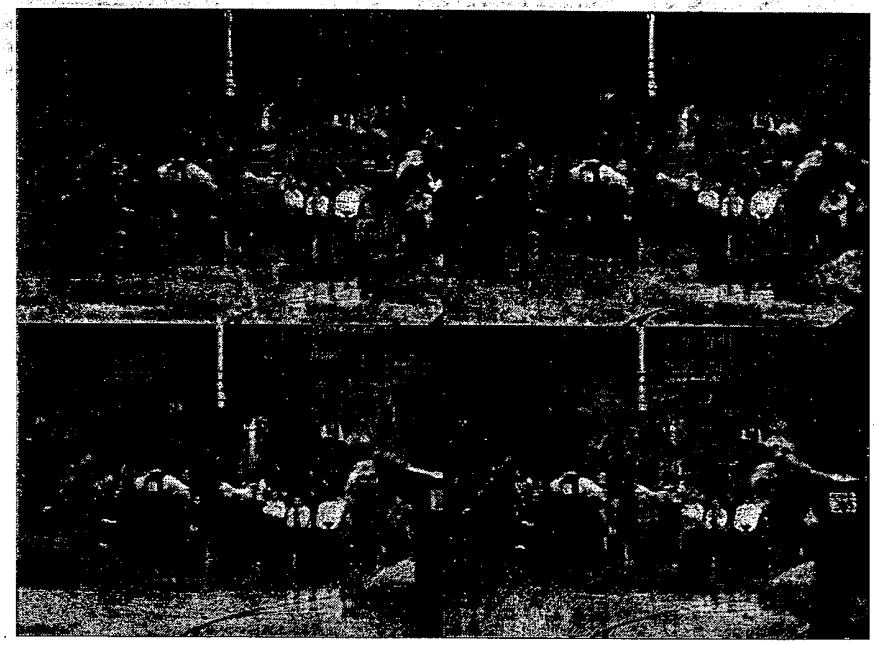


FIG. 1

200

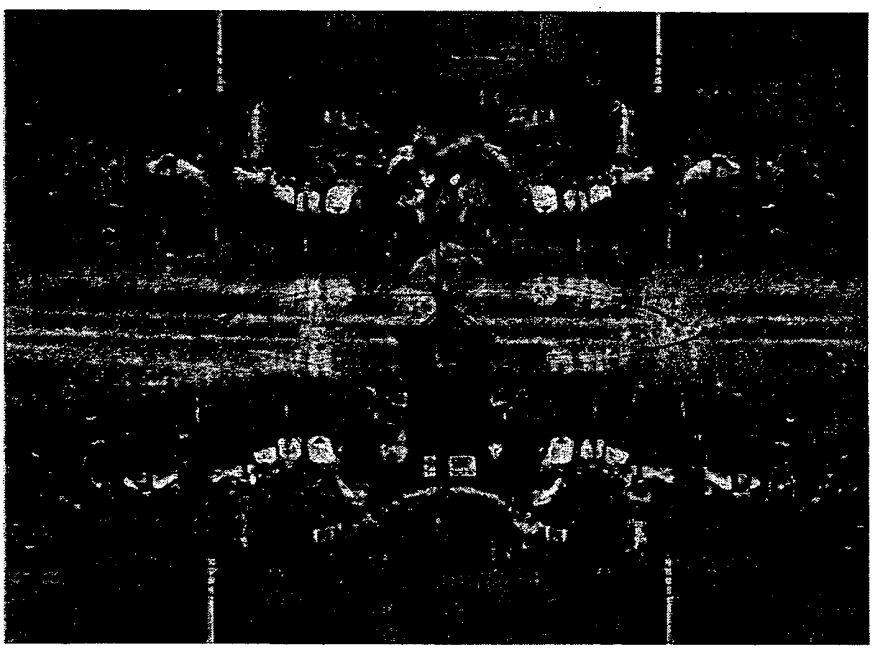


FIG. 2

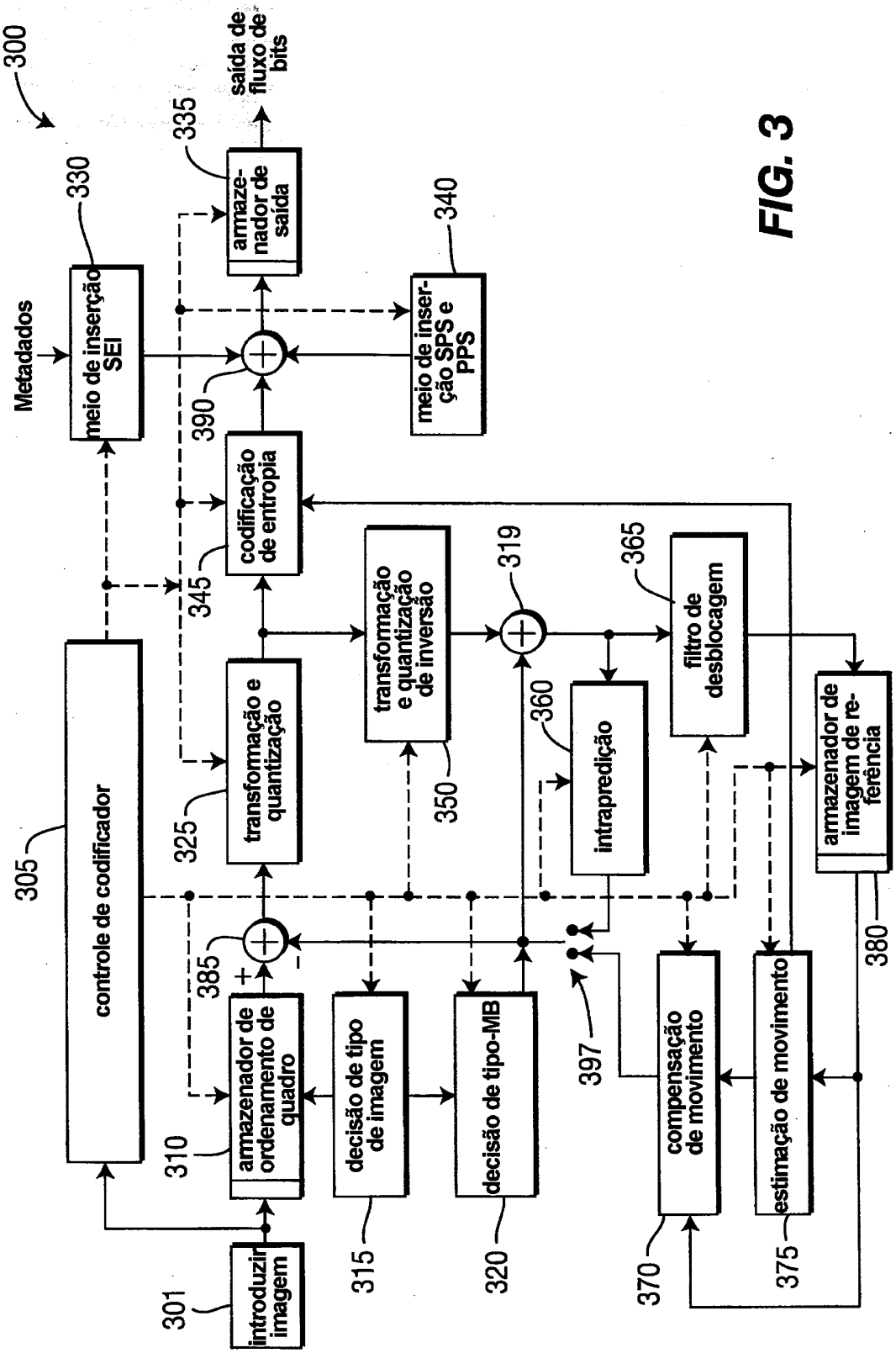

300
saída de fluxo de bits
335
armazenador de saída
330
Metadados
meio de inserção SEI
340
meio de inserção SPS e PPS
390
codificação de entropia
345
transformação e quantização
325
transformação e quantização de inversão
350
319
365
filtro de desblocagem
360
intrapredição
armazenador de imagem de referência
380
305
controle de codificador
385
+
-
310
armazenador de ordenamento de quadro
315
decisão de tipo de imagem
320
decisão de tipo-MB
397
370
compensação de movimento
375
estimação de movimento
301
introduzir imagem

FIG. 3

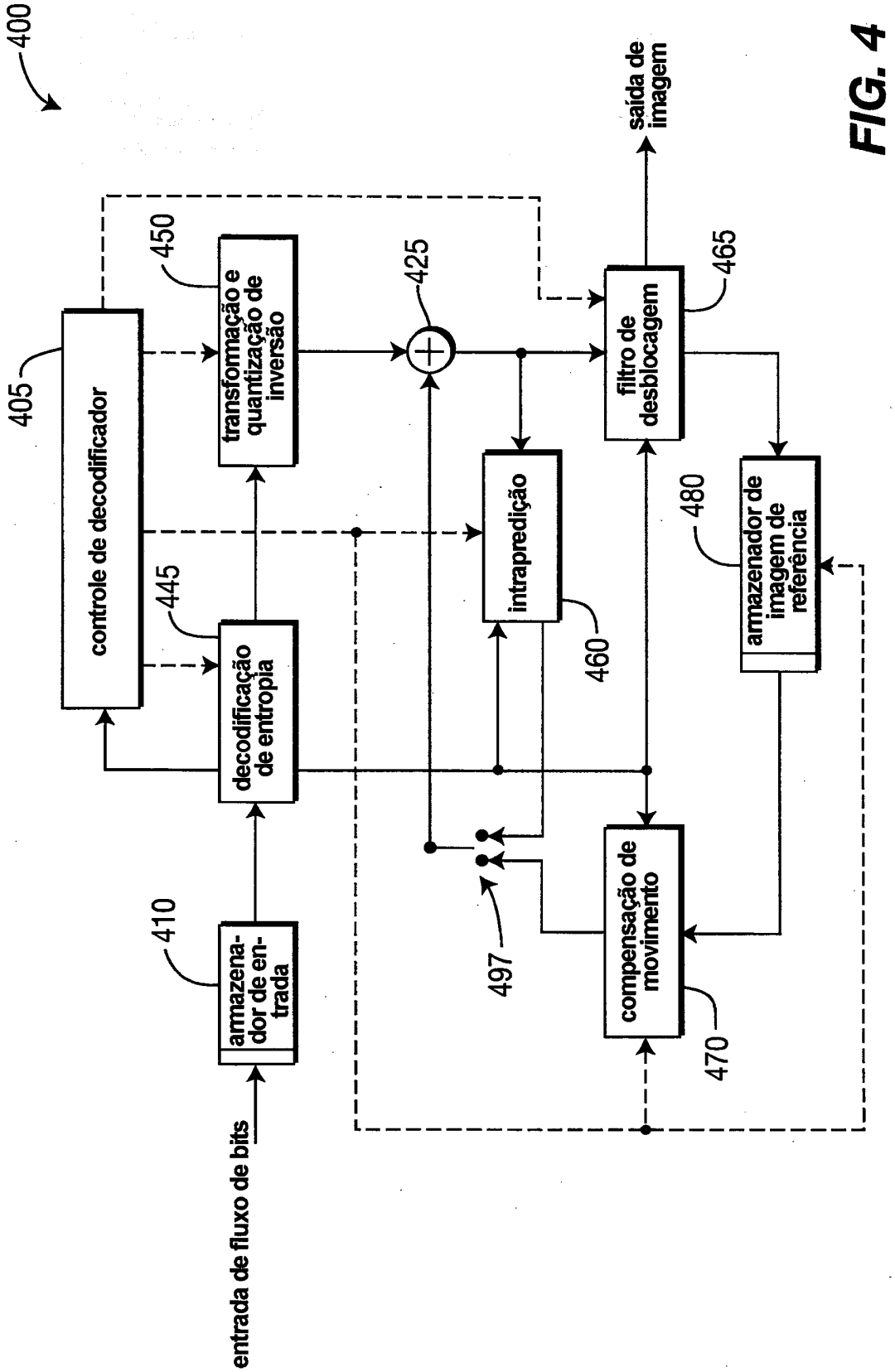

400
saída de imagem
450
425
465
405
transformação e quantização de inversão
filtro de desblocagem
controle de decodificador
intrapredição
480
445
armazenador de imagem de referência
460
decodificação de entropia
410
497
armazenador de entrada
compensação de movimento
470
entrada de fluxo de bits

