(12) BẢN MÔ TẢ SÁNG CHẾ THUỘC BẰNG ĐỘC QUYỀN SÁNG CHẾ

(19) **Cộng hòa xã hội chủ nghĩa Việt Nam (VN)**
CỤC SỞ HỮU TRÍ TUỆ

(11) **1-0052892**

(51)[2020.01] **H04N 19/46**

(13) **B**

(21) 1-2022-02495
(22) 19/10/2020
(86) PCT/CN2020/121767 19/10/2020
(87) WO 2021/073630 22/04/2021
(30) PCT/CN2019/111807 18/10/2019 CN
(45) 25/11/2025 452
(43) 25/08/2022 413A
(73) 1. BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. (CN)
Room B-0035, 2/F, No.3 Building No.30, Shixing Road, Shijingshan District Beijing 100041, P.R. China
2. BYTEDANCE INC. (US)
12655 West Jefferson Boulevard Sixth Floor, Suite No. 137 Los Angeles, California 90066, United States of America
(72) ZHANG, Kai (CN); DENG, Zhipin (CN); LIU, Hongbin (CN); ZHANG, Li (CN); XU, Jizheng (CN).
(74) Công ty Luật TNHH Phạm và Liên danh (PHAM & ASSOCIATES)

(54) PHƯƠNG PHÁP VÀ THIẾT BỊ XỬ LÝ DỮ LIỆU VIĐEO, VÀ VẬT GHI

(21)1-2022-02495

(57) Sáng chế đề xuất phương pháp xử lý viđeo được nêu làm ví dụ bao gồm bước thực hiện sự biến đổi giữa ảnh của viđeo và biểu diễn dòng bit của viđeo. Ảnh bao gồm một hoặc nhiều ảnh con, và biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra rằng chiều dài của phần tử cú pháp là bằng Ceil(Log2(SS)) bit. SS là lớn hơn 0, và phần tử cú pháp chỉ báo vị trí ngang hoặc dọc của góc trên cùng bên trái của đơn vị cây mã hóa của ảnh con của ảnh.

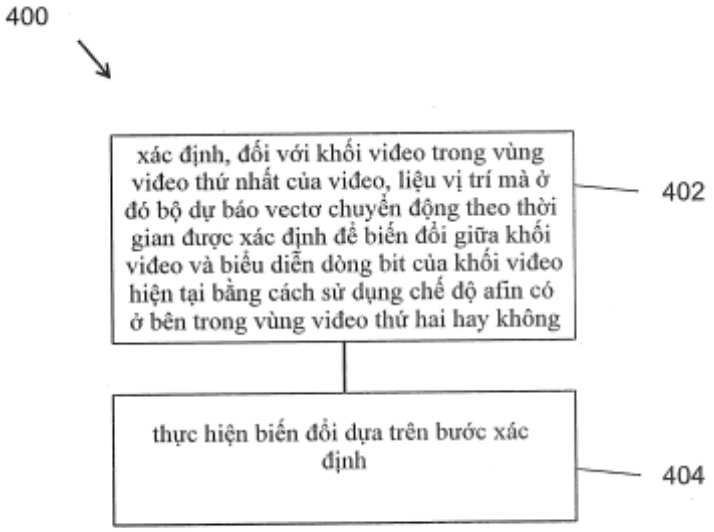

FIG. 4

Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập tới các công nghệ mã hóa và giải mã viđeo và hình ảnh.

Tình trạng kỹ thuật của sáng chế

Viđeo kỹ thuật số sử dụng băng thông lớn nhất trên mạng internet và các mạng truyền thông kỹ thuật số khác. Do số lượng các thiết bị người dùng được kết nối có khả năng nhận và hiển thị viđeo tăng lên, nên dự kiến rằng yêu cầu băng thông dành cho việc sử dụng viđeo kỹ thuật số sẽ tiếp tục tăng.

Bản chất kỹ thuật của sáng chế

Các kỹ thuật theo sáng chế có thể được sử dụng bởi các phương án bộ giải mã hoặc bộ tạo mã viđeo hoặc hình ảnh mà trong đó việc mã hóa hoặc giải mã dựa trên ảnh con được thực hiện.

Theo một khía cạnh được nêu làm ví dụ, phương pháp xử lý viđeo được bộc lộ. Phương pháp này bao gồm bước thực hiện sự biến đổi giữa ảnh của viđeo và biểu diễn dòng bit của viđeo. Ảnh bao gồm một hoặc nhiều ảnh con, và biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra rằng chiều dài của phần tử cú pháp là bằng Ceil(Log2(SS)) bit. SS là lớn hơn 0, và phần tử cú pháp chỉ báo vị trí ngang hoặc dọc của góc trên cùng bên trái của đơn vị cây mã hóa của ảnh con của ảnh.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo được bộc lộ. Phương pháp này bao gồm bước thực hiện sự biến đổi giữa ảnh của viđeo và biểu diễn dòng bit của viđeo, trong đó ảnh bao gồm một hoặc nhiều ảnh con. Biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra rằng các ảnh con khác nhau có các bộ nhận dạng khác nhau.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo được bộc lộ. Phương pháp này bao gồm bước xác định, đối với khối viđeo trong vùng

viđeo thứ nhất của viđeo, liệu vị trí mà ở đó bộ dự báo vectơ chuyển động theo thời gian được xác định để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại bằng cách sử dụng chế độ afin có ở bên trong vùng viđeo thứ hai hay không; và thực hiện sự biến đổi dựa trên xác định này.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo khác được bộc lộ. Phương pháp này bao gồm bước xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó mẫu nguyên trong ảnh tham chiếu được tìm nạp để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại có ở bên trong vùng viđeo thứ hai hay không, trong đó ảnh tham chiếu không được sử dụng trong quá trình nội suy trong khi biến đổi; và thực hiện sự biến đổi dựa trên xác định này.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo khác được bộc lộ. Phương pháp này bao gồm bước xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó giá trị mẫu độ sáng được tái tạo được tìm nạp để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại có ở bên trong vùng viđeo thứ hai hay không; và thực hiện sự biến đổi dựa trên xác định này.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo khác được bộc lộ. Phương pháp này bao gồm bước xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó việc kiểm tra liên quan đến tách, dẫn xuất sâu hoặc báo tín hiệu cờ tách đối với khối viđeo được thực hiện trong khi biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại có ở bên trong vùng viđeo thứ hai hay không; và thực hiện sự biến đổi dựa trên xác định này.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo khác được bộc lộ. Phương pháp này bao gồm bước thực hiện sự biến đổi giữa viđeo bao gồm một hoặc nhiều ảnh viđeo bao gồm một hoặc nhiều khối viđeo, và sự biểu diễn được mã hóa của viđeo, trong đó sự biểu diễn được mã hóa phù hợp với yêu cầu cú pháp mã hóa rằng sự biến đổi không được sử dụng mã hóa/giải mã ảnh con và công cụ mã hóa/giải mã biến đổi độ phân giải động hoặc công cụ lấy mẫu lại ảnh tham chiếu bên trong đơn vị viđeo.

Theo một khía cạnh khác được nêu làm ví dụ, phương pháp xử lý viđeo khác được bộc lộ. Phương pháp này bao gồm bước thực hiện sự biến đổi giữa viđeo bao gồm một hoặc nhiều ảnh viđeo bao gồm một hoặc nhiều khối viđeo, và sự biểu diễn được mã hóa của viđeo, trong đó sự biểu diễn được mã hóa phù hợp với yêu cầu cú pháp mã hóa rằng phần tử cú pháp thứ nhất subpic_grid_idx[i][j] không lớn hơn phần tử cú pháp thứ hai max_subpics_minus1.

Theo một khía cạnh được nêu làm ví dụ khác nữa, phương pháp nêu trên có thể được thực hiện bởi thiết bị tạo mã viđeo mà bao gồm bộ xử lý.

Theo một khía cạnh được nêu làm ví dụ khác nữa, phương pháp nêu trên có thể được thực hiện bởi thiết bị giải mã viđeo mà bao gồm bộ xử lý.

Theo một khía cạnh được nêu làm ví dụ khác nữa, các phương pháp này có thể được thực hiện dưới dạng các lệnh thực thi được bởi bộ xử lý và được lưu trữ trong vật ghi chương trình máy tính đọc được.

Các khía cạnh này và các khía cạnh khác được mô tả tiếp trong phần mô tả này.

Mô tả vắn tắt các hình vẽ

FIG. 1 là hình vẽ thể hiện ví dụ về điều kiện ràng buộc vùng trong dự báo vectơ chuyển động theo thời gian (temporal motion vector prediction - TMVP) và khối con TMVP.

FIG. 2 là hình vẽ thể hiện ví dụ về sơ đồ ước tính chuyển động phân cấp.

FIG. 3 là sơ đồ khối của ví dụ về nền tảng phần cứng được sử dụng để thực hiện các kỹ thuật được mô tả trong phần mô tả.

FIG. 4 là lưu đồ của phương pháp xử lý viđeo được nêu làm ví dụ.

FIG. 5 là hình vẽ thể hiện ví dụ về ảnh với 18 nhân 12 CTU độ sáng mà được phân chia thành 12 ô ảnh và 3 lát quét mành (mang tính thông tin).

FIG. 6 là hình vẽ thể hiện ví dụ về ảnh với 18 nhân 12 CTU độ sáng mà được phân chia thành 24 ô ảnh và 9 lát chữ nhật (mang tính thông tin).

FIG. 7 là hình vẽ thể hiện ví dụ về ảnh mà được phân chia thành 4 ô ảnh, 11 ô gạch, và 4 lát chữ nhật (mang tính thông tin).

FIG. 8 là sơ đồ khối thể hiện hệ thống xử lý viđeo được nêu làm ví dụ mà trong đó các kỹ thuật khác nhau được bộc lộ ở đây có thể được thực hiện.

FIG. 9 là sơ đồ khối mà thể hiện hệ thống mã hóa viđeo được nêu làm ví dụ.

FIG. 10 là sơ đồ khối thể hiện bộ tạo mã theo một số phương án theo sáng chế.

FIG. 11 là sơ đồ khối thể hiện bộ giải mã theo một số phương án theo sáng chế.

FIG. 12 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế.

FIG. 13 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác theo công nghệ theo sáng chế.

FIG. 14 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác theo công nghệ theo sáng chế.

FIG. 15 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác theo công nghệ theo sáng chế.

FIG. 16 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác theo công nghệ theo sáng chế.

FIG. 17 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác theo công nghệ theo sáng chế.

FIG. 18 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác theo công nghệ theo sáng chế.

FIG. 19 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo khác nữa theo công nghệ theo sáng chế.

Mô tả chi tiết sáng chế

Sáng chế đề xuất các kỹ thuật khác nhau mà có thể được sử dụng bởi bộ giải mã của các dòng bit hình ảnh hoặc viđeo để cải thiện chất lượng của các viđeo hoặc hình ảnh kỹ thuật số được nén hoặc được giải mã. Để ngắn gọn, thuật ngữ "viđeo" được sử dụng ở đây để bao gồm cả dãy của các ảnh (theo truyền thống được gọi là viđeo) lẫn các hình ảnh riêng lẻ. Hơn nữa, bộ tạo mã viđeo cũng có

thể thực hiện các kỹ thuật này trong suốt quá trình tạo mã nhằm tái tạo các khung được giải mã được sử dụng để tạo mã tiếp.

Các tiêu đề của các phần được sử dụng trong phần mô tả để làm cho dễ hiểu và không giới hạn các phương án và các kỹ thuật ở các phần tương ứng. Như vậy, các phương án từ một phần có thể được kết hợp với các phương án từ các phần khác.

1. Tổng quan

Sáng chế đề cập đến các công nghệ mã hóa viđeo. Cụ thể là, sáng chế đề cập tới mã hóa bảng màu có sử dụng sự biểu diễn dựa trên các màu cơ bản trong mã hóa viđeo. Sáng chế có thể được áp dụng cho chuẩn mã hóa viđeo hiện có như HEVC, hoặc chuẩn VVC (Versatile Video Coding) đang được hoàn thiện. Sáng chế cũng có thể áp dụng được cho các chuẩn mã hóa viđeo hoặc bộ mã hóa-giải mã viđeo tương lai.

2. Mở đầu

Các chuẩn mã hóa viđeo đã tiến hóa chủ yếu thông qua phát triển các chuẩn ITU-T và ISO/IEC được biết đến rộng rãi. ITU-T đã đề xuất H.261 và H.263, ISO/IEC đã đề xuất MPEG-1 và MPEG-4 Visual, và hai tổ chức này cùng nhau đề xuất các chuẩn H.262/MPEG-2 Video và H.264/MPEG-4 Mã hóa Viđeo Cao cấp (Advanced Video Coding - AVC) và H.265/HEVC [1,2]. Kể từ chuẩn H.262, các chuẩn mã hóa viđeo dựa trên cấu trúc mã hóa viđeo lai trong đó dự báo theo thời gian cộng với mã hóa chuyển đổi được sử dụng. Để nghiên cứu các công nghệ mã hóa viđeo tương lai sau HEVC, Joint Viđeo Exploration Team (JVET) đã được đồng thành lập bởi VCEG và MPEG trong năm 2015. Từ đó đến nay, nhiều phương pháp mới đã được chấp nhận bởi JVET và được đưa vào phần mềm tham chiếu tên là Joint Exploration Model (JEM). Vào tháng tư, năm 2018, Joint Video Expert Team (JVET) giữa VCEG (Q6/16) và ISO/IEC JTC1 SC29/WG11 (MPEG) đã được thành lập để làm việc với chuẩn VVC nhằm làm giảm 50% tốc độ bit so với HEVC.

2.1 Điều kiện ràng buộc vùng trong TMVP và khối con TMVP trong VVC

FIG. 1 thể hiện ví dụ về điều kiện ràng buộc vùng trong TMVP và khối con TMVP. Trong TMVP và khối con TMVP, điều kiện ràng buộc là MV theo thời gian chỉ có thể được tìm nạp từ CTU được sắp xếp cộng với cột của 4×4 khối như được thể hiện trên FIG. 1.

2.2 Ảnh con ví dụ

Theo một số phương án, các kỹ thuật mã hóa dựa trên ảnh con dựa trên cách tiếp cận tạo ô ảnh linh hoạt có thể được thực hiện. Tổng quan của các kỹ thuật mã hóa dựa trên ảnh con bao gồm các phần sau đây:

(1) Các ảnh có thể được chia thành các ảnh con.

(2) Chỉ báo về sự tồn tại của các ảnh con được chỉ báo trong SPS, cùng với thông tin bậc dãy khác của các ảnh con.

(3) Việc liệu ảnh con có được xử lý như ảnh trong quá trình giải mã (ngoại trừ các hoạt động lọc trong vòng) hay không có thể được điều khiển bởi dòng bit.

(4) Việc liệu lọc trong vòng ngang qua các đường biên ảnh con có không được phép hay không có thể được điều khiển bởi dòng bit đối với mỗi ảnh con. Các quá trình DBF, SAO, và ALF được cập nhật để điều khiển các hoạt động lọc trong vòng ngang qua các đường biên ảnh con.

(5) Nhằm đơn giản hóa, để làm điểm bắt đầu, chiều rộng, chiều cao, dịch vị ngang, và dịch vị dọc của ảnh con được báo tín hiệu trong các đơn vị của các mẫu độ sáng trong SPS. Các đường biên ảnh con bị ràng buộc là các đường biên lát.

(6) Việc xử lý ảnh con như ảnh trong quá trình giải mã (ngoại trừ các hoạt động lọc trong vòng) được chỉ ra bằng cách cập nhật không nhiều cú pháp coding_tree_unit(), và cập nhật tới các quá trình giải mã sau đây:

- Quá trình dẫn xuất đối với dự báo vectơ chuyển động độ sáng theo thời gian (cao cấp)
- Quá trình nội suy song tuyến tính mẫu độ sáng
- Quá trình lọc nội suy 8 nhánh mẫu độ sáng
- Quá trình nội suy mẫu sắc độ

(7) Các ID ảnh con được chỉ ra một cách rõ ràng trong SPS và được bao gồm trong các tiêu đề nhóm ô ảnh để cho phép trích ra các dãy ảnh con mà không cần thay đổi các đơn vị VCL NAL.

(8) Các tập ảnh con đầu ra (output sub-picture set - OSPS) được yêu cầu chỉ ra các điểm tương thích và trích ra định mức đối với các ảnh con và các tập của chúng.

2.3 Các ảnh con ví dụ trong mã hóa viđeo vạn năng

Cú pháp tập thông số dãy RBSP

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
...	
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
subpics_present_flag	u(1)
if(subpics_present_flag) {	
max_subpics_minus1	u(8)
subpic_grid_col_width_minus1	u(v)
subpic_grid_row_height_minus1	u(v)
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
subpic_grid_idx[i][j]	u(v)
for(i = 0; i <= NumSubPics; i++) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	
...	
}	

subpics_present_flag bằng 1 chỉ báo rằng các thông số ảnh con hiện nay đang có mặt trong cú pháp SPS RBSP. subpics_present_flag bằng 0 chỉ báo rằng các thông số ảnh con hiện nay không có mặt trong cú pháp SPS RBSP.

Ghi chú 2: Khi dòng bit là kết quả của quá trình trích ra dòng bit con và chỉ chứa tập con của các ảnh con của dòng bit đầu vào tới quá trình trích ra dòng bit con, thì có thể cần phải thiết lập giá trị của subpics_present_flag bằng 1 trong RBSP của các SPS.

max_subpics_minus1 plus 1 chỉ ra số lượng tối đa của các ảnh con mà có thể có mặt trong CVS. max_subpics_minus1 sẽ nằm trong khoảng từ 0 tới 254. Trị số 255 được bảo lưu để sử dụng trong tương lai bởi ITU-T | ISO/IEC.

subpic_grid_col_width_minus1 plus 1 chỉ ra chiều rộng của mỗi phần tử của lưới bộ nhận dạng ảnh con trong các đơn vị bằng 4 mẫu. Chiều dài của phần tử cú pháp là Ceil(Log2(pic_width_max_in_luma_samples / 4)) bit.

Biến số NumSubPicGridCols được rút ra như sau:

NumSubPicGridCols = (

pic_width_max_in_luma_samples + subpic_grid_col_width_minus 1 * 4 + 3) /

(subpic_grid_col_width_minus1 * 4 + 4) (7-5)

subpic_grid_row_height_minus1 plus 1 chỉ ra chiều cao của mỗi phần tử của lưới bộ nhận dạng ảnh con trong các đơn vị bằng 4 mẫu. Chiều dài của phần tử cú pháp bằng Ceil(Log2(pic_height_max_in_luma_samples / 4)) bit.

Biến số NumSubPicGridRows được rút ra như sau:

NumSubPicGridRows = (

pic_height_max_in_luma_samples + subpic_grid_row_height_min us1 * 4 + 3)/

(subpic_grid_row_height_minus1 * 4 + 4) (7-6)

subpic_grid_idx[i][j] chỉ ra chỉ số ảnh con của vị trí lưới (i, j). Chiều dài của phần tử cú pháp bằng Ceil(Log2(max_subpics_minus1 + 1)) bit.

Các biến số SubPicTop[subpic_grid_idx[i][j]], SubPicLeft[subpic_grid_idx[i][j]], SubPicWidth[subpic_grid_idx [i][j]], SubPicHeight[subpic_grid_idx[i][j]], và NumSubPics được rút ra như sau:

```
NumSubPics = 0
for(i = 0; i. < NumSubPicGridRows; i++) {
for(j = 0; j < NumSubPicGridCols; j++) {
    if (i = = 0)
    SubPicTop[subpic_grid_idx[i][j]] = 0
    else if(subpic_grid_idx[i][j] != subpic_grid_idx[i – 1][j]) {
            SubPicTop[subpic_grid_idx[i][j]] = i

SubPicHeight[subpic_grid_idx[i – 1][j]] =
```

```
i − SubPicTop[subpic_grid_idx[i − 1][j]]
                }
                if (j = = 0)
                SubPicLeft[subpic_grid_idx[i][j]] = 0
                else if (subpic_grid_idx[i][j] != subpic_grid_idx[i][j − 1]) {
                SubPicLeft[subpic_grid_idx[i][j]] = j
        SubPicWidth[subpic_grid_idx[i][j]] =
j − SubPicLeft[subpic_grid_idx[i][j − 1]]
        }
                if (i = = NumSubPicGridRows − 1)
SubPicHeight[subpic_grid_idx[i][j]] =
i − SubPicTop[subpic_grid_idx[i − 1][j]] + 1

        if (j = = NumSubPicGridRows − 1)
SubPicWidth[subpic_grid_idx[i][j]] =
j − SubPicLeft[subpic_grid_idx[i][j − 1]] + 1
                if(subpic_grid_idx[i][j] > NumSubPics)
                NumSubPics = subpic_grid_idx[i][j]
        }
}
```

subpic_treated_as_pic_flag[i] bằng 1 chỉ ra rằng ảnh con thứ i của mỗi ảnh được mã hóa trong CVS được xử lý như ảnh trong quá trình giải mã ngoại trừ các hoạt động lọc trong vòng. subpic_treated_as_pic_flag[i] bằng 0 chỉ ra rằng ảnh con thứ i của mỗi ảnh được mã hóa trong CVS không được xử lý như ảnh trong quá trình giải mã ngoại trừ các hoạt động lọc trong vòng. Khi không có mặt, giá trị của subpic_treated_as_pic_flag[i] được suy ra bằng 0.

loop_filter_across_subpic_enabled_flag[i] bằng 1 chỉ ra rằng các hoạt động lọc trong vòng có thể được thực hiện ngang qua các đường biên của ảnh con thứ i trong mỗi ảnh được mã hóa trong CVS. loop_filter_across_subpic_enabled_flag[i] bằng 0 chỉ ra rằng các hoạt động lọc trong vòng không được thực hiện ngang qua các đường biên của ảnh con thứ i

trong mỗi ảnh được mã hóa trong CVS. Khi không có mặt, giá trị của loop_filter_across_subpic_enabled_pic_flag[i] được suy ra bằng 1.

Sự tương thích dòng bit yêu cầu rằng các điều kiện ràng buộc sau đây được áp dụng:

–Đối với hai ảnh con bất kỳ subpicA và subpicB, khi chỉ số của subpicA nhỏ hơn chỉ số của subpicB, đơn vị NAL được mã hóa bất kỳ của subPicA sẽ tiếp theo đơn vị NAL được mã hóa bất kỳ của subPicB trong thứ tự giải mã.

–Các hình dạng của các ảnh con sẽ sao cho mỗi ảnh con, khi được giải mã, sẽ có toàn bộ biên trái và toàn bộ biên trên của nó bao gồm các đường biên ảnh hoặc bao gồm các đường biên của các ảnh con được giải mã trước đó.

Danh sách CtbToSubPicIdx[ctbAddrRs] đối với ctbAddrRs nằm trong khoảng từ 0 tới PicSizeInCtbsY − 1, bao gồm các giá trị biên, chỉ ra sự biến đổi từ địa chỉ CTB trong quét mành ảnh thành chỉ số ảnh con, được rút ra như sau:

```
for(ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++) {
posX = ctbAddrRs % PicWidthInCtbsY * CtbSizeY
posY = ctbAddrRs / PicWidthInCtbsY * CtbSizeY
CtbToSubPicIdx[ctbAddrRs] = −1
for(i = 0; CtbToSubPicIdx[ctbAddrRs] < 0 && i < NumSubPics;
i++) {
if((posX >=
SubPicLeft[i] * (subpic_grid_col_width_minus1 + 1) * 4) &&
(posX < (SubPicLeft[i] + SubPicWidth[i]) *

(subpic_grid_col_width_minus1 + 1) * 4) &&
(posY >= SubPicTop[i] *

(subpic_grid_row_height_minus1 + 1) * 4) &&
(posY < (SubPicTop[i] + SubPicHeight[i]) *
(subpic_grid_row_height_minus1 + 1) * 4))
CtbToSubPicIdx[ctbAddrRs] = i
```

```
    }
  }
```

num_bricks_in_slice_minus1, khi có mặt, chỉ ra số lượng các ô gạch trong lát trừ 1. Giá trị của num_bricks_in_slice_minus1 sẽ nằm trong khoảng từ 0 tới NumBricksInPic − 1, bao gồm các giá trị biên. Khi rect_slice_flag là bằng 0 và single_brick_per_slice_flag là bằng 1, thì giá trị của num_bricks_in_slice_minus1 được suy ra bằng 0. Khi single_brick_per_slice_flag là bằng 1, thì giá trị của num_bricks_in_slice_minus1 được suy ra bằng 0.

Biến số NumBricksInCurrSlice, mà chỉ ra số lượng các ô gạch trong lát hiện tại, và SliceBrickIdx[i], mà chỉ ra chỉ số ô gạch của ô gạch thứ i trong lát hiện tại, được rút ra như sau:

```
if(rect_slice_flag) {
  sliceIdx = 0
  while(slice_address != slice_id[sliceIdx])
      sliceIdx++
NumBricksInCurrSlice = NumBricksInSlice[sliceIdx]
  brickIdx = TopLeftBrickIdx[sliceIdx]
  for(bIdx = 0; brickIdx <= BottomRightBrickIdx[sliceIdx];
brickIdx++)    (7-92)
  if(BricksToSliceMap[brickIdx] == sliceIdx)
      SliceBrickIdx[bIdx++] = brickIdx
} else {
  NumBricksInCurrSlice = num_bricks_in_slice_minus1 + 1
  SliceBrickIdx[0] = slice_address
  for(i = 1; i < NumBricksInCurrSlice; i++)
      SliceBrickIdx[i] = SliceBrickIdx[i − 1] + 1
}
```

Các biến số SubPicIdx, SubPicLeftBoundaryPos, SubPicTopBoundaryPos, SubPicRightBoundaryPos, và SubPicBotBoundaryPos được rút ra như sau:

SubPicIdx =

CtbToSubPicIdx[CtbAddrBsToRs[FirstCtbAddrBs[SliceBrickIdx[0]]]]

if(subpic_treated_as_pic_flag[SubPicIdx]) {

SubPicLeftBoundaryPos =

SubPicLeft[SubPicIdx] * (subpic_grid_col_width_minus1 + 1) * 4

SubPicRightBoundaryPos =

(SubPicLeft[SubPicIdx] + SubPicWidth[SubPicIdx]) *

(subpic_grid_col_width_minus1 + 1) * 4

SubPicTopBoundaryPos =

SubPicTop[SubPicIdx] * (subpic_grid_row_height_minus1 + 1)* 4

SubPicBotBoundaryPos =

(SubPicTop[SubPicIdx] + SubPicHeight[SubPicIdx]) *

(subpic_grid_row_height_minus1 + 1) * 4

}

...

Quá trình dẫn xuất để dự báo vectơ chuyển động độ sáng theo thời gian

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCb, yCb) của mẫu trên cùng bên trái của khối mã hóa độ sáng hiện tại so với mẫu độ sáng trên cùng bên trái của ảnh hiện tại,

– biến số cbWidth chỉ ra chiều rộng của khối mã hóa hiện tại trong các mẫu độ sáng,

– biến số cbHeight chỉ ra chiều cao của khối mã hóa hiện tại trong các mẫu độ sáng,

– chỉ số tham chiếu refIdxLX, với X bằng 0 hoặc 1.

Các đầu ra của quá trình này là:

– dự báo vectơ chuyển động mvLXCol có độ chính xác 1/16 mẫu,

– cờ khả dụng availableFlagLXCol.

Biến số currCb chỉ ra khối mã hóa độ sáng hiện tại tại vị trí độ sáng (xCb, yCb).