FIG. 4

FIG. 5

FIG. 5A | FIG. 5B

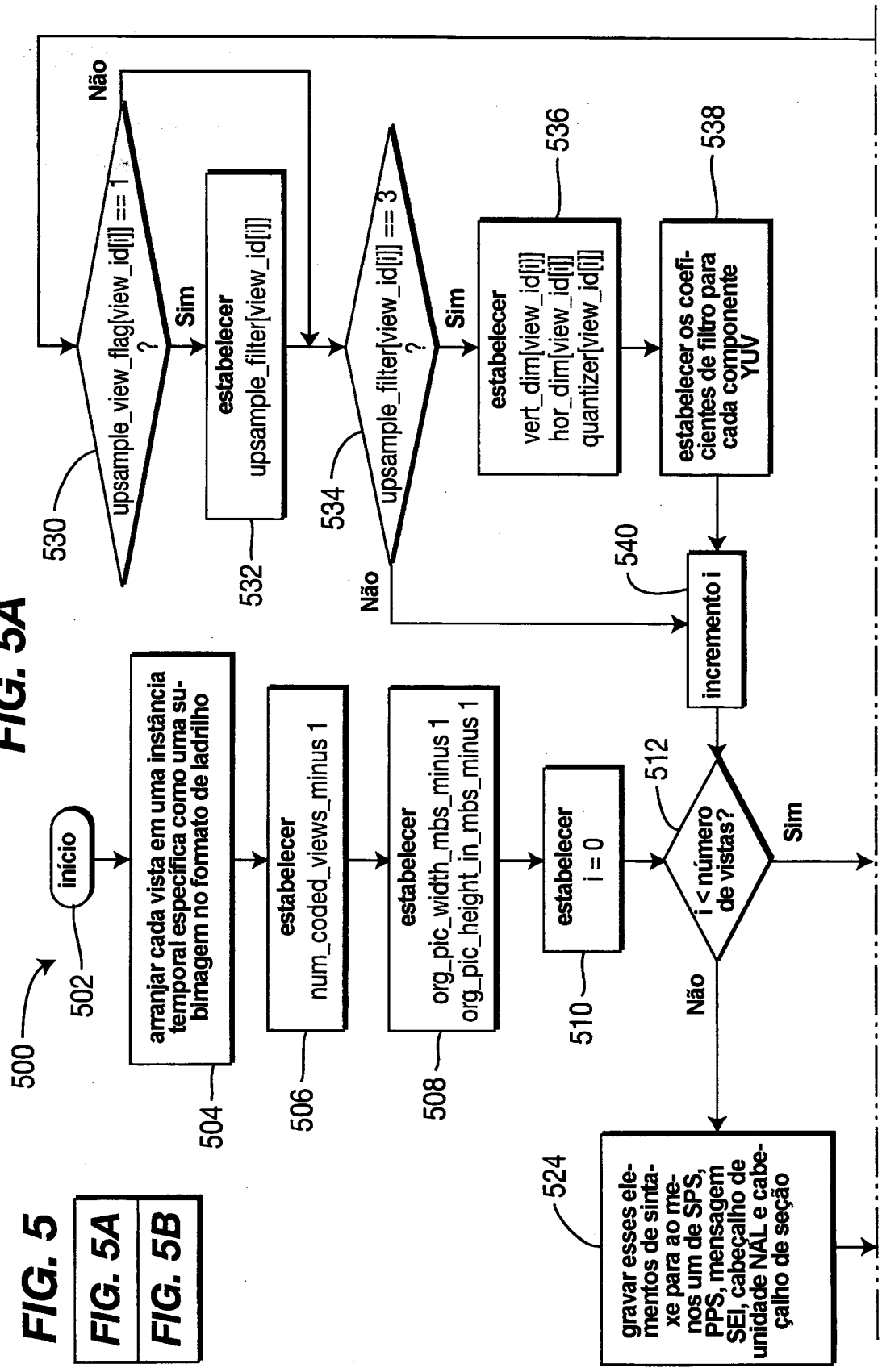


FIG. 5A

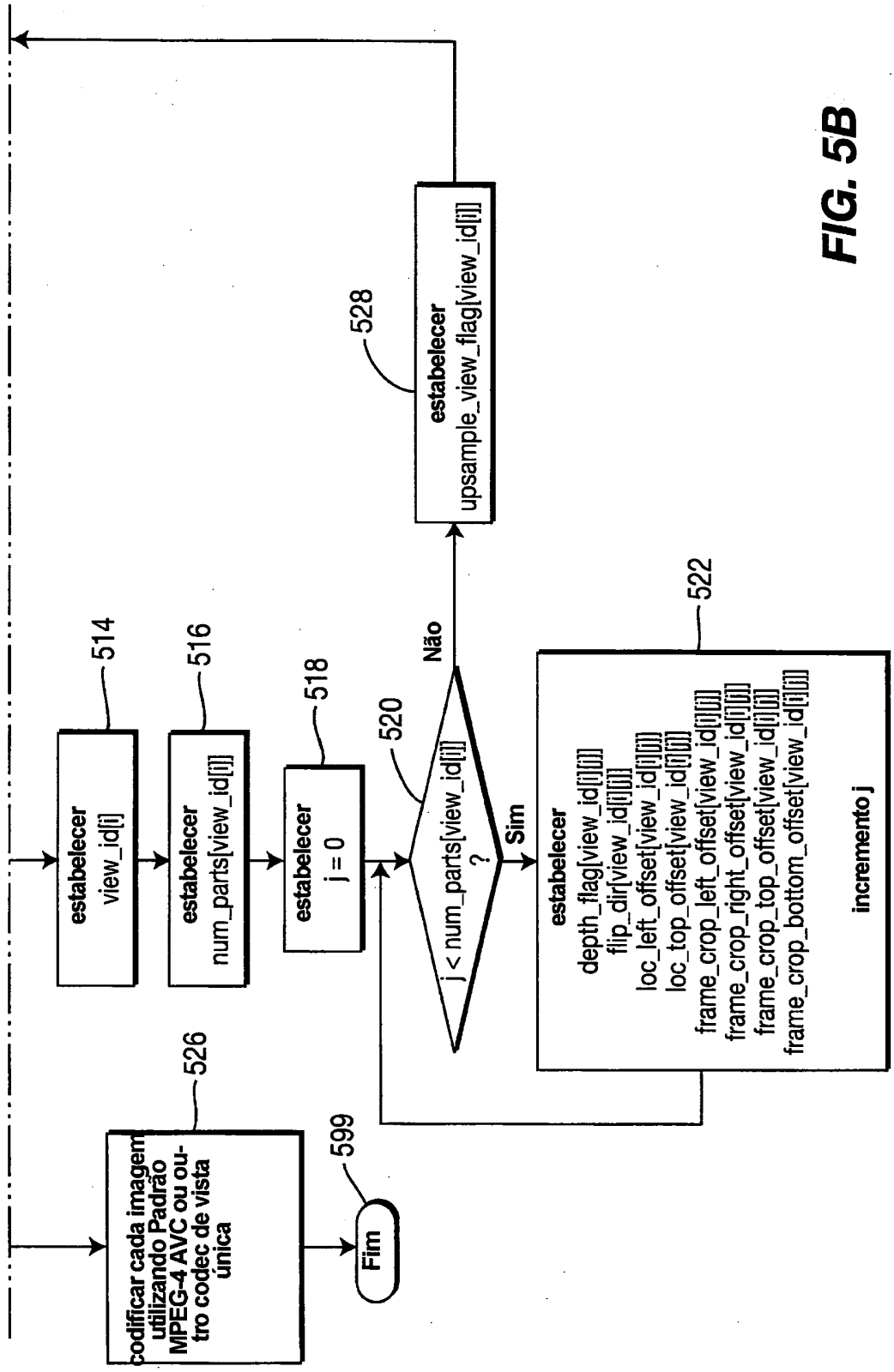

codificar cada imagem utilizando Padrão MPEG-4 AVC ou outro codec de vista única
526
Fim
599
estabelecer view_id[i]
514
estabelecer num_parts[view_id[i]]
516
estabelecer j = 0
518
j < num_parts[view_id[i]] ?
520
Não
Sim
estabelecer upsample_view_flag[view_id[i]]
528
estabelecer depth_flag[view_id[i][j]]
flip_dir[view_id[i][j]]
loc_left_offset[view_id[i][j]]
loc_top_offset[view_id[i][j]]
frame_crop_left_offset[view_id[i][j]]
frame_crop_right_offset[view_id[i][j]]
frame_crop_top_offset[view_id[i][j]]
frame_crop_bottom_offset[view_id[i][j]]
incremento j
522

FIG. 5B

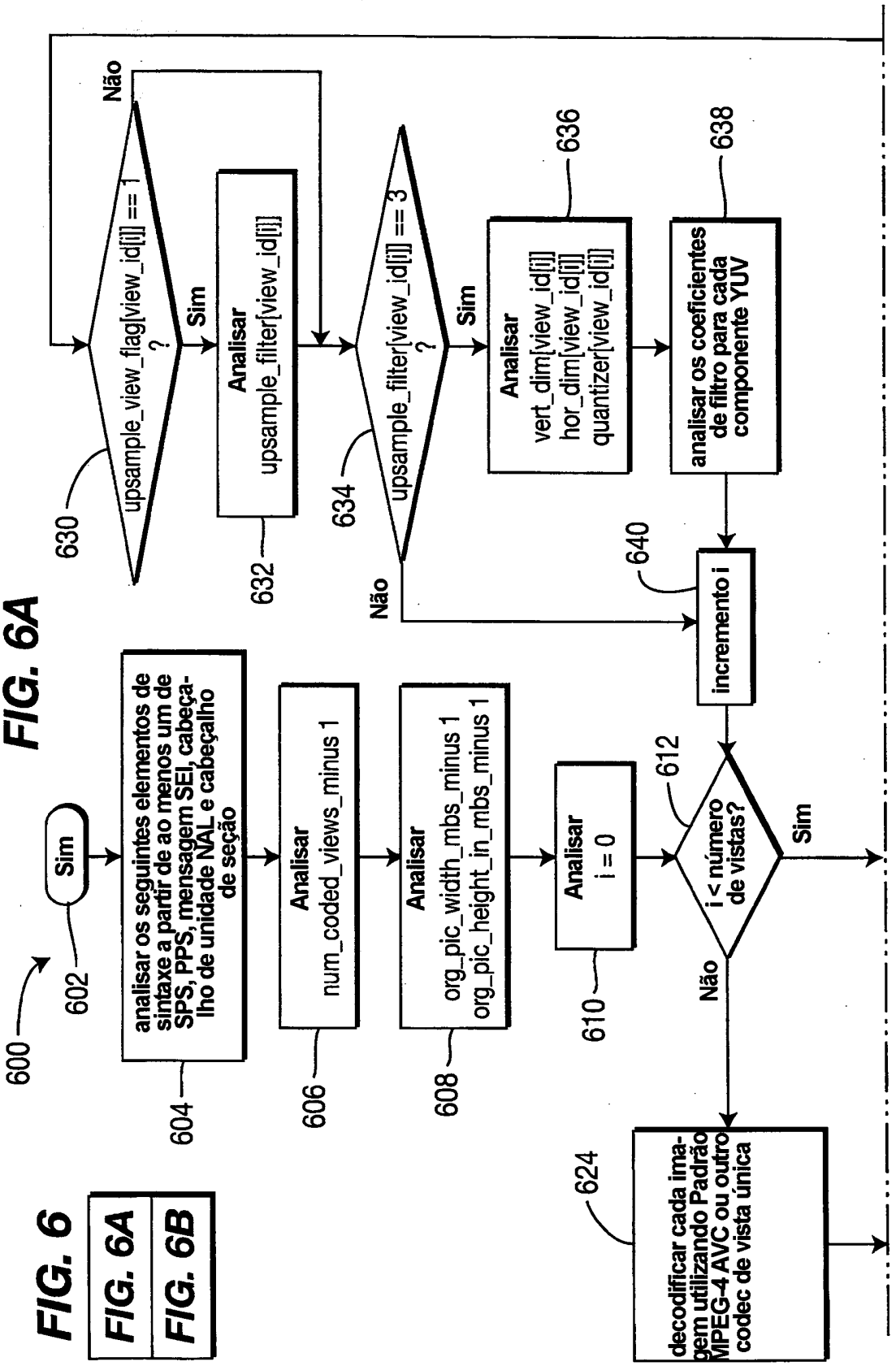


FIG. 6
FIG. 6A
FIG. 6B
FIG. 6A
600
Sim
602
analisar os seguintes elementos de sintaxe a partir de ao menos um de SPS, PPS, mensagem SEI, cabeçalho de unidade NAL e cabeçalho de seção
604
Analisar num_coded_views_minus 1
606
Analisar org_pic_width_mbs_minus 1 org_pic_height_in_mbs_minus 1
608
Analisar i = 0
610
i < número de vistas?
612
Não
Sim
decodificar cada imagem utilizando Padrão MPEG-4 AVC ou outro codec de vista única
624
upsample_view_flag[view_id[i]] == 1 ?
630
Não
Sim
Analisar upsample_filter[view_id[i]]
632
upsample_filter[view_id[i]] == 3 ?
634
Não
Sim
Analisar vert_dim[view_id[i]] hor_dim[view_id[i]] quantizer[view_id[i]]
636
analisar os coeficientes de filtro para cada componente YUV
638
incremento i
640

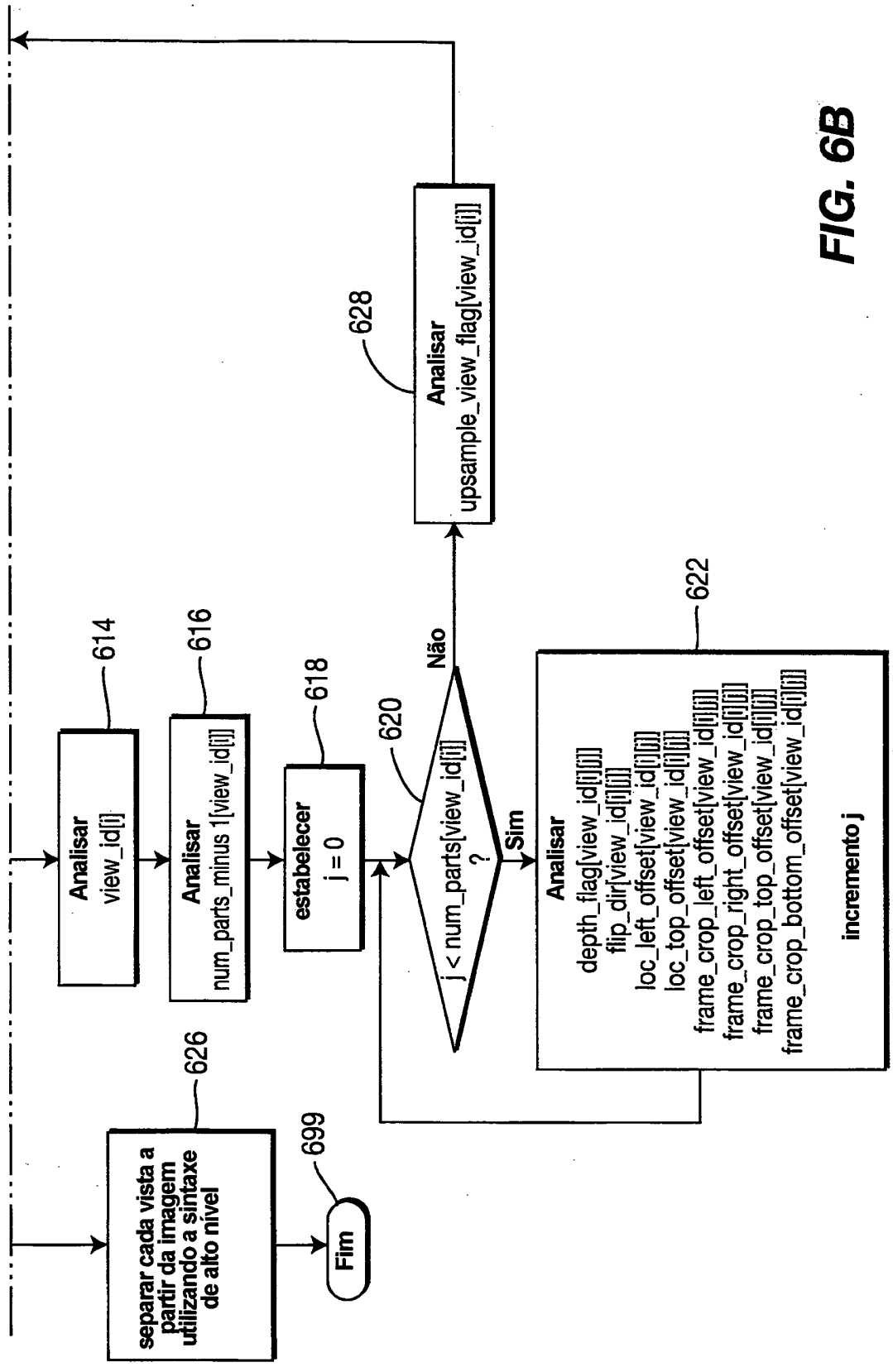

separar cada vista a partir da imagem utilizando a sintaxe de alto nível
626
Fim
699
Analisar view_id[i]
614
Analisar num_parts_minus 1[view_id[i]]
616
estabelecer j = 0
618
j < num_parts[view_id[i]] ?
620
Não
Sim
Analisar upsample_view_flag[view_id[i]]
628
Analisar
depth_flag[view_id[j][i]]
flip_dir[view_id[i][i]]
loc_left_offset[view_id[i][i]]
loc_top_offset[view_id[i][i]]
frame_crop_left_offset[view_id[i][i]]
frame_crop_right_offset[view_id[i][i]]
frame_crop_top_offset[view_id[i][i]]
frame_crop_bottom_offset[view_id[i][i]]
incremento j
622