Các biến số mvLXCol và availableFlagLXCol được rút ra như sau:

– Nếu slice_temporal_mvp_enabled_flag là bằng 0 hoặc (cbWidth * cbHeight) nhỏ hơn hoặc bằng 32, cả hai thành phần của

mvLXCol được thiết lập bằng 0 và availableFlagLXCol được thiết lập bằng 0.

– Ngược lại (slice_temporal_mvp_enabled_flag là bằng 1), các bước được xếp thứ tự sau đây được áp dụng:

1. Vectơ chuyển động được sắp xếp dưới cùng bên phải và các vị trí mẫu biên dưới cùng và bên phải được rút ra như sau:

xColBr = xCb + cbWidth (8-421)

yColBr = yCb + cbHeight (8-422)

rightBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ?
SubPicRightBoundaryPos : pic_width_in_luma_samples − 1 (8-423)

botBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ?
SubPicBotBoundaryPos : pic_height_in_luma_samples − 1 (8-424)

– Nếu yCb >> CtbLog2SizeY là bằng yColBr >> CtbLog2SizeY, yColBr nhỏ hơn hoặc bằng botBoundaryPos và xColBr nhỏ hơn hoặc bằng rightBoundaryPos, thì các điều sau đây được áp dụng:

– Biến số colCb chỉ ra khối mã hóa độ sáng bao phủ vị trí được cải biến được cho bởi ((xColBr >> 3) << 3, (yColBr >> 3) << 3) bên trong ảnh được sắp xếp được chỉ ra bởi ColPic.

– Vị trí độ sáng (xColCb, yColCb) được thiết lập bằng mẫu trên cùng bên trái của khối mã hóa độ sáng được sắp xếp được chỉ ra bởi colCb so với mẫu độ sáng trên cùng bên trái của ảnh được sắp xếp được chỉ ra bởi ColPic.

– Quá trình dẫn xuất đối với các vectơ chuyển động được sắp xếp như được chỉ ra trong mục 8.5.2.12 được gọi ra với currCb, colCb, (xColCb, yColCb), refIdxLX và sbFlag được thiết lập bằng 0 là các đầu vào, và đầu ra được gán cho mvLXCol và availableFlagLXCol.

Ngược lại, cả hai thành phần của mvLXCol được thiết lập bằng 0 và availableFlagLXCol được thiết lập bằng 0.

...

Quá trình nội suy song tuyến tính mẫu độ sáng

Các đầu vào của quá trình này là:

–vị trí độ sáng trong các đơn vị mẫu đủ ($xInt_L$, $yInt_L$),

–vị trí độ sáng trong các đơn vị mẫu phân đoạn ($xFrac_L$, $yFrac_L$),

–mảng mẫu tham chiếu độ sáng $refPicLX_L$.

Đầu ra của quá trình này là giá trị mẫu độ sáng được dự báo $predSampleLX_L$

Các biến số shift1, shift2, shift3, shift4, offset1, offset2 và offset3 được rút ra như sau:

$shift1 = BitDepth_Y - 6$ (8-453)

$offset1 = 1 << (shift1 - 1)$ (8-454)

$shift2 = 4$ (8-455)

$offset2 = 1 << (shift2 - 1)$ (8-456)

$shift3 = 10 - BitDepth_Y$ (8-457)

$shift4 = BitDepth_Y - 10$ (8-458)

$offset4 = 1 << (shift4 - 1)$ (8-459)

Biến số picW được thiết lập bằng pic_width_in_luma_samples và biến số picH được thiết lập bằng pic_height_in_luma_samples.

Các hệ số lọc nội suy độ sáng $fb_L[p]$ đối với mỗi vị trí mẫu 1/16 p bằng $xFrac_L$ hoặc $yFrac_L$ được chỉ ra trong Bảng 8-10.

Các vị trí độ sáng trong các đơn vị mẫu đủ ($xInt_i$, $yInt_i$) được rút ra như sau đối với i = 0..1:

–Nếu subpic_treated_as_pic_flag[SubPicIdx] là bằng 1, các điều sau đây được áp dụng:

$xInt_i = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xInt_L + i)$ (8-460)

$yInt_i = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yInt_L + i)$ (8-461)

–Ngược lại (subpic_treated_as_pic_flag[SubPicIdx] là bằng 0), các điều sau đây được áp dụng:

$xInt_i$ = Clip3(0, picW − 1, sps_ref_wraparound_enabled_flag ?

ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW , ($xInt_L$ + i)) : (8-462)

$xInt_L$ + i)

$yInt_i$ = Clip3(0, picH − 1, $yInt_L$ + i) (8-463)

...

Quá trình dẫn xuất đối với các ứng viên hợp nhất theo thời gian dựa trên khối con

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCb, yCb) của mẫu trên cùng bên trái của khối mã hóa độ sáng hiện tại so với mẫu độ sáng trên cùng bên trái của ảnh hiện tại,

– biến số cbWidth chỉ ra chiều rộng của khối mã hóa hiện tại trong các mẫu độ sáng,

– biến số cbHeight chỉ ra chiều cao của khối mã hóa hiện tại trong các mẫu độ sáng.

– cờ khả dụng $availableFlagA_1$ của đơn vị mã hóa lân cận,

– chỉ số tham chiếu $refIdxLXA_1$ của đơn vị mã hóa lân cận với X bằng 0 hoặc 1,

– cờ sử dụng danh sách dự báo $predFlagLXA_1$ của đơn vị mã hóa lân cận với X bằng 0 hoặc 1,

– vectơ chuyển động có độ chính xác 1/16 mẫu $mvLXA_1$ của đơn vị mã hóa lân cận với X bằng 0 hoặc 1.

Các đầu ra của quá trình này là:

– cờ khả dụng availableFlagSbCol,

– số lượng các khối con mã hóa độ sáng theo hướng ngang numSbX và theo hướng dọc numSbY,

– các chỉ số tham chiếu refIdxL0SbCol và refIdxL1SbCol,

– các vectơ chuyển động độ sáng có độ chính xác 1/16 mẫu mvL0SbCol[xSbIdx][ySbIdx] và mvL1SbCol[xSbIdx][ySbIdx] với xSbIdx = 0..numSbX − 1, ySbIdx = 0 .. numSbY − 1,

– các cờ sử dụng danh sách dự báo predFlagL0SbCol[xSbIdx][ySbIdx] và predFlagL1SbCol[xSbIdx][ySbIdx] with xSbIdx = 0..numSbX − 1, ySbIdx = 0 .. numSbY − 1.

Cờ khả dụng availableFlagSbCol được rút ra như sau.

– Nếu một hoặc nhiều trong số các điều kiện sau đây là đúng, availableFlagSbCol được thiết lập bằng 0.

– slice_temporal_mvp_enabled_flag là bằng 0.

– sps_sbtmvp_enabled_flag là bằng 0.

– cbWidth nhỏ hơn 8.

– cbHeight nhỏ hơn 8.

– Ngược lại, các bước được xếp thứ tự sau đây được áp dụng:

1. Vị trí (xCtb, yCtb) của mẫu trên cùng bên trái của khối cây mã hóa độ sáng mà chứa khối mã hóa hiện tại và vị trí (xCtr, yCtr) của mẫu giữa dưới bên phải của khối mã hóa độ sáng hiện tại được rút ra như sau:

xCtb = (xCb >> CtuLog2Size) << CtuLog2Size (8-542)

yCtb = (yCb >> CtuLog2Size) << CtuLog2Size (8-543)

xCtr = xCb + (cbWidth / 2) (8-544)

yCtr = yCb + (cbHeight / 2) (8-545)

2. Vị trí độ sáng (xColCtrCb, yColCtrCb) được thiết lập bằng mẫu trên cùng bên trái của khối mã hóa độ sáng được sắp xếp bao phủ vị trí được cho bởi (xCtr, yCtr) bên trong ColPic so với mẫu độ sáng trên cùng bên trái của ảnh được sắp xếp được chỉ ra bởi ColPic.

3. Quá trình dẫn xuất đối với dữ liệu chuyển động cơ sở hợp nhất theo thời gian dựa trên khối con như được chỉ ra trong mục 8.5.5.4 được gọi ra với vị trí (xCtb, yCtb), vị trí (xColCtrCb, yColCtrCb), cờ khả dụng availableFlagA_1, và cờ sử dụng danh sách dự báo predFlagLXA_1, và chỉ số tham chiếu refIdxLXA_1, và vectơ chuyển động mvLXA_1, với X bằng 0 và 1 là các đầu vào và các vectơ chuyển động ctrMvLX, và các cờ sử

dụng danh sách dự báo ctrPredFlagLX của khối được sắp xếp, với X bằng 0 và 1, và vectơ chuyển động theo thời gian tempMv là các đầu ra.

4. Biến số availableFlagSbCol được rút ra như sau:

– Nếu cả ctrPredFlagL0 lẫn ctrPredFlagL1 đều bằng 0, availableFlagSbCol được thiết lập bằng 0.

– Ngược lại, availableFlagSbCol được thiết lập bằng 1.

Khi availableFlagSbCol là bằng 1, các điều sau đây được áp dụng:

– Các biến số numSbX, numSbY, sbWidth, sbHeight và refIdxLXSbCol được rút ra như sau:

numSbX = cbWidth >> 3 (8-546)

numSbY = cbHeight >> 3 (8-547)

sbWidth = cbWidth / numSbX (8-548)

sbHeight = cbHeight / numSbY (8-549)

refIdxLXSbCol = 0 (8-550)

– Đối với xSbIdx = 0..numSbX − 1 và ySbIdx = 0 .. numSbY − 1, các vectơ chuyển động mvLXSbCol[xSbIdx][ySbIdx] và các cờ sử dụng danh sách dự báo predFlagLXSbCol[xSbIdx][ySbIdx] được rút ra như sau:

– Vị trí độ sáng (xSb, ySb) chỉ ra mẫu trên cùng bên trái của khối con mã hóa hiện tại so với mẫu độ sáng trên cùng bên trái của ảnh hiện tại được rút ra như sau:

xSb = xCb + xSbIdx * sbWidth + sbWidth / 2 (8-551)

ySb = yCb + ySbIdx * sbHeight + sbHeight / 2 (8-552)

– Vị trí (xColSb, yColSb) của khối con được sắp xếp bên trong ColPic được rút ra như sau.

– Các điều sau đây được áp dụng:

yColSb = Clip3(yCtb, Min(CurPicHeightInSamplesY − 1, yCtb + (1 << CtbLog2SizeY) − 1), (8-553) ySb + (tempMv[1] >> 4))

– Nếu subpic_treated_as_pic_flag[SubPicIdx] là bằng 1, các điều sau đây được áp dụng:

xColSb = Clip3(xCtb, Min(SubPicRightBoundaryPos, xCtb + (1 << CtbLog2SizeY) + 3), (8-554) xSb + (tempMv[0] >> 4))

– Ngược lại (subpic_treated_as_pic_flag[SubPicIdx] là bằng 0), các điều sau đây được áp dụng:

xColSb = Clip3(xCtb, Min(CurPicWidthInSamplesY − 1, xCtb + (1 << CtbLog2SizeY) + 3), (8-555) xSb + (tempMv[0] >> 4))

...

Quá trình dẫn xuất đối với dữ liệu chuyển động cơ sở hợp nhất theo thời gian dựa trên khối con

Các đầu vào của quá trình này là:

– vị trí (xCtb, yCtb) của mẫu trên cùng bên trái của khối cây mã hóa độ sáng mà chứa khối mã hóa hiện tại,

– vị trí (xColCtrCb, yColCtrCb) của mẫu trên cùng bên trái của khối mã hóa độ sáng được sắp xếp mà bao phủ mẫu giữa dưới bên phải.

– cờ khả dụng availableFlagA_1 của đơn vị mã hóa lân cận,

– chỉ số tham chiếu refIdxLXA_1 của đơn vị mã hóa lân cận,

– cờ sử dụng danh sách dự báo predFlagLXA_1 của đơn vị mã hóa lân cận,

– vectơ chuyển động có độ chính xác 1/16 mẫu mvLXA_1 của đơn vị mã hóa lân cận.

Các đầu ra của quá trình này là:

– các vectơ chuyển động ctrMvL0 và ctrMvL1,

– các cờ sử dụng danh sách dự báo ctrPredFlagL0 và ctrPredFlagL1,

– vectơ chuyển động theo thời gian tempMv.

Biến số tempMv được thiết lập như sau:

tempMv[0] = 0 (8-558)

tempMv[1] = 0 (8-559)

Biến số currPic chỉ ra ảnh hiện tại.

Khi availableFlagA_1 là bằng TRUE, các điều sau đây được áp dụng:

– Nếu tất cả các điều kiện sau đây là đúng, tempMv được thiết lập bằng mvL0A_1:

– predFlagL0A_1 là bằng 1,

– DiffPicOrderCnt(ColPic, RefPicList[0][refIdxL0A_1]) là bằng 0,

– Ngược lại, nếu tất cả các điều kiện sau đây là đúng, tempMv được thiết lập bằng mvL1A_1:

– slice_type là bằng B,

– predFlagL1A_1 là bằng 1,

– DiffPicOrderCnt(ColPic, RefPicList[1][refIdxL1A_1]) là bằng 0.

Vị trí (xColCb, yColCb) của khối được sắp xếp bên trong ColPic được rút ra như sau.

– Các điều sau đây được áp dụng:

yColCb = Clip3(yCtb, Min(CurPicHeightInSamplesY − 1, yCtb + (1 << CtbLog2SizeY) − 1), (8-560) yColCtrCb + (tempMv[1] >> 4))

– Nếu subpic_treated_as_pic_flag[SubPicIdx] là bằng 1, các điều sau đây được áp dụng:

xColCb = Clip3(xCtb, Min(SubPicRightBoundaryPos, xCtb + (1 << CtbLog2SizeY) + 3), (8-561) xColCtrCb + (tempMv[0] >> 4))

– Ngược lại (subpic_treated_as_pic_flag[SubPicIdx] là bằng 0, các điều sau đây được áp dụng:

xColCb = Clip3(xCtb, Min(CurPicWidthInSamplesY − 1, xCtb + (1 << CtbL

og2SizeY) + 3), (8-562)

xColCtrCb + (tempMv[0] >> 4))

...

Quá trình lọc nội suy mẫu độ sáng

Các đầu vào của quá trình này là:

- vị trí độ sáng trong các đơn vị mẫu đủ ($xInt_L$, $yInt_L$),
- vị trí độ sáng trong các đơn vị mẫu phân đoạn ($xFrac_L$, $yFrac_L$),
- vị trí độ sáng trong các đơn vị mẫu đủ ($xSbInt_L$, $ySbInt_L$) chỉ ra mẫu trên cùng bên trái của khối giới hạn để đệm mẫu tham chiếu so với mẫu độ sáng trên cùng bên trái của ảnh tham chiếu,
- mảng mẫu tham chiếu độ sáng $refPicLX_L$,
- chỉ số lọc nội suy nửa mẫu hpelIfIdx,
- biến số sbWidth chỉ ra chiều rộng của khối con hiện tại,
- biến số sbHeight chỉ ra chiều cao của khối con hiện tại,
- vị trí độ sáng (xSb, ySb) chỉ ra mẫu trên cùng bên trái của khối con hiện tại so với mẫu độ sáng trên cùng bên trái của ảnh hiện tại,

Đầu ra của quá trình này là giá trị mẫu độ sáng được dự báo $predSampleLX_L$

Các biến số shift1, shift2 và shift3 được rút ra như sau:

- Biến số shift1 được thiết lập bằng Min(4, $BitDepth_Y$ − 8), biến số shift2 được thiết lập bằng 6 và biến số shift3 được thiết lập bằng Max(2, 14 − $BitDepth_Y$).
- Biến số picW được thiết lập bằng pic_width_in_luma_samples và biến số picH được thiết lập bằng pic_height_in_luma_samples.

Các hệ số lọc nội suy độ sáng $f_L[p]$ đối với mỗi vị trí mẫu 1/16 p bằng $xFrac_L$ hoặc $yFrac_L$ được rút ra như sau:

- If MotionModelIdc[xSb][ySb] là lớn hơn 0, và cả hai sbWidth và sbHeight đều bằng 4, các hệ số lọc nội suy độ sáng $f_L[p]$ được chỉ ra trong Bảng 8-12.
- Ngược lại, các hệ số lọc nội suy độ sáng $f_L[p]$ được chỉ ra trong Bảng 8-11 phụ thuộc vào hpelIfIdx.

Các vị trí độ sáng trong các đơn vị mẫu đủ ($xInt_i$, $yInt_i$) được rút ra như sau đối với i = 0..7:

– Nếu subpic_treated_as_pic_flag[SubPicIdx] là bằng 1, các điều sau đây được áp dụng:

$xInt_i$ = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, $xInt_L$ + i − 3) (8-771)

$yInt_i$ = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, $yInt_L$ + i − 3) (8-772)

– Ngược lại (subpic_treated_as_pic_flag[SubPicIdx] là bằng 0), các điều sau đây được áp dụng:

$xInt_i$ = Clip3(0, picW − 1, sps_ref_wraparound_enabled_flag ?

ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW , $xInt_L$ + i − 3) : (8-773)

$xInt_L$ + i − 3)

$yInt_i$ = Clip3(0, picH − 1, $yInt_L$ + i − 3) (8-774)

...

Quá trình nội suy mẫu sắc độ

Các đầu vào của quá trình này là:

–vị trí sắc độ trong các đơn vị mẫu đủ ($xInt_C$, $yInt_C$),

–vị trí sắc độ trong các đơn vị mẫu phân đoạn 1/32 ($xFrac_C$, $yFrac_C$),

–vị trí sắc độ trong các đơn vị mẫu đủ (xSbIntC, ySbIntC) chỉ ra mẫu trên cùng bên trái của khối giới hạn để đệm mẫu tham chiếu so với mẫu sắc độ trên cùng bên trái của ảnh tham chiếu,

– biến số sbWidth chỉ ra chiều rộng của khối con hiện tại,

– biến số sbHeight chỉ ra chiều cao của khối con hiện tại,

–mảng mẫu tham chiếu sắc độ $refPicLX_C$.

Đầu ra của quá trình này là giá trị mẫu sắc độ được dự báo $predSampleLX_C$

Các biến số shift1, shift2 và shift3 được rút ra như sau:

–Biến số shift1 được thiết lập bằng Min(4, $BitDepth_C$ − 8), biến số shift2 được thiết lập bằng 6 và biến số shift3 được thiết lập bằng Max(2, 14 − $BitDepth_C$).

–Biến số $picW_C$ được thiết lập bằng pic_width_in_luma_samples / SubWidthC và biến số $picH_C$ được thiết lập bằng pic_height_in_luma_samples / SubHeightC.

Các hệ số lọc nội suy sắc độ $f_C[p]$ đối với mỗi vị trí mẫu phân đoạn 1/32 p bằng $xFrac_C$ hoặc $yFrac_C$ được chỉ ra trong Bảng 8-13.

Biến số xOffset được thiết lập bằng (sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY) / SubWidthC.

Các vị trí sắc độ trong các đơn vị mẫu đủ ($xInt_i$, $yInt_i$) được rút ra như sau đối với i = 0..3:

– Nếu subpic_treated_as_pic_flag[SubPicIdx] là bằng 1, các điều sau đây được áp dụng:

$xInt_i$ = Clip3(SubPicLeftBoundaryPos / SubWidthC, SubPicRightBoundaryPos / SubWidthC, $xInt_L$ + i) (8-785)

$yInt_i$ = Clip3(SubPicTopBoundaryPos / SubHeightC, SubPicBotBoundaryPos / SubHeightC, $yInt_L$ + i) (8-786)

– Ngược lại (subpic_treated_as_pic_flag[SubPicIdx] là bằng 0), các điều sau đây được áp dụng:

$xInt_i$ = Clip3(0, $picW_C$ − 1, sps_ref_wraparound_enabled_flag ? ClipH(xOffset, $picW_C$, $xInt_C$ + i − 1) : $xInt_C$ + i − 1) (8-787)

$yInt_i$ = Clip3(0, $picH_C$ − 1, $yInt_C$ + i − 1) (8-788)

2.4 Ví dụ về bộ lọc thời gian dựa trên GOP chỉ tạo mã

Theo một số phương án, bộ lọc thời gian chỉ tạo mã có thể được thực hiện. Bước lọc được thực hiện ở phía bộ tạo mã như bước tiền xử lý. Các ảnh nguồn trước và sau ảnh được chọn để tạo mã được đọc và phương pháp bù chuyển động dựa trên khối so với ảnh được chọn được áp dụng vào các ảnh nguồn này. Các

mẫu trong ảnh được chọn được lọc theo thời gian bằng cách sử dụng các giá trị mẫu sau bù chuyển động.

Sức mạnh bộ lọc tổng thể được thiết lập phụ thuộc vào lớp con thời gian của ảnh được chọn cũng như QP. Chỉ các ảnh ở các lớp con thời gian 0 và 1 là được lọc và các ảnh của lớp 0 được lọc bởi bộ lọc mạnh hơn các ảnh của lớp 1. Sức mạnh bộ lọc theo mẫu được điều chỉnh phụ thuộc vào hiệu số giữa giá trị mẫu trong ảnh được chọn và các mẫu cùng vị trí trong các ảnh được bù chuyển động sao cho các hiệu số nhỏ giữa ảnh được bù chuyển động và ảnh được chọn được lọc mạnh hơn là các hiệu số lớn hơn.

Bộ lọc thời gian dựa trên GOP

Bộ lọc thời gian được đề xuất trực tiếp sau khi đọc ảnh và trước khi tạo mã. Dưới đây là các bước được mô tả chi tiết hơn.

Hoạt động 1: Các ảnh được đọc bởi bộ tạo mã

Hoạt động 2: Nếu ảnh là đủ thấp trong phân cấp mã hóa, nó được lọc trước khi tạo mã. Ngược lại ảnh được tạo mã mà không lọc. Các ảnh RA với POC % 8 == 0 được lọc cũng như các ảnh LD với POC % 4 == 0. Các ảnh AI không bao giờ được lọc.

Sức mạnh bộ lọc tổng thể, s_o, được thiết lập theo phương trình dưới đây đối với RA.

$$s_o(n) = \begin{cases} 1{,}5, & n \bmod 16 = 0 \\ 0{,}95, & n \bmod 16 \neq 0 \end{cases}$$

trong đó n là số lượng của các ảnh được đọc.

Đối với trường hợp LD, $s_o(n) = 0{,}95$ được sử dụng.

Hoạt động 3: Hai ảnh trước và/hoặc sau ảnh được chọn (sau đây được gọi là ảnh gốc) được đọc. Trong các trường hợp biên, ví dụ nếu đó là ảnh thứ nhất hoặc gần tới ảnh cuối cùng, thì chỉ các ảnh khả dụng được đọc.

Hoạt động 4: Chuyển động của các ảnh được đọc trước và sau, so với ảnh gốc được ước tính theo khối 8x8 ảnh.

Sơ đồ ước tính chuyển động phân cấp được sử dụng và các lớp L0, L1 và L2, được thể hiện trên FIG. 2. Các ảnh được lấy mẫu con được tạo ra bằng cách lấy trung bình mỗi khối 2x2 đối với tất cả các ảnh được đọc và ảnh gốc, ví dụ L1

trên FIG. 1. L2 được rút ra từ L1 bằng cách sử dụng cùng phương pháp lấy mẫu con.

FIG. 2 là hình vẽ thể hiện các ví dụ về các lớp khác nhau của ước tính chuyển động phân cấp. L0 là độ phân giải gốc. L1 là phiên bản được lấy mẫu con của L0. L2 là phiên bản được lấy mẫu con của L1.

Đầu tiên, việc ước tính chuyển động được thực hiện đối với mỗi khối 16x16 trong L2. Hiệu số bình phương được tính toán đối với mỗi vectơ chuyển động được chọn và vectơ chuyển động tương ứng với hiệu số nhỏ nhất được chọn. Khi đó vectơ chuyển động được chọn được sử dụng làm giá trị ban đầu khi ước tính chuyển động trong L1. Sau đó các bước giống như vậy được thực hiện để ước tính chuyển động trong L0. Ở bước cuối cùng, chuyển động điểm ảnh con được ước tính đối với mỗi khối 8x8 bằng cách sử dụng bộ lọc nội suy trên L0.

Bộ lọc nội suy 6 nhánh VTM có thể được sử dụng:

0:	0,	0,	64,	0,	0,	0
1:	1,	-3,	64,	4,	-2,	0
2:	1,	-6,	62,	9,	-3,	1
3:	2,	-8,	60,	14,	-5,	1
4:	2,	-9,	57,	19,	-7,	2
5:	3,	-10,	53,	24,	-8,	2
6:	3,	-11,	50,	29,	-9,	2
7:	3,	-11,	44,	35,	-10,	3
8:	1,	-7,	38,	38,	-7,	1
9:	3,	-10,	35,	44,	-11,	3
10:	2,	-9,	29,	50,	-11,	3
11:	2,	-8,	24,	53,	-10,	3
12:	2,	-7,	19,	57,	-9,	2
13:	1,	-5,	14,	60,	-8,	2
14:	1,	-3,	9,	62,	-6,	1
15:	0,	-2,	4,	64,	-3,	1

Hoạt động 5: Bù chuyển động được áp dụng vào các ảnh trước và sau ảnh gốc theo chuyển động so khớp nhất đối với mỗi khối, ví dụ, sao cho các tọa độ

mẫu của ảnh gốc trong mỗi khối có các tọa độ so khớp nhất trong các ảnh được tham chiếu.

Hoạt động 6: Các mẫu được xử lý từng mẫu một đối với các kênh độ sáng và sắc độ như được mô tả trong các bước sau đây.

Hoạt động 7: giá trị mẫu mới, I_n, được tính toán bằng cách sử dụng công thức sau đây.

$$I_n = \frac{I_o + \sum_{i=0}^{3} w_r(i,a) I_r(i)}{1 + \sum_{i=0}^{3} w_r(i,a)}$$

Trong đó I_o là giá trị mẫu của mẫu gốc, $I_r(i)$ là mật độ của mẫu tương ứng của ảnh được bù chuyển động i và $w_r(i,a)$ là trọng số của ảnh được bù chuyển động i khi số lượng của các ảnh khả dụng được bù chuyển động là a.

Trong kênh độ sáng, các trọng số, $w_r(i,a)$, được xác định như sau:

$$w_r(i,a) = s_l s_o(n) s_r(i,a) e^{-\frac{\Delta I(i)^2}{2\sigma_l(QP)^2}}$$

Trong đó

$$s_l = 0{,}4$$

$$s_r(i,2) = \begin{cases} 1{,}2, & i = 0 \\ 1{,}0, & i = 1 \end{cases}$$

$$s_r(i,4) = \begin{cases} 0{,}60, & i = 0 \\ 0{,}85, & \mathrm{i} = 1 \\ 0{,}85, & \mathrm{i} = 2 \\ 0{,}60, & i = 3 \end{cases}$$

Đối với tất cả các trường hợp khác của i, và a: $s_r(i,a) = 0{,}3$

$$\sigma_l(QP) = 3 * (QP - 10)$$

$$\Delta I(i) = I_r(i) - I_o$$

Đối với các kênh sắc độ, các trọng số, $w_r(i,a)$, được xác định như sau:

$$w_r(i,a) = s_c s_o(n) s_r(i,a) e^{-\frac{\Delta I(i)^2}{2\sigma_c{}^2}}$$

Trong đó $s_c = 0{,}55$ và $\sigma_c = 30$

Hoạt động 8: bộ lọc được áp dụng đối với mẫu hiện tại. giá trị mẫu thu được được lưu trữ riêng biệt.

Hoạt động 9: ảnh được lọc được tạo mã.

2.5 Ví dụ về các phân chia ảnh (các ô ảnh, các ô gạch, các lát)

Theo một số phương án, ảnh được chia thành một hoặc nhiều hàng ô ảnh và một hoặc nhiều cột ô ảnh. Ô ảnh là dãy của các CTU mà bao phủ vùng hình chữ nhật của ảnh.