FIG. 6B

FIG. 7

FIG. 7A
FIG. 7B

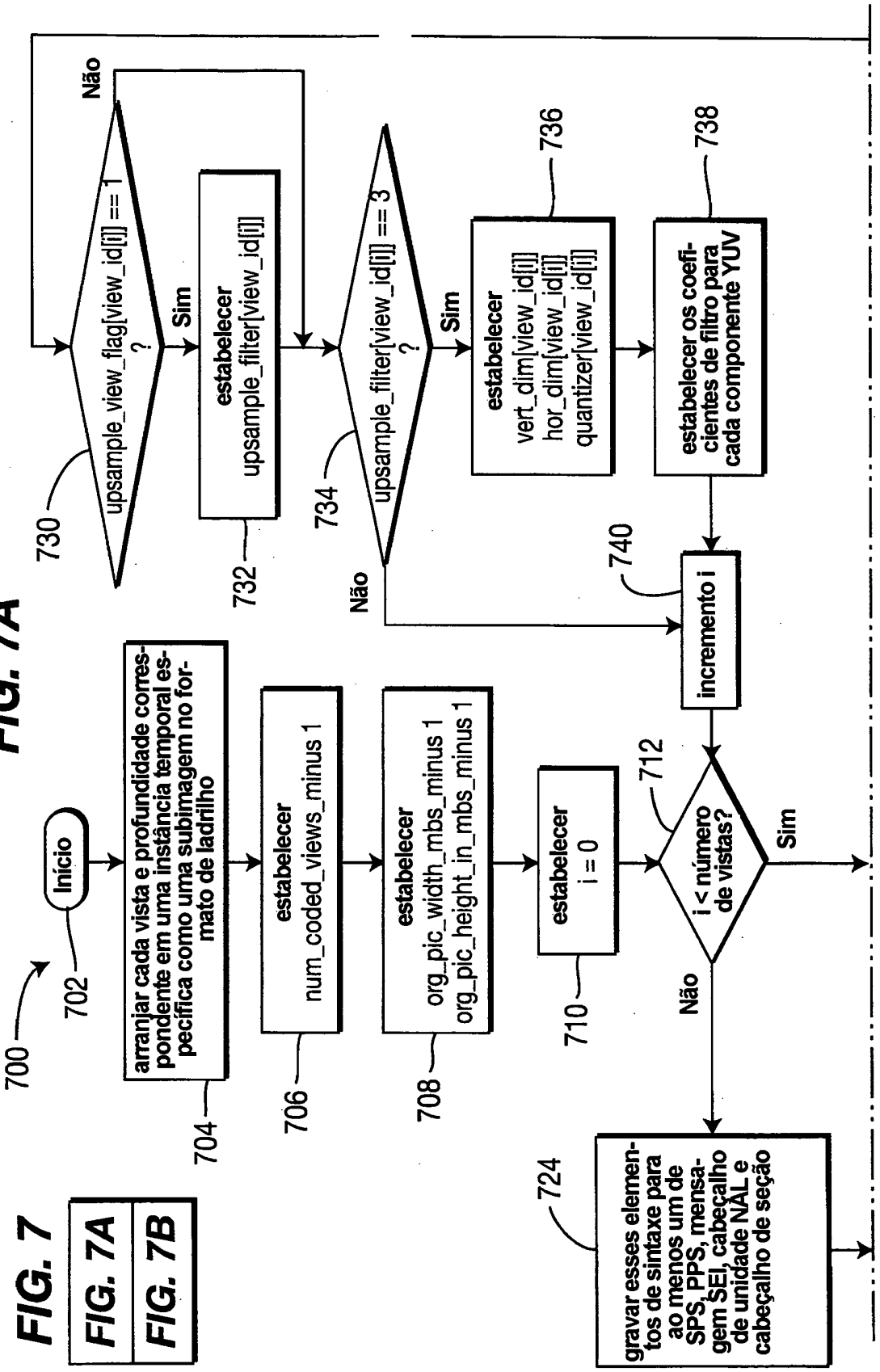


FIG. 7A

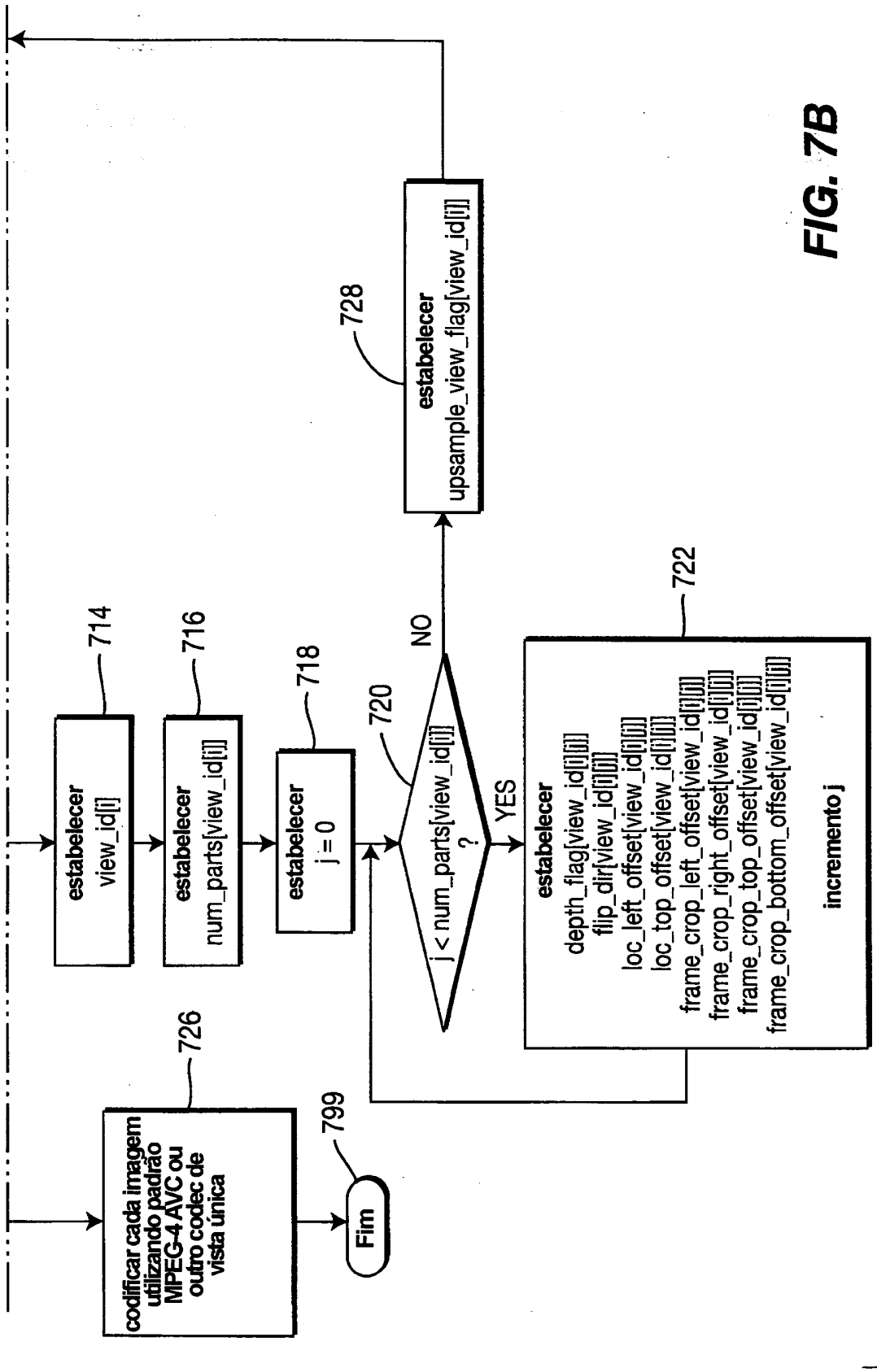

codificar cada imagem utilizando padrão MPEG-4 AVC ou outro codec de vista única
726
Fim
799
estabelecer view_id[i]
714
estabelecer num_parts[view_id[i]]
716
estabelecer j = 0
718
j < num_parts[view_id[i]] ?
720
NO
YES
estabelecer upsample_view_flag[view_id[i]]
728
estabelecer
depth_flag[view_id[i][j]]
flip_dir[view_id[i][j]]
loc_left_offset[view_id[i][j]]
loc_top_offset[view_id[i][j]]
frame_crop_left_offset[view_id[i][j]]
frame_crop_right_offset[view_id[i][j]]
frame_crop_top_offset[view_id[i][j]]
frame_crop_bottom_offset[view_id[i][j]]
incremento j
722

FIG. 7B

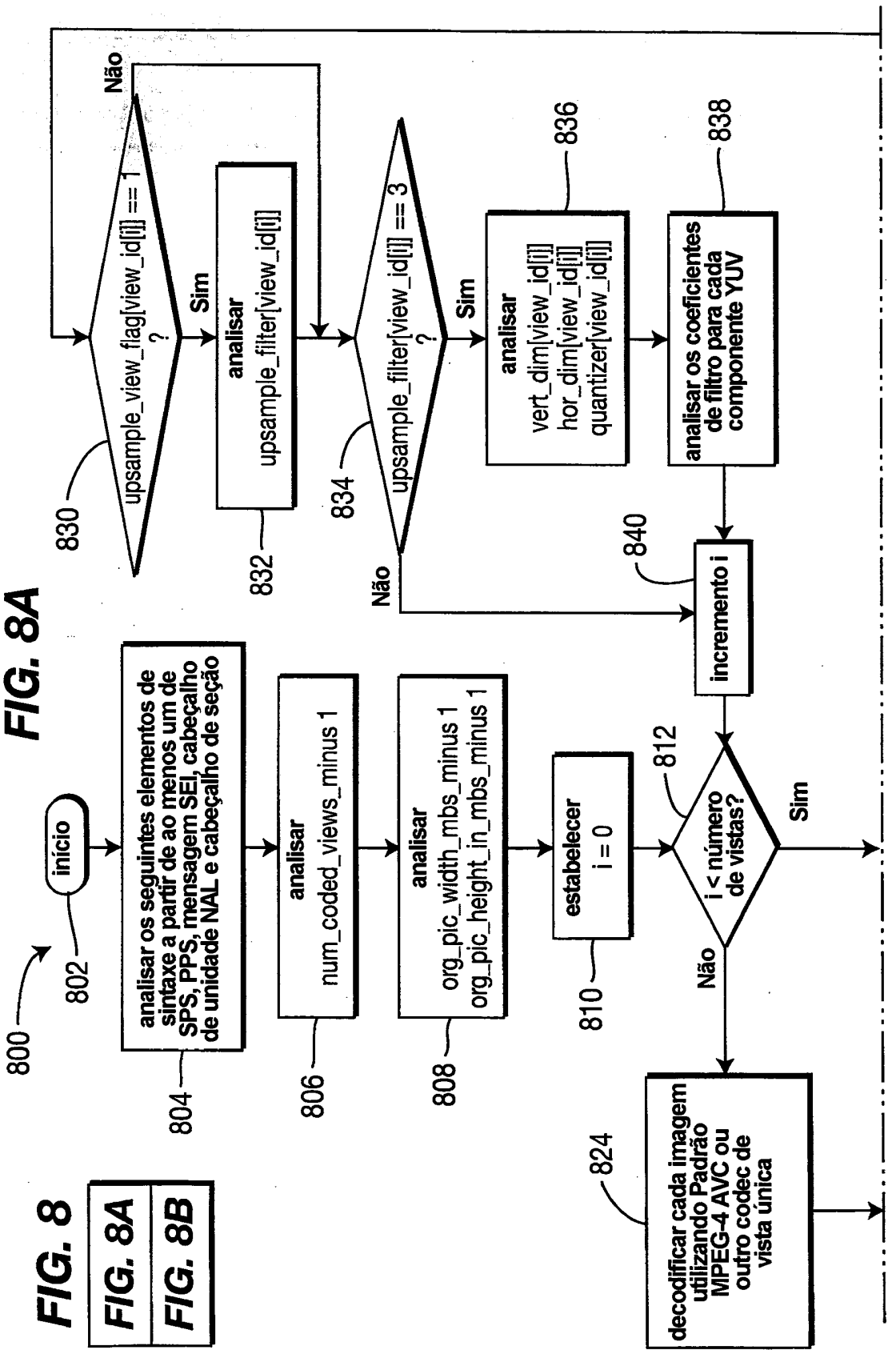


FIG. 8A

FIG. 8

FIG. 8A
FIG. 8B

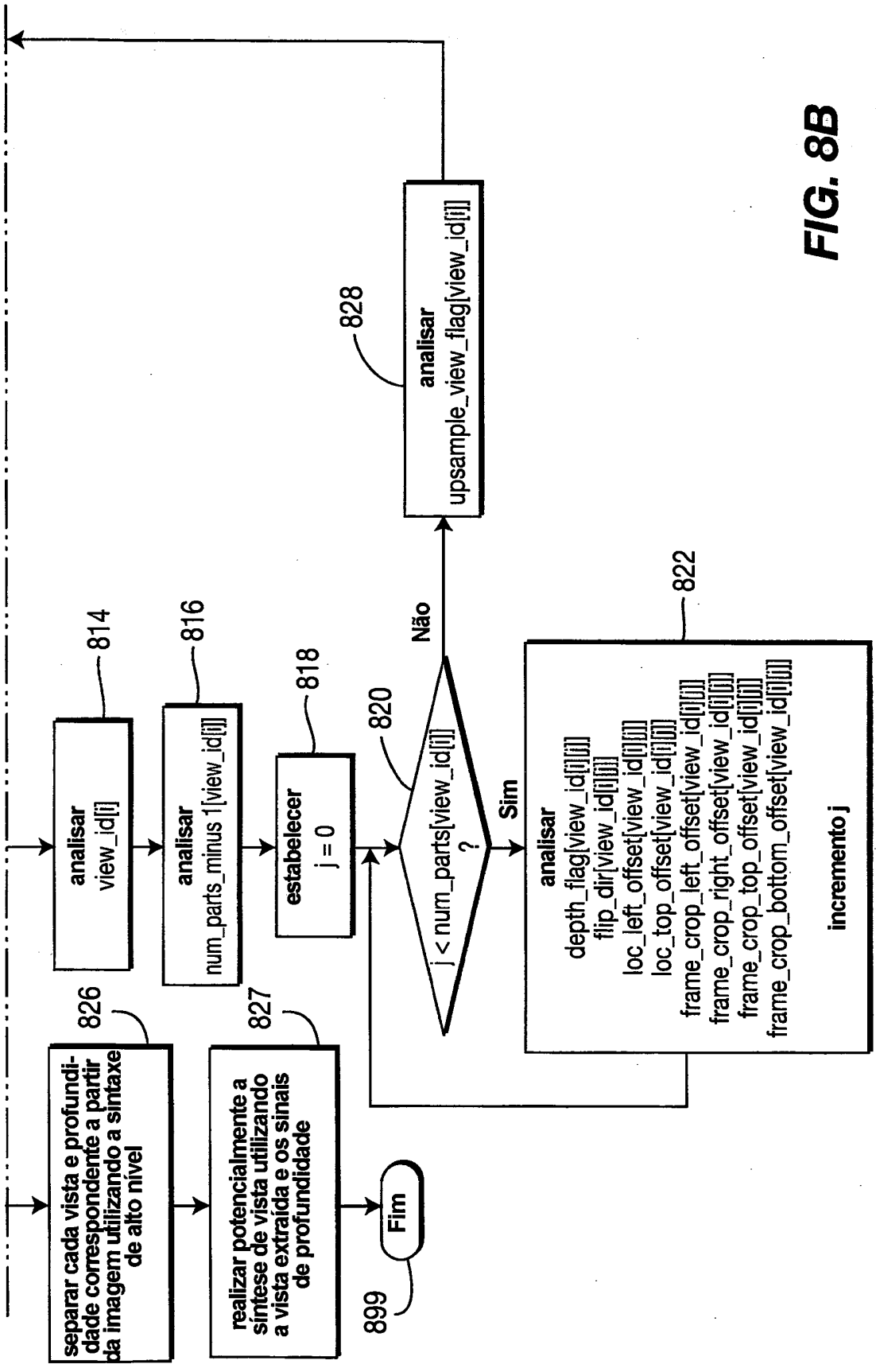

separar cada vista e profundidade correspondente a partir da imagem utilizando a sintaxe de alto nivel
826
realizar potencialmente a sintese de vista utilizando a vista extraida e os sinais de profundidade
827
Fim
899
analisar
view_id[i]
814
analisar
num_parts_minus 1[view_id[i]]
816
estabelecer
j = 0
818
j < num_parts[view_id[i]] ?
820
Não
Sim
analisar
upsample_view_flag[view_id[i]]
828
analisar
depth_flag[view_id[i][j]]
flip_dir[view_id[i][j]]
loc_left_offset[view_id[i][j]]
loc_top_offset[view_id[i][j]]
frame_crop_left_offset[view_id[i][j]]
frame_crop_right_offset[view_id[i][j]]
frame_crop_top_offset[view_id[i][j]]
frame_crop_bottom_offset[view_id[i][j]]
incremento j
822

FIG. 8B

900

FIG. 9

1000

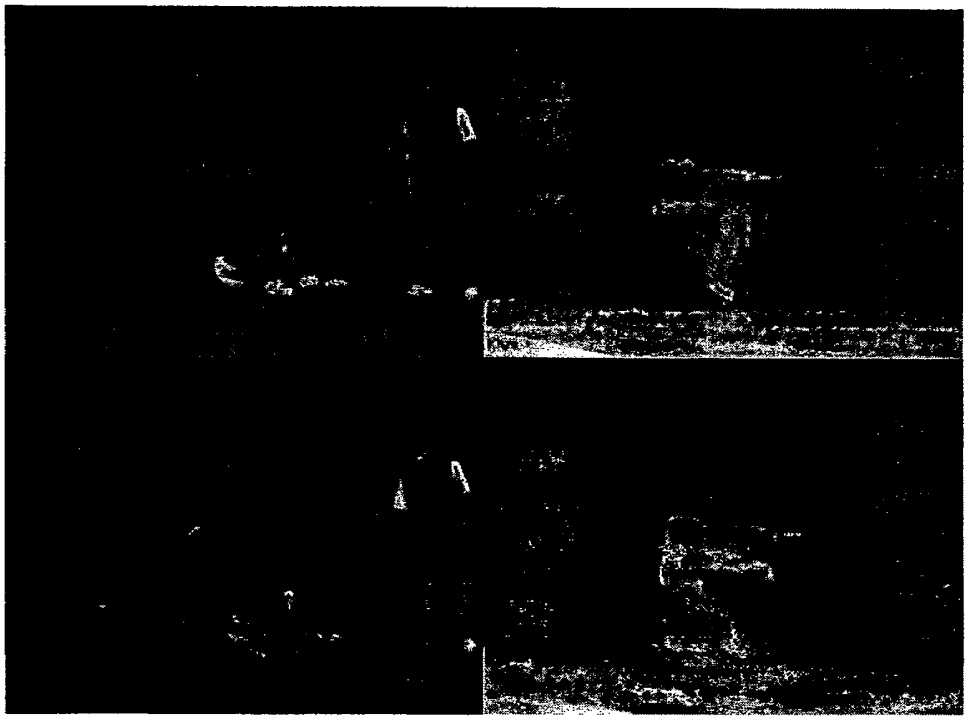


FIG. 10

1100

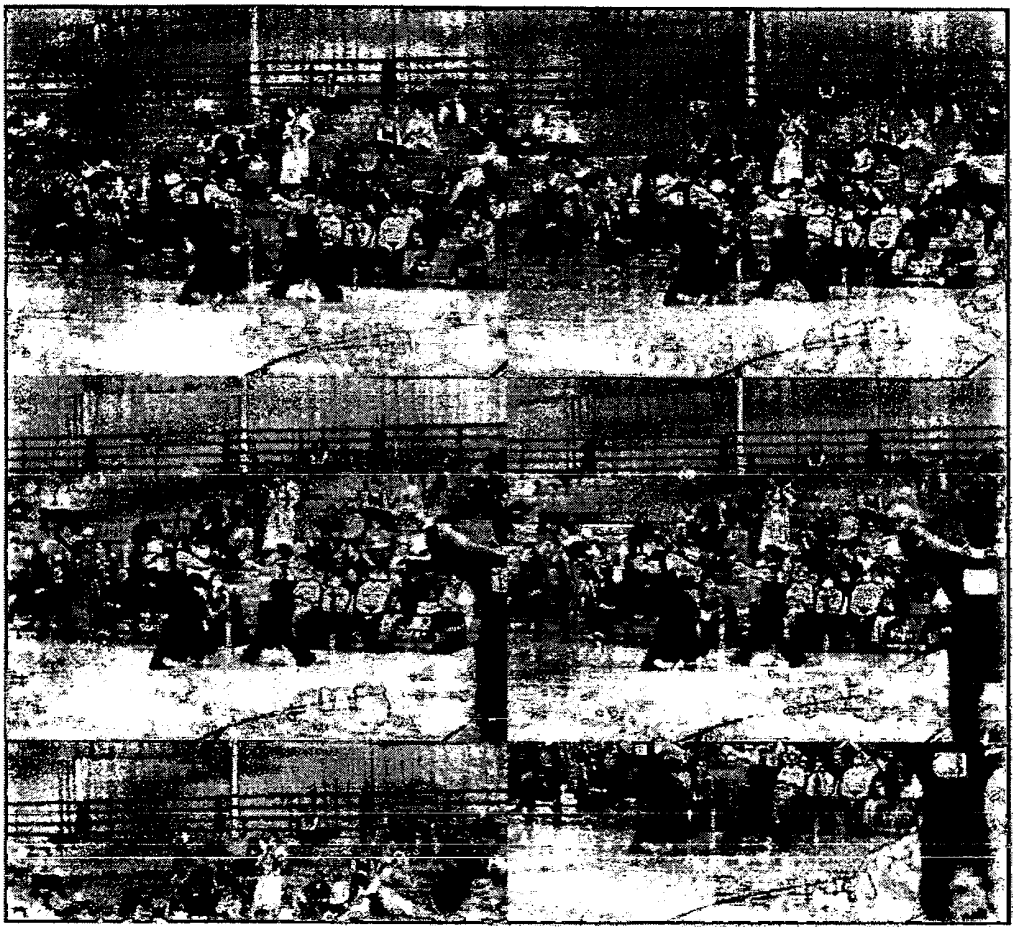


FIG. 11

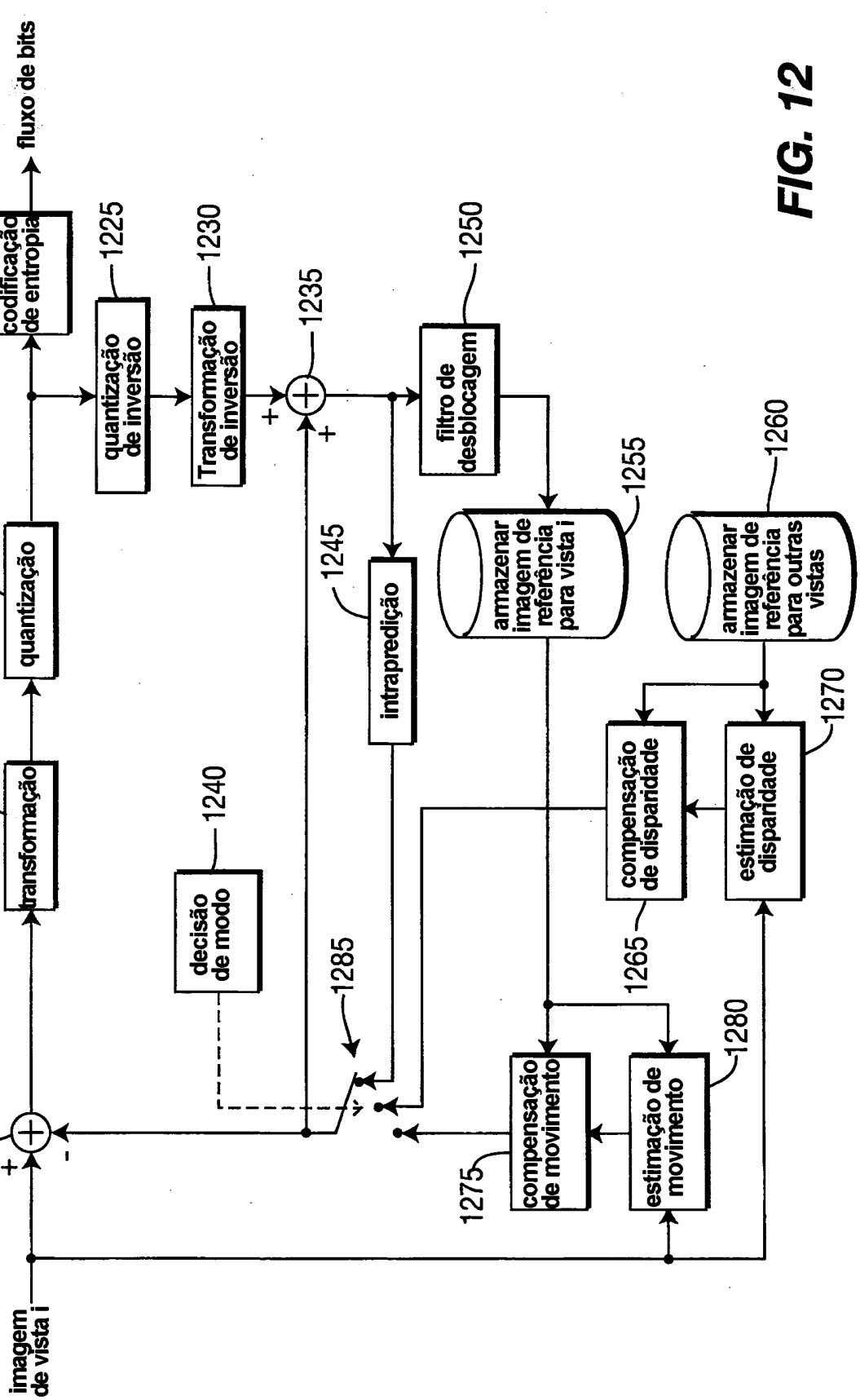

1200
fluxo de bits
1220
codificação de entropia
1225
quantização de inversão
1230
Transformação de inversão
1235
1250
filtro de desblocagem
1255
armazenar imagem de referência para vista i
1260
armazenar imagem de referência para outras vistas
1215
quantização
1245
intrapredição
1210
transformação
1240
decisão de modo
1270
compensação de disparidade
estimação de disparidade
1265
1285
1280
1205
compensação de movimento
estimação de movimento
1275
imagem de vista i

FIG. 12

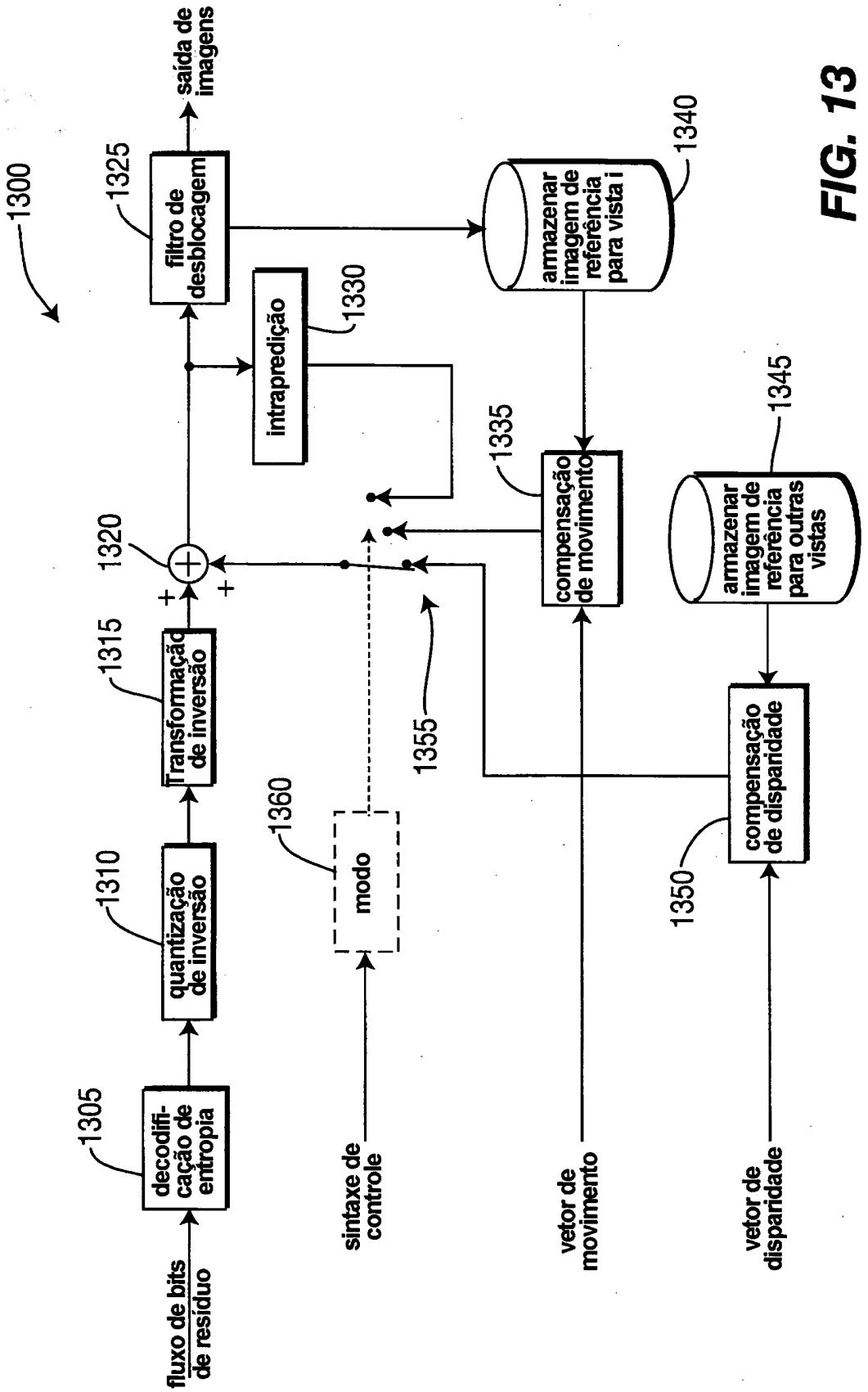

1300
1325
filtro de desblocagem
saída de imagens
1340
armazenar imagem de referência para vista i
1330
intrapredição
1335
compensação de movimento
1345
armazenar imagem de referência para outras vistas
1320
+
+
1315
Transformação de inversão
1310
quantização de inversão
1360
modo
1355
1305
decodificação de entropia
1350
compensação de disparidade
fluxo de bits de resíduo
sintaxe de controle
vetor de movimento
vetor de disparidade