Ô ảnh được chia thành một hoặc nhiều ô gạch, mỗi ô trong số đó bao gồm một số lượng các hàng CTU bên trong ô ảnh.

Ô ảnh mà không được phân chia thành nhiều ô gạch cũng được gọi là ô gạch. Tuy nhiên, ô gạch mà là tập con đúng của ô ảnh thì không được gọi là ô ảnh.

Lát chứa hoặc số lượng các ô ảnh của ảnh hoặc số lượng các ô gạch của ô ảnh.

Ảnh con chứa một hoặc nhiều lát mà cùng nhau bao phủ vùng hình chữ nhật của ảnh.

Hai chế độ của các lát được hỗ trợ, mà chính là chế độ lát quét mành và chế độ lát chữ nhật. Trong chế độ lát quét mành, lát chứa dãy của các ô ảnh trong quét mành ô ảnh của ảnh. Trong chế độ lát chữ nhật, lát chứa số lượng các ô gạch của ảnh mà cùng nhau tạo thành vùng hình chữ nhật của ảnh. Các ô gạch nằm bên trong lát chữ nhật theo thứ tự của quét mành ô gạch của lát.

FIG. 5 là hình vẽ thể hiện ví dụ về phân chia lát quét mành của ảnh, trong đó ảnh được chia thành 12 ô ảnh và 3 lát quét mành.

FIG. 6 là hình vẽ thể hiện ví dụ về phân chia lát chữ nhật của ảnh, trong đó ảnh được chia thành 24 ô ảnh (6 cột ô ảnh và 4 hàng ô ảnh) và 9 lát chữ nhật.

FIG. 7 là hình vẽ thể hiện ví dụ về ảnh được phân chia thành các ô ảnh, các ô gạch, và các lát chữ nhật, trong đó ảnh được chia thành 4 ô ảnh (2 cột ô ảnh và 2 hàng ô ảnh), 11 ô gạch (ô ảnh trên cùng bên trái chứa 1 ô gạch, ô ảnh trên cùng bên phải chứa 5 ô gạch, ô ảnh dưới cùng bên trái chứa 2 ô gạch, và ô ảnh dưới cùng bên phải chứa 3 ô gạch), và 4 lát chữ nhật.

Cú pháp tập thông số ảnh RBSP

pic_parameter_set_rbsp() {	**Descriptor**
...	
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
uniform_tile_spacing_flag	u(1)
if(uniform_tile_spacing_flag) {	
tile_cols_width_minus1	ue(v)

tile_rows_height_minus1	ue(v)
} else {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
brick_splitting_present_flag	u(1)
if(uniform_tile_spacing_flag && brick_splitting_present_flag)	
num_tiles_in_pic_minus1	ue(v)
for(i = 0; brick_splitting_present_flag && i <= num_tiles_in_pic_minus1 + 1; i++) {	
if(RowHeight[i] > 1)	
brick_split_flag[i]	u(1)
if(brick_split_flag[i]) {	
if(RowHeight[i] > 2)	
uniform_brick_spacing_flag[i]	u(1)
if(uniform_brick_spacing_flag[i])	
brick_height_minus1[i]	ue(v)
else {	
num_brick_rows_minus2[i]	ue(v)
for(j = 0; j <= num_brick_rows_minus2[i]; j++)	
brick_row_height_minus1[i][j]	ue(v)
}	
}	
}	
single_brick_per_slice_flag	u(1)
if(!single_brick_per_slice_flag)	
rect_slice_flag	u(1)
if(rect_slice_flag && !single_brick_per_slice_flag) {	
num_slices_in_pic_minus1	ue(v)
bottom_right_brick_idx_length_minus1	ue(v)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
bottom_right_brick_idx_delta[i]	u(v)
brick_idx_delta_sign_flag[i]	u(1)
}	
}	
loop_filter_across_bricks_enabled_flag	u(1)
if(loop_filter_across_bricks_enabled_flag)	
loop_filter_across_slices_enabled_flag	u(1)
}	
if(rect_slice_flag) {	
signalled_slice_id_flag	u(1)
if(signalled_slice_id_flag) {	
signalled_slice_id_length_minus1	ue(v)
for(i = 0; i <= num_slices_in_pic_minus1; i++)	
slice_id[i]	u(v)
}	
}	
...	

slice_header() {	Descriptor
slice_pic_parameter_set_id	ue(v)
if(rect_slice_flag || NumBricksInPic > 1)	
slice_address	u(v)
if(!rect_slice_flag && !single_brick_per_slice_flag)	
num_bricks_in_slice_minus1	ue(v)
non_reference_picture_flag	u(1)
slice_type	ue(v)
...	

single_tile_in_pic_flag bằng 1 chỉ ra rằng chỉ có một ô ảnh trong mỗi ảnh tham chiếu tới PPS. single_tile_in_pic_flag bằng 0 chỉ ra rằng có nhiều hơn một ô ảnh trong mỗi ảnh tham chiếu tới PPS.

Ghi chú: Khi không có sự tách tiếp ô gạch bên trong ô ảnh, thì toàn bộ ô ảnh được gọi là ô gạch. Khi ảnh chỉ chứa một ô ảnh mà không tách tiếp ô gạch, thì được gọi là ô gạch đơn.

Sự tương thích dòng bit yêu cầu rằng giá trị của single_tile_in_pic_flag sẽ giống nhau đối với tất cả các PPS mà được tham chiếu tới bởi các ảnh được mã hóa bên trong CVS.

uniform_tile_spacing_flag bằng 1 chỉ ra rằng các đường biên cột ô ảnh và tương tự các đường biên hàng ô ảnh được phân bố đều ngang qua ảnh và được báo tín hiệu bằng cách sử dụng các phần tử cú pháp tile_cols_width_minus1 và tile_rows_height_minus1. uniform_tile_spacing_flag bằng 0 chỉ ra rằng các đường biên cột ô ảnh và tương tự các đường biên hàng ô ảnh có thể có hoặc có thể không được phân bố đều ngang qua ảnh và được báo tín hiệu bằng cách sử dụng các phần tử cú pháp num_tile_columns_minus1 và num_tile_rows_minus1 và danh sách của các cặp phần tử cú pháp tile_column_width_minus1[i] và tile_row_height_minus1[i]. Khi không có mặt, giá trị của uniform_tile_spacing_flag được suy ra bằng 1.

tile_cols_width_minus1 plus 1 chỉ ra chiều rộng của các cột ô ảnh ngoại trừ cột ô ảnh ngoài cùng bên phải của ảnh trong các đơn vị của các CTB khi uniform_tile_spacing_flag là bằng 1. Giá trị của tile_cols_width_minus1 sẽ trong khoảng từ 0 tới PicWidthInCtbsY − 1, bao gồm các giá trị biên. Khi không

có mặt, giá trị của tile_cols_width_minus1 được suy ra bằng PicWidthInCtbsY − 1.

tile_rows_height_minus1 plus 1 chỉ ra chiều cao của các hàng ô ảnh ngoại trừ hàng ô ảnh dưới cùng của ảnh trong các đơn vị của các CTB khi uniform_tile_spacing_flag là bằng 1. Giá trị của tile_rows_height_minus1 sẽ trong khoảng từ 0 tới PicHeightInCtbsY − 1, bao gồm các giá trị biên. Khi không có mặt, giá trị của tile_rows_height_minus1 được suy ra bằng PicHeightInCtbsY − 1.

num_tile_columns_minus1 plus 1 chỉ ra số lượng của các cột ô ảnh mà phân chia ảnh khi uniform_tile_spacing_flag là bằng 0. Giá trị của num_tile_columns_minus1 sẽ trong khoảng từ 0 tới PicWidthInCtbsY − 1, bao gồm các giá trị biên. Nếu single_tile_in_pic_flag là bằng 1, giá trị của num_tile_columns_minus1 được suy ra bằng 0. Ngược lại, khi uniform_tile_spacing_flag là bằng 1, giá trị của num_tile_columns_minus1 được suy ra như được chỉ ra trong mục 6.5.1.

num_tile_rows_minus1 plus 1 chỉ ra số lượng của các hàng ô ảnh mà phân chia ảnh khi uniform_tile_spacing_flag là bằng 0. Giá trị của num_tile_rows_minus1 sẽ trong khoảng từ 0 tới PicHeightInCtbsY − 1, bao gồm các giá trị biên. Nếu single_tile_in_pic_flag là bằng 1, giá trị của num_tile_rows_minus1 được suy ra bằng 0. Ngược lại, khi uniform_tile_spacing_flag là bằng 1, giá trị của num_tile_rows_minus1 được suy ra như được chỉ ra trong mục 6.5.1.

Biến số NumTilesInPic được thiết lập bằng
(num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1).

Khi single_tile_in_pic_flag là bằng 0, NumTilesInPic sẽ lớn hơn 1.

tile_column_width_minus1[i] plus 1 chỉ ra chiều rộng của cột ô ảnh thứ i trong các đơn vị của các CTB.

tile_row_height_minus1[i] plus 1 chỉ ra chiều cao của hàng ô ảnh thứ i trong các đơn vị của các CTB.

brick_splitting_present_flag bằng 1 chỉ ra rằng một hoặc nhiều ô ảnh của các ảnh tham chiếu tới PPS có thể được chia thành hai hoặc nhiều ô gạch.

brick_splitting_present_flag bằng 0 chỉ ra rằng không ô ảnh nào của các ảnh tham chiếu tới PPS được chia thành hai hoặc nhiều ô gạch.

num_tiles_in_pic_minus1 plus 1 chỉ ra số lượng của các ô ảnh trong mỗi ảnh tham chiếu tới PPS. Giá trị của num_tiles_in_pic_minus1 sẽ bằng NumTilesInPic – 1. Khi không có mặt, giá trị của num_tiles_in_pic_minus1 được suy ra bằng NumTilesInPic − 1.

brick_split_flag[i] bằng 1 chỉ ra rằng ô ảnh thứ i được chia thành hai hoặc nhiều ô gạch. brick_split_flag[i] bằng 0 chỉ ra rằng ô ảnh thứ i không được chia thành hai hoặc nhiều ô gạch. Khi không có mặt, giá trị của brick_split_flag[i] được suy ra bằng 0. Theo một số phương án, sự phụ thuộc phân tách PPS vào SPS được đưa vào bằng cách bổ sung điều kiện cú pháp "if(RowHeight[i] > 1)" (ví dụ, tương tự đối với uniform_brick_spacing_flag[i]).

uniform_brick_spacing_flag[i] bằng 1 chỉ ra rằng các đường biên ô gạch ngang được phân bố đều ngang qua ô ảnh thứ i và được báo tín hiệu bằng cách sử dụng phần tử cú pháp brick_height_minus1[i]. uniform_brick_spacing_flag[i] bằng 0 chỉ ra rằng các đường biên ô gạch ngang có thể có hoặc có thể không được phân bố đều ngang qua ô ảnh thứ i và được báo tín hiệu bằng cách sử dụng phần tử cú pháp num_brick_rows_minus2[i] và danh sách của các phần tử cú pháp brick_row_height_minus1[i][j]. Khi không có mặt, giá trị của uniform_brick_spacing_flag[i] được suy ra bằng 1.

brick_height_minus1[i] plus 1 chỉ ra chiều cao của các hàng ô gạch ngoại trừ ô gạch dưới cùng trong ô ảnh thứ i trong các đơn vị của các CTB khi uniform_brick_spacing_flag[i] là bằng 1. Khi có mặt, giá trị của brick_height_minus1 sẽ trong khoảng từ 0 tới RowHeight[i] − 2, bao gồm các giá trị biên. Khi không có mặt, giá trị của brick_height_minus1[i] được suy ra bằng RowHeight[i] − 1.

num_brick_rows_minus2[i] plus 2 chỉ ra số lượng các ô gạch mà phân chia ô ảnh thứ i khi uniform_brick_spacing_flag[i] là bằng 0. Khi có mặt, giá trị của num_brick_rows_minus2[i] sẽ trong khoảng từ 0 tới RowHeight[i] − 2, bao gồm các giá trị biên. Nếu brick_split_flag[i] là bằng 0, giá trị của num_brick_rows_minus2[i] được suy ra bằng −1. Ngược lại, khi

uniform_brick_spacing_flag[i] là bằng 1, giá trị của num_brick_rows_minus2[i] được suy ra như được chỉ ra trong mục 6.5.1.

brick_row_height_minus1[i][j] plus 1 chỉ ra chiều cao của ô gạch thứ j trong ô ảnh thứ i trong các đơn vị của các CTB khi uniform_tile_spacing_flag là bằng 0.

Các biến số sau đây được rút ra, và, khi uniform_tile_spacing_flag là bằng 1, các giá trị của num_tile_columns_minus1 và num_tile_rows_minus1 được suy ra, và, đối với mỗi i nằm trong khoảng từ 0 tới NumTilesInPic − 1, bao gồm các giá trị biên, khi uniform_brick_spacing_flag[i] là bằng 1, giá trị của num_brick_rows_minus2[i] được suy ra, bằng cách gọi ra quá trình biến đổi quét mành CTB và quét ô gạch như được chỉ ra trong mục 6.5.1:

– danh sách RowHeight[j] với j nằm trong khoảng từ 0 tới num_tile_rows_minus1, bao gồm các giá trị biên, chỉ ra chiều cao của hàng ô ảnh thứ j trong các đơn vị của các CTB,

– danh sách CtbAddrRsToBs[ctbAddrRs] đối với ctbAddrRs nằm trong khoảng từ 0 tới PicSizeInCtbsY − 1, bao gồm các giá trị biên, chỉ ra biến đổi từ địa chỉ CTB trong quét mành CTB của ảnh thành địa chỉ CTB trong quét ô gạch,

– danh sách CtbAddrBsToRs[ctbAddrBs] đối với ctbAddrBs nằm trong khoảng từ 0 tới PicSizeInCtbsY − 1, bao gồm các giá trị biên, chỉ ra biến đổi từ địa chỉ CTB trong quét ô gạch thành địa chỉ CTB trong quét mành CTB của ảnh,

– danh sách BrickId[ctbAddrBs] đối với ctbAddrBs nằm trong khoảng từ 0 tới PicSizeInCtbsY − 1, bao gồm các giá trị biên, chỉ ra biến đổi từ địa chỉ CTB trong quét ô gạch thành ID ô gạch,

– danh sách NumCtusInBrick[brickIdx] đối với brickIdx nằm trong khoảng từ 0 tới NumBricksInPic − 1, bao gồm các giá trị biên, chỉ ra biến đổi từ chỉ số ô gạch thành số lượng của các CTU trong ô gạch,

– danh sách FirstCtbAddrBs[brickIdx] đối với brickIdx nằm trong khoảng từ 0 tới NumBricksInPic − 1, bao gồm các giá trị biên, chỉ ra biến

đổi từ ID ô gạch thành địa chỉ CTB trong quét ô gạch của CTB thứ nhất trong ô gạch.

single_brick_per_slice_flag bằng 1 chỉ ra rằng mỗi lát mà tham chiếu tới PPS này bao gồm một ô gạch. single_brick_per_slice_flag bằng 0 chỉ ra rằng lát mà tham chiếu tới PPS này có thể bao gồm nhiều hơn một ô gạch. Khi không có mặt, giá trị của single_brick_per_slice_flag được suy ra bằng 1.

rect_slice_flag bằng 0 chỉ ra rằng các ô gạch bên trong mỗi lát theo thứ tự quét mành và thông tin lát không được báo tín hiệu trong PPS. rect_slice_flag bằng 1 chỉ ra rằng các ô gạch bên trong mỗi lát bao phủ vùng hình chữ nhật của ảnh và thông tin lát được báo tín hiệu trong PPS. Khi brick_splitting_present_flag là bằng 1, giá trị của rect_slice_flag sẽ bằng 1. Khi không có mặt, rect_slice_flag được suy ra bằng 1.

num_slices_in_pic_minus1 plus 1 chỉ ra số lượng của các lát trong mỗi ảnh tham chiếu tới PPS. Giá trị của num_slices_in_pic_minus1 sẽ trong khoảng từ 0 tới NumBricksInPic − 1, bao gồm các giá trị biên. Khi không có mặt và single_brick_per_slice_flag là bằng 1, giá trị của num_slices_in_pic_minus1 được suy ra bằng NumBricksInPic − 1.

bottom_right_brick_idx_length_minus1 plus 1 chỉ ra số lượng của các bit được sử dụng để biểu diễn phần tử cú pháp bottom_right_brick_idx_delta[i]. Giá trị của bottom_right_brick_idx_length_minus1 sẽ trong khoảng từ 0 tới Ceil(Log2(NumBricksInPic)) − 1, bao gồm các giá trị biên.

bottom_right_brick_idx_delta[i] khi i là lớn hơn 0 chỉ ra hiệu số giữa chỉ số ô gạch của ô gạch nằm ở góc dưới cùng bên phải của lát thứ i và chỉ số ô gạch của góc dưới cùng bên phải của lát thứ (i – 1). bottom_right_brick_idx_delta[0] chỉ ra chỉ số ô gạch của góc dưới cùng bên phải của lát thứ 0. Khi single_brick_per_slice_flag là bằng 1, giá trị của bottom_right_brick_idx_delta[i] được suy ra bằng 1. Giá trị của BottomRightBrickIdx[num_slices_in_pic_minus1] được suy ra bằng NumBricksInPic – 1. Chiều dài của phần tử cú pháp bottom_right_brick_idx_delta[i] là bottom_right_brick_idx_length_minus1 + 1 bit.

brick_idx_delta_sign_flag[i] bằng 1 chỉ báo dấu dương cho bottom_right_brick_idx_delta[i]. sign_bottom_right_brick_idx_delta[i] bằng 0 chỉ báo dấu âm cho bottom_right_brick_idx_delta[i].

Sự tương thích dòng bit yêu cầu rằng lát phải bao gồm hoặc là một số lượng các ô ảnh hoàn chỉnh hoặc chỉ dãy liên tiếp của các ô gạch hoàn chỉnh của một ô ảnh.

Biến số TopLeftBrickIdx[i], BottomRightBrickIdx[i], NumBricksInSlice[i] và BricksToSliceMap[j], mà chỉ ra chỉ số ô gạch của ô gạch nằm ở góc trên cùng bên trái của lát thứ i, chỉ số ô gạch của ô gạch nằm ở góc dưới cùng bên phải của lát thứ i, số lượng các ô gạch trong lát thứ i và ánh xạ của các ô gạch tới các lát, được rút ra như sau:

```
for(j = 0; i == 0 && j < NumBricksInPic; j++)
  BricksToSliceMap[j] = −1
NumBricksInSlice[i] = 0
BottomRightBrickIdx[i] = bottom_right_brick_idx_delta[i]] +((i == 0) ?
0 :
        (brick_idx_delta_sign_flag[i] ? BottomRightBrickIdx[i − 1] :
−BottomRightBrickIdx[i−1])
for(j = BottomRightBrickIdx[i]; j >= 0; j− −) {
    if(BrickColBd[j] <= BrickColBd[BottomRightBrickIdx[i]] && (7-
43)
        BrickRowBd[j] <= BrickRowBd[BottomRightBrickIdx[i]]
&&                                  BricksToSliceMap[j] == −1) {
      TopLeftBrickIdx[i] = j
      NumBricksInSlice[i]++
      BricksToSliceMap[j] = i
    }
}
```

Các ngữ nghĩa tiêu đề lát tổng quát

Khi có mặt, giá trị của mỗi phần tử trong số của các phần tử cú pháp tiêu đề lát slice_pic_parameter_set_id, non_reference_picture_flag, colour_plane_id,

slice_pic_order_cnt_lsb, recovery_poc_cnt, no_output_of_prior_pics_flag, pic_output_flag, và slice_temporal_mvp_enabled_flag sẽ giống như trong tất cả các tiêu đề lát của ảnh được mã hóa.

Biến số CuQpDeltaVal, chỉ ra hiệu số giữa thông số lượng tử hóa độ sáng đối với đơn vị mã hóa chứa cu_qp_delta_abs và dự báo của nó, được thiết lập bằng 0. Các biến số $CuQpOffset_{Cb}$, $CuQpOffset_{Cr}$, và $CuQpOffset_{CbCr}$, chỉ ra các giá trị cần được sử dụng khi xác định các giá trị tương ứng của các thông số lượng tử hóa Qp'_{Cb}, Qp'_{Cr}, và Qp'_{CbCr} đối với đơn vị mã hóa chứa cu_chroma_qp_offset_flag, tất cả đều được được thiết lập bằng 0.

slice_pic_parameter_set_id chỉ ra giá trị của pps_pic_parameter_set_id đối với PPS đang được sử dụng. giá trị của slice_pic_parameter_set_id sẽ trong khoảng từ 0 tới 63, bao gồm các giá trị biên.

Sự tương thích dòng bit yêu cầu rằng giá trị của TemporalId của ảnh hiện tại sẽ lớn hơn hoặc bằng giá trị của TemporalId của PPS mà có pps_pic_parameter_set_id bằng slice_pic_parameter_set_id.

slice_address chỉ ra địa chỉ lát của lát. Khi không có mặt, giá trị của slice_address được suy ra bằng 0.

Nếu rect_slice_flag là bằng 0, các điều sau đây được áp dụng:

- Địa chỉ lát là ID ô gạch như được chỉ ra bởi Phương trình (7-59).
- Chiều dài của slice_address là bằng Ceil(Log2 (NumBricksInPic)) bit.
- Giá trị của slice_address sẽ trong khoảng từ 0 tới NumBricksInPic − 1, bao gồm các giá trị biên.

Ngược lại (rect_slice_flag là bằng 1), các điều sau đây được áp dụng:

- Địa chỉ lát là ID lát của lát.
- Chiều dài của slice_address là bằng signalled_slice_id_length_minus1 + 1 bit.
- Nếu signalled_slice_id_flag là bằng 0, giá trị của slice_address sẽ trong khoảng từ 0 tới num_slices_in_pic_minus1, bao gồm các giá trị biên. Ngược lại, giá trị của slice_address sẽ trong khoảng từ 0 tới $2^{(signalled\_slice\_id\_length\_minus1 + 1)} - 1$, bao gồm các giá trị biên.

Sự tương thích dòng bit yêu cầu rằng các điều kiện ràng buộc sau đây áp dụng:

– Giá trị của slice_address sẽ không bằng giá trị của slice_address của đơn vị NAL lát được mã hóa bất kỳ nào khác của cùng ảnh được mã hóa.

– Khi rect_slice_flag là bằng 0, các lát của ảnh sẽ theo thứ tự tăng của các giá trị slice_address của chúng.

– Các hình dạng của các lát của ảnh sẽ sao cho mỗi ô gạch, khi được giải mã, sẽ có toàn bộ biên trái và toàn bộ biên trên của nó bao gồm biên ảnh hoặc bao gồm các đường biên của (các) ô gạch được giải mã trước đó.

num_bricks_in_slice_minus1, khi có mặt, chỉ ra số lượng các ô gạch trong lát trừ 1. Giá trị của num_bricks_in_slice_minus1 sẽ trong khoảng từ 0 tới NumBricksInPic − 1, bao gồm các giá trị biên. Khi rect_slice_flag là bằng 0 và single_brick_per_slice_flag là bằng 1, giá trị của num_bricks_in_slice_minus1 được suy ra bằng 0. Khi single_brick_per_slice_flag là bằng 1, giá trị của num_bricks_in_slice_minus1 được suy ra bằng 0.

Biến số NumBricksInCurrSlice, mà chỉ ra số lượng các ô gạch trong lát hiện tại, và SliceBrickIdx[i], mà chỉ ra chỉ số ô gạch của ô gạch thứ i trong lát hiện tại, được rút ra như sau:

```
if(rect_slice_flag) {
 sliceIdx = 0
 while(slice_address != slice_id[sliceIdx])
   sliceIdx++
 NumBricksInCurrSlice = NumBricksInSlice[sliceIdx]
 brickIdx = TopLeftBrickIdx[sliceIdx]
 for(bIdx = 0; brickIdx <= BottomRightBrickIdx[sliceIdx];
brickIdx++) (7-92)
   if(BricksToSliceMap[brickIdx] == sliceIdx)
     SliceBrickIdx[bIdx++] = brickIdx
} else {
```

NumBricksInCurrSlice = num_bricks_in_slice_minus1 + 1

SliceBrickIdx[0] = slice_address

for(i = 1; i < NumBricksInCurrSlice; i++)

SliceBrickIdx[i] = SliceBrickIdx[i − 1] + 1

}

Các biến số SubPicIdx, SubPicLeftBoundaryPos, SubPicTopBoundaryPos, SubPicRightBoundaryPos, và SubPicBotBoundaryPos được rút ra như sau:

SubPicIdx =

CtbToSubPicIdx[CtbAddrBsToRs[FirstCtbAddrBs[SliceBrickIdx[0]]]]

if(subpic_treated_as_pic_flag[SubPicIdx]) {

SubPicLeftBoundaryPos =

SubPicLeft[SubPicIdx] * (subpic_grid_col_width_minus1 + 1) * 4

SubPicRightBoundaryPos =

(SubPicLeft[SubPicIdx] + SubPicWidth[SubPicIdx]) *

(subpic_grid_col_width_minus1 + 1) * 4 (7-93)