FIG. 13

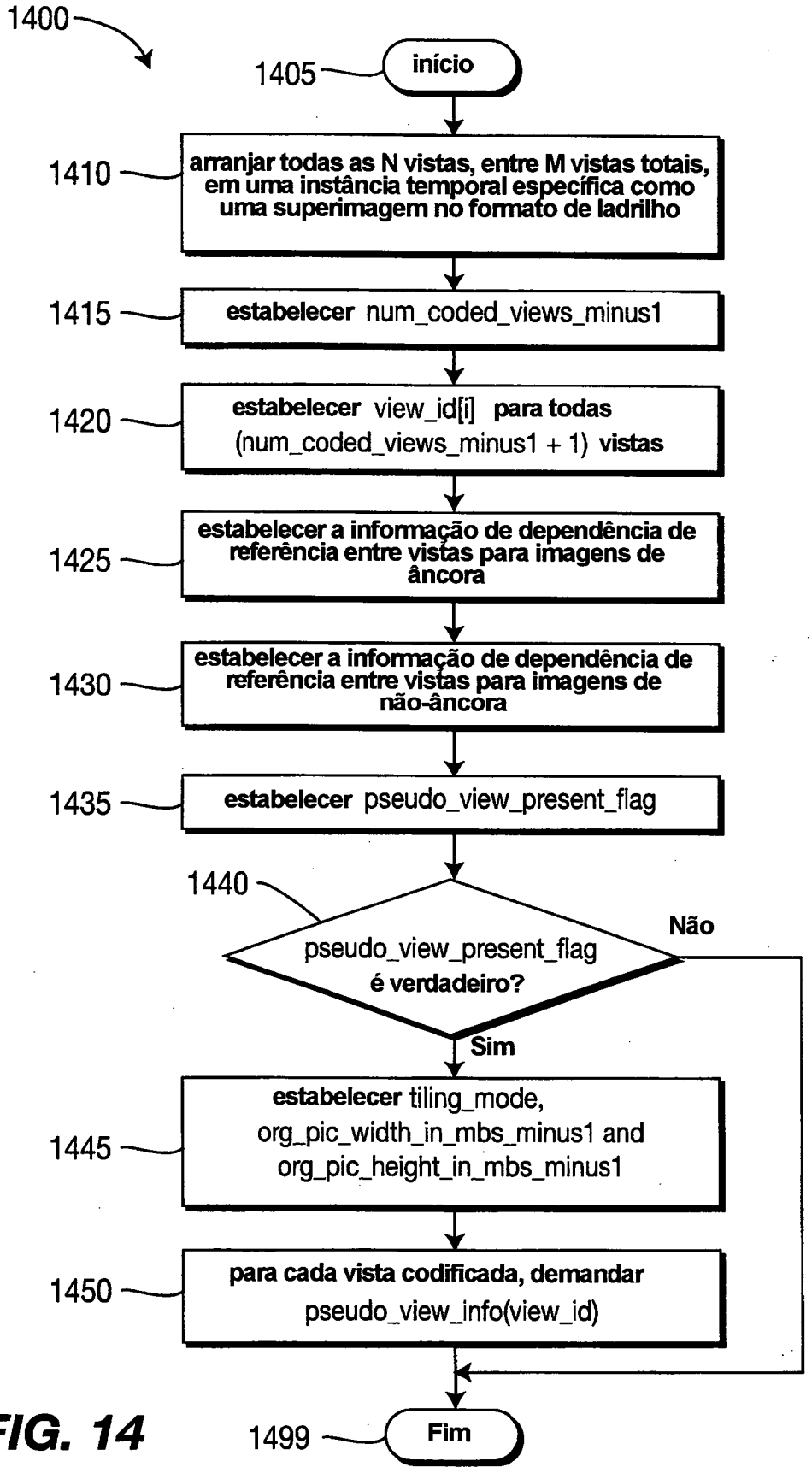
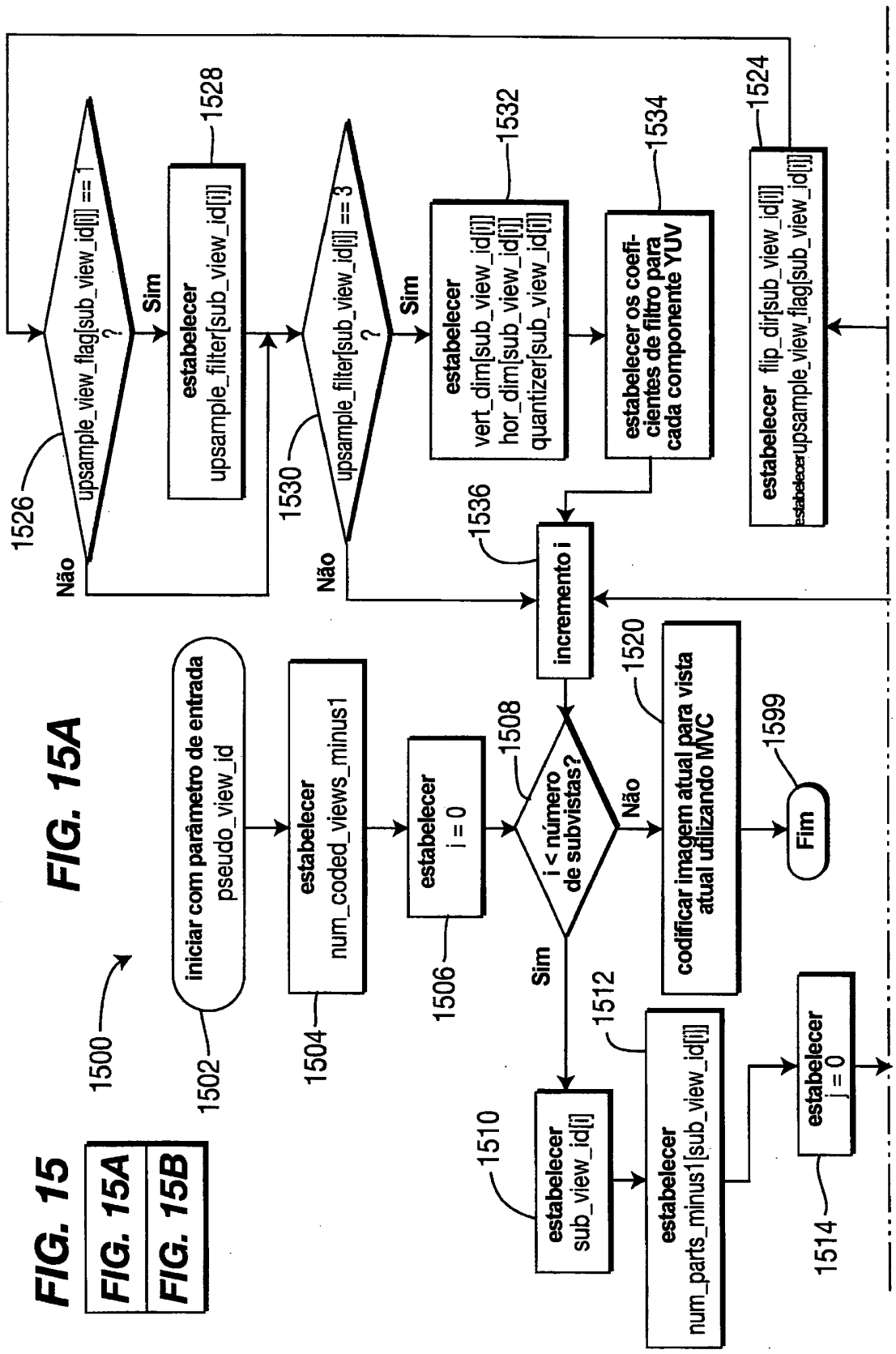

1400
1405
início
1410
arranjar todas as N vistas, entre M vistas totais, em uma instância temporal específica como uma superimagem no formato de ladrilho
1415
estabelecer num_coded_views_minus1
1420
estabelecer view_id[i] para todas (num_coded_views_minus1 + 1) vistas
1425
estabelecer a informação de dependência de referência entre vistas para imagens de âncora
1430
estabelecer a informação de dependência de referência entre vistas para imagens de não-âncora
1435
estabelecer pseudo_view_present_flag
1440
pseudo_view_present_flag é verdadeiro?
Não
Sim
1445
estabelecer tiling_mode, org_pic_width_in_mbs_minus1 and org_pic_height_in_mbs_minus1
1450
para cada vista codificada, demandar pseudo_view_info(view_id)
1499
Fim

FIG. 14

FIG. 15

FIG. 15A
FIG. 15B

FIG. 15A



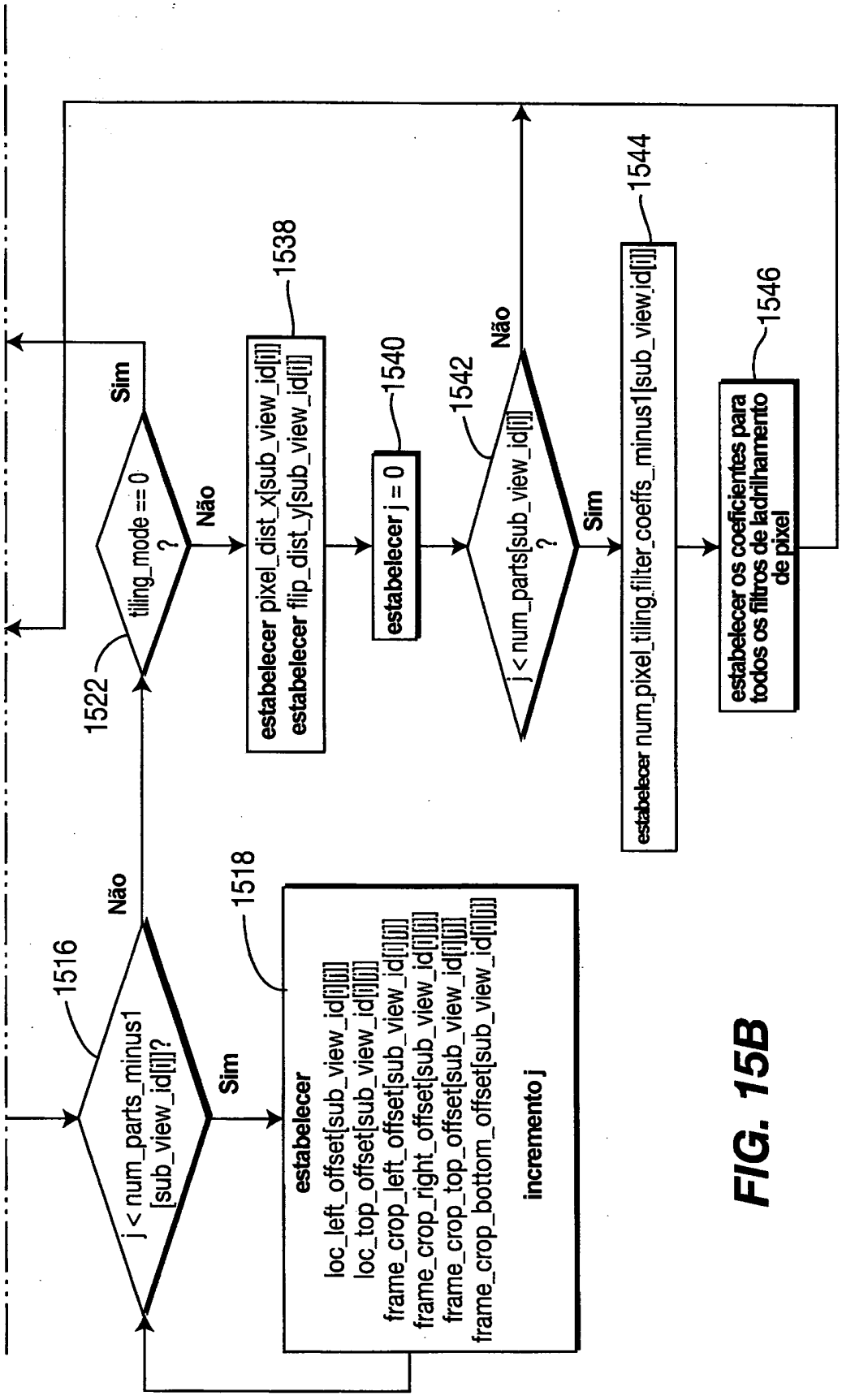

1516
j < num_parts_minus1 [sub_view_id[i]]?
Não
Sim
1518
estabelecer
loc_left_offset[sub_view_id[i]][j]
loc_top_offset[sub_view_id[i]][j]
frame_crop_left_offset[sub_view_id[i]][j]
frame_crop_right_offset[sub_view_id[i]][j]
frame_crop_top_offset[sub_view_id[i]][j]
frame_crop_bottom_offset[sub_view_id[i]][j]
incremento j
1522
tiling_mode == 0 ?
Sim
Não
1538
estabelecer pixel_dist_x[sub_view_id[i]]
estabelecer flip_dist_y[sub_view_id[i]]
1540
estabelecer j = 0
1542
j < num_parts[sub_view_id[i]] ?
Não
Sim
1544
estabelecer num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]]
1546
estabelecer os coeficientes para todos os filtros de ladrilhamento de pixel

FIG. 15B

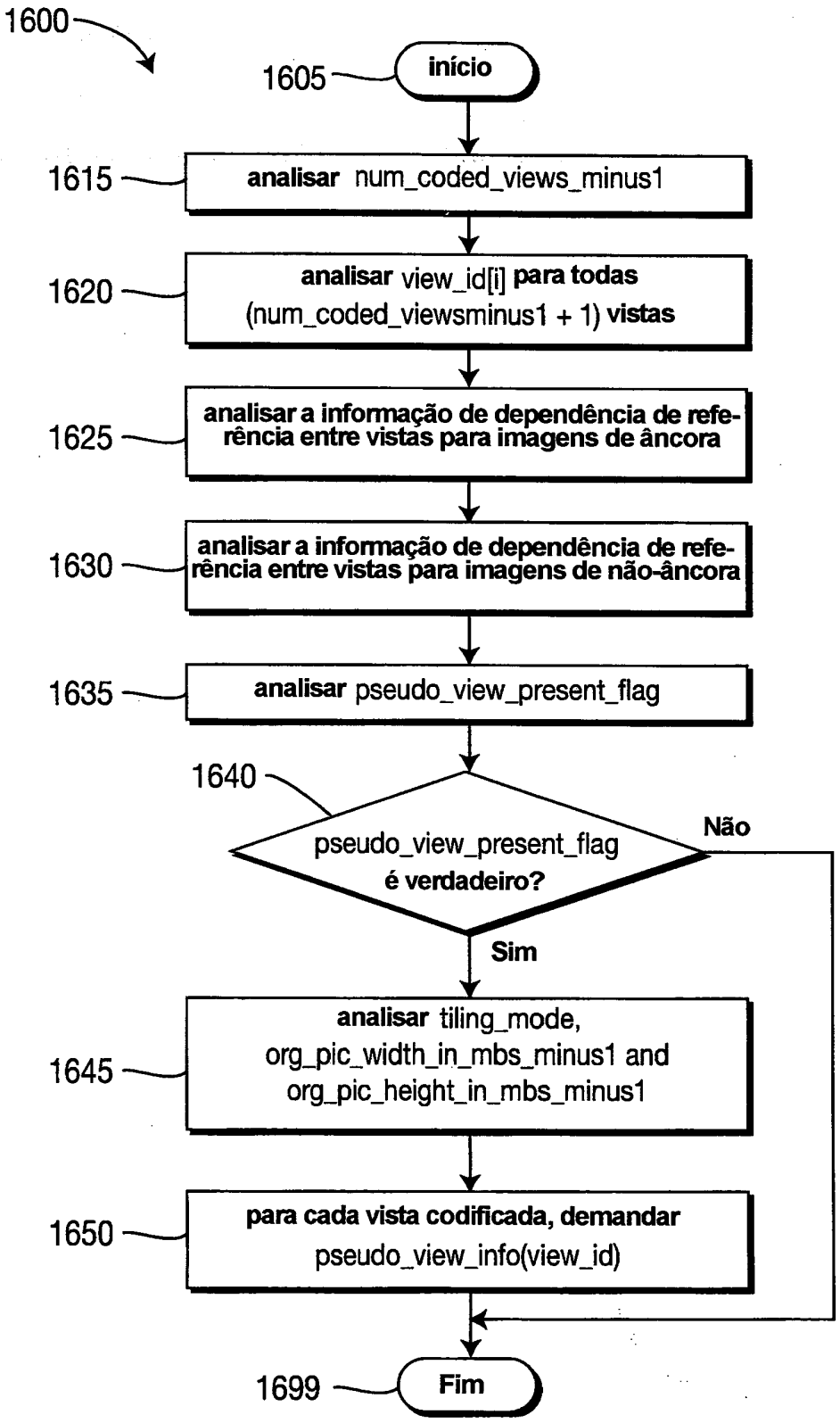

1600
1605
início
1615
analisar num_coded_views_minus1
1620
analisar view_id[i] para todas (num_coded_viewsminus1 + 1) vistas
1625
analisar a informação de dependência de referência entre vistas para imagens de âncora
1630
analisar a informação de dependência de referência entre vistas para imagens de não-âncora
1635
analisar pseudo_view_present_flag
1640
pseudo_view_present_flag é verdadeiro?
Não
Sim
1645
analisar tiling_mode, org_pic_width_in_mbs_minus1 and org_pic_height_in_mbs_minus1
1650
para cada vista codificada, demandar pseudo_view_info(view_id)
1699
Fim

FIG. 16

FIG. 17

FIG. 17A
FIG. 17B

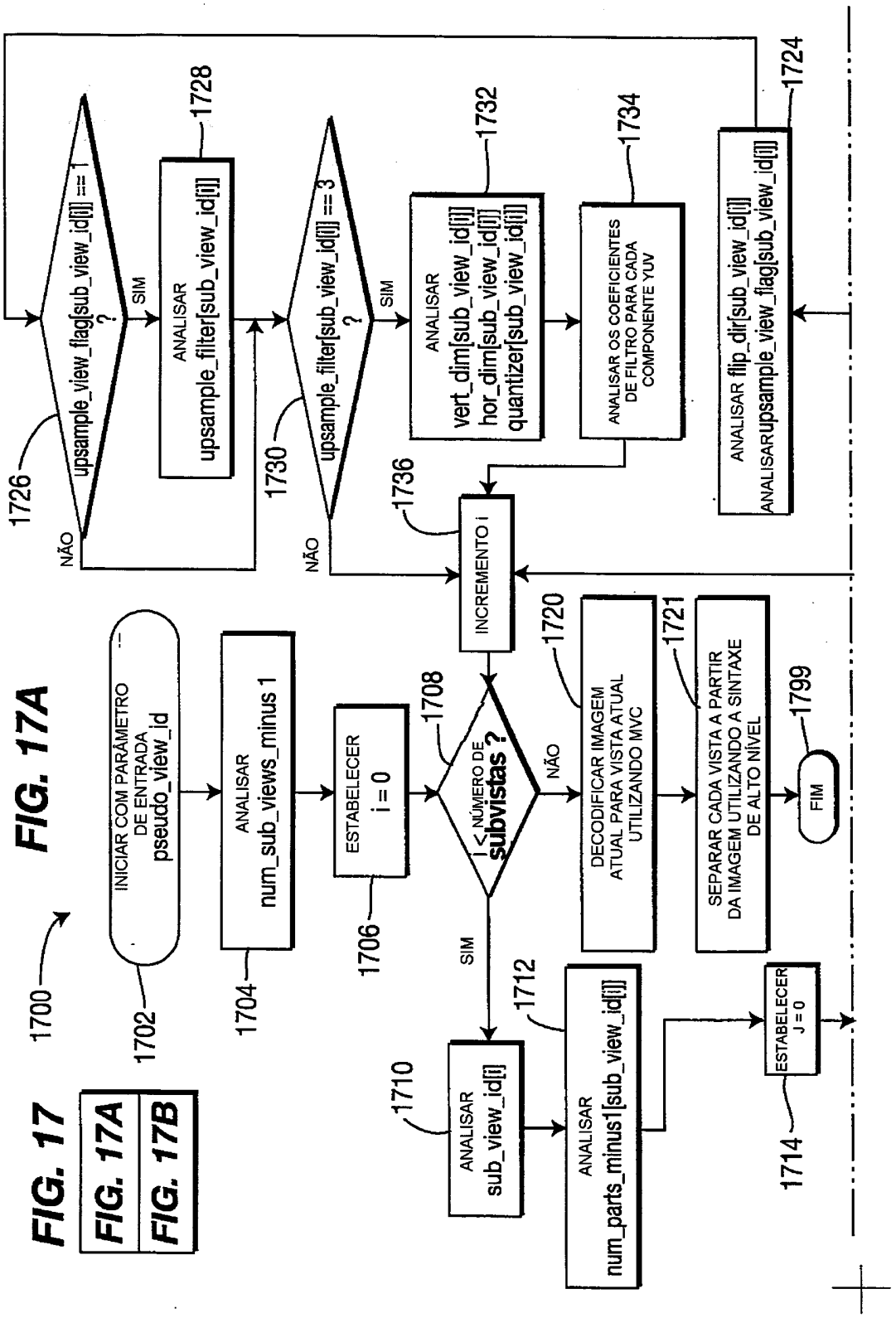


FIG. 17A

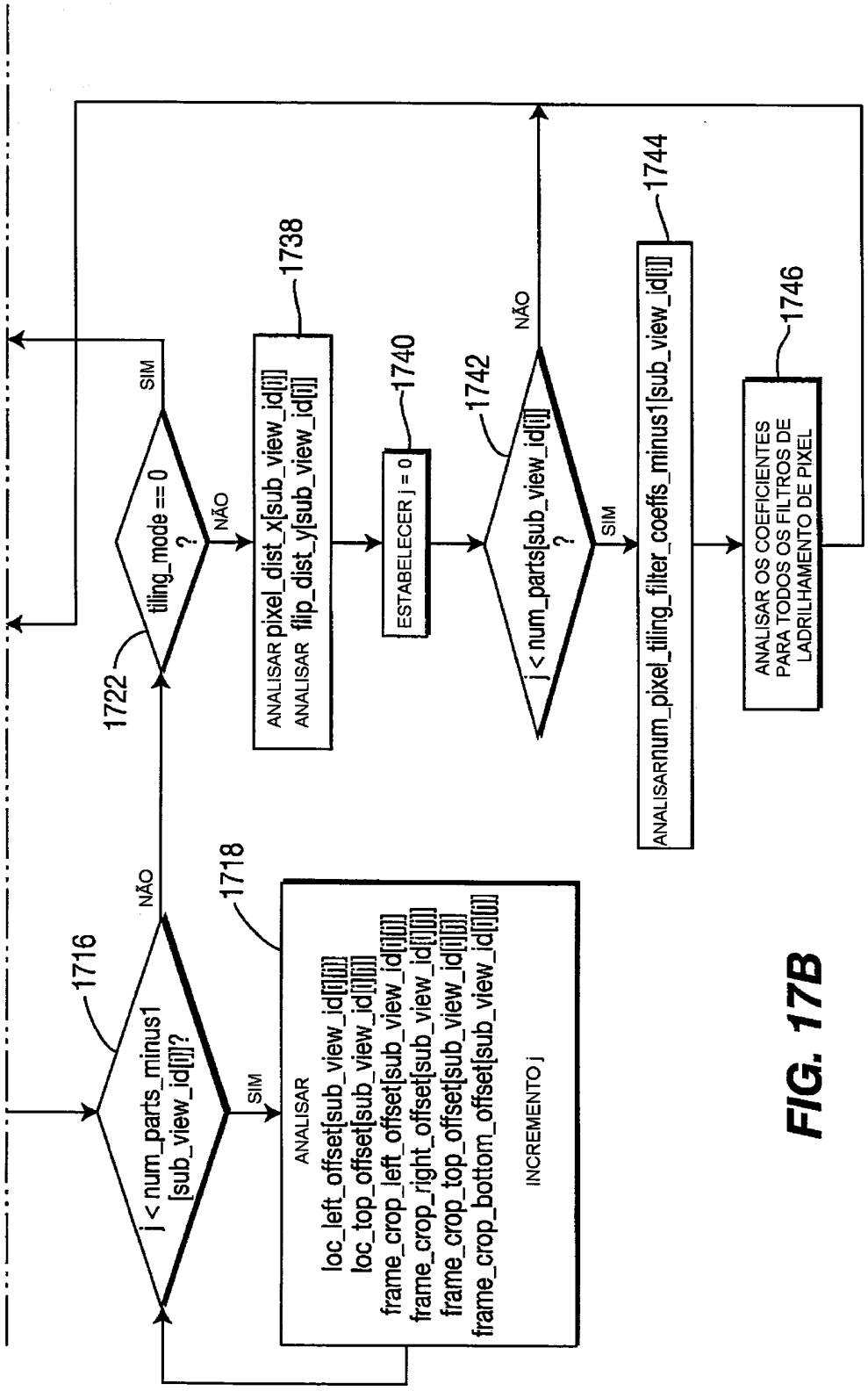

1716
j < num_parts_minus1 [sub_view_id[i]]?
NÃO
SIM
1718
ANALISAR
loc_left_offset[sub_view_id[i]][j]
loc_top_offset[sub_view_id[i]][j]
frame_crop_left_offset[sub_view_id[i]][j]
frame_crop_right_offset[sub_view_id[i]][j]
frame_crop_top_offset[sub_view_id[i]][j]
frame_crop_bottom_offset[sub_view_id[i]][j]
INCREMENTO j
1722
tiling_mode == 0 ?
SIM
NÃO
1738
ANALISAR pixel_dist_x[sub_view_id[i]]
ANALISAR flip_dist_y[sub_view_id[i]]
1740
ESTABELECER j = 0
1742
j < num_parts[sub_view_id[i]] ?
NÃO
SIM
1744
ANALISARnum_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]]
1746
ANALISAR OS COEFICIENTES PARA TODOS OS FILTROS DE LADRILHAMENTO DE PIXEL

FIG. 17B

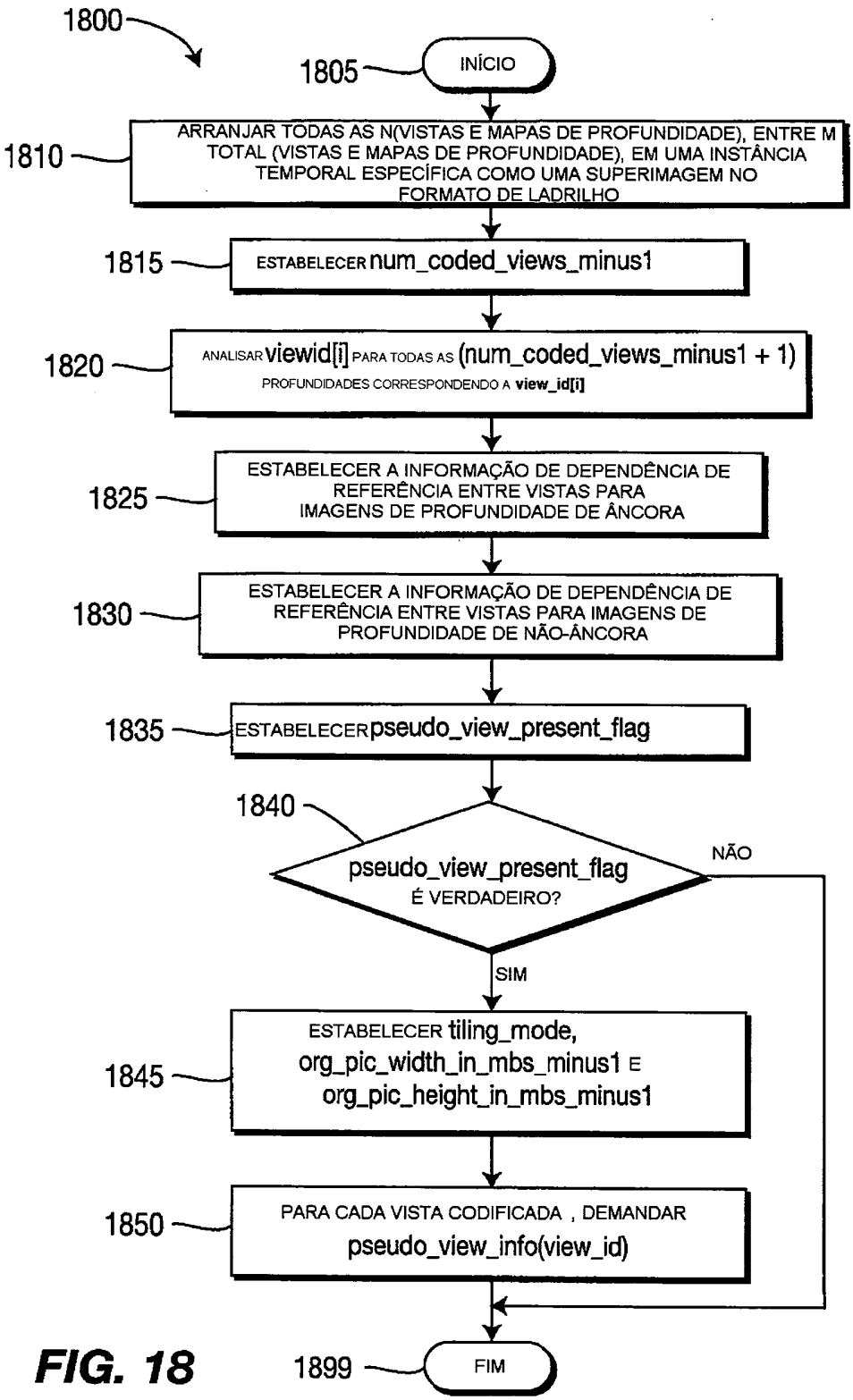

1800
1805
INÍCIO
1810
ARRANJAR TODAS AS N(VISTAS E MAPAS DE PROFUNDIDADE), ENTRE M TOTAL (VISTAS E MAPAS DE PROFUNDIDADE), EM UMA INSTÂNCIA TEMPORAL ESPECÍFICA COMO UMA SUPERIMAGEM NO FORMATO DE LADRILHO
1815
ESTABELECER num_coded_views_minus1
1820
ANALISAR viewid[i] PARA TODAS AS (num_coded_views_minus1 + 1) PROFUNDIDADES CORRESPONDENDO A view_id[i]
1825
ESTABELECER A INFORMAÇÃO DE DEPENDÊNCIA DE REFERÊNCIA ENTRE VISTAS PARA IMAGENS DE PROFUNDIDADE DE ÂNCORA
1830
ESTABELECER A INFORMAÇÃO DE DEPENDÊNCIA DE REFERÊNCIA ENTRE VISTAS PARA IMAGENS DE PROFUNDIDADE DE NÃO-ÂNCORA
1835
ESTABELECER pseudo_view_present_flag
1840
pseudo_view_present_flag É VERDADEIRO?
NÃO
SIM
1845
ESTABELECER tiling_mode, org_pic_width_in_mbs_minus1 E org_pic_height_in_mbs_minus1
1850
PARA CADA VISTA CODIFICADA , DEMANDAR pseudo_view_info(view_id)
1899
FIM