SubPicTopBoundaryPos =

SubPicTop[SubPicIdx] * (subpic_grid_row_height_minus1 + 1)* 4

SubPicBotBoundaryPos =

(SubPicTop[SubPicIdx] + SubPicHeight[SubPicIdx]) *

(subpic_grid_row_height_minus1 + 1) * 4

}

2.6 Ví dụ về cú pháp và các ngữ nghĩa

Cú pháp tập thông số dãy RBSP

seq_parameter_set_rbsp() {	**Mô tả**
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_5bits	u(5)
protệp tin_tier_level(sps_max_sub_layers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	u(4)
chroma_format_idc	u(2)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
ref_pic_resampling_enabled_flag	u(1)
sps_seq_parameter_set_id	ue(v)

chroma_format_idc	ue(v)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
sps_log2_ctu_size_minus5	u(2)
subpics_present_flag	*u(1)*
sps_num_subpics_minus1	*u(8)*
for(i = 0; i <= sps_num_subpics_minus1; i++) {	
subpic_ctu_top_left_x[i]	*u(v)*
subpic_ctu_top_left_y[i]	*u(v)*
subpic_width_minus1[i]	*u(v)*
subpic_height_minus1[i]	*u(v)*
subpic_treated_as_pic_flag[i]	*u(1)*
loop_filter_across_subpic_enabled_flag[i]	*u(1)*
}	
}	
sps_subpic_id_present_flag	*u(1)*
if(sps_subpics_id_present_flag) {	
sps_subpic_id_signalling_present_flag	*u(1)*
if(sps_subpics_id_signalling_present_flag) {	
sps_subpic_id_len_minus1	*ue(v)*
for(i = 0; i <= sps_num_subpics_minus1; i++)	
sps_subpic_id[i]	*u(v)*
}	
}	
bit_depth_minus8	ue(v)
min_qp_prime_ts_minus4	ue(v)
sps_weighted_pred_flag	u(1)
sps_weighted_bipred_flag	u(1)
log2_max_pic_order_cnt_lsb_minus4	u(4)
if(sps_max_sub_layers_minus1 > 0)	
sps_sub_layer_ordering_info_present_flag	u(1)
for(i = (sps_sub_layer_ordering_info_present_flag ? 0 : sps_max_sub_layers_minus1); i <= sps_max_sub_layers_minus1; i++) {	
sps_max_dec_pic_buffering_minus1[i]	ue(v)
sps_max_num_reorder_pics[i]	ue(v)
sps_max_latency_increase_plus1[i]	ue(v)
}	
long_term_ref_pics_flag	u(1)
inter_layer_ref_pics_present_flag	u(1)
sps_idr_rpl_present_flag	u(1)
rpl1_same_as_rpl0_flag	u(1)
for(i = 0; i < !rpl1_same_as_rpl0_flag ? 2 : 1; i++) {	
num_ref_pic_lists_in_sps[i]	ue(v)
for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)	
ref_pic_list_struct(i, j)	
}	
if(ChromaArrayType != 0)	
qtbtt_dual_tree_intra_flag	u(1)
log2_min_luma_coding_block_size_minus2	ue(v)
partition_constraints_override_enabled_flag	u(1)

sps_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
sps_log2_diff_min_qt_min_cb_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(sps_max_mtt_hierarchy_depth_inter_slice != 0) {	
sps_log2_diff_max_bt_min_qt_inter_slice	ue(v)
sps_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
sps_max_luma_transform_size_64_flag	u(1)
sps_joint_cbcr_enabled_flag	u(1)
if(ChromaArrayType != 0) {	
same_qp_table_for_chroma	u(1)
numQpTables = same_qp_table_for_chroma ? 1 : (sps_joint_cbcr_enabled_flag ? 3 : 2)	
for(i = 0; i < numQpTables; i++) {	
qp_table_start_minus26[i]	se(v)
num_points_in_qp_table_minus1[i]	ue(v)
for(j = 0; j <= num_points_in_qp_table_minus1[i]; j++) {	
delta_qp_in_val_minus1[i][j]	ue(v)
delta_qp_diff_val[i][j]	ue(v)
}	
}	
}	
sps_sao_enabled_flag	u(1)
sps_alf_enabled_flag	u(1)
sps_transform_skip_enabled_flag	u(1)
if(sps_transform_skip_enabled_flag)	
sps_bdpcm_enabled_flag	u(1)
if(sps_bdpcm_enabled_flag && chroma_format_idc == 3)	
sps_bdpcm_chroma_enabled_flag	u(1)
sps_ref_wraparound_enabled_flag	u(1)
if(sps_ref_wraparound_enabled_flag)	
sps_ref_wraparound_offset_minus1	ue(v)
sps_temporal_mvp_enabled_flag	u(1)
if(sps_temporal_mvp_enabled_flag)	
sps_sbtmvp_enabled_flag	u(1)
sps_amvr_enabled_flag	u(1)
sps_bdof_enabled_flag	u(1)
if(sps_bdof_enabled_flag)	
sps_bdof_pic_present_flag	u(1)
sps_smvd_enabled_flag	u(1)

sps_dmvr_enabled_flag	u(1)
if(sps_dmvr_enabled_flag)	
sps_dmvr_pic_present_flag	u(1)
sps_mmvd_enabled_flag	u(1)
sps_isp_enabled_flag	u(1)
sps_mrl_enabled_flag	u(1)
sps_mip_enabled_flag	u(1)
if(ChromaArrayType != 0)	
sps_cclm_enabled_flag	u(1)
if(sps_cclm_enabled_flag && chroma_format_idc == 1)	
sps_cclm_colocated_chroma_flag	u(1)
sps_mts_enabled_flag	u(1)
if(sps_mts_enabled_flag) {	
sps_explicit_mts_intra_enabled_flag	u(1)
sps_explicit_mts_inter_enabled_flag	u(1)
}	
sps_sbt_enabled_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag) {	
sps_affine_type_flag	u(1)
sps_affine_amvr_enabled_flag	u(1)
sps_affine_prof_enabled_flag	u(1)
if(sps_affine_prof_enabled_flag)	
sps_prof_pic_present_flag	u(1)
}	
if(chroma_format_idc == 3) {	
sps_palette_enabled_flag	u(1)
sps_act_enabled_flag	u(1)
}	
sps_bcw_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
sps_ciip_enabled_flag	u(1)
if(sps_mmvd_enabled_flag)	
sps_fpel_mmvd_enabled_flag	u(1)
sps_triangle_enabled_flag	u(1)
sps_lmcs_enabled_flag	u(1)
sps_lfnst_enabled_flag	u(1)
sps_ladf_enabled_flag	u(1)
if(sps_ladf_enabled_flag) {	
sps_num_ladf_intervals_minus2	u(2)
sps_ladf_lowest_interval_qp_offset	se(v)
for(i = 0; i < sps_num_ladf_intervals_minus2 + 1; i++) {	
sps_ladf_qp_offset[i]	se(v)
sps_ladf_delta_threshold_minus1[i]	ue(v)
}	
}	
sps_scaling_list_enabled_flag	u(1)
sps_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
sps_num_ver_virtual_boundaries	u(2)
for(i = 0; i < sps_num_ver_virtual_boundaries; i++)	
sps_virtual_boundaries_pos_x[i]	u(13)
sps_num_hor_virtual_boundaries	u(2)

for(i = 0; i < sps_num_hor_virtual_boundaries; i++)	
sps_virtual_boundaries_pos_y[i]	u(13)
}	
general_hrd_parameters_present_flag	u(1)
if(general_hrd_parameters_present_flag) {	
num_units_in_tick	u(32)
time_scale	u(32)
sub_layer_cpb_parameters_present_flag	u(1)
if(sub_layer_cpb_parameters_present_flag)	
general_hrd_parameters(0, sps_max_sub_layers_minus1)	
Else	
general_hrd_parameters(sps_max_sub_layers_minus1, sps_max_sub_layers_minus1)	
}	
vui_parameters_present_flag	u(1)
if(vui_parameters_present_flag)	
vui_parameters()	
sps_extension_flag	u(1)
if(sps_extension_flag)	
while(more_rbsp_data())	
sps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

Cú pháp tập thông số ảnh RBSP

pic_parameter_set_rbsp() {	**Mô tả**
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	u(4)
pps_seq_parameter_set_id	ue(v)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
scaling_window_flag	u(1)
if(scaling_window_flag) {	
scaling_win_left_offset	ue(v)
scaling_win_right_offset	ue(v)
scaling_win_top_offset	ue(v)
scaling_win_bottom_offset	ue(v)
}	
output_flag_present_flag	u(1)
mixed_nalu_types_in_pic_flag	u(1)
pps_subpic_id_signalling_present_flag	***u(1)***
if(pps_subpics_id_signalling_present_flag) {	
pps_num_subpics_minus1	***ue(v)***

pps_subpic_id_len_minus1	*ue(v)*
for(i = 0; i <= pps_num_subpic_minus1; i++)	
pps_subpic_id[i]	*u(v)*
}	
no_pic_partition_flag	u(1)
if(!no_pic_partition_flag) {	
pps_log2_ctu_size_minus5	u(2)
num_exp_tile_columns_minus1	ue(v)
num_exp_tile_rows_minus1	ue(v)
for(i = 0; i <= num_exp_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i <= num_exp_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
rect_slice_flag	u(1)
if(rect_slice_flag)	
single_slice_per_subpic_flag	u(1)
if(rect_slice_flag && !single_slice_per_subpic_flag) {	
num_slices_in_pic_minus1	ue(v)
tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
slice_width_in_tiles_minus1[i]	ue(v)
slice_height_in_tiles_minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0) {	
num_slices_in_tile_minus1[i]	ue(v)
numSlicesInTileMinus1 = num_slices_in_tile_minus1[i]	
for(j = 0; j < numSlicesInTileMinus1; j++)	
slice_height_in_ctu_minus1[i++]	ue(v)
}	
if(tile_idx_delta_present_flag && i < num_slices_in_pic_minus1)	
tile_idx_delta[i]	se(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
loop_filter_across_slices_enabled_flag	u(1)
}	
entropy_coding_sync_enabled_flag	u(1)
if(!no_pic_partition_flag || entropy_coding_sync_enabled_flag)	
entry_point_offsets_present_flag	u(1)
cabac_init_present_flag	u(1)
for(i = 0; i < 2; i++)	
num_ref_idx_default_active_minus1[i]	ue(v)
rpl1_idx_present_flag	u(1)
init_qp_minus26	se(v)
if(sps_transform_skip_enabled_flag)	
log2_transform_skip_max_size_minus2	*ue(v)*
cu_qp_delta_enabled_flag	u(1)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbcr_qp_offset_present_flag	u(1)
if(pps_joint_cbcr_qp_offset_present_flag)	
pps_joint_cbcr_qp_offset_giá trị	se(v)

pps_slice_chroma_qp_offsets_present_flag	u(1)
cu_chroma_qp_offset_enabled_flag	u(1)
if(cu_chroma_qp_offset_enabled_flag) {	
chroma_qp_offset_list_len_minus1	ue(v)
for(i = 0; i <= chroma_qp_offset_list_len_minus1; i++) {	
cb_qp_offset_list[i]	se(v)
cr_qp_offset_list[i]	se(v)
if(pps_joint_cbcr_qp_offset_present_flag)	
joint_cbcr_qp_offset_list[i]	se(v)
}	
}	
pps_weighted_pred_flag	u(1)
pps_weighted_bipred_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
deblocking_filter_override_enabled_flag	u(1)
pps_deblocking_filter_disabled_flag	u(1)
if(!pps_deblocking_filter_disabled_flag) {	
pps_beta_offset_div2	se(v)
pps_tc_offset_div2	se(v)
}	
}	
constant_slice_header_params_enabled_flag	u(1)
if(constant_slice_header_params_enabled_flag) {	
pps_dep_quant_enabled_idc	u(2)
for(i = 0; i < 2; i++)	
pps_ref_pic_list_sps_idc[i]	u(2)
pps_mvd_l1_zero_idc	u(2)
pps_collocated_from_l0_idc	u(2)
pps_six_minus_max_num_merge_cand_plus1	ue(v)
pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1	ue(v)
}	
picture_header_extension_present_flag	u(1)
slice_header_extension_present_flag	u(1)
pps_extension_flag	u(1)
if(pps_extension_flag)	
while(more_rbsp_data())	
pps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

Cú pháp tiêu đề ảnh RBSP

picture_header_rbsp() {	**Mô tả**
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	

ph_subpic_id_signalling_present_flag	***u(1)***
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	***ue(v)***
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	***u(v)***
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	ue(v)
if(partition_constraints_override_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_log2_diff_min_qt_min_cb_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)

pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	
pic_cu_qp_delta_subdiv_intra_slice	ue(v)
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(cu_chroma_qp_offset_enabled_flag) {	
pic_cu_chroma_qp_offset_subdiv_intra_slice	ue(v)
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
}	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbcr_enabled_flag)	
pic_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	

pic_sao_chroma_enabled_flag	u(1)
}	
}	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
if(!pps_dep_quant_enabled_flag)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	
pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	

rbsp_trailing_bits()	
}	

subpics_present_flag bằng 1 chỉ báo rằng các thông số ảnh con hiện nay đang có mặt trong cú pháp SPS RBSP. subpics_present_flag bằng 0 chỉ báo rằng các thông số ảnh con hiện nay không có mặt trong cú pháp SPS RBSP.

Ghi chú 2: Khi dòng bit là kết quả của quá trình trích ra dòng bit con và chỉ chứa tập con của các ảnh con của dòng bit đầu vào tới quá trình trích ra dòng bit con, có thể cần phải thiết lập giá trị của subpics_present_flag bằng 1 trong RBSP của các SPS.

sps_num_subpics_minus1 plus 1 chỉ ra số lượng của các ảnh con. sps_num_subpics_minus1 sẽ trong khoảng từ 0 tới 254. Khi không có mặt, giá trị của sps_num_subpics_minus1 được suy ra bằng 0.

subpic_ctu_top_left_x[i] chỉ ra vị trí ngang của CTU trên cùng bên trái của ảnh con thứ i trong đơn vị của CtbSizeY. Chiều dài của phần tử cú pháp bằng ***Ceil(Log2(pic_width_max_in_luma_samples / CtbSizeY))*** bit. Khi không có mặt, giá trị của subpic_ctu_top_left_x[i] được suy ra bằng 0.

subpic_ctu_top_left_y[i] chỉ ra vị trí dọc của CTU trên cùng bên trái của ảnh con thứ i trong đơn vị của CtbSizeY. Chiều dài của phần tử cú pháp bằng ***Ceil(Log2(pic_height_max_in_luma_samples / CtbSizeY))*** bit. Khi không có mặt, giá trị của subpic_ctu_top_left_y[i] được suy ra bằng 0.

subpic_width_minus1[i] plus 1 chỉ ra chiều rộng của ảnh con thứ i trong các đơn vị của CtbSizeY. Chiều dài của phần tử cú pháp bằng Ceil(Log2(pic_width_max_in_luma_samples / CtbSizeY)) bit. Khi không có mặt, giá trị của subpic_width_minus1[i] được suy ra bằng ***Ceil(pic_width_max_in_luma_samples / CtbSizeY)*** − 1.

subpic_ height_minus1[i] plus 1 chỉ ra chiều cao của ảnh con thứ i trong các đơn vị của CtbSizeY. Chiều dài của phần tử cú pháp bằng Ceil(Log2(pic_height_max_in_luma_samples / CtbSizeY)) bit. Khi không có mặt, giá trị của subpic_height_minus1[i] được suy ra bằng ***Ceil(pic_height_max_in_luma_samples / CtbSizeY)*** − 1.

subpic_treated_as_pic_flag[i] bằng 1 chỉ ra rằng ảnh con thứ i của mỗi ảnh được mã hóa trong CVS được xử lý như ảnh trong quá trình giải mã ngoại trừ các hoạt động lọc trong vòng. subpic_treated_as_pic_flag[i] bằng 0 chỉ ra rằng ảnh con thứ i của mỗi ảnh được mã hóa trong CVS không được xử lý như ảnh trong quá trình giải mã ngoại trừ các hoạt động lọc trong vòng. Khi không có mặt, giá trị của subpic_treated_as_pic_flag[i] được suy ra bằng 0.

loop_filter_across_subpic_enabled_flag[i] bằng 1 chỉ ra rằng các hoạt động lọc trong vòng có thể được thực hiện ngang qua các đường biên của ảnh con thứ i trong mỗi ảnh được mã hóa trong CVS. loop_filter_across_subpic_enabled_flag[i] bằng 0 chỉ ra rằng các hoạt động lọc trong vòng không được thực hiện ngang qua các đường biên của ảnh con thứ i trong mỗi ảnh được mã hóa trong CVS. Khi không có mặt, giá trị của loop_filter_across_subpic_enabled_pic_flag[i] được suy ra bằng 1.

Sự tương thích dòng bit yêu cầu rằng các điều kiện ràng buộc sau đây áp dụng:

– Đối với hai ảnh con bất kỳ subpicA và subpicB, khi chỉ số của subpicA nhỏ hơn chỉ số của subpicB, thì đơn vị NAL được mã hóa bất kỳ của subPicA sẽ tiếp theo đơn vị NAL được mã hóa bất kỳ của subPicB theo thứ tự giải mã.

– Các hình dạng của các ảnh con sẽ sao cho mỗi ảnh con, khi được giải mã, sẽ có toàn bộ biên trái và toàn bộ biên trên của nó bao gồm các đường biên ảnh hoặc bao gồm các đường biên của các ảnh con được giải mã trước đó.

sps_subpic_id_present_flag bằng 1 chỉ ra rằng ánh xạ Id ảnh con có mặt trong SPS. sps_subpic_id_present_flag bằng 0 chỉ ra rằng ánh xạ Id ảnh con không có mặt trong SPS.

sps_subpic_id_signalling_present_flag bằng 1 chỉ ra rằng ánh xạ Id ảnh con được báo tín hiệu trong SPS. sps_subpic_id_signalling_present_flag bằng 0 chỉ ra rằng ánh xạ Id ảnh con không được báo tín hiệu trong SPS. Khi không có mặt, giá trị của sps_subpic_id_signalling_present_flag được suy ra bằng 0.

sps_subpic_id_len_minus1 plus 1 chỉ ra số lượng của các bit được sử dụng để biểu diễn phần tử cú pháp sps_subpic_id[i]. Giá trị của sps_subpic_id_len_minus1 sẽ trong khoảng từ 0 tới 15, bao gồm các giá trị biên.

sps_subpic_id[i] chỉ ra rằng Id ảnh con của ảnh con thứ i. Chiều dài của phần tử cú pháp sps_subpic_id[i] bằng sps_subpic_id_len_minus1 + 1 bit. Khi không có mặt, và khi sps_subpic_id_present_flag bằng 0, giá trị của sps_subpic_id[i] được suy ra bằng i, đối với mỗi i trong khoảng từ 0 tới sps_num_subpics_minus1, bao gồm các giá trị biên.

ph_pic_parameter_set_id chỉ ra giá trị của pps_pic_parameter_set_id đối với PPS đang được sử dụng. Giá trị của ph_pic_parameter_set_id sẽ trong khoảng từ 0 tới 63, bao gồm các giá trị biên.

Sự tương thích dòng bit yêu cầu rằng giá trị của TemporalId của tiêu đề ảnh sẽ lớn hơn hoặc bằng giá trị của TemporalId của PPS mà có pps_pic_parameter_set_id bằng ph_pic_parameter_set_id.

ph_subpic_id_signalling_present_flag bằng 1 chỉ ra rằng ánh xạ Id ảnh con được báo tín hiệu trong tiêu đề ảnh. ph_subpic_id_signalling_present_flag bằng 0 chỉ ra rằng ánh xạ Id ảnh con không được báo tín hiệu trong tiêu đề ảnh.

ph_subpic_id_len_minus1 plus 1 chỉ ra số lượng của các bit được sử dụng để biểu diễn phần tử cú pháp ph_subpic_id[i]. Giá trị của pic_subpic_id_len_minus1 sẽ trong khoảng từ 0 tới 15, bao gồm các giá trị biên.

Sự tương thích dòng bit yêu cầu rằng giá trị của ph_subpic_id_len_minus1 sẽ giống nhau đối với tất cả các tiêu đề ảnh mà được tham chiếu tới bởi các ảnh được mã hóa trong CVS.

ph_subpic_id[i] chỉ ra rằng Id ảnh con của ảnh con thứ i. Chiều dài của phần tử cú pháp ph_subpic_id[i] bằng ph_subpic_id_len_minus1 + 1 bit.

Danh sách SubpicIdList[i] được rút ra như sau:

```
for(i = 0; i <= sps_num_subpics_minus1; i++)
        SubpicIdList[i] = sps_subpic_id_present_flag ? (7-39)
        (sps_subpic_id_signalling_present_flag ? sps_subpic_id[i] :
        (ph_subpic_id_signalling_present_flag ? ph_subpic_id[i] :
pps_subpic_id[i])) : i
```

Quá trình lọc giải khối

Tổng quát

Các đầu vào của quá trình này là ảnh được tái tạo trước khi giải khối, tức là, mảng $recPicture_L$ và, khi ChromaArrayType không bằng 0, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$.

Các đầu ra của quá trình này là ảnh được tái tạo được cải biến sau giải khối, tức là, mảng $recPicture_L$ và, khi ChromaArrayType không bằng 0, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$.

Các mép dọc trong ảnh được lọc trước tiên. Sau đó các mép ngang trong ảnh được lọc với các mẫu được cải biến bởi quá trình lọc mép dọc làm đầu vào. Các mép dọc và ngang trong các CTB của mỗi CTU được xử lý riêng rẽ trên cơ sở đơn vị mã hóa. Các mép dọc của các khối mã hóa trong đơn vị mã hóa được lọc bắt đầu bằng mép ở phía tay trái của các khối mã hóa tiếp tục qua các mép hướng phía tay phải của các khối mã hóa theo thứ tự hình học của chúng. Các mép ngang của các khối mã hóa trong đơn vị mã hóa được lọc bắt đầu bằng mép ở trên cùng của các khối mã hóa tiếp tục qua các mép hướng về đáy của các khối mã hóa theo thứ tự hình học của chúng.

Ghi chú: Mặc dù quá trình lọc được chỉ ra trên cơ sở ảnh trong phần mô tả này, quá trình lọc có thể được thực hiện trên cơ sở đơn vị mã hóa với kết quả tương đương, miễn là bộ giải mã tính đến thứ tự phụ thuộc xử lý một cách chính xác để đưa ra các giá trị đầu ra giống nhau.

Quá trình lọc giải khối được áp dụng cho tất cả các mép khối con mã hóa và các mép khối chuyển đổi của ảnh, ngoại trừ các mép thuộc các kiểu sau đây:

- Các mép mà ở biên của ảnh,
- ***Các mép mà trùng với các đường biên của ảnh con mà đối với nó loop_filter_across_subpic_enabled_flag[SubPicIdx] là bằng 0,***
- Các mép mà trùng với các đường biên ảo của ảnh khi pps_loop_filter_across_virtual_boundaries_disabled_flag là bằng 1,
- Các mép mà trùng với các đường biên ô ảnh khi loop_filter_across_tiles_enabled_flag là bằng 0,

– Các mép mà trùng với các đường biên lát khi loop_filter_across_slices_enabled_flag là bằng 0,

– Các mép mà trùng với các đường biên trên hoặc trái của các lát với slice_deblocking_filter_disabled_flag bằng 1,

– Các mép bên trong các lát với slice_deblocking_filter_disabled_flag bằng 1,

– Các mép mà không tương ứng với các đường biên lưới mẫu 4x4 của thành phần độ sáng,

– Các mép mà không tương ứng với các đường biên lưới mẫu 8x8 của thành phần sắc độ,

– Các mép bên trong thành phần độ sáng mà đối với chúng cả hai bên của mép có intra_bdpcm_luma_flag bằng 1,

– Các mép bên trong các thành phần sắc độ mà đối với chúng cả hai bên của mép có intra_bdpcm_chroma_flag bằng 1,

– Các mép của các khối con sắc độ mà không phải là các mép của đơn vị chuyển đổi liên kết.

...

Quá trình lọc giải khối đối với một hướng

Các đầu vào của quá trình này là:

– biến số treeType chỉ ra liệu thành phần độ sáng (DUAL_TREE_LUMA) hoặc các thành phần sắc độ (DUAL_TREE_CHROMA) có đang được xử lý hay không,

– khi treeType là bằng DUAL_TREE_LUMA, ảnh được tái tạo trước khi giải khối, tức là, mảng $recPicture_L$,

– khi ChromaArrayType không bằng 0 và treeType là bằng DUAL_TREE_CHROMA, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$,

– biến số edgeType chỉ ra liệu mép dọc (EDGE_VER) hoặc ngang (EDGE_HOR) có được lọc hay không.

Các đầu ra của quá trình này là ảnh được tái tạo được cải biến sau giải khối, tức là:

– khi treeType là bằng DUAL_TREE_LUMA, mảng $recPicture_L$,

– khi ChromaArrayType không bằng 0 và treeType là bằng DUAL_TREE_CHROMA, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$.

Các biến số firstCompIdx và lastCompIdx được rút ra như sau:

firstCompIdx = (treeType = = DUAL_TREE_CHROMA) ? 1 : 0 (8-1010)

lastCompIdx = (treeType = = DUAL_TREE_LUMA || ChromaArrayType = = 0) ? 0 : 2 (8-1011)

Đối với mỗi đơn vị mã hóa và mỗi khối mã hóa trên thành phần màu của đơn vị mã hóa được chỉ báo bởi chỉ số thành phần màu cIdx nằm trong khoảng từ firstCompIdx tới lastCompIdx, bao gồm các giá trị biên, với chiều rộng khối mã hóa nCbW, chiều cao khối mã hóa nCbH và vị trí của mẫu trên cùng bên trái của khối mã hóa (xCb, yCb), khi cIdx là bằng 0, hoặc khi cIdx không bằng 0 và edgeType là bằng EDGE_VER và xCb % 8 là bằng 0, hoặc khi cIdx không bằng 0 và edgeType là bằng EDGE_HOR và yCb % 8 là bằng 0, các mép được lọc bởi các bước được xếp thứ tự sau đây:

1. Biến số filterEdgeFlag được rút ra như sau:

– Nếu edgeType là bằng EDGE_VER và một hoặc nhiều trong số các điều kiện sau đây là đúng, filterEdgeFlag được thiết lập bằng 0:

– Biên trái của khối mã hóa hiện tại là biên trái của ảnh.

– ***Biên trái của khối mã hóa hiện tại là biên trái hoặc phải của ảnh con và loop_filter_across_subpic_enabled_flag[SubPicIdx] là bằng 0.***

– Biên trái của khối mã hóa hiện tại là biên trái của ô ảnh và loop_filter_across_tiles_enabled_flag là bằng 0.

– Biên trái của khối mã hóa hiện tại là biên trái của lát và loop_filter_across_slices_enabled_flag là bằng 0.

– Biên trái của khối mã hóa hiện tại là một trong số các đường biên ảo dọc của ảnh và VirtualBoundariesDisabledFlag là bằng 1.

– Ngược lại, nếu edgeType là bằng EDGE_HOR và một hoặc nhiều trong số các điều kiện sau đây là đúng, biến số filterEdgeFlag được thiết lập bằng 0:

– Biên trên của khối mã hóa độ sáng hiện tại là biên trên của ảnh.

– ***Bên trên của khối mã hóa hiện tại là biên trên cùng hoặc biên đáy của ảnh con và loop_filter_across_subpic_enabled_flag[SubPicIdx] là bằng 0.***

– Biên trên cùng của khối mã hóa hiện tại là biên trên cùng của ô ảnh và loop_filter_across_tiles_enabled_flag là bằng 0.

– Biên trên cùng của khối mã hóa hiện tại là biên trên cùng của lát và loop_filter_across_slices_enabled_flag là bằng 0.

– Biên trên cùng của khối mã hóa hiện tại là một trong số các đường biên ảo ngang của ảnh và VirtualBoundariesDisabledFlag là bằng 1.

– Ngược lại, filterEdgeFlag được thiết lập bằng 1.

2.7 Ví dụ về TPM, HMVP và GEO

Chế độ dự báo tam giác TPM (triangular Prediction Mode) trong VVC chia khối thành hai tam giác với thông tin chuyển động khác nhau.

Dự báo vectơ chuyển động dựa trên tiến trình HMVP (History-based Motion vector Prediction) trong VVC duy trì bảng thông tin chuyển động cần được sử dụng để dự báo vectơ chuyển động. Bảng này được cập nhật sau khi giải mã khối được mã hóa ngoài ảnh, nhưng không được cập nhật nếu khối được mã hóa ngoài ảnh là mã hóa TPM.

Chế độ phân chia hình học GEO (geometry partition mode) là mở rộng của TPM. Với GEO, khối có thể được chia bởi đường thẳng thành hai phần chia, mà có thể là hoặc có thể không là các tam giác.

2.8 ALF, CC-ALF và biên ảo

Bộ lọc vòng thích ứng ALF (Adaptive Loop-Filter) trong VVC được áp dụng sau khi ảnh đã được giải mã, để tăng cường chất lượng ảnh.

Biên ảo (VB) được chấp nhận trong VVC để làm cho ALF thân thiện với thiết kế phần cứng. Với VB, ALF được tiến hành trong đơn vị xử lý ALF được giới hạn bởi hai đường biên ảo ALF.

CC-ALF (cross-component ALF) dưới dạng các bộ lọc các mẫu sắc độ bằng cách tham chiếu tới thông tin của các mẫu độ sáng.

3. Các ví dụ về các vấn đề kỹ thuật được giải quyết bởi các phương án theo sáng chế

(1) Có một số thiết kế mà có thể vi phạm điều kiện ràng buộc ảnh con.

A. TMVP trong các ứng viên được xây dựng afin có thể tìm nạp MV trong ảnh được sắp xếp ra ngoài khoảng của ảnh con hiện tại.

B. Khi dẫn xuất các gradien trong dòng quang học hai hướng (Bi-Directional Optical Flow - BDOF) và dòng quang học tinh chỉnh dự báo (Prediction Refinement Optical Flow - PROF), hai hàng mở rộng và hai cột mở rộng của các mẫu tham chiếu nguyên cần phải được tìm nạp. Các mẫu tham chiếu này có thể ra ngoài khoảng của ảnh con hiện tại.

C. Khi dẫn xuất hệ số định tỷ lệ độ dư sắc độ trong định tỷ lệ sắc độ ánh xạ độ sáng (luma mapping chroma scaling - LMCS), các mẫu độ sáng được tái tạo được truy cập có thể ra ngoài khoảng của ảnh con hiện tại.

D. Khối lân cận có thể ra ngoài khoảng của ảnh con hiện tại, khi dẫn xuất chế độ dự báo trong ảnh độ sáng, các mẫu tham chiếu để dự báo trong ảnh, các mẫu tham chiếu đối với CCLM, sự khả dụng khối lân cận đối với các ứng viên lân cận không gian đối với hợp nhất/AMVP/CIIP/IBC/LMCS, các thông số lượng tử hóa, quá trình khởi tạo CABAC, dẫn xuất ctxInc bằng cách sử dụng các phần tử cú pháp bên trái và trên, và ctxInc đối với phần tử cú pháp mtt_split_cu_vertical_flag. Sự biểu diễn của ảnh con có thể dẫn tới ảnh con với các CTU không hoàn chỉnh. Các phần chia CTU và quá trình tách CU có thể cần phải xem xét các CTU không hoàn chỉnh.

(2) Các phần tử cú pháp được báo tín hiệu liên quan đến ảnh con có thể lớn tùy ý, nên có thể gây ra vấn đề tràn.

(3) Sự biểu diễn của các ảnh con có thể dẫn tới các ảnh con không phải hình chữ nhật.

(4) Hiện tại ảnh con và lưới ảnh con được xác định trong các đơn vị bằng 4 mẫu. Và chiều dài của phần tử cú pháp phụ thuộc vào chiều cao ảnh được chia cho 4. Tuy nhiên, do pic_width_in_luma_samples và pic_height_in_luma_samples hiện tại phải là bội số nguyên của Max(8, MinCbSizeY), nên lưới ảnh con có thể cần phải được xác định trong các đơn vị bằng 8 mẫu.

(5) Cú pháp SPS, pic_width_max_in_luma_samples và pic_height_max_in_luma_samples có thể cần phải được giới hạn không nhỏ hơn 8.

(6) Sự tương tác giữa lấy mẫu lại/khả năng mở rộng ảnh tham chiếu và ảnh con không được xem xét trong thiết kế hiện tại.

(7) Trong khi lọc theo thời gian, có thể cần đến các mẫu ngang qua các ảnh con khác nhau.

(8) Khi báo tín hiệu các lát, thông tin có thể được suy ra mà không báo tín hiệu trong một số trường hợp.

(9) Có thể là tất cả các lát được xác định không thể bao phủ toàn bộ ảnh hoặc ảnh con.

(1) Các ID của hai ảnh con có thể giống hệt nhau.

(11) pic_width_max_in_luma_samples / CtbSizeY có thể bằng 0, dẫn đến hoạt động Log2() vô nghĩa.

(12) ID trong PH là ưu tiên hơn trong PPS, nhưng ít ưu tiên hơn trong SPS, nên không thống nhất.

(13) log2_transform_skip_max_size_minus2 trong PPS được phân tách phụ thuộc vào sps_transform_skip_enabled_flag trong SPS, dẫn đến sự phụ thuộc phân tách.

(14) loop_filter_across_subpic_enabled_flag để giải khối chỉ xem xét ảnh con hiện tại, mà không xem xét ảnh con lân cận.

4. Ví dụ về các kỹ thuật và các phương án

Phần liệt kê chi tiết dưới đây cần phải được xem xét như các ví dụ để giải thích các khái niệm chung. Các hạng mục này không được diễn giải theo cách hẹp. Hơn nữa, các hạng mục này có thể được kết hợp theo cách bất kỳ. Từ đây

về sau, bộ lọc thời gian được sử dụng để biểu diễn các bộ lọc mà yêu cầu các mẫu trong các ảnh khác. Max(x, y) trở lại giá trị lớn hơn trong số x và y. Min(x, y) trở lại giá trị nhỏ hơn trong số x và y.

1. Vị trí (gọi là vị trí RB) mà ở đó bộ dự báo MV theo thời gian được tìm nạp trong ảnh để tạo ra các ứng viên chuyển động afin (ví dụ ứng viên hợp nhất afin được xây dựng) phải ở trong ảnh con được yêu cầu, khi giả định tọa độ góc trên cùng bên trái của ảnh con được yêu cầu là (xTL, yTL) và tọa độ dưới cùng bên phải của ảnh con được yêu cầu là (xBR, yBR).

a. Trong một ví dụ, ảnh con được yêu cầu là ảnh con bao phủ khối hiện tại.

b. Trong một ví dụ, nếu vị trí RB với tọa độ (x, y) là ra ngoài ảnh con được yêu cầu, thì bộ dự báo MV theo thời gian được xử lý như không khả dụng.

i. Trong một ví dụ, vị trí RB là ra ngoài ảnh con được yêu cầu nếu x > xBR.

ii. Trong một ví dụ, vị trí RB là ra ngoài ảnh con được yêu cầu nếu y > yBR.

iii. Trong một ví dụ, vị trí RB là ra ngoài ảnh con được yêu cầu nếu x < xTL.

iv. Trong một ví dụ, vị trí RB là ra ngoài ảnh con được yêu cầu nếu y < yTL.

c. Trong một ví dụ, vị trí RB, nếu ở bên ngoài ảnh con được yêu cầu, thì sự thay thế RB được sử dụng.

i. Hơn nữa, theo cách khác, vị trí thay thế sẽ ở trong ảnh con được yêu cầu.

d. Trong một ví dụ, vị trí RB được xén để ở bên trong ảnh con được yêu cầu.

i. Trong một ví dụ, x được xén thành x = Min(x, xBR).

ii. Trong một ví dụ, y được xén thành y = Min(y, yBR).

iii. Trong một ví dụ, x được xén thành x = Max(x, xTL).

iv. Trong một ví dụ, y được xén thành y = Max(y, yTL).

e. Trong một ví dụ, vị trí RB có thể là vị trí dưới cùng bên phải bên trong khối tương ứng của khối hiện tại trong ảnh được sắp xếp.

f. Phương pháp được đề xuất có thể được sử dụng trong các công cụ mã hóa khác mà yêu cầu truy nhập thông tin chuyển động từ ảnh khác với ảnh hiện tại.

g. Trong một ví dụ, việc liệu các phương pháp trên đây có được áp dụng hay không (ví dụ, vị trí RB phải ở trong ảnh con được yêu cầu (ví dụ thực hiện như nêu trong 1.a và/hoặc 1.b)) có thể phụ thuộc vào một hoặc nhiều phần tử cú pháp được báo tín hiệu trong VPS/DPS/SPS/PPS/APS/tiêu đề lát/tiêu đề nhóm ô ảnh. Ví dụ, phần tử cú pháp có thể là subpic_treated_as_pic_flag[SubPicIdx], trong đó SubPicIdx là chỉ số ảnh con của ảnh con bao phủ khối hiện tại.

2. Vị trí (được gọi là vị trí S) mà ở đó mẫu nguyên được tìm nạp khi tham chiếu không được sử dụng trong quá trình nội suy phải ở trong ảnh con được yêu cầu, khi giả định tọa độ góc trên cùng bên trái của ảnh con được yêu cầu là (xTL, yTL) và tọa độ dưới cùng bên phải của ảnh con được yêu cầu là (xBR, yBR).

a. Trong một ví dụ, ảnh con được yêu cầu là ảnh con bao phủ khối hiện tại.

b. Trong một ví dụ, nếu vị trí S với tọa độ (x, y) là ra ngoài ảnh con được yêu cầu, thì mẫu tham chiếu được xử lý như không khả dụng.

i. Trong một ví dụ, vị trí S là ra ngoài ảnh con được yêu cầu nếu x > xBR.

ii. Trong một ví dụ, vị trí S là ra ngoài ảnh con được yêu cầu nếu y > yBR.

iii. Trong một ví dụ, vị trí S là ra ngoài ảnh con được yêu cầu nếu x < xTL.

iv. Trong một ví dụ, vị trí S là ra ngoài ảnh con được yêu cầu nếu y < yTL.

c. Trong một ví dụ, vị trí S được xén để ở trong ảnh con được yêu cầu.

i. Trong một ví dụ, x được xén thành x = Min(x, xBR).

ii. Trong một ví dụ, y được xén thành y = Min(y, yBR).

iii. Trong một ví dụ, x được xén thành x = Max(x, xTL).

iv. Trong một ví dụ, y được xén thành y = Max(y, yTL).

d. Trong một ví dụ, việc liệu vị trí S có phải ở trong ảnh con được yêu cầu hay không (ví dụ thực hiện như nêu trong 2.a và/hoặc 2.b) có thể phụ thuộc vào một hoặc nhiều phần tử cú pháp được báo tín hiệu trong VPS/DPS/SPS/PPS/APS/tiêu đề lát/tiêu đề nhóm ô ảnh. Ví dụ, phần tử cú pháp có thể là subpic_treated_as_pic_flag[SubPicIdx], trong đó SubPicIdx là chỉ số ảnh con của ảnh con bao phủ khối hiện tại.

e. Trong một ví dụ, mẫu nguyên được tìm nạp được sử dụng để tạo ra các gradien trong BDOF và/hoặc PORF.

3. Vị trí (được gọi là vị trí R) mà ở đó giá trị mẫu độ sáng được tái tạo được tìm nạp có thể ở trong ảnh con được yêu cầu, khi giả định tọa độ góc trên cùng bên trái của ảnh con được yêu cầu là (xTL, yTL) và tọa độ dưới cùng bên phải của ảnh con được yêu cầu là (xBR, yBR).

a. Trong một ví dụ, ảnh con được yêu cầu là ảnh con bao phủ khối hiện tại.

b. Trong một ví dụ, nếu vị trí R với tọa độ (x, y) là ra ngoài ảnh con được yêu cầu, thì mẫu tham chiếu được xử lý như không khả dụng.

i. Trong một ví dụ, vị trí R là ra ngoài ảnh con được yêu cầu nếu x > xBR.

ii. Trong một ví dụ, vị trí R là ra ngoài ảnh con được yêu cầu nếu y > yBR.

iii. Trong một ví dụ, vị trí R là ra ngoài ảnh con được yêu cầu nếu x < xTL.

iv. Trong một ví dụ, vị trí R là ra ngoài ảnh con được yêu cầu nếu y < yTL.

c. Trong một ví dụ, vị trí R được xén để ở trong ảnh con được yêu cầu.

i. Trong một ví dụ, x được xén thành x = Min(x, xBR).

ii. Trong một ví dụ, y được xén thành y = Min(y, yBR).

iii. Trong một ví dụ, x được xén thành x = Max(x, xTL).

iv. Trong một ví dụ, y được xén thành y = Max(y, yTL).

d. Trong một ví dụ, việc liệu vị trí R có phải ở trong ảnh con được yêu cầu hay không (ví dụ thực hiện như nêu trong 4.a và/hoặc 4.b) có thể phụ thuộc vào một hoặc nhiều phần tử cú pháp được báo tín hiệu trong VPS/DPS/SPS/PPS/APS/tiêu đề lát/tiêu đề nhóm ô ảnh. Ví dụ, phần tử cú pháp có thể là subpic_treated_as_pic_flag[SubPicIdx], trong đó SubPicIdx là chỉ số ảnh con của ảnh con bao phủ khối hiện tại.

e. Trong một ví dụ, mẫu độ sáng được tìm nạp được sử dụng để rút ra hệ số định tỷ lệ đối với (các) thành phần sắc độ trong LMCS.

4. Vị trí (được gọi là vị trí N) mà ở đó việc kiểm tra biên ảnh đối với tách BT/TT/QT, dẫn xuất sâu BT/TT/QT, và/hoặc báo tín hiệu của cờ tách CU phải ở trong ảnh con được yêu cầu, khi giả định tọa độ góc trên cùng bên trái của ảnh con được yêu cầu là (xTL, yTL) và tọa độ dưới cùng bên phải của ảnh con được yêu cầu là (xBR, yBR).

a. Trong một ví dụ, ảnh con được yêu cầu là ảnh con bao phủ khối hiện tại.

b. Trong một ví dụ, nếu vị trí N với tọa độ (x, y) là ra ngoài ảnh con được yêu cầu, thì mẫu tham chiếu được xử lý như không khả dụng.

i. Trong một ví dụ, vị trí N là ra ngoài ảnh con được yêu cầu nếu x > xBR.

ii. Trong một ví dụ, vị trí N là ra ngoài ảnh con được yêu cầu nếu y > yBR.

iii. Trong một ví dụ, vị trí N là ra ngoài ảnh con được yêu cầu nếu x < xTL.

iv. Trong một ví dụ, vị trí N là ra ngoài ảnh con được yêu cầu nếu y < yTL.

c. Trong một ví dụ, vị trí N được xén để ở trong ảnh con được yêu cầu.

i. Trong một ví dụ, x được xén thành x = Min(x, xBR).

ii. Trong một ví dụ, y được xén thành y = Min(y, yBR).

iii. Trong một ví dụ, x được xén thành x = Max(x, xTL).

d. Trong một ví dụ, y được xén thành y = Max(y, yTL). Trong một ví dụ, việc liệu vị trí N có phải ở trong ảnh con được yêu

cầu hay không (ví dụ thực hiện như nêu trong 5.a và/hoặc 5.b) có thể phụ thuộc vào một hoặc nhiều phần tử cú pháp được báo tín hiệu trong VPS/DPS/SPS/PPS/APS/tiêu đề lát/tiêu đề nhóm ô ảnh. Ví dụ, phần tử cú pháp có thể là subpic_treated_as_pic_flag[SubPicIdx], trong đó SubPicIdx là chỉ số ảnh con của ảnh con bao phủ khối hiện tại.

5. Bảng dự báo vectơ chuyển động dựa trên tiến trình (HMVP) có thể được thiết lập lại trước khi giải mã ảnh con mới trong một ảnh.

a. Trong một ví dụ, bảng HMVP được sử dụng để mã hóa IBC có thể được thiết lập lại

b. Trong một ví dụ, bảng HMVP được sử dụng để mã hóa ngoài ảnh có thể được thiết lập lại

c. Trong một ví dụ, bảng HMVP được sử dụng để mã hóa trong ảnh có thể được thiết lập lại

6. Các phần tử cú pháp ảnh con có thể được xác định trong các đơn vị của N (chẳng hạn N = 8, 32, và v.v.) mẫu.

a. Trong một ví dụ, chiều rộng của mỗi phần tử của lưới bộ nhận dạng ảnh con trong các đơn vị của N mẫu.

b. Trong một ví dụ, chiều cao của mỗi phần tử của lưới bộ nhận dạng ảnh con trong các đơn vị của N mẫu.

c. Trong một ví dụ, N được thiết lập bằng chiều rộng và/hoặc chiều cao của CTU.

7. Phần tử cú pháp của chiều rộng ảnh và chiều cao ảnh có thể được giới hạn không nhỏ hơn K (K>=8).

a. Trong một ví dụ, chiều rộng ảnh có thể cần phải được giới hạn không nhỏ hơn 8.

b. Trong một ví dụ, chiều cao ảnh có thể cần phải được giới hạn không nhỏ hơn 8.

8. Dòng bit tương thích sẽ thỏa mãn điều kiện rằng mã hóa ảnh con và Biến đổi độ phân giải thích ứng (Adaptive resolution conversion - ARC)/ Biến đổi độ phân giải động (Dynamic resolution conversion -

DRC)/ Ảnh tham chiếu lấy mẫu lại (Reference picture resampling - RPR) không được cho phép là được cấp phép đối với một đơn vị viđeo (ví dụ, dãy).

a. Trong một ví dụ, việc báo tín hiệu về việc cho phép mã hóa ảnh con có thể theo các điều kiện về việc không cho phép ARC/DRC/RPR.

i. Trong một ví dụ, khi ảnh con là được phép, chẳng hạn **subpics_present_flag** bằng 1, **pic_width_in_luma_samples** đối với tất cả các ảnh mà đối với nó SPS này là hoạt động là bằng max_width_in_luma_samples.

b. Theo cách khác, cả việc mã hóa ảnh con lẫn ARC/DRC/RPR có thể được phép đối với một đơn vị viđeo (ví dụ, dãy).

i. Trong một ví dụ, dòng bit tương thích sẽ thỏa mãn rằng ảnh con được giảm kích thước mẫu do ARC/DRC/RPR sẽ vẫn dưới dạng K các CTU theo chiều rộng và M các CTU theo chiều cao trong đó cả K và M đều là các số nguyên.

ii. Trong một ví dụ, dòng bit tương thích sẽ thỏa mãn rằng đối với các ảnh con không nằm ở các đường biên ảnh (ví dụ, biên phải và/hoặc biên đáy), ảnh con được giảm kích thước mẫu do ARC/DRC/RPR sẽ vẫn dưới dạng K các CTU theo chiều rộng và M các CTU theo chiều cao trong đó cả K và M đều là các số nguyên.

iii. Trong một ví dụ, các kích thước CTU có thể được thay đổi một cách thích ứng dựa trên độ phân giải ảnh.

1) Trong một ví dụ, kích thước CTU tối đa có thể được báo tín hiệu trong SPS. Đối với mỗi ảnh với độ phân giải thấp hơn, kích thước CTU có

thể thay đổi một cách tương ứng dựa trên độ phân giải được giảm đi.

2) Trong một ví dụ, kích thước CTU có thể được báo tín hiệu trong SPS và PPS, và/hoặc bậc ảnh con.

9. Phần tử cú pháp subpic_grid_col_width_minus1 và subpic_grid_row_height_minus1 có thể bị ràng buộc.

a. Trong một ví dụ, subpic_grid_col_width_minus1 phải không lớn hơn (hoặc không nhỏ hơn) T1.

b. Trong một ví dụ, subpic_grid_row_height_minus1 phải không lớn hơn (hoặc không nhỏ hơn) T2.

c. Trong một ví dụ, trong dòng bit tương thích, subpic_grid_col_width_minus1 và/hoặc subpic_grid_row_height_minus1 phải theo điều kiện ràng buộc chẳng hạn tiểu mục 3.a hoặc 3.b.

d. Trong một ví dụ, T1 trong 3.a và/hoặc T2 trong 3.b có thể phụ thuộc vào các lược tả/các bậc/các cấp của chuẩn mã hóa viđeo.

e. Trong một ví dụ, T1 trong 3.a có thể phụ thuộc vào chiều rộng ảnh.

i. Ví dụ, T1 là bằng pic_width_max_in_luma_samples/4 hoặc pic_width_max_in_luma_samples/4+Off. Off có thể là 1, 2, -1, -2, v.v..

f. Trong một ví dụ, T2 trong 3.b có thể phụ thuộc vào chiều rộng ảnh.

i. Ví dụ, T2 là bằng pic_height_max_in_luma_samples/4 hoặc pic_height_max_in_luma_samples /4-1+Off. Off có thể là 1, 2, -1, -2, v.v..

10. Điều kiện ràng buộc là biên giữa hai ảnh con phải là biên giữa hai CTU.

a. Nói cách khác, CTU không thể được bao phủ bởi nhiều hơn một ảnh con.

b. Trong một ví dụ, đơn vị của subpic_grid_col_width_minus1 có thể là chiều rộng CTU (chẳng hạn 32, 64, 128), thay vì 4 như trong VVC. Chiều rộng lưới ảnh con cần phải bằng (subpic_grid_col_width_minus1+1) * CTU width.

c. Trong một ví dụ, đơn vị của subpic_grid_col_height_minus1 có thể bằng chiều cao CTU (chẳng hạn 32, 64, 128), thay vì 4 như trong VVC. Chiều cao lưới ảnh con cần phải bằng (subpic_grid_col_height_minus1+1) * CTU height.

d. Trong một ví dụ, trong dòng bit tương thích, điều kiện ràng buộc phải được thỏa mãn nếu cách tiếp cận ảnh con được áp dụng.

11. Điều kiện ràng buộc là hình dạng của ảnh con phải là hình chữ nhật.

a. Trong một ví dụ, trong dòng bit tương thích, điều kiện ràng buộc phải được thỏa mãn nếu cách tiếp cận ảnh con được áp dụng.

b. Ảnh con có thể chỉ chứa các lát chữ nhật. Ví dụ, trong dòng bit tương thích, điều kiện ràng buộc phải được thỏa mãn nếu cách tiếp cận ảnh con được áp dụng.

12. Điều kiện ràng buộc là hai ảnh con không thể bị xếp chồng.

a. Trong một ví dụ, trong dòng bit tương thích, điều kiện ràng buộc phải được thỏa mãn nếu cách tiếp cận ảnh con được áp dụng.

b. Theo cách khác, hai ảnh con có thể được xếp chồng với nhau.

13. Điều kiện ràng buộc là vị trí bất kỳ trong ảnh phải được bao phủ bởi một và chỉ một ảnh con.

a. Trong một ví dụ, trong dòng bit tương thích, điều kiện ràng buộc phải được thỏa mãn nếu cách tiếp cận ảnh con được áp dụng.

b. Theo cách khác, một mẫu có thể không thuộc về ảnh con bất kỳ.

c. Theo cách khác, một mẫu thuộc về nhiều hơn một ảnh con.

14. Có thể bị ràng buộc rằng các ảnh con được xác định trong SPS được ánh xạ tới mọi độ phân giải có mặt trong cùng dãy phải theo các ràng buộc về vị trí và/hoặc kích thước được nêu trên.

a. Trong một ví dụ, chiều rộng và chiều cao của ảnh con được xác định trong SPS được ánh xạ tới độ phân giải có mặt trong cùng dãy, cần phải là bội số nguyên lần của N (chẳng hạn 8, 16, 32) các mẫu độ sáng.

b. Trong một ví dụ, các ảnh con có thể được xác định đối với lớp đã biết và có thể được ánh xạ tới các lớp khác.

i. Ví dụ, các ảnh con có thể được xác định đối với lớp với độ phân giải cao nhất trong dãy.

ii. Ví dụ, các ảnh con có thể được xác định đối với lớp với độ phân giải thấp nhất trong dãy.

iii. Việc các ảnh con được xác định đối với lớp nào có thể được báo tín hiệu trong SPS/VPS/PPS/tiêu đề lát.

c. Trong một ví dụ, khi cả các ảnh con và các độ phân giải khác nhau đều được áp dụng, tất cả các độ phân giải (ví dụ, chiều rộng hoặc/và chiều cao) có thể là bội số nguyên của độ phân giải cho trước.

d. Trong một ví dụ, chiều rộng và/hoặc chiều cao của ảnh con được xác định trong SPS có thể là bội số nguyên lần (ví dụ, M) của kích thước CTU.

e. Theo cách khác, các ảnh con và các độ phân giải khác nhau trong dãy có thể không được cho phép đồng thời.

15. Các ảnh con có thể chỉ áp dụng cho (các) lớp đã biết

a. Trong một ví dụ, các ảnh con được xác định trong SPS có thể chỉ áp dụng cho lớp với độ phân giải cao nhất trong dãy.

b. Trong một ví dụ, các ảnh con được xác định trong SPS có thể chỉ áp dụng cho lớp với id theo thời gian thấp nhất trong dãy.

c. Việc các ảnh con có thể được áp dụng cho (các) lớp nào có thể được chỉ báo bởi một hoặc nhiều phần tử cú pháp trong SPS/VPS/PPS.

d. Việc các ảnh con không thể được áp dụng cho (các) lớp nào có thể được chỉ báo bởi một hoặc nhiều phần tử cú pháp trong SPS/VPS/PPS.

16. Trong một ví dụ, vị trí và/hoặc các kích cỡ của ảnh con có thể được báo tín hiệu mà không sử dụng subpic_grid_idx.

a. Trong một ví dụ, vị trí trên cùng bên trái của ảnh con có thể được báo tín hiệu.

b. Trong một ví dụ, vị trí dưới cùng bên phải của ảnh con có thể được báo tín hiệu.

c. Trong một ví dụ, chiều rộng của ảnh con có thể được báo tín hiệu.

d. Trong một ví dụ, chiều cao của ảnh con có thể được báo tín hiệu.

17. Đối với bộ lọc thời gian, khi thực hiện việc lọc thời gian của mẫu, thì chỉ các mẫu bên trong cùng ảnh con mà mẫu hiện tại thuộc về là có thể được sử dụng. Các mẫu được yêu cầu có thể ở trong cùng ảnh mà mẫu hiện tại thuộc về hoặc trong các ảnh khác.

18. Trong một ví dụ, việc liệu có áp dụng và/hoặc cách thức áp dụng phương pháp phân chia (chẳng hạn QT, BT ngang, BT dọc, TT ngang, TT dọc, hoặc không tách, v.v.) có thể phụ thuộc vào việc liệu khối hiện

tại (hoặc phần chia) đi ngang qua một hoặc nhiều biên của ảnh con hay không.

a. Trong một ví dụ, phương pháp xử lý biên ảnh để phân chia trong VVC cũng có thể được áp dụng khi biên ảnh được thay thế bởi biên ảnh con.

b. Trong một ví dụ, việc liệu có phân tách phần tử cú pháp (ví dụ flag) mà biểu diễn phương pháp phân chia (chẳng hạn QT, BT ngang, BT dọc, TT ngang, TT dọc, hoặc không tách, v.v.) hay không có thể phụ thuộc vào việc liệu khối hiện tại (hoặc phân chia) có đi ngang qua một hoặc nhiều biên của ảnh con hay không.

19. Thay vì tách một ảnh thành nhiều ảnh con nhờ mã hóa độc lập của mỗi ảnh con, đề xuất rằng tách ảnh thành ít nhất hai nhóm các vùng con, với nhóm thứ nhất bao gồm nhiều ảnh con và nhóm thứ hai bao gồm tất cả các mẫu còn lại.

a. Trong một ví dụ, mẫu trong nhóm thứ hai không có trong các ảnh con bất kỳ.

b. Theo cách khác, hơn nữa, nhóm thứ hai có thể được tạo mã/được giải mã dựa trên thông tin của nhóm thứ nhất.

c. Trong một ví dụ, giá trị mặc định có thể được sử dụng để đánh dấu liệu mẫu/vùng con MxK có thuộc về nhóm thứ hai.

i. Trong một ví dụ, giá trị mặc định có thể được thiết lập bằng (**max_subpics_minus1** + K) trong đó K là số nguyên lớn hơn 1.

ii. Giá trị mặc định có thể được gán cho subpic_grid_idx[i][j] để chỉ báo rằng lưới thuộc về nhóm thứ hai.

20. Đề xuất rằng phần tử cú pháp subpic_grid_idx[i][j] không thể lớn hơn max_subpics_minus1.

a. Ví dụ, điều kiện ràng buộc là trong dòng bit tương thích, subpic_grid_idx[i][j] không thể lớn hơn max_subpics_minus1.

b. Ví dụ, từ mã để mã hóa subpic_grid_idx[i][j] không thể lớn hơn max_subpics_minus1.

21. Đề xuất rằng, số nguyên bất kỳ từ 0 tới max_subpics_minus1 phải bằng ít nhất một subpic_grid_idx[i][j].

22. Bộ đệm ảo IBC có thể được thiết lập lại trước khi giải mã ảnh con mới trong một ảnh.

a. Trong một ví dụ, tất cả các mẫu trong bộ đệm ảo IBC có thể được thiết lập lại tới -1.

23. Danh sách mục nhập bảng màu có thể được thiết lập lại trước khi giải mã ảnh con mới trong một ảnh.

a. Trong một ví dụ, PredictorPaletteSize có thể được thiết lập bằng 0 trước khi giải mã ảnh con mới trong một ảnh.

24. Việc liệu có báo tín hiệu thông tin của các lát (ví dụ số lượng của các lát và/hoặc các khoảng của các lát) hay không có thể phụ thuộc vào số lượng của các ô ảnh và/hoặc số lượng các ô gạch.

a. Trong một ví dụ, nếu số lượng các ô gạch trong ảnh là một, thì num_slices_in_pic_minus1 không được báo tín hiệu và được suy ra bằng 0.

b. Trong một ví dụ, nếu số lượng các ô gạch trong ảnh là một, thì thông tin của các lát (ví dụ số lượng của các lát và/hoặc các khoảng của các lát) có thể không được báo tín hiệu.

c. Trong một ví dụ, nếu số lượng các ô gạch trong ảnh là một, thì số lượng của các lát có thể được suy ra bằng một. và lát này bao phủ toàn bộ ảnh.Trong một ví dụ, nếu số lượng các ô gạch trong ảnh là một, thì single_brick_per_slice_flag không được báo tín hiệu và được suy ra bằng một.

i. Theo cách khác, nếu số lượng các ô gạch trong ảnh là một, thì single_brick_per_slice_flag phải là một.

d. Thiết kế cú pháp ví dụ là như dưới đây:

pic_parameter_set_rbsp() {	**Descriptor**
...	
if(NumBricksInPic > 1){	

single_brick_per_slice_flag	u(1)
if(!single_brick_per_slice_flag)	
rect_slice_flag	u(1)
if(rect_slice_flag && !single_brick_per_slice_flag) {	
num_slices_in_pic_minus1	ue(v)
bottom_right_brick_idx_length_minus1	ue(v)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
bottom_right_brick_idx_delta[i]	u(v)
brick_idx_delta_sign_flag[i]	u(1)
}	
}	
}	
loop_filter_across_bricks_enabled_flag	u(1)
if(loop_filter_across_bricks_enabled_flag)	
loop_filter_across_slices_enabled_flag	u(1)
}	
...	

25. Việc liệu có báo tín hiệu slice_address có thể được tách rời khỏi việc liệu các lát có được báo tín hiệu là các hình chữ nhật (ví dụ liệu rect_slice_flag là bằng 0 hoặc 1).

a. Thiết kế cú pháp ví dụ là như dưới đây:

if([[rect_slice_flag ||]] NumBricksInPic > 1)	
slice_address	u(v)

26. Việc liệu có báo tín hiệu slice_address có thể phụ thuộc vào số lượng của các lát khi các lát được báo tín hiệu là các hình chữ nhật.

if((rect_slice_flag && num_slices_in_pic_minus1> 0) || (!rect_slice_flag && NumBricksInPic > 1))	
slice_address	u(v)

27. Việc liệu có báo tín hiệu num_bricks_in_slice_minus1 có thể phụ thuộc vào slice_address và/hoặc số lượng các ô gạch trong ảnh.

a. Thiết kế cú pháp ví dụ là như dưới đây:

if(!rect_slice_flag && !single_brick_per_slice_flag ***&& slice_address < NumBricksInPic -1***)	
num_bricks_in_slice_minus1	ue(v)

28. Việc liệu có báo tín hiệu loop_filter_across_bricks_enabled_flag có thể phụ thuộc vào số lượng của các ô ảnh và/hoặc số lượng các ô gạch.

a. Trong một ví dụ, loop_filter_across_bricks_enabled_flag không được báo tín hiệu nếu số lượng các ô gạch nhỏ hơn 2.

b. Thiết kế cú pháp ví dụ là như dưới đây:

pic_parameter_set_rbsp() {	**Descriptor**
...	

if(NumBricksInPic > 1)	
loop_filter_across_bricks_enabled_flag	u(1)
if(loop_filter_across_bricks_enabled_flag)	
loop_filter_across_slices_enabled_flag	u(1)
...	

29. Sự tương thích dòng bit yêu cầu rằng tất cả các lát của ảnh phải bao phủ toàn bộ ảnh.

a. Yêu cầu này phải được thỏa mãn khi các lát được báo tín hiệu là các hình chữ nhật (ví dụ rect_slice_flag là bằng 1).

30. Sự tương thích dòng bit yêu cầu rằng tất cả các lát của ảnh con phải bao phủ toàn bộ ảnh con.

a. Yêu cầu này phải được thỏa mãn khi các lát được báo tín hiệu là các hình chữ nhật (ví dụ rect_slice_flag là bằng 1).

31. Sự tương thích dòng bit yêu cầu rằng lát không thể được xếp chồng với nhiều hơn một ảnh con.

32. Sự tương thích dòng bit yêu cầu rằng ô ảnh không thể được xếp chồng với nhiều hơn một ảnh con.

33. Sự tương thích dòng bit yêu cầu rằng ô gạch không thể được xếp chồng với nhiều hơn một ảnh con.

Trong phần sau đây, khối đơn vị cơ bản (basic unit block - BUB) với các kích cỡ CW × CH là vùng chữ nhật. Ví dụ, BUB có thể là khối cây mã hóa (Coding Tree Block - CTB).

34. Trong một ví dụ, số lượng của các ảnh con (được ký hiệu là N) có thể được báo tín hiệu.

a. Có thể yêu cầu đối với dòng bit tương thích rằng có ít nhất hai ảnh con trong ảnh nếu các ảnh con được sử dụng (ví dụ subpics_present_flag là bằng 1).

b. Theo cách khác, N minus d (tức là, N-d) có thể được báo tín hiệu, trong đó d là số nguyên chẳng hạn 0, 1, hoặc 2.

c. Ví dụ, N-d có thể được mã hóa với mã hóa chiều dài cố định ví dụ u(x).

i. Trong một ví dụ, x có thể là số cố định chẳng hạn 8.

ii. Trong một ví dụ, x hoặc x-dx có thể được báo tín hiệu trước khi N-d được báo tín hiệu, trong đó dx là số nguyên chẳng hạn 0, 1 hoặc 2. x được báo tín hiệu không thể lớn hơn giá trị tối đa trong dòng bit tương thích.

iii. Trong một ví dụ, x có thể được rút ra trong tiến trình.

1) Ví dụ, x có thể được rút ra dưới dạng hàm số của tổng số lượng (được ký hiệu là M) của các BUB trong ảnh. Ví dụ, x = Ceil(log2(M+d0))+d1, trong đó d0 và d1 là hai số nguyên, chẳng hạn -2, -1, 0, 1, 2, v.v. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

2) M có thể được rút ra dưới dạng M = Ceiling(W/CW) × Ceiling(H/ CH), trong đó W và H biểu diễn chiều rộng và chiều cao của ảnh, và CW và CH biểu diễn chiều rộng và chiều cao của BUB.

d. Ví dụ, N-d có thể được mã hóa bằng mã đơn phân hoặc mã đơn phân được rút ngắn.

e. Trong một ví dụ, giá trị tối đa được cho phép của N-d có thể là số cố định.

i. Theo cách khác, giá trị tối đa được cho phép của N-d có thể được rút ra dưới dạng hàm số của tổng số lượng (được ký hiệu là M) của các BUB trong ảnh. Ví dụ, x = Ceil(log2(M+d0))+d1, trong đó d0 và d1 là hai số nguyên, chẳng hạn -2, -1, 0, 1, 2, v.v. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

35. Trong một ví dụ, ảnh con có thể được báo tín hiệu bằng các chỉ báo của một hoặc nhiều vị trí được chọn của nó (ví dụ, vị trí trên cùng

bên trái/trên cùng bên phải/dưới cùng bên trái/ dưới cùng bên phải) và/hoặc chiều rộng và/hoặc chiều cao của nó.

a. Trong một ví dụ, vị trí trên cùng bên trái của ảnh con có thể được báo tín hiệu trong chi tiết của khối đơn vị cơ bản (BUB) với các kích cỡ CW × CH.

i. Ví dụ, cột chỉ số (được ký hiệu là Col) dưới dạng các BUB trong số BUB trên cùng bên trái của ảnh con có thể được báo tín hiệu.

1) Ví dụ, Col - d có thể được báo tín hiệu, trong đó d là số nguyên chẳng hạn 0, 1, hoặc 2.

a) Theo cách khác, d có thể bằng Col của ảnh con được mã hóa trước đó, được bổ sung bởi d1, trong đó d1 là số nguyên chẳng hạn -1,0, hoặc 1.

b) Dấu của Col -d có thể được báo tín hiệu.

ii. Ví dụ, chỉ số hàng (được ký hiệu là Row) dưới dạng các BUB trong số BUB trên cùng bên trái của ảnh con có thể được báo tín hiệu.

1) Ví dụ, Row - d có thể được báo tín hiệu, trong đó d là số nguyên chẳng hạn 0, 1, hoặc 2.

a) Theo cách khác, d có thể bằng Row của ảnh con được mã hóa trước đó, được bổ sung bởi d1, trong đó d1 là số nguyên chẳng hạn -1,0, hoặc 1.

b) Dấu của Row -d có thể được báo tín hiệu.

iii. Chỉ số hàng/cột (được ký hiệu là Row) được nêu trên có thể được biểu diễn trong đơn vị khối cây mã hóa (Coding Tree Block - CTB), ví dụ, tọa độ x hoặc y so với

vị trí trên cùng bên trái của ảnh có thể được chia cho kích thước CTB và được báo tín hiệu.

iv. Trong một ví dụ, việc liệu có báo tín hiệu vị trí của ảnh con có thể phụ thuộc vào chỉ số ảnh con.

1) Trong một ví dụ, đối với ảnh con thứ nhất bên trong ảnh, vị trí trên cùng bên trái có thể không được báo tín hiệu.

a) Theo cách khác, hơn nữa, vị trí trên cùng bên trái có thể được suy ra, ví dụ, bằng (0, 0).

2) Trong một ví dụ, đối với ảnh con cuối cùng bên trong ảnh, vị trí trên cùng bên trái có thể không được báo tín hiệu.

a) Vị trí trên cùng bên trái có thể được suy ra phụ thuộc vào thông tin của các ảnh con được báo tín hiệu trước đó.

b. Trong một ví dụ, các chỉ báo về chiều rộng/chiều cao/vị trí được chọn của ảnh con có thể được báo tín hiệu bằng mã hóa đơn phân được cắt ngắn/nhị phân được cắt ngắn/đơn phân/chiều dài cố định/EG thứ K (ví dụ, K=0, 1, 2, 3).

c. Trong một ví dụ, chiều rộng của ảnh con có thể được báo tín hiệu trong chi tiết của BUB với các kích cỡ CW × CH.

i. Ví dụ, số lượng các cột của các BUB trong ảnh con (được ký hiệu là W) có thể được báo tín hiệu.

ii. Ví dụ, W-d có thể được báo tín hiệu, trong đó d là số nguyên chẳng hạn 0, 1, hoặc 2.

1) Theo cách khác, d có thể bằng W của ảnh con được mã hóa trước đó, được bổ sung bởi d1, trong đó d1 là số nguyên chẳng hạn -1,0, hoặc 1.

2) Dấu của W-d có thể được báo tín hiệu.

d. Trong một ví dụ, chiều cao của ảnh con có thể được báo tín hiệu trong chi tiết của BUB với các kích cỡ CW × CH.

i. Ví dụ, số lượng của các hàng của các BUB trong ảnh con (được ký hiệu là H) có thể được báo tín hiệu.

ii. Ví dụ, H-d có thể được báo tín hiệu, trong đó d là số nguyên chẳng hạn 0, 1, hoặc 2.

1) Theo cách khác, d có thể bằng H của ảnh con được mã hóa trước đó, được bổ sung bởi d1, trong đó d1 là số nguyên chẳng hạn -1,0, hoặc 1.

2) Dấu của H-d có thể được báo tín hiệu.

e. Trong một ví dụ, Col-d có thể được mã hóa bằng mã hóa chiều dài cố định ví dụ u(x).

i. Trong một ví dụ, x có thể là số cố định chẳng hạn 8.

ii. Trong một ví dụ, x hoặc x-dx có thể được báo tín hiệu trước khi Col-d được báo tín hiệu, trong đó dx là số nguyên chẳng hạn 0, 1 hoặc 2. x được báo tín hiệu không thể lớn hơn giá trị tối đa trong dòng bit tương thích.

iii. Trong một ví dụ, x có thể được rút ra trong tiến trình.

1) Ví dụ, x có thể được rút ra dưới dạng hàm số của tổng số lượng (được ký hiệu là M) của các cột BUB trong ảnh. Ví dụ, x = Ceil(log2(M+d0))+d1, trong đó d0 và d1 là hai số nguyên, chẳng hạn -2, -1, 0, 1, 2, v.v. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

2) M có thể được rút ra dưới dạng M = Ceiling(W/CW), trong đó W biểu diễn chiều rộng của ảnh, và CW biểu diễn chiều rộng của BUB.

f. Trong một ví dụ, Row-d có thể được mã hóa bằng mã hóa chiều dài cố định ví dụ u(x).

i. Trong một ví dụ, x có thể là số cố định chẳng hạn 8.

ii. Trong một ví dụ, x hoặc x-dx có thể được báo tín hiệu trước khi Row-d được báo tín hiệu, trong đó dx là số nguyên chẳng hạn 0, 1 hoặc 2. x được báo tín hiệu không thể lớn hơn giá trị tối đa trong dòng bit tương thích.

iii. Trong một ví dụ, x có thể được rút ra trong tiến trình.

1) Ví dụ, x có thể được rút ra dưới dạng hàm số của tổng số lượng (được ký hiệu là M) của các hàng BUB trong ảnh. Ví dụ, x = Ceil(log2(M+d0))+d1, trong đó d0 và d1 là hai số nguyên, chẳng hạn -2, -1, 0, 1, 2, v.v. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

2) M có thể được rút ra dưới dạng M = Ceiling(H/CH), trong đó H biểu diễn chiều cao của ảnh, và CH biểu diễn chiều cao của BUB.

g. Trong một ví dụ, W-d có thể được mã hóa bằng mã hóa chiều dài cố định ví dụ u(x).

i. Trong một ví dụ, x có thể là số cố định chẳng hạn 8.

ii. Trong một ví dụ, x hoặc x-dx có thể được báo tín hiệu trước khi W-d được báo tín hiệu, trong đó dx là số nguyên chẳng hạn 0, 1 hoặc 2. x được báo tín hiệu không thể lớn hơn giá trị tối đa trong dòng bit tương thích.

iii. Trong một ví dụ, x có thể được rút ra trong tiến trình.

1) Ví dụ, x có thể được rút ra dưới dạng hàm số của tổng số lượng (được ký hiệu là M) của các cột BUB trong ảnh. Ví dụ, x = Ceil(log2(M+d0))+d1, trong đó d0 và d1 là hai số nguyên, chẳng hạn -2, -1, 0, 1, 2, v.v. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

2) M có thể được rút ra dưới dạng M = Ceiling(W/CW), trong đó W biểu diễn chiều rộng của ảnh, và CW biểu diễn chiều rộng của BUB.

h. Trong một ví dụ, H-d có thể được mã hóa bằng mã hóa chiều dài cố định ví dụ u(x).

i. Trong một ví dụ, x có thể là số cố định chẳng hạn 8.

ii. Trong một ví dụ, x hoặc x-dx có thể được báo tín hiệu trước khi H-d được báo tín hiệu, trong đó dx là số nguyên chẳng hạn 0, 1 hoặc 2. x được báo tín hiệu không thể lớn hơn giá trị tối đa trong dòng bit tương thích.

iii. Trong một ví dụ, x có thể được rút ra trong tiến trình.

1) Ví dụ, x có thể được rút ra dưới dạng hàm số của tổng số lượng (được ký hiệu là M) của các hàng BUB trong ảnh. Ví dụ, x = Ceil(log2(M+d0))+d1, trong đó d0 và d1 là hai số nguyên, chẳng hạn -2, -1, 0, 1, 2, v.v. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

2) M có thể được rút ra dưới dạng M = Ceiling(H/CH), trong đó H biểu diễn chiều cao của ảnh, và CH biểu diễn chiều cao của BUB.

i. Col-d và/hoặc Row-d có thể được báo tín hiệu đối với tất cả các ảnh con.

i. Theo cách khác, Col-d và/hoặc Row-d có thể không được báo tín hiệu đối với tất cả các ảnh con.

1) Col-d và/hoặc Row-d có thể không được báo tín hiệu nếu số lượng của các ảnh con là nhỏ hơn 2. (bằng 1).

2) Ví dụ, Col-d và/hoặc Row-d có thể không được báo tín hiệu đối với ảnh con thứ nhất (ví dụ bằng chỉ số ảnh con (hoặc Id ảnh con) bằng 0).

a) Khi chúng không được báo tín hiệu, chúng có thể được suy ra bằng 0.

3) Ví dụ, Col-d và/hoặc Row-d có thể không được báo tín hiệu đối với ảnh con cuối cùng (ví dụ bằng chỉ số ảnh con (hoặc Id ảnh con) bằng NumSubPics-1).

a) Khi chúng không được báo tín hiệu, chúng có thể được suy ra phụ thuộc vào các vị trí và các kích cỡ của các ảnh con đã được báo tín hiệu.

j. W-d và/hoặc H-d có thể được báo tín hiệu đối với tất cả các ảnh con.

i. Theo cách khác, W-d và/hoặc H-d có thể không được báo tín hiệu đối với tất cả các ảnh con.

1) W-d và/hoặc H-d có thể không được báo tín hiệu nếu số lượng của các ảnh con là nhỏ hơn 2. (bằng 1).

2) Ví dụ, W-d và/hoặc H-d có thể không được báo tín hiệu đối với ảnh con cuối cùng (ví dụ bằng chỉ số ảnh con (hoặc Id ảnh con) bằng NumSubPics-1).

a) Khi chúng không được báo tín hiệu, chúng có thể được suy ra phụ thuộc vào các vị trí và các kích cỡ của các ảnh con đã được báo tín hiệu.

k. Trong các tiểu mục trên đây, BUB có thể là khối cây mã hóa (CTB).

36. Trong một ví dụ, thông tin của các ảnh con cần phải được báo tín hiệu sau khi thông tin về kích thước CTB (ví dụ log2_ctu_size_minus5) đã được báo tín hiệu.

37. subpic_treated_as_pic_flag[i] có thể không được báo tín hiệu đối với mỗi ảnh con. Thay vào đó, một subpic_treated_as_pic_flag được báo tín hiệu để điều khiển việc liệu ảnh con có được xử lý như ảnh đối với tất cả các ảnh con.

38. loop_filter_across_subpic_enabled_flag [i] có thể không được báo tín hiệu đối với mỗi ảnh con. Thay vào đó, một loop_filter_across_subpic_enabled_flag được báo tín hiệu để điều khiển việc liệu các bộ lọc vòng có thể được áp dụng ngang qua các ảnh con đối với tất cả các ảnh con.

39. subpic_treated_as_pic_flag[i] (subpic_treated_as_pic_flag) và/hoặc loop_filter_across_subpic_enabled_flag[i] (loop_filter_across_subpic_enabled_flag) có thể được báo tín hiệu một cách có điều kiện.

a. Trong một ví dụ, subpic_treated_as_pic_flag[i] và/hoặc loop_filter_across_subpic_enabled_flag[i] có thể không được báo tín hiệu nếu số lượng của các ảnh con là nhỏ hơn 2. (bằng 1).

40. RPR có thể được áp dụng khi các ảnh con được sử dụng.

a. Trong một ví dụ, tỷ lệ mở rộng trong RPR có thể bị ràng buộc phải là nhóm được giới hạn khi các ảnh con được sử dụng, chẳng hạn { 1:1, 1:2 và/hoặc 2:1}, hoặc { 1:1, 1:2 và/hoặc 2:1, 1:4 và/hoặc 4:1 }, { 1:1, 1:2 và/hoặc 2:1, 1:4 và/hoặc 4:1, 1:8 và/hoặc 8:1}.

b. Trong một ví dụ, kích thước CTB của ảnh A và kích thước CTB của ảnh B có thể khác nhau nếu độ phân giải của ảnh A và ảnh B là khác nhau.

c. Trong một ví dụ, giả sử ảnh con SA với các kích cỡ SAW×SAH là ở trong ảnh A và ảnh con SB với các kích cỡ SBW×SBH là ở trong ảnh B, SA tương ứng với SB, và các tỷ lệ mở rộng giữa ảnh A và ảnh B là Rw và Rh theo các hướng ngang và dọc, thì

i. SAW/SBW hoặc SBW/SAW cần phải bằng Rw.

ii. SAH/SBH hoặc SBH/SAH cần phải bằng Rh.

41. Khi các ảnh con được sử dụng (ví dụ sub_pics_present_flag là đúng), chỉ số ảnh con (hoặc Id ảnh con) có thể được báo tín hiệu trong tiêu đề lát, và địa chỉ lát bị gián đoạn dưới dạng địa chỉ trong ảnh con thay vì toàn bộ ảnh.

42. Yêu cầu rằng Id ảnh con của ảnh con thứ nhất là phải khác với Id ảnh con của ảnh con thứ hai, nếu ảnh con thứ nhất và ảnh con thứ hai không phải là cùng một ảnh con.

a. Trong một ví dụ, yêu cầu trong dòng bit tương thích là sps_subpic_id[i] là phải khác với sps_subpic_id[j], nếu i không bằng j.

b. Trong một ví dụ, yêu cầu trong dòng bit tương thích là pps_subpic_id[i] là phải khác với pps_subpic_id[j], nếu i không bằng j.

c. Trong một ví dụ, yêu cầu trong dòng bit tương thích là ph_subpic_id[i] là phải khác với ph_subpic_id[j], nếu i không bằng j.

d. Trong một ví dụ, yêu cầu trong dòng bit tương thích là SubpicIdList[i] là phải khác với SubpicIdList[j], nếu i không bằng j.

e. Trong một ví dụ, hiệu số được ký hiệu là D[i] bằng X_subpic_id[i]- X_subpic_id[i -P] có thể được báo tín hiệu.

i. Ví dụ, X có thể là sps, pps hoặc ph.

ii. Ví dụ, P là bằng 1.

iii. Ví dụ, i > P.

iv. Ví dụ, D[i] phải lớn hơn 0.

v. Ví dụ, D[i]-1 có thể được báo tín hiệu.

43. Đề xuất rằng chiều dài của phần tử cú pháp chỉ ra vị trí ngang hoặc dọc của CTU trên cùng bên trái (ví dụ **subpic_ctu_top_left_x** hoặc **subpic_ctu_top_left_y)** có thể được rút ra bằng Ceil(Log2(SS)) bit, trong đó SS là phải lớn hơn 0. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

a. Trong một ví dụ, SS = (pic_width_max_in_luma_samples + RR) / CtbSizeY khi phần tử cú pháp chỉ ra vị trí ngang của CTU trên cùng bên trái (ví dụ **subpic_ctu_top_left_x).**

b. Trong một ví dụ, SS = (pic_height_max_in_luma_samples + RR) / CtbSizeY khi phần tử cú pháp chỉ ra vị trí dọc của CTU trên cùng bên trái (ví dụ **subpic_ctu_top_left_y).**

c. Trong một ví dụ, RR là số nguyên khác không chẳng hạn CtbSizeY-1.

44. Đề xuất rằng chiều dài của phần tử cú pháp chỉ ra vị trí ngang hoặc dọc của CTU trên cùng bên trái của ảnh con (ví dụ **subpic_ctu_top_left_x** hoặc **subpic_ctu_top_left_y)** có thể được rút ra bằng Ceil(Log2(SS)) bit, trong đó SS là phải lớn hơn 0. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

a. Trong một ví dụ, SS = (pic_width_max_in_luma_samples + RR) / CtbSizeY khi phần tử cú pháp chỉ ra vị trí ngang của CTU trên cùng bên trái của ảnh con (ví dụ **subpic_ctu_top_left_x).**

b. Trong một ví dụ, SS = (pic_height_max_in_luma_samples + RR) / CtbSizeY khi phần

tử cú pháp chỉ ra vị trí dọc của CTU trên cùng bên trái của ảnh con (ví dụ **subpic_ctu_top_left_y).**

c. Trong một ví dụ, RR là số nguyên khác không chẳng hạn CtbSizeY-1.

45. Đề xuất rằng giá trị mặc định của chiều dài của phần tử cú pháp (mà có thể cộng dịch vị P chẳng hạn 1) chỉ ra chiều rộng hoặc chiều cao của ảnh con (ví dụ **subpic_width_minus1** hoặc **subpic_height_minus1)** có thể được rút ra bằng Ceil(Log2(SS))-P, trong đó SS là phải lớn hơn 0. Ở đây, hàm số Ceil() trở lại giá trị nguyên nhỏ nhất mà lớn hơn hoặc bằng giá trị đầu vào.

a. Trong một ví dụ, SS = (pic_width_max_in_luma_samples + RR) / CtbSizeY khi phần tử cú pháp chỉ ra chiều rộng mặc định (mà có thể cộng dịch vị P) của ảnh con (ví dụ **subpic_width_minus1).**

b. Trong một ví dụ, SS = (pic_height_max_in_luma_samples + RR) / CtbSizeY khi phần tử cú pháp chỉ ra chiều cao mặc định (mà có thể cộng dịch vị P) của ảnh con (ví dụ **subpic_height_minus1).**

c. Trong một ví dụ, RR là số nguyên khác không chẳng hạn CtbSizeY-1.

46. Đề xuất rằng, thông tin của các ID của các ảnh con cần phải được báo tín hiệu ít nhất trong một trong số SPS, PPS, và tiêu đề ảnh nếu xác định được rằng thông tin này cần phải được báo tín hiệu.

a. Trong một ví dụ, yêu cầu trong dòng bit tương thích là ít nhất một trong số sps_subpic_id_signalling_present_flag, pps_subpic_id_signalling_present_flag và ph_subpic_id_signalling_present_flag cần phải bằng 1 nếu sps_subpic_id_present_flag là bằng 1.

47. Đề xuất rằng, nếu thông tin của các ID của các ảnh con không được báo tín hiệu trong một tín hiệu bất kỳ trong số SPS, PPS, và tiêu

đề ảnh, nhưng xác định được rằng thông tin này cần phải được báo tín hiệu thì các ID mặc định cần phải được gán.

a. Trong một ví dụ, nếu tất cả sps_subpic_id_signalling_present_flag, pps_subpic_id_signalling_present_flag và ph_subpic_id_signalling_present_flag đều bằng 0 và sps_subpic_id_present_flag là bằng 1, thì SubpicIdList[i] phải được thiết lập bằng i+P, trong đó P là dịch vị chẳng hạn 0. Phần mô tả ví dụ là như dưới đây:

for(i = 0; i <= sps_num_subpics_minus1; i++)
SubpicIdList[i] = sps_subpic_id_present_flag ?
(sps_subpic_id_signalling_present_flag ? sps_subpic_id[i] :
(ph_subpic_id_signalling_present_flag ? ph_subpic_id[i] :
(pps_subpic_id_signalling_present_flag ? pps_subpic_id[i]: i))) : i

48. Đề xuất rằng thông tin của các Id ảnh con không được báo tín hiệu trong tiêu đề ảnh nếu chúng được báo tín hiệu trong PPS tương ứng.

a. Thiết kế cú pháp ví dụ là như dưới đây,

picture_header_rbsp() {	**Mô tả**
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag && ! pps_subpic_id_signalling_flag) {	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
...	

b. Trong một ví dụ, các Id ảnh con được thiết lập theo thông tin của các Id ảnh con được báo tín hiệu trong SPS nếu chúng

được báo tín hiệu trong SPS; Ngược lại, các Id ảnh con được thiết lập theo thông tin của các Id ảnh con được báo tín hiệu trong PPS nếu chúng được báo tín hiệu trong PPS, Ngược lại, các Id ảnh con được thiết lập theo thông tin của các Id ảnh con được báo tín hiệu trong tiêu đề ảnh nếu chúng được báo tín hiệu trong tiêu đề ảnh. Phần mô tả ví dụ là như dưới đây,

```
for(i = 0; i <= sps_num_subpics_minus1; i++)
SubpicIdList[i] = sps_subpic_id_present_flag ?
    (sps_subpic_id_signalling_present_flag ? sps_subpic_id[i] :
    (pps_subpic_id_signalling_present_flag? pps_subpic_id[i]:
(ph_subpic_id_signalling_present_flag? ph_subpic_id[i]: i ))) : i
```

c. Trong một ví dụ, các Id ảnh con được thiết lập theo thông tin của các Id ảnh con được báo tín hiệu trong tiêu đề ảnh nếu chúng được báo tín hiệu trong tiêu đề ảnh; Ngược lại, các Id ảnh con được thiết lập theo thông tin của các Id ảnh con được báo tín hiệu trong PPS nếu chúng được báo tín hiệu trong PPS, Ngược lại, các Id ảnh con được thiết lập theo thông tin của các Id ảnh con được báo tín hiệu trong SPS nếu chúng được báo tín hiệu trong SPS. Phần mô tả ví dụ là như dưới đây,

```
for(i = 0; i <= sps_num_subpics_minus1; i++)
SubpicIdList[i] = sps_subpic_id_present_flag ?
    (ph_subpic_id_signalling_present_flag ? ph_subpic_id[i] :
    (pps_subpic_id_signalling_present_flag? pps_subpic_id[i]:
(sps_subpic_id_signalling_present_flag? sps_subpic_id[i]: i ))) : i
```

49. Đề xuất rằng quá trình giải khối trên mép E phải phụ thuộc vào việc xác định xem liệu việc lọc vòng có được cho phép ngang qua các đường biên ảnh con (ví dụ được xác định bởi loop_filter_across_subpic_enabled_flag) trên cả hai phía (được ký hiệu là phía P và phía Q) của mép. Phía P biểu diễn phía trong khối hiện tại, và phía Q biểu diễn phía trong khối lân cận, mà thuộc về ảnh con khác nhau. Trong phần mô tả sau đây, giả sử rằng phía P và phía Q thuộc về

hai ảnh con khác nhau. loop_filter_across_subpic_enabled_flag[P] =0/1 có nghĩa là việc lọc vòng không được cho phép/được cho phép ngang qua các đường biên ảnh con của ảnh con chứa phía P. loop_filter_across_subpic_enabled_flag[Q] =0/1 có nghĩa là việc lọc vòng không được cho phép/được cho phép ngang qua các đường biên ảnh con của ảnh con chứa phía Q.

a. Trong một ví dụ, E không được lọc nếu loop_filter_across_subpic_enabled_flag[P] là bằng 0 hoặc loop_filter_across_subpic_enabled_flag[Q] là bằng 0.

b. Trong một ví dụ, E không được lọc nếu loop_filter_across_subpic_enabled_flag[P] là bằng 0 và loop_filter_across_subpic_enabled_flag[Q] là bằng 0.

c. Trong một ví dụ, việc liệu có lọc hai phía của E hay không được điều khiển riêng rẽ.

i. Ví dụ, phía P của E được lọc nếu và chỉ nếu loop_filter_across_subpic_enabled_flag[P] là bằng 1.

ii. Ví dụ, phía Q của E được lọc nếu và chỉ nếu loop_filter_across_subpic_enabled_flag[Q] là bằng 1.

50. Đề xuất rằng, việc báo tín hiệu/phân tách của phần tử cú pháp SE trong PPS chỉ ra kích thước khối tối đa được sử dụng để bỏ qua chuyển đổi (chẳng hạn log2_transform_skip_max_size_minus2) cần phải được tách rời khỏi phần tử cú pháp bất kỳ trong SPS (chẳng hạn sps_transform_skip_enabled_flag).

a. Sự thay đổi cú pháp ví dụ là như dưới đây:

pic_parameter_set_rbsp() {	
...	
[[if(sps_transform_skip_enabled_flag)]]	
log2_transform_skip_max_size_minus2	ue(v)
...	

b. Theo cách khác, SE có thể được báo tín hiệu trong SPS, chẳng hạn:

seq_parameter_set_rbsp() {	
...	
if(sps_transform_skip_enabled_flag)	

log2_transform_skip_max_size_minus2	ue(v)
...	

c. Theo cách khác, SE có thể được báo tín hiệu trong tiêu đề ảnh, chẳng hạn:

picture_header_rbsp() {	
...	
if(sps_transform_skip_enabled_flag)	
log2_transform_skip_max_size_minus2	ue(v)
...	

51. Việc liệu có cập nhật và/hoặc cách thức cập nhật bảng HMVP (hoặc được gọi là danh sách/lưu trữ/ánh xạ v.v.) sau khi giải mã khối thứ nhất có thể phụ thuộc vào việc liệu khối thứ nhất có được mã hóa bằng GEO hay không.

a. Trong một ví dụ, bảng HMVP có thể không được cập nhật sau khi giải mã khối thứ nhất nếu khối thứ nhất được mã hóa bằng GEO.

b. Trong một ví dụ, bảng HMVP có thể được cập nhật sau khi giải mã khối thứ nhất nếu khối thứ nhất được mã hóa bằng GEO.

i. Trong một ví dụ, bảng HMVP có thể được cập nhật với thông tin chuyển động của một phân chia được chia cho GEO.

ii. Trong một ví dụ, bảng HMVP có thể được cập nhật với thông tin chuyển động của nhiều phân chia được chia cho GEO

52. Trong CC-ALF, các mẫu độ sáng ra ngoài đơn vị xử lý hiện tại (ví dụ, đơn vị xử lý ALF được giới hạn bởi hai đường biên ảo ALF) được loại trừ khỏi việc lọc trên các mẫu sắc độ trong đơn vị xử lý tương ứng.

a. Các mẫu độ sáng được đệm ra ngoài đơn vị xử lý hiện tại có thể được sử dụng để lọc các mẫu sắc độ trong đơn vị xử lý tương ứng.

i. Phương pháp đệm bất kỳ được bộc lộ trong phần mô tả này có thể được sử dụng để đệm các mẫu độ sáng.

b. Theo cách khác, các mẫu độ sáng ra ngoài đơn vị xử lý hiện tại có thể được sử dụng để lọc các mẫu sắc độ trong đơn vị xử lý tương ứng.

5. Các phương án

Trong các phương án sau đây, các đoạn văn bản mới được bổ sung được in đậm, nghiêng và các đoạn văn bản bị xóa được đánh dấu bằng "[[]]".

5.1 Phương án 1: điều kiện ràng buộc ảnh con về các ứng viên hợp nhất xây dựng afin

8.5.5.6 Quá trình dẫn xuất đối với các ứng viên hợp nhất vectơ chuyển động điểm điều khiển afin được xây dựng

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCb, yCb) chỉ ra mẫu trên cùng bên trái của khối mã hóa độ sáng hiện tại so với mẫu độ sáng trên cùng bên trái của ảnh hiện tại,

– hai biến số cbWidth và cbHeight chỉ ra chiều rộng và chiều cao của khối mã hóa độ sáng hiện tại,

– các cờ khả dụng $availableA_0$, $availableA_1$, $availableA_2$, $availableB_0$, $availableB_1$, $availableB_2$, $availableB_3$,

– các vị trí mẫu ($xNbA_0$, $yNbA_0$), ($xNbA_1$, $yNbA_1$), ($xNbA_2$, $yNbA_2$), ($xNbB_0$, $yNbB_0$), ($xNbB_1$, $yNbB_1$), ($xNbB_2$, $yNbB_2$) và ($xNbB_3$, $yNbB_3$).

Đầu ra của quá trình này là:

– cờ khả dụng của các ứng viên hợp nhất vectơ chuyển động điểm điều khiển afin được xây dựng availableFlagConstK, với K = 1..6,

– các chỉ số tham chiếu refIdxLXConstK, với K = 1..6, X bằng 0 hoặc 1,

– các cờ sử dụng danh sách dự báo predFlagLXConstK, với K = 1..6, X bằng 0 hoặc 1,

– các chỉ số mô hình chuyển động afin MotionModelIdcConstK, với K = 1..6,

– các chỉ số trọng số dự báo đôi bcwIdxConstK, với K = 1..6,

– các vectơ chuyển động điểm điều khiển afin được xây dựng cpMvLXConstK[cpIdx] với cpIdx = 0..2, K = 1..6 và X bằng 0 hoặc 1.

...

Vectơ chuyển động điểm điều khiển (được sắp xếp dưới cùng bên phải) thứ tư cpMvLXCorner[3], chỉ số tham chiếu refIdxLXCorner[3], cờ sử dụng danh sách dự báo predFlagLXCorner[3] và cờ khả dụng availableFlagCorner[3] với X bằng 0 và 1 được rút ra như sau:

– Các chỉ số tham chiếu đối với ứng viên hợp nhất theo thời gian, refIdxLXCorner[3], với X bằng 0 hoặc 1, được thiết lập bằng 0.

– Các biến số mvLXCol và availableFlagLXCol, với X bằng 0 hoặc 1, được rút ra như sau:

– Nếu slice_temporal_mvp_enabled_flag là bằng 0, cả hai thành phần của mvLXCol được thiết lập bằng 0 và availableFlagLXCol được thiết lập bằng 0.

– Ngược lại (slice_temporal_mvp_enabled_flag là bằng 1), các điều sau đây được áp dụng:

xColBr = xCb + cbWidth(8-601)

yColBr = yCb + cbHeight(8-602)

rightBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos : pic_width_in_luma_samples – 1

botBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos : pic_height_in_luma_samples – 1

– Nếu yCb >> CtbLog2SizeY là bằng yColBr >> CtbLog2SizeY, ***yColBr nhỏ hơn hoặc bằng botBoundaryPos và xColBr nhỏ hơn hoặc bằng rightBoundaryPos, các điều sau đây được áp dụng***:

– Biến số colCb chỉ ra khối mã hóa độ sáng bao phủ vị trí được cải biến được cho bởi

((xColBr >> 3) << 3, (yColBr >> 3) << 3) bên trong ảnh được sắp xếp được chỉ ra bởi ColPic.

– Vị trí độ sáng (xColCb, yColCb) được thiết lập bằng mẫu trên cùng bên trái của khối mã hóa độ sáng được sắp xếp được chỉ ra bởi colCb so với mẫu độ sáng trên cùng bên trái của ảnh được sắp xếp được chỉ ra bởi ColPic.

– Quá trình dẫn xuất đối với các vectơ chuyển động được sắp xếp như được chỉ ra trong mục 8.5.2.12 được gọi ra với currCb, colCb, (xColCb, yColCb), refIdxLXCorner[3] và sbFlag được thiết lập bằng 0 làm các đầu vào, và đầu ra được gán cho mvLXCol và availableFlagLXCol.

– Ngược lại, cả hai thành phần của mvLXCol được thiết lập bằng 0 và availableFlagLXCol được thiết lập bằng 0.

...

5.2 Phương án 2: điều kiện ràng buộc ảnh con về các ứng viên hợp nhất xây dựng afin

8.5.5.6 Quá trình dẫn xuất đối với các ứng viên hợp nhất vectơ chuyển động điểm điều khiển afin được xây dựng

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCb, yCb) chỉ ra mẫu trên cùng bên trái của khối mã hóa độ sáng hiện tại so với mẫu độ sáng trên cùng bên trái của ảnh hiện tại,

– hai biến số cbWidth và cbHeight chỉ ra chiều rộng và chiều cao của khối mã hóa độ sáng hiện tại,

– các cờ khả dụng availableA$_0$, availableA$_1$, availableA$_2$, availableB$_0$, availableB$_1$, availableB$_2$, availableB$_3$,

– các vị trí mẫu (xNbA$_0$, yNbA$_0$), (xNbA$_1$, yNbA$_1$), (xNbA$_2$, yNbA$_2$), (xNbB$_0$, yNbB$_0$), (xNbB$_1$, yNbB$_1$), (xNbB$_2$, yNbB$_2$) và (xNbB$_3$, yNbB$_3$).

Đầu ra của quá trình này là:

– cờ khả dụng của các ứng viên hợp nhất vectơ chuyển động điểm điều khiển afin được xây dựng availableFlagConstK, với K = 1..6,

– các chỉ số tham chiếu refIdxLXConstK, với K = 1..6, X bằng 0 hoặc 1,

– các cờ sử dụng danh sách dự báo predFlagLXConstK, với K = 1..6, X bằng 0 hoặc 1,

– các chỉ số mô hình chuyển động afin MotionModelIdcConstK, với K = 1..6,

– các chỉ số trọng số dự báo đôi bcwIdxConstK, với K = 1..6,

– các vectơ chuyển động điểm điều khiển afin được xây dựng cpMvLXConstK[cpIdx] với cpIdx = 0..2, K = 1..6 và X bằng 0 hoặc 1.

...

Vectơ chuyển động điểm điều khiển (được sắp xếp dưới cùng bên phải) thứ tư cpMvLXCorner[3], chỉ số tham chiếu refIdxLXCorner[3], cờ sử dụng danh sách dự báo predFlagLXCorner[3] và cờ khả dụng availableFlagCorner[3] với X bằng 0 và 1 được rút ra như sau:

– Các chỉ số tham chiếu đối với ứng viên hợp nhất theo thời gian, refIdxLXCorner[3], với X bằng 0 hoặc 1, được thiết lập bằng 0.

– Các biến số mvLXCol và availableFlagLXCol, với X bằng 0 hoặc 1, được rút ra như sau:

– Nếu slice_temporal_mvp_enabled_flag là bằng 0, cả hai thành phần của mvLXCol được thiết lập bằng 0 và availableFlagLXCol được thiết lập bằng 0.

– Ngược lại (slice_temporal_mvp_enabled_flag là bằng 1), các điều sau đây được áp dụng:

xColBr = xCb + cbWidth (8-601)

yColBr = yCb + cbHeight (8-602)

rightBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos : pic_width_in_luma_samples − 1

botBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos : pic_height_in_luma_samples − 1

xColBr = Min(rightBoundaryPos , xColBr)

yColBr = Min (botBoundaryPos , yColBr)

– Nếu yCb >> CtbLog2SizeY là bằng yColBr >> CtbLog2SizeY, [[yColBr nhỏ hơn pic_height_in_luma_samples và xColBr nhỏ hơn pic_width_in_luma_samples, các điều sau đây được áp dụng]]:

– Biến số colCb chỉ ra khối mã hóa độ sáng bao phủ vị trí được cải biến được cho bởi ((xColBr >> 3) << 3, (yColBr >> 3) << 3) bên trong ảnh được sắp xếp được chỉ ra bởi ColPic.

– Vị trí độ sáng (xColCb, yColCb) được thiết lập bằng mẫu trên cùng bên trái của khối mã hóa độ sáng được sắp xếp được chỉ ra bởi colCb so với mẫu độ sáng trên cùng bên trái của ảnh được sắp xếp được chỉ ra bởi ColPic.

– Quá trình dẫn xuất đối với các vectơ chuyển động được sắp xếp như được chỉ ra trong mục 8.5.2.12 được gọi ra với currCb, colCb, (xColCb, yColCb), refIdxLXCorner[3] và sbFlag được thiết lập bằng 0 làm các đầu vào, và đầu ra được gán cho mvLXCol và availableFlagLXCol.

– Ngược lại, cả hai thành phần của mvLXCol được thiết lập bằng 0 và availableFlagLXCol được thiết lập bằng 0.

...

5.3 Phương án 3: Tìm nạp các mẫu nguyên theo điều kiện ràng buộc ảnh con

8.5.6.3.3 Quá trình tìm nạp mẫu nguyên độ sáng

Các đầu vào của quá trình này là:

– vị trí độ sáng trong các đơn vị mẫu đủ ($xInt_L$, $yInt_L$),

– mảng mẫu tham chiếu độ sáng $refPicLX_L$,

Đầu ra của quá trình này là giá trị mẫu độ sáng được dự báo $predSampleLX_L$

Biến số dịch chuyển được thiết lập bằng Max(2, 14 − $BitDepth_Y$).

Biến số picW được thiết lập bằng pic_width_in_luma_samples và biến số picH được thiết lập bằng pic_height_in_luma_samples.

Các vị trí độ sáng trong các đơn vị mẫu đủ (xInt, yInt) được rút ra như sau:

– If subpic_treated_as_pic_flag[SubPicIdx] là bằng 1, các điều sau đây được áp dụng:

xInt = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xInt)

yInt = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yInt)

– Ngược lại:

xInt = Clip3(0, picW − 1, sps_ref_wraparound_enabled_flag ? (8-782)

ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW , $xInt_L$) : $xInt_L$)

yInt = Clip3(0, picH − 1, $yInt_L$)(8-783)

Giá trị mẫu độ sáng được dự báo $predSampleLX_L$ được rút ra như sau:

$predSampleLX_L$ = $refPicLX_L$[xInt][yInt] << shift3 (8-784)

5.4 Phương án 4: dẫn xuất biến số invAvgLuma trong định tỷ lệ phần dư sắc độ của LMCS

8.7.5.3 Tái tạo ảnh với quá trình định tỷ lệ phần dư sắc độ phụ thuộc độ sáng đối với các mẫu sắc độ

Các đầu vào của quá trình này là:

– vị trí sắc độ (xCurr, yCurr) của mẫu sắc độ trên cùng bên trái của khối chuyển đổi sắc độ hiện tại so với mẫu sắc độ trên cùng bên trái của ảnh hiện tại,

– biến số nCurrSw chỉ ra chiều rộng khối chuyển đổi sắc độ,

– biến số nCurrSh chỉ ra chiều cao khối chuyển đổi sắc độ,

– biến số tuCbfChroma chỉ ra cờ khối được mã hóa của khối chuyển đổi sắc độ hiện tại,

– mảng (nCurrSw)x(nCurrSh) predSamples chỉ ra các mẫu dự báo sắc độ của khối hiện tại,

– mảng (nCurrSw)x(nCurrSh) resSamples chỉ ra các mẫu phần dư sắc độ của khối hiện tại,

Đầu ra của quá trình này là mảng mẫu ảnh sắc độ được tái tạo recSamples.

Biến số sizeY được thiết lập bằng Min(CtbSizeY, 64).

Mẫu ảnh sắc độ được tái tạo recSamples được rút ra như sau đối với i = 0..nCurrSw − 1, j = 0..nCurrSh − 1:

– …

– Ngược lại, các điều sau đây được áp dụng:

– …

– Biến số currPic chỉ ra mảng của các mẫu độ sáng được tái tạo trong ảnh hiện tại.

– Để dẫn xuất biến số varScale, các bước được xếp thứ tự sau đây được áp dụng:

1. Biến số invAvgLuma được rút ra như sau:

–Mảng recLuma[i] với i=0..(2 * sizeY − 1) và biến số cnt được rút ra như sau:

– Biến số cnt được thiết lập bằng 0.

– ***Biến số rightBoundaryPos và botBoundaryPos được rút ra như sau:***

rightBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos : pic_width_in_luma_samples − 1

botBoundaryPos = subpic_treated_as_pic_flag[SubPicIdx] ?

SubPicBotBoundaryPos : pic_height_in_luma_samples – 1

– Khi availL là bằng TRUE, mảng recLuma[i] với i = 0..sizeY − 1 được thiết lập bằng currPic[xCuCb − 1][Min(yCuCb + i, [[pic_height_in_luma_samples – 1]] ***botBoundaryPos***)] với i = 0..sizeY − 1, và cnt được thiết lập bằng sizeY

– Khi availT là bằng TRUE, mảng recLuma[cnt + i] với i = 0..sizeY – 1 được thiết lập bằng

currPic[Min(xCuCb + i, [[pic_width_in_luma_samples − 1]] ***rightBoundaryPos***)][yCuCb − 1] với i = 0..sizeY − 1, và cnt được thiết lập bằng (cnt + sizeY)

–Biến số invAvgLuma được rút ra như sau:

– Nếu cnt là lớn hơn 0, các điều sau đây được áp dụng:

invAvgLuma = $\text{Clip1}_Y((\sum_{k=0}^{cnt-1} \text{recLuma}[k] + (cnt >> 1)) >> \text{Log2}(cnt))$(8-1013)

– Ngược lại (cnt là bằng 0), các điều sau đây được áp dụng:

invAvgLuma = 1 << (BitDepth_Y – 1)(8-1014)

5.5 Phương án 5: ví dụ về xác định phần tử ảnh con trong đơn vị của N (chẳng hạn N=8 hoặc 32) khác với 4 mẫu

7.4.3.3 Các ngữ nghĩa tập thông số dãy RBSP

subpic_grid_col_width_minus1 plus 1 chỉ ra chiều rộng của mỗi phần tử của lưới bộ nhận dạng ảnh con trong các đơn vị của ***4-N*** mẫu. Chiều dài của phần tử cú pháp này là Ceil(Log2(pic_width_max_in_luma_samples / ***4-N***)) bit.

Biến số NumSubPicGridCols được rút ra như sau:

NumSubPicGridCols = (pic_width_max_in_luma_samples + subpic_grid_col_width_minus1 *[[4+ 3]] ***N*** ***+N-1***) / (subpic_grid_col_width_minus1 * [[4+ 3]] ***N + N-1***) (7-5)

subpic_grid_row_height_minus1 plus 1 chỉ ra chiều cao của mỗi phần tử của lưới bộ nhận dạng ảnh con trong các đơn vị của 4 mẫu. Chiều dài của phần tử cú pháp này là Ceil(Log2(pic_height_max_in_luma_samples / ***4-N***)) bit.

Biến số NumSubPicGridRows được rút ra như sau:

NumSubPicGridRows = (pic_height_max_in_luma_samples + subpic_grid_row_height_minus1 * ***4 N + N-1***) / (subpic_grid_row_height_minus1 *[[4+ 3] ***N + N-1***)

7.4.7.1 Các ngữ nghĩa tiêu đề lát tổng quát

Các biến số SubPicIdx, SubPicLeftBoundaryPos, SubPicTopBoundaryPos, SubPicRightBoundaryPos, và SubPicBotBoundaryPos được rút ra như sau:

SubPicIdx =
CtbToSubPicIdx[CtbAddrBsToRs[FirstCtbAddrBs[SliceBrickIdx[0]]]]
if(subpic_treated_as_pic_flag[SubPicIdx]) {
SubPicLeftBoundaryPos =
SubPicLeft[SubPicIdx] * (subpic_grid_col_width_minus1 + 1) * ***4 N***
SubPicRightBoundaryPos =
(SubPicLeft[SubPicIdx] + SubPicWidth[SubPicIdx]) *
(subpic_grid_col_width_minus1 + 1) * ***4 N*** (7-93)
SubPicTopBoundaryPos =
SubPicTop[SubPicIdx] * (subpic_grid_row_height_minus1 + 1)* ***4 N***
SubPicBotBoundaryPos =
(SubPicTop[SubPicIdx] + SubPicHeight[SubPicIdx]) *
(subpic_grid_row_height_minus1 + 1) * ***4N***
}

5.6 Phương án 6: giới hạn chiều rộng ảnh và chiều cao ảnh phải bằng hoặc lớn hơn 8

7.4.3.3 Các ngữ nghĩa tập thông số dãy RBSP

pic_width_max_in_luma_samples chỉ ra chiều rộng tối đa, trong các đơn vị của các mẫu độ sáng, của mỗi ảnh được giải mã tham chiếu tới SPS. pic_width_max_in_luma_samples sẽ không bằng 0 và sẽ là bội số nguyên của [[MinCbSizeY]] ***Max(8, MinCbSizeY)***.

pic_height_max_in_luma_samples chỉ ra chiều cao tối đa, trong các đơn vị của các mẫu độ sáng, của mỗi ảnh được giải mã tham chiếu tới SPS. pic_height_max_in_luma_samples sẽ không bằng 0 và sẽ bội số nguyên của [[MinCbSizeY]] ***Max(8, MinCbSizeY)***.

5.7 Phương án 7: kiểm tra biên ảnh con để tách BT/TT/QT, dẫn xuất sâu BT/TT/QT, và/hoặc báo tín hiệu cờ tách CU

6.4.2 Quá trình tách nhị phân được cho phép

Biến số allowBtSplit được rút ra như sau:

– ...

– Ngược lại, nếu tất cả các điều kiện sau đây là đúng, allowBtSplit được thiết lập bằng FALSE

– btSplit là bằng SPLIT_BT_VER

– y0 + cbHeight là lớn hơn [[pic_height_in_luma_samples]]***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***.

– Ngược lại, nếu tất cả các điều kiện sau đây là đúng, allowBtSplit được thiết lập bằng FALSE

– btSplit là bằng SPLIT_BT_VER

– cbHeight là lớn hơn MaxTbSizeY

– x0 + cbWidth là lớn hơn [[pic_width_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples***

– Ngược lại, nếu tất cả các điều kiện sau đây là đúng, allowBtSplit được thiết lập bằng FALSE

– btSplit là bằng SPLIT_BT_HOR

– cbWidth là lớn hơn MaxTbSizeY

– y0 + cbHeight là lớn hơn [[pic_height_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples.***

– Ngược lại, nếu tất cả các điều kiện sau đây là đúng, allowBtSplit được thiết lập bằng FALSE

– x0 + cbWidth là lớn hơn [[pic_width_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples***

– y0 + cbHeight là lớn hơn [[pic_height_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***.

– cbWidth là lớn hơn minQtSize

– Ngược lại, nếu tất cả các điều kiện sau đây là đúng, allowBtSplit được thiết lập bằng FALSE

– btSplit là bằng SPLIT_BT_HOR

– x0 + cbWidth là lớn hơn [[pic_width_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples***

– y0 + cbHeight nhỏ hơn hoặc bằng [[pic_height_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples.***

6.4.2 Quá trình tách tam phân được cho phép

Biến số allowTtSplit được rút ra như sau:

– Nếu một hoặc nhiều trong số các điều kiện sau đây là đúng, allowTtSplit được thiết lập bằng FALSE:

– CbSize nhỏ hơn hoặc bằng 2 * MinTtSizeY

– cbWidth là lớn hơn Min(MaxTbSizeY, maxTtSize)

– cbHeight là lớn hơn Min(MaxTbSizeY, maxTtSize)

– mttDepth là lớn hơn hoặc bằng maxMttDepth

– x0 + cbWidth là lớn hơn [[pic_width_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples***

– y0 + cbHeight là lớn hơn [[pic_height_in_luma_samples]] ***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***.

– treeType là bằng DUAL_TREE_CHROMA và (cbWidth / SubWidthC) * (cbHeight / SubHeightC) nhỏ hơn hoặc bằng 32

– treeType là bằng DUAL_TREE_CHROMA và modeType là bằng MODE_TYPE_INTRA

–Ngược lại, allowTtSplit được thiết lập bằng TRUE.

7.3.8.2 Cú pháp đơn vị cây mã hóa

dual_tree_implicit_qt_split(x0, y0, CbSize, cqtDepth) {	Mô tả
...	
if(x1 < [[pic_width_in_luma_samples]] ***(subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples)***)	
dual_tree_implicit_qt_split(x1, y0, CbSize / 2, cqtDepth + 1)	
if(y1 < [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***))	
dual_tree_implicit_qt_split(x0, y1, CbSize / 2, cqtDepth + 1)	
if(x1 < [[pic_width_in_luma_samples]] ***(subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples)*** && y1 < [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***))	
dual_tree_implicit_qt_split(x1, y1, CbSize / 2, cqtDepth + 1)	
} else {	
...	
}	
}	

7.3.8.4 Cú pháp cây mã hóa

coding_tree(x0, y0, cbWidth, cbHeight, qgOnY, qgOnC, cbSubdiv, cqtDepth, mttDepth, depthOffset, partIdx, treeTypeCurr, modeTypeCurr) {	**Descriptor**
if((allowSplitBtVer || allowSplitBtHor || allowSplitTtVer || allowSplitTtHor || allowSplitQT) &&(x0 + cbWidth <= [[pic_width_in_luma_samples]] ***(subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples)***) && (y0 + cbHeight <= [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***)))	
split_cu_flag	ae(v)
if(cu_qp_delta_enabled_flag && qgOnY && cbSubdiv <= cu_qp_delta_subdiv) {	
…	
depthOffset += (y0 + cbHeight > [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***)) ? 1 : 0	
y1 = y0 + (cbHeight / 2)	
coding_tree(x0, y0, cbWidth, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 0, treeType, modeType)	
if(y1 < [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***))	
coding_tree(x0, y1, cbWidth, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 1, cqtDepth, mttDepth + 1, depthOffset, 1, treeType, modeType)	
…	
if(x1 < [[pic_width_in_luma_samples]] ***(subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples)***)	
coding_tree(x1, y0, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 1, treeType, modeType)	
if(y1 < [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***))	
coding_tree(x0, y1, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 2, treeType, modeType)	
if(y1 < [[pic_height_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicBotBoundaryPos + 1: pic_height_in_luma_samples***) && x1 < [[pic_width_in_luma_samples]] (***subpic_treated_as_pic_flag[SubPicIdx] ? SubPicRightBoundaryPos + 1 : pic_width_in_luma_samples***))	
coding_tree(x1, y1, cbWidth / 2, cbHeight / 2, qgOnY, qgOnC, cbSubdiv + 2, cqtDepth + 1, 0, 0, 3, treeType, modeType)	

5.8 Phương án 8: ví dụ về xác định các ảnh con

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
...	
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
[[subpics_present_flag	u(1)
if(subpics_present_flag) {	
max_subpics_minus1	u(8)
subpic_grid_col_width_minus1	u(v)
subpic_grid_row_height_minus1	u(v)
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
subpic_grid_idx[i][j]	u(v)
for(i = 0; i <= NumSubPics; i++) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}]]	
bit_depth_luma_minus8	ue(v)
...	
log2_ctu_size_minus5	u(2)
...	
subpics_present_flag	***u(1)***
if(subpics_present_flag) {	
num_subpics_minus1	***u(8)***
for(i = 0; i <= num_subpics_minus1; i++) {	
subpic_ctb_addr_x[i]	***u(8)***
subpic_ctb_addr_y[i]	***u(8)***
subpic_ctb_width_minus1[i]	***u(8)***
subpic_ctb_height_minus1[i]	***u(8)***
subpic_treated_as_pic_flag[i]	***u(1)***
loop_filter_across_subpic_enabled_flag[i]	***u(1)***
}	
...	

5.9 Phương án 9: ví dụ về xác định các ảnh con

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
...	
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
[[subpics_present_flag	u(1)
if(subpics_present_flag) {	
max_subpics_minus1	u(8)
subpic_grid_col_width_minus1	u(v)
subpic_grid_row_height_minus1	u(v)
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
subpic_grid_idx[i][j]	u(v)

for(i = 0; i <= NumSubPics; i++) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}]]	
bit_depth_luma_minus8	ue(v)
...	
log2_ctu_size_minus5	u(2)
...	
subpics_present_flag	*u(1)*
if(subpics_present_flag) {	
num_subpics_minus1	*ue(v)*
for(i = 0; i <= num_subpics_minus1; i++) {	
subpic_ctb_addr_x[i]	*u(8)*
subpic_ctb_addr_y[i]	*u(8)*
subpic_ctb_width_minus1[i]	*u(8)*
subpic_ctb_height_minus1[i]	*u(8)*
subpic_treated_as_pic_flag[i]	*u(1)*
loop_filter_across_subpic_enabled_flag[i]	*u(1)*
}	
...	

5.10 Phương án 10: ví dụ về xác định các ảnh con

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
...	
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
[[subpics_present_flag	u(1)
if(subpics_present_flag) {	
max_subpics_minus1	u(8)
subpic_grid_col_width_minus1	u(v)
subpic_grid_row_height_minus1	u(v)
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
subpic_grid_idx[i][j]	u(v)
for(i = 0; i <= NumSubPics; i++) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}]]	
...	
log2_ctu_size_minus5	u(2)
...	
subpics_present_flag	*u(1)*
if(subpics_present_flag) {	
num_subpics_minus2	*u(v)*
subpic_addr_x_length_minus1	*ue(v)*
subpic_addr_y_length_minus1	*ue(v)*
for(i = 0; i < NumSubPics; i++) {	
subpic_ctb_addr_x[i]	*u(v)*

subpic_ctb_addr_y[i]	*u(v)*
subpic_ctb_width_minus1[i]	*u(v)*
subpic_ctb_height_minus1[i]	*u(v)*
subpic_treated_as_pic_flag[i]	*u(1)*
loop_filter_across_subpic_enabled_flag[i]	*u(1)*
}	
...	

5.11 Phương án 11: ví dụ về xác định các ảnh con

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
...	
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
[[subpics_present_flag	u(1)
if(subpics_present_flag) {	
max_subpics_minus1	u(8)
subpic_grid_col_width_minus1	u(v)
subpic_grid_row_height_minus1	u(v)
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
subpic_grid_idx[i][j]	u(v)
for(i = 0; i <= NumSubPics; i++) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}]]	
...	
log2_ctu_size_minus5	u(2)
...	
subpics_present_flag	*u(1)*
if(subpics_present_flag) {	
num_subpics_minus2	*u(v)*
subpic_addr_x_length_minus1	*ue(v)*
subpic_addr_y_length_minus1	*ue(v)*
for(i = 0; i < NumSubPics; i++) {	
if(i < NumSubPics – 1){	
subpic_ctb_addr_x[i]	*u(v)*
subpic_ctb_addr_y[i]	*u(v)*
subpic_ctb_width_minus1[i]	*u(v)*
subpic_ctb_height_minus1[i]	*u(v)*
}	
subpic_treated_as_pic_flag[i]	*u(1)*
loop_filter_across_subpic_enabled_flag[i]	*u(1)*
}	
...	

NumSubPics= num_subpics_minus2+2.

5.12 Phương án: giải khối xem xét các ảnh con

Quá trình lọc giải khối

Tổng quan

Các đầu vào của quá trình này là ảnh được tái tạo trước khi giải khối, tức là, mảng $recPicture_L$ và, khi ChromaArrayType không bằng 0, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$.

Các đầu ra của quá trình này là ảnh được tái tạo được cải biến sau khi giải khối, tức là, mảng $recPicture_L$ và, khi ChromaArrayType không bằng 0, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$.

Các mép dọc trong ảnh được lọc đầu tiên. Sau đó các mép ngang trong ảnh được lọc với các mẫu được cải biến bởi quá trình lọc mép dọc làm đầu vào. Các mép dọc và ngang trong các CTB của mỗi CTU được xử lý riêng rẽ trên cơ sở đơn vị mã hóa. Các mép dọc của các khối mã hóa trong đơn vị mã hóa được lọc bắt đầu bằng mép ở phía tay trái của các khối mã hóa tiếp tục qua các mép hướng về phía tay phải của các khối mã hóa theo thứ tự hình học của chúng. Các mép ngang của các khối mã hóa trong đơn vị mã hóa được lọc bắt đầu bằng mép ở trên cùng của các khối mã hóa tiếp tục qua các mép hướng về đáy của các khối mã hóa theo thứ tự hình học của chúng.

Ghi chú: Mặc dù quá trình lọc được chỉ ra trên cơ sở ảnh trong phần mô tả này, quá trình lọc có thể được thực hiện trên cơ sở đơn vị mã hóa với kết quả tương đương, miễn là bộ giải mã tính đúng thứ tự phụ thuộc xử lý để đưa ra các giá trị đầu ra giống nhau.

Quá trình lọc giải khối được áp dụng cho tất cả các mép khối con mã hóa và các mép khối chuyển đổi của ảnh, ngoại trừ các mép thuộc các kiểu sau đây:

- Các mép mà ở biên của ảnh,
- [[Các mép mà trùng với các đường biên của ảnh con mà đối với nó loop_filter_across_subpic_enabled_flag[SubPicIdx] là bằng 0,]]
- Các mép mà trùng với các đường biên ảo của ảnh khi pps_loop_filter_across_virtual_boundaries_disabled_flag là bằng 1,
- Các mép mà trùng với các đường biên ô ảnh khi loop_filter_across_tiles_enabled_flag là bằng 0,

– Các mép mà trùng với các đường biên lát khi loop_filter_across_slices_enabled_flag là bằng 0,

– Các mép mà trùng với các đường biên trên hoặc trái của các lát với slice_deblocking_filter_disabled_flag bằng 1,

– Các mép bên trong các lát với slice_deblocking_filter_disabled_flag bằng 1,

– Các mép mà không tương ứng với các đường biên lưới mẫu 4x4 của thành phần độ sáng,

– Các mép mà không tương ứng với các đường biên lưới mẫu 8x8 của thành phần sắc độ,

– Các mép bên trong thành phần độ sáng mà đối với nó cả hai bên của mép có intra_bdpcm_luma_flag bằng 1,

– Các mép bên trong các thành phần sắc độ mà đối với nó cả hai bên của mép có intra_bdpcm_chroma_flag bằng 1,

– Các mép của các khối con sắc độ mà không phải là các mép của đơn vị chuyển đổi liên kết.

...

Quá trình lọc giải khối đối với một hướng

Các đầu vào của quá trình này là:

– biến số treeType chỉ ra liệu các thành phần độ sáng (DUAL_TREE_LUMA) hoặc các thành phần sắc độ (DUAL_TREE_CHROMA) có đang được xử lý,

– khi treeType là bằng DUAL_TREE_LUMA, ảnh được tái tạo trước khi giải khối, tức là, mảng $recPicture_L$,

– khi ChromaArrayType không bằng 0 và treeType là bằng DUAL_TREE_CHROMA, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$,

– biến số edgeType chỉ ra liệu mép dọc (EDGE_VER) hoặc mép ngang (EDGE_HOR) có được lọc hay không.

Các đầu ra của quá trình này là ảnh được tái tạo được cải biến sau giải khối, tức là:

– khi treeType là bằng DUAL_TREE_LUMA, mảng $recPicture_L$,

– khi ChromaArrayType không bằng 0 và treeType là bằng DUAL_TREE_CHROMA, các mảng $recPicture_{Cb}$ và $recPicture_{Cr}$.

Các biến số firstCompIdx và lastCompIdx được rút ra như sau:

firstCompIdx = (treeType = = DUAL_TREE_CHROMA) ? 1 : 0 (8-1010)

lastCompIdx = (treeType = = DUAL_TREE_LUMA || ChromaArrayType = = 0) ? 0 : 2 (8-1011)

Đối với mỗi đơn vị mã hóa và mỗi khối mã hóa trên thành phần màu của đơn vị mã hóa được chỉ báo bởi chỉ số thành phần màu cIdx nằm trong khoảng từ firstCompIdx tới lastCompIdx, bao gồm các giá trị biên, với chiều rộng khối mã hóa nCbW, chiều cao khối mã hóa nCbH và vị trí của mẫu trên cùng bên trái của khối mã hóa (xCb, yCb), khi cIdx là bằng 0, hoặc khi cIdx không bằng 0 và edgeType là bằng EDGE_VER và xCb % 8 là bằng 0, hoặc khi cIdx không bằng 0 và edgeType là bằng EDGE_HOR và yCb % 8 là bằng 0, các mép được lọc bởi các bước được xếp thứ tự sau đây:

2. Biến số filterEdgeFlag được rút ra như sau:

– Nếu edgeType là bằng EDGE_VER và một hoặc nhiều trong số các điều kiện sau đây là đúng, filterEdgeFlag được thiết lập bằng 0:

– Biên trái của khối mã hóa hiện tại là biên trái của ảnh.

– [[Biên trái của khối mã hóa hiện tại là biên trái hoặc phải của ảnh con và loop_filter_across_subpic_enabled_flag[SubPicIdx] là bằng 0.]]

– Biên trái của khối mã hóa hiện tại là biên trái của ô ảnh và loop_filter_across_tiles_enabled_flag là bằng 0.

– Biên trái của khối mã hóa hiện tại là biên trái của lát và loop_filter_across_slices_enabled_flag là bằng 0.

– Biên trái của khối mã hóa hiện tại là một trong số các đường biên ảo dọc của ảnh và VirtualBoundariesDisabledFlag là bằng 1.

– Ngược lại, nếu edgeType là bằng EDGE_HOR và một hoặc nhiều trong số các điều kiện sau đây là đúng, biến số filterEdgeFlag được thiết lập bằng 0:

– Biên trên của khối mã hóa độ sáng hiện tại là biên trên của ảnh.

– [[Biên trên của khối mã hóa hiện tại là biên trên hoặc biên đáy của ảnh con và loop_filter_across_subpic_enabled_flag[SubPicIdx] là bằng 0.]]

– Biên trên của khối mã hóa hiện tại là biên trên của ô ảnh và loop_filter_across_tiles_enabled_flag là bằng 0.

– Biên trên của khối mã hóa hiện tại là biên trên của lát và loop_filter_across_slices_enabled_flag là bằng 0.

– Biên trên của khối mã hóa hiện tại là một trong số các đường biên ảo ngang của ảnh và VirtualBoundariesDisabledFlag là bằng 1.

– Ngược lại, filterEdgeFlag được thiết lập bằng 1.

...

Quá trình lọc đối với mẫu độ sáng bằng cách sử dụng các bộ lọc ngắn

Các đầu vào của quá trình này là:

– các giá trị mẫu p_i và q_i với i = 0..3,

– các vị trí của p_i và q_i, (xP_i, yP_i) và (xQ_i, yQ_i) với i = 0..2,

– biến số dE,

– các biến số dEp và dEq chứa các quyết định để lọc lần lượt các mẫu p1 và q1,

– biến số t_C.

Các đầu ra của quá trình này là:

– số lượng các mẫu được lọc nDp và nDq,

– các giá trị mẫu được lọc p_i' và q_j' với i = 0..nDp − 1, j = 0..nDq − 1.

Phụ thuộc vào giá trị của dE, các điều sau đây được áp dụng:

– Nếu biến số dE là bằng 2, cả nDp và nDq đều được thiết lập bằng 3 và việc lọc mạnh sau đây được áp dụng:

p_0' =

$Clip3(p_0 - 3 * t_C, p_0 + 3 * t_C, (p_2 + 2 * p_1 + 2 * p_0 + 2 * q_0 + q_1 + 4) >> 3)$ (8-1150)

$p_1' = Clip3(p_1 - 2 * t_C, p_1 + 2 * t_C, (p_2 + p_1 + p_0 + q_0 + 2) >> 2)$ (8-1151)

p_2' =

$Clip3(p_2 - 1 * t_C, p_2 + 1 * t_C, (2 * p_3 + 3 * p_2 + p_1 + p_0 + q_0 + 4) >> 3)$ (8-1152)

q_0' =

$Clip3(q_0 - 3 * t_C, q_0 + 3 * t_C, (p_1 + 2 * p_0 + 2 * q_0 + 2 * q_1 + q_2 + 4) >> 3)$ (8-1153)

$q_1' = Clip3(q_1 - 2 * t_C, q_1 + 2 * t_C, (p_0 + q_0 + q_1 + q_2 + 2) >> 2)$ (8-1154)

$q_2'=$

$Clip3(q_2 - 1 * t_C, q_2 + 1 * t_C, (p_0 + q_0 + q_1 + 3 * q_2 + 2 * q_3 + 4) >> 3)$ (8-1155)

– Ngược lại, cả nDp và nDq đều được thiết lập bằng 0 và việc lọc yếu sau đây được áp dụng:

– Các điều sau đây được áp dụng:

$\Delta = (9 * (q_0 - p_0) - 3 * (q_1 - p_1) + 8) >> 4$ (8-1156)

– Khi Abs(Δ) nhỏ hơn $t_C * 10$, các bước được xếp thứ tự sau đây được áp dụng:

– Các giá trị mẫu được lọc p_0' và q_0' được chỉ ra như sau đây:

$\Delta = Clip3(-t_C, t_C, \Delta)$ (8-1157)

$p_0' = Clip1(p_0 + \Delta)$ (8-1158)

$q_0' = Clip1(q_0 - \Delta)$ (8-1159)

–Khi dEp là bằng 1, giá trị mẫu được lọc p_1' được chỉ ra như sau đây:

$$\Delta p = \text{Clip3}(-(t_C >> 1), t_C >> 1, (((p_2 + p_0 + 1) >> 1) - p_1 + \Delta) >> 1) \quad (8\text{-}1160)$$

$$p_1' = \text{Clip1}(p_1 + \Delta p) \quad (8\text{-}1161)$$

–Khi dEq là bằng 1, giá trị mẫu được lọc q_1' được chỉ ra như sau đây:

$$\Delta q = \text{Clip3}(-(t_C >> 1), t_C >> 1, (((q_2 + q_0 + 1) >> 1) - q_1 - \Delta) >> 1) \quad (8\text{-}1162)$$

$$q_1' = \text{Clip1}(q_1 + \Delta q) \quad (8\text{-}1163)$$

–nDp được thiết lập bằng dEp + 1 và nDq được thiết lập bằng dEq + 1.

Khi nDp là lớn hơn 0 và pred_mode_plt_flag của đơn vị mã hóa mà bao gồm khối mã hóa chứa mẫu p_0 là bằng 1, nDp được thiết lập bằng 0

Khi nDq là lớn hơn 0 và pred_mode_plt_flag của đơn vị mã hóa mà bao gồm khối mã hóa chứa mẫu q_0 là bằng 1, nDq được thiết lập bằng 0

Khi nDp là lớn hơn 0 và loop_filter_across_subpic_enabled_flag[subPicIdxP] là bằng 0, nDp được thiết lập bằng 0, trong đó subPicIdxP là chỉ số ảnh con của ảnh con chứa mẫu p_0.

Khi nDq là lớn hơn 0 và loop_filter_across_subpic_enabled_flag[subPicIdxQ] là bằng 0, nDq được thiết lập bằng 0, trong đó subPicIdxQ là chỉ số ảnh con của ảnh con chứa mẫu q_0.

Quá trình lọc đối với mẫu độ sáng bằng cách sử dụng các bộ lọc dài

Các đầu vào của quá trình này là:

– các biến số maxFilterLengthP và maxFilterLengthQ,

– các giá trị mẫu p_i và q_j với i = 0..maxFilterLengthP và j = 0..maxFilterLengthQ,

– các vị trí của p_i và q_j, (xP_i, yP_i) và (xQ_j, yQ_j) với i = 0..maxFilterLengthP − 1 và j = 0..maxFilterLengthQ − 1,

– biến số t_C.

Các đầu ra của quá trình này là:

– các giá trị mẫu được lọc p_i' và q_j' với i = 0..maxFilterLengthP − 1, j = 0..maxFilterLenghtQ − 1.

Biến số refMiddle được rút ra như sau:

– Nếu maxFilterLengthP là bằng maxFilterLengthQ và maxFilterLengthP là bằng 5, các điều sau đây được áp dụng:

$$refMiddle = (p_4 + p_3 + 2*(p_2 + p_1 + p_0 + q_0 + q_1 + q_2) + q_3 + q_4 + 8) >> 4 \quad (8\text{-}1164)$$

– Ngược lại, nếu maxFilterLengthP là bằng maxFilterLengthQ và maxFilterLengthP không bằng 5, các điều sau đây được áp dụng:

$$refMiddle = (p_6 + p_5 + p_4 + p_3 + p_2 + p_1 + 2*(p_0 + q_0) + q_1 + q_2 + q_3 + q_4 + q_5 + q_6 + 8) >> 4 \quad (8\text{-}1165)$$

– Ngược lại, nếu một trong số các điều kiện sau đây là đúng,

– maxFilterLengthQ là bằng 7 và maxFilterLengthP là bằng 5,

– maxFilterLengthQ là bằng 5 và maxFilterLengthP là bằng 7,

các điều sau đây được áp dụng:

$$refMiddle = (p_5 + p_4 + p_3 + p_2 + 2*(p_1 + p_0 + q_0 + q_1) + q_2 + q_3 + q_4 + q_5 + 8) >> 4 \quad (8\text{-}1166)$$

– Ngược lại, nếu một trong số các điều kiện sau đây là đúng,

– maxFilterLengthQ là bằng 5 và maxFilterLengthP là bằng 3,

– maxFilterLengthQ là bằng 3 và maxFilterLengthP là bằng 5,

các điều sau đây được áp dụng:

$$refMiddle = (p_3 + p_2 + p_1 + p_0 + q_0 + q_1 + q_2 + q_3 + 4) >> 3 \quad (8\text{-}1167)$$

– Ngược lại, nếu maxFilterLengthQ là bằng 7 và maxFilterLengthP là bằng 3, các điều sau đây được áp dụng:

refMiddle =

$(2 * (p_2 + p_1 + p_0 + q_0) + p_0 + p_1 + q_1 + q_2 + q_3 + q_4 + q_5 + q_6 + 8) >> 4$ (8-1168)

– Ngược lại, các điều sau đây được áp dụng:

refMiddle =

$(p_6 + p_5 + p_4 + p_3 + p_2 + p_1 + 2*(q_2 + q_1 + q_0 + p_0) + q_0 + q_1 + 8) >> 4$ (8-1169)

Các biến số refP và refQ được rút ra như sau:

$refP = (p_{maxFilterLengtP} + p_{maxFilterLengthP-1} + 1) >> 1$ (8-1179)

$refQ = (q_{maxFilterLengtQ} + q_{maxFilterLengthQ-1} + 1) >> 1$ (8-1171)

Các biến số f_i và $tcPD_i$ được xác định như sau đây:

– Nếu maxFilterLengthP là bằng 7, các điều sau đây được áp dụng:

$f_{0..6} = \{ 59, 50, 41, 32, 23, 14, 5 \}$ (8-1172)

$tcPD_{0..6} = \{ 6, 5, 4, 3, 2, 1, 1 \}$ (8-1173)

– Ngược lại, nếu maxFilterLengthP là bằng 5, các điều sau đây được áp dụng:

$f_{0..4} = \{ 58, 45, 32, 19, 6 \}$ (8-1174)

$tcPD_{0..4} = \{ 6, 5, 4, 3, 2 \}$ (8-1175)

– Ngược lại, các điều sau đây được áp dụng:

$f_{0..2} = \{ 53, 32, 11 \}$ (8-1176)

$tcPD_{0..2} = \{ 6, 4, 2\}$ (8-1177)

Các biến số g_j và $tcQD_j$ được xác định như sau đây:

– Nếu maxFilterLengthQ là bằng 7, các điều sau đây được áp dụng:

$g_{0..6} = \{ 59, 50, 41, 32, 23, 14, 5 \}$ (8-1178)

$tcQD_{0..6} = \{ 6, 5, 4, 3, 2, 1, 1 \}$ (8-1179)

– Ngược lại, nếu maxFilterLengthQ là bằng 5, các điều sau đây được áp dụng:

$g_{0..4} = \{ 58, 45, 32, 19, 6 \}$ (8-1180)

$tcQD_{0..4} = \{ 6, 5, 4, 3, 2 \}$ (8-1181)

– Ngược lại, các điều sau đây được áp dụng:

$g_{0..2} = \{ 53, 32, 11 \}$ (8-1182)

$tcQD_{0..2} = \{ 6, 4, 2 \}$ (8-1183)

Các giá trị mẫu được lọc p_i' và q_j' với i = 0..maxFilterLengthP − 1 và j = 0..maxFilterLengthQ − 1 được rút ra như sau:

$p_i' = Clip3(p_i - (tc*tcPD_i) >> 1, p_i + (tc*tcPD_i) >> 1, (refMiddle*f_i + refP*(64 - f_i) + 32) >> 6)$ (8-1184)

$q_j' = Clip3(q_j - (tc*tcQD_j) >> 1, q_j + (tc*tcQD_j) >> 1, (refMiddle*g_j + refQ*(64 - g_j) + 32) >> 6)$ (8-1185)

Khi pred_mode_plt_flag của đơn vị mã hóa mà bao gồm khối mã hóa chứa mẫu p_i là bằng 1, giá trị mẫu được lọc, p_i' được thay thế bằng giá trị mẫu đầu vào tương ứng p_i với i = 0..maxFilterLengthP − 1.

Khi pred_mode_plt_flag của đơn vị mã hóa mà bao gồm khối mã hóa chứa mẫu q_i là bằng 1, giá trị mẫu được lọc, q_i' được thay thế bằng giá trị mẫu đầu vào tương ứng q_j với j = 0..maxFilterLengthQ − 1.

Khi loop_filter_across_subpic_enabled_flag[subPicIdxP] là bằng 0, trong đó subPicIdxP là chỉ số ảnh con của ảnh con chứa mẫu p_0, giá trị mẫu được lọc, p_i' được thay thế bằng giá trị mẫu đầu vào tương ứng p_i với i = 0..maxFilterLengthP − 1.

Khi loop_filter_across_subpic_enabled_flag[subPicIdxQ] là bằng 0, trong đó subPicIdxQ là chỉ số ảnh con của ảnh con chứa mẫu q_0, giá trị mẫu được lọc, q_i' được thay thế bằng giá trị mẫu đầu vào tương ứng q_j với j = 0..maxFilterLengthQ − 1.

Quá trình lọc đối với mẫu sắc độ

Quá trình này chỉ được gọi ra khi ChromaArrayType không bằng 0.

Các đầu vào của quá trình này là:

- biến số maxFilterLength,
- các giá trị mẫu sắc độ p_i và q_i với i = 0..maxFilterLengthCbCr,
- các vị trí sắc độ của p_i và q_i, (xP_i, yP_i) và (xQ_i, yQ_i) với i = 0..maxFilterLengthCbCr − 1,
- biến số tc.

Các đầu ra của quá trình này là các giá trị mẫu được lọc p_i' và q_i' với i = 0..maxFilterLengthCbCr − 1.

Các giá trị mẫu được lọc p_i' và q_i' với i = 0..maxFilterLengthCbCr − 1 được rút ra như sau:

– Nếu maxFilterLengthCbCr là bằng 3, việc lọc mạnh sau đây được áp dụng:

$$p_0' = \text{Clip3}(p_0 - t_C, p_0 + t_C, (p_3 + p_2 + p_1 + 2 * p_0 + q_0 + q_1 + q_2 + 4) >> 3) \quad (8\text{-}1186)$$

$$p_1' = \text{Clip3}(p_1 - t_C, p_1 + t_C, (2 * p_3 + p_2 + 2 * p_1 + p_0 + q_0 + q_1 + 4) >> 3) \quad (8\text{-}1187)$$

$$p_2' = \text{Clip3}(p_2 - t_C, p_2 + t_C, (3 * p_3 + 2 * p_2 + p_1 + p_0 + q_0 + 4) >> 3) \quad (8\text{-}1188)$$

$$q_0' = \text{Clip3}(q_0 - t_C, q_0 + t_C, (p_2 + p_1 + p_0 + 2 * q_0 + q_1 + q_2 + q_3 + 4) >> 3) \quad (8\text{-}1189)$$

$$q_1' = \text{Clip3}(q_1 - t_C, q_1 + t_C, (p_1 + p_0 + q_0 + 2 * q_1 + q_2 + 2 * q_3 + 4) >> 3) \quad (8\text{-}1190)$$

$$q_2' = \text{Clip3}(q_2 - t_C, q_2 + t_C, (p_0 + q_0 + q_1 + 2 * q_2 + 3 * q_3 + 4) >> 3) \quad (8\text{-}1191)$$

– Ngược lại, việc lọc yếu sau đây