FIG. 18

FIG. 19

FIG. 19A
FIG. 19B

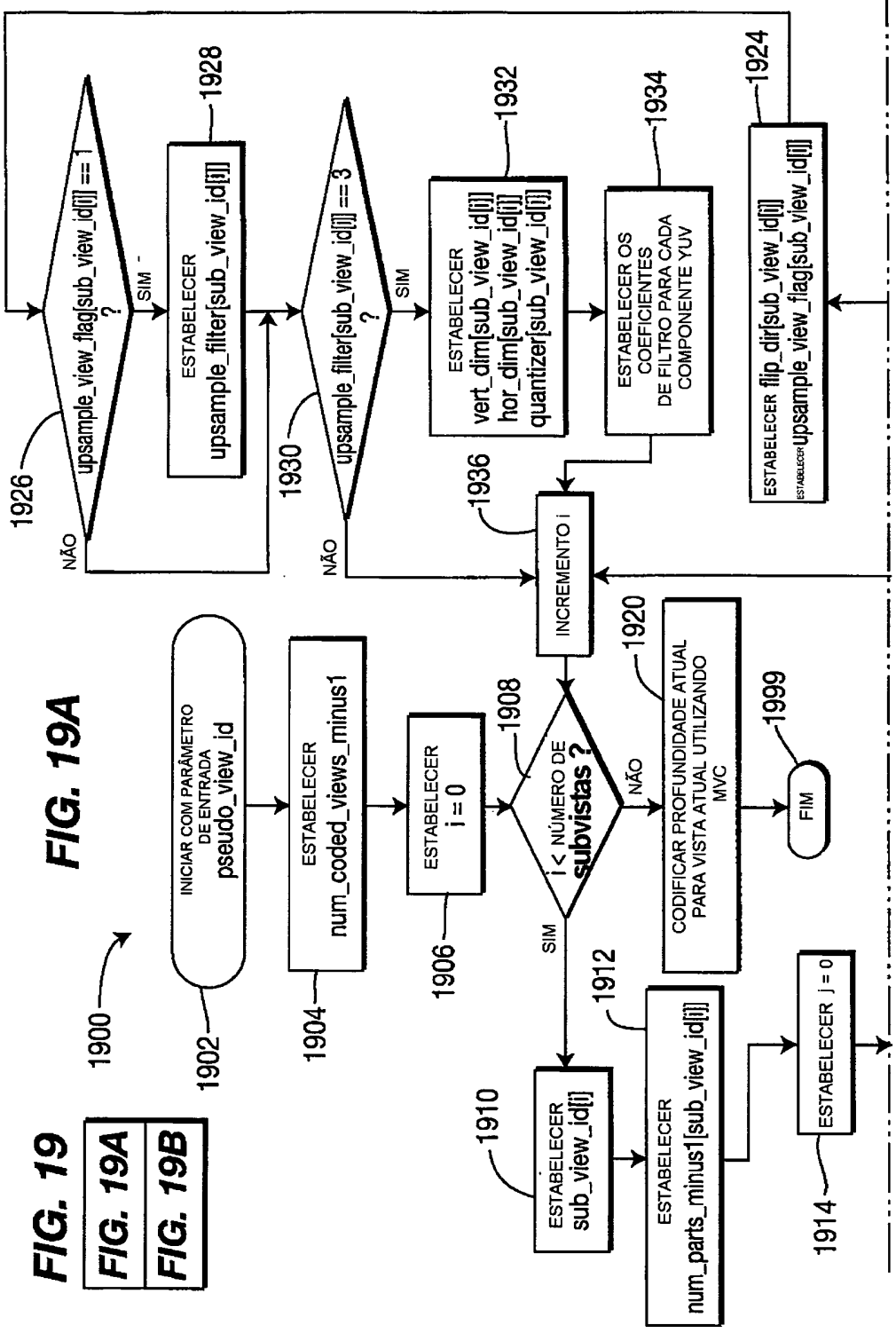


FIG. 19A

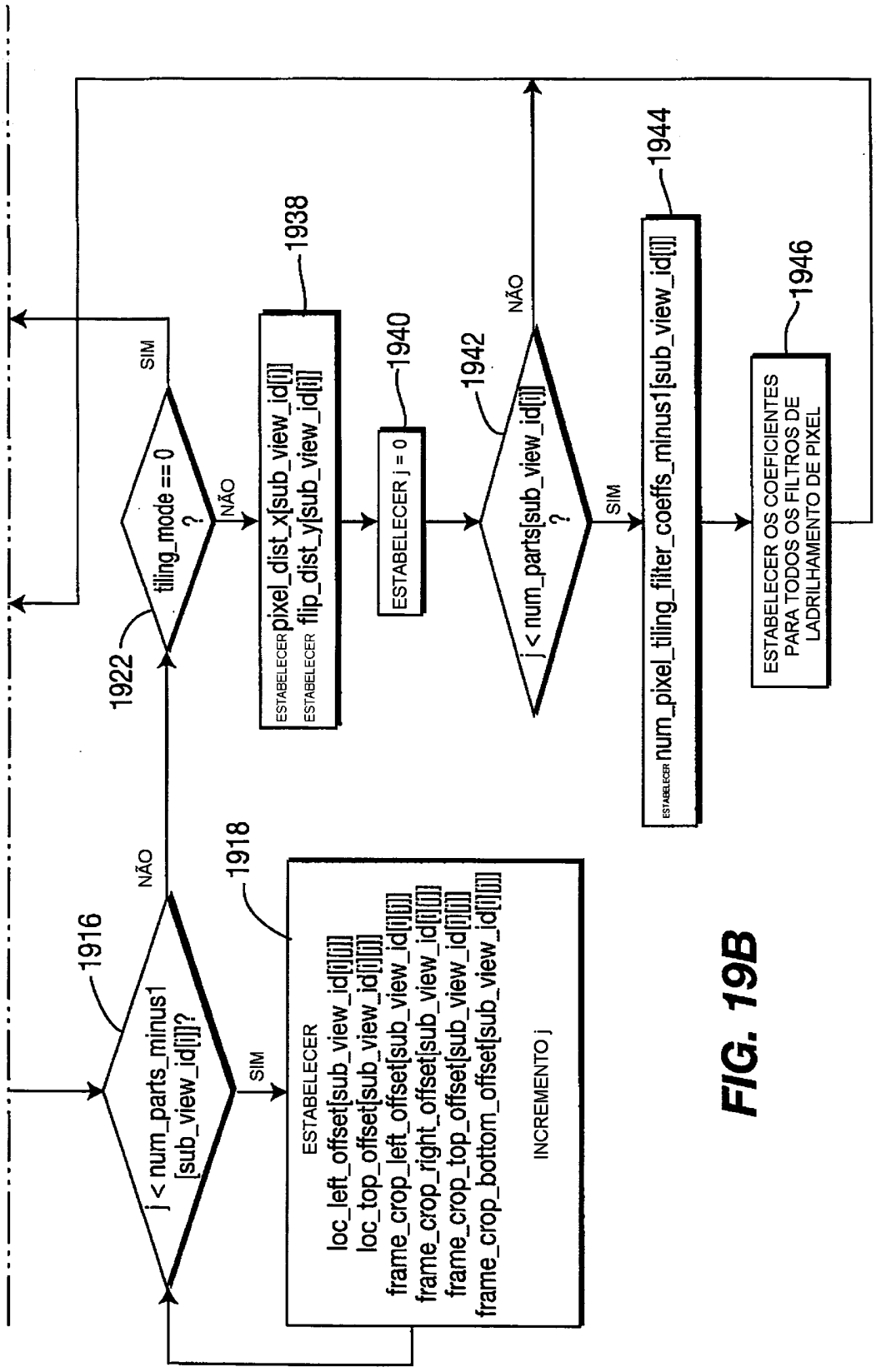

1916
j < num_parts_minus1 [sub_view_id[i]]?
NÃO
SIM
1918
ESTABELECER
loc_left_offset[sub_view_id[i][j]]
loc_top_offset[sub_view_id[i][j]]
frame_crop_left_offset[sub_view_id[i][j]]
frame_crop_right_offset[sub_view_id[i][j]]
frame_crop_top_offset[sub_view_id[i][j]]
frame_crop_bottom_offset[sub_view_id[i][j]]
INCREMENTO j
1922
tiling_mode == 0 ?
SIM
NÃO
1938
ESTABELECER pixel_dist_x[sub_view_id[i]]
ESTABELECER flip_dist_y[sub_view_id[i]]
1940
ESTABELECER j = 0
1942
j < num_parts[sub_view_id[i]] ?
NÃO
SIM
1944
ESTABELECER num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]]
1946
ESTABELECER OS COEFICIENTES PARA TODOS OS FILTROS DE LADRILHAMENTO DE PIXEL

FIG. 19B

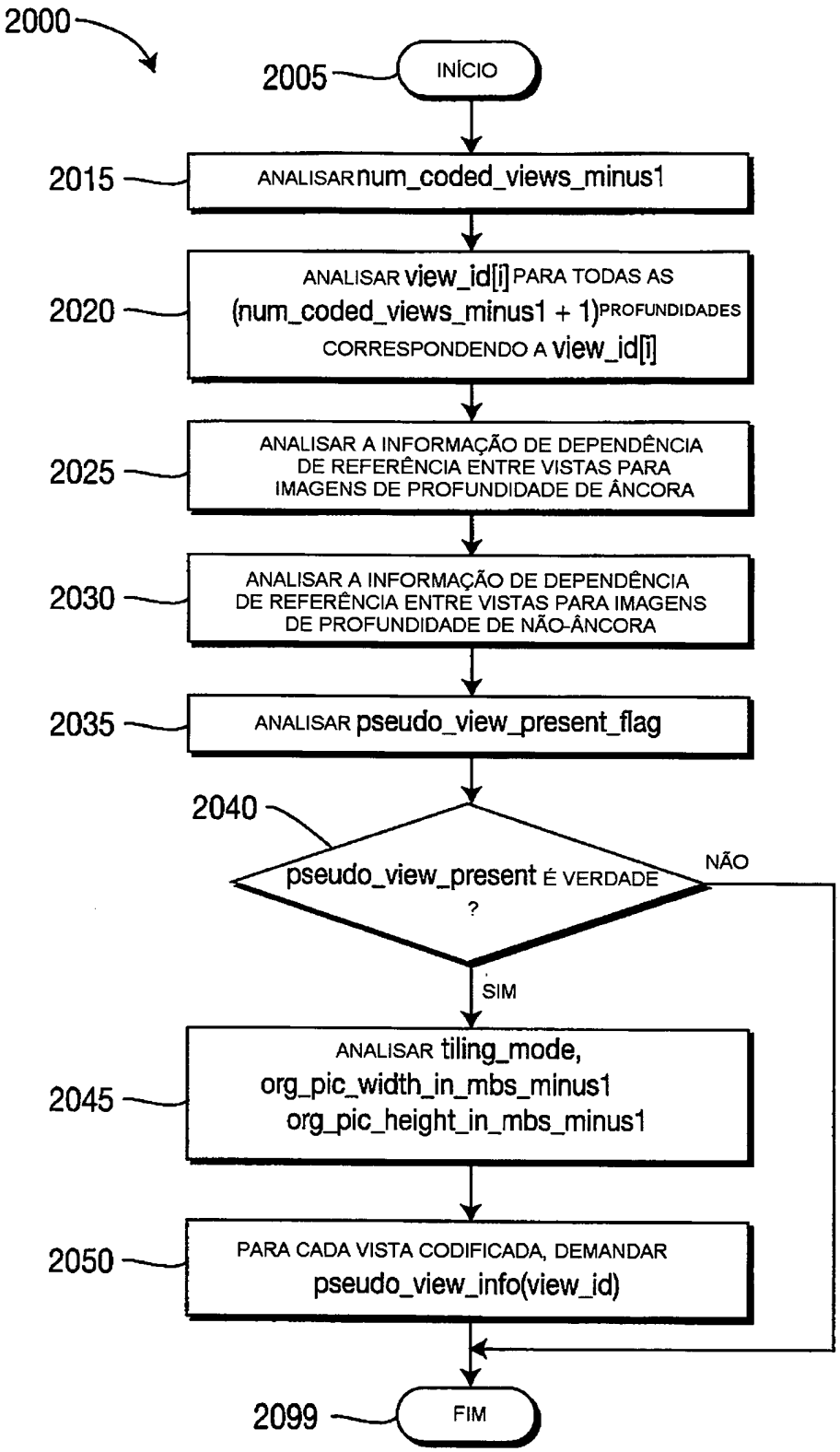
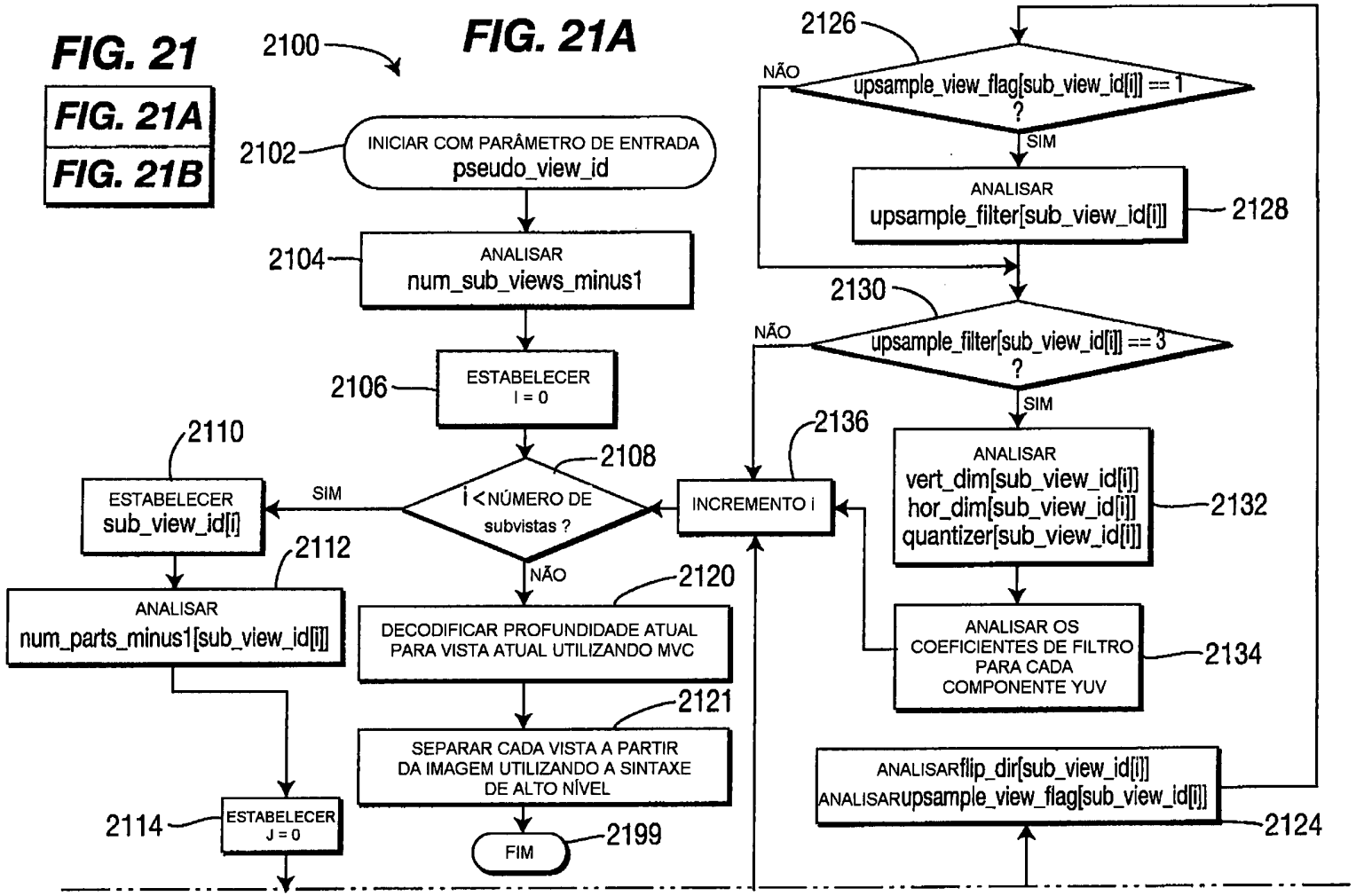

2000
2005
INÍCIO
2015
ANALISAR num_coded_views_minus1
2020
ANALISAR view_id[i] PARA TODAS AS (num_coded_views_minus1 + 1) PROFUNDIDADES CORRESPONDENDO A view_id[i]
2025
ANALISAR A INFORMAÇÃO DE DEPENDÊNCIA DE REFERÊNCIA ENTRE VISTAS PARA IMAGENS DE PROFUNDIDADE DE ÂNCORA
2030
ANALISAR A INFORMAÇÃO DE DEPENDÊNCIA DE REFERÊNCIA ENTRE VISTAS PARA IMAGENS DE PROFUNDIDADE DE NÃO-ÂNCORA
2035
ANALISAR pseudo_view_present_flag
2040
pseudo_view_present É VERDADE ?
NÃO
SIM
2045
ANALISAR tiling_mode, org_pic_width_in_mbs_minus1 org_pic_height_in_mbs_minus1
2050
PARA CADA VISTA CODIFICADA, DEMANDAR pseudo_view_info(view_id)
2099
FIM

FIG. 20

FIG. 21

FIG. 21A

FIG. 21B



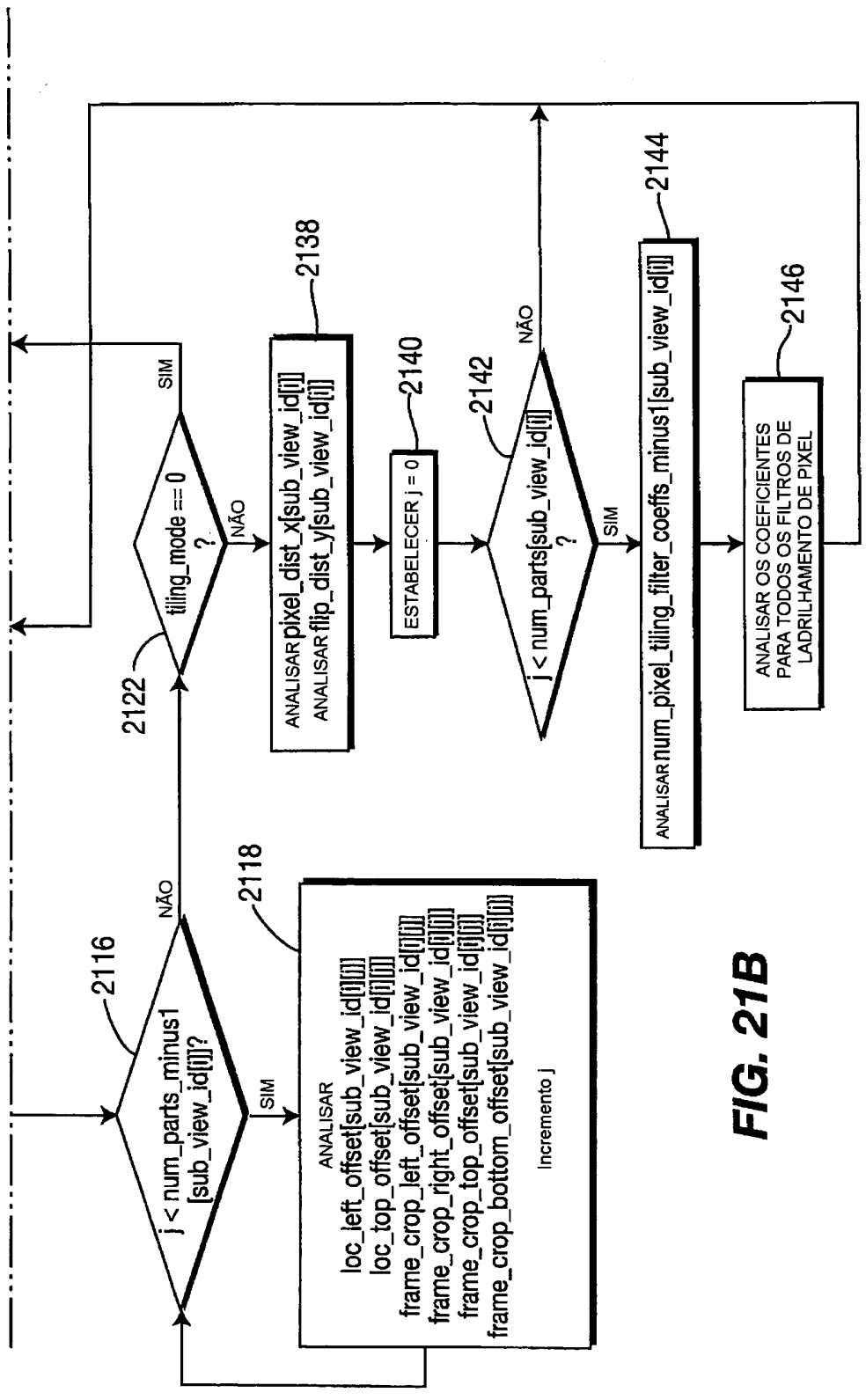

2116
j < num_parts_minus1 [sub_view_id[i]]?
NÃO
SIM
2118
ANALISAR
loc_left_offset[sub_view_id[i]][j]
loc_top_offset[sub_view_id[i]][j]
frame_crop_left_offset[sub_view_id[i]][j]
frame_crop_right_offset[sub_view_id[i]][j]
frame_crop_top_offset[sub_view_id[i]][j]
frame_crop_bottom_offset[sub_view_id[i]][j]
Incremento j
2122
tiling_mode == 0 ?
SIM
NÃO
2138
ANALISAR pixel_dist_x[sub_view_id[i]]
ANALISAR flip_dist_y[sub_view_id[i]]
2140
ESTABELECER j = 0
2142
j < num_parts[sub_view_id[i]] ?
NÃO
SIM
2144
ANALISAR num_pixel_tiling_filter_coeffs_minus1[sub_view_id[i]]
2146
ANALISAR OS COEFICIENTES PARA TODOS OS FILTROS DE LADRILHAMENTO DE PIXEL

FIG. 21B

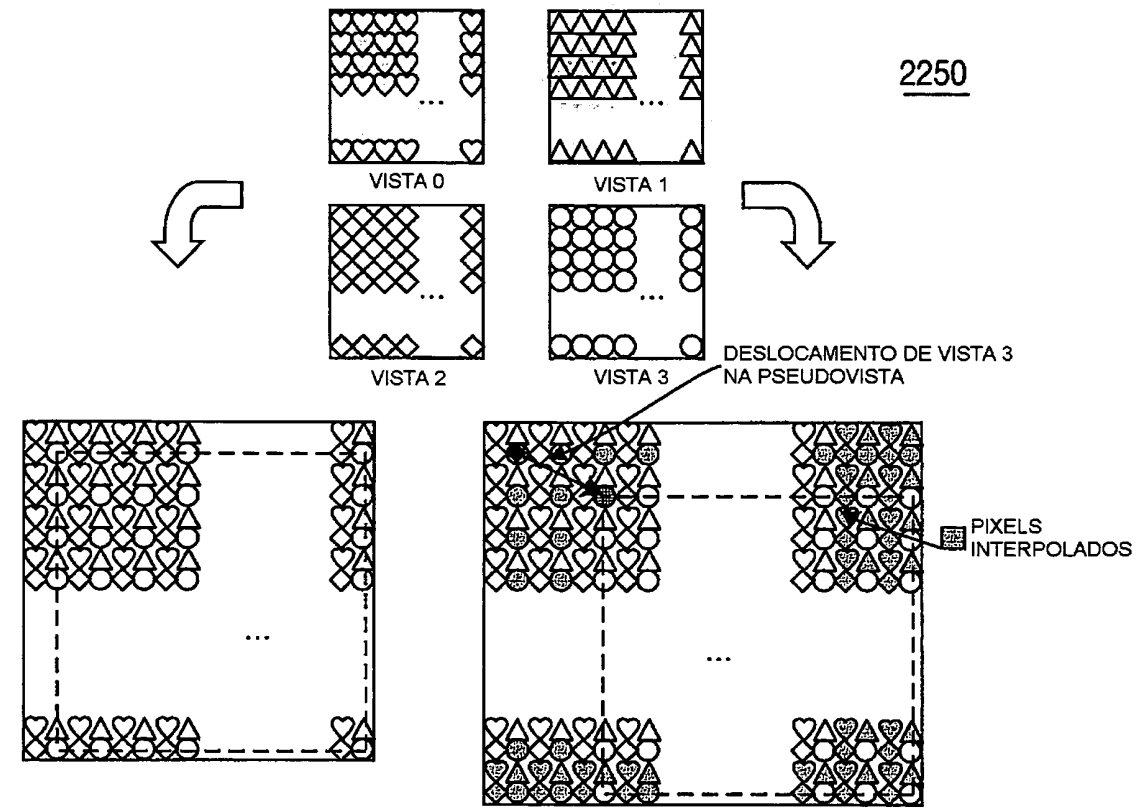

2250
VISTA 0
VISTA 1
VISTA 2
VISTA 3
DESLOCAMENTO DE VISTA 3 NA PSEUDOVISTA
PIXELS INTERPOLADOS
EXEMPLO 1: LADRILHAMENTO EM NÍVEL DE PIXEL
2210
EXEMPLO 2: LADRILHAMENTO EM NÍVEL DE PIXEL
2220

FIG. 22

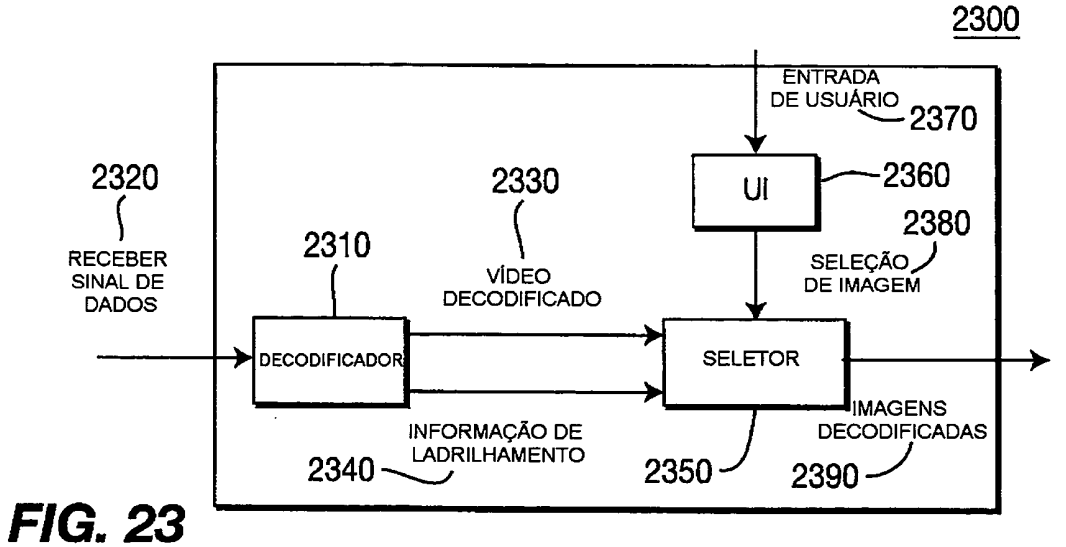

2300
ENTRADA DE USUÁRIO
2370
UI
2360
2380
SELEÇÃO DE IMAGEM
2320
RECEBER SINAL DE DADOS
2330
VÍDEO DECODIFICADO
2310
DECODIFICADOR
SELETOR
IMAGENS DECODIFICADAS
INFORMAÇÃO DE LADRILHAMENTO
2340
2350
2390

FIG. 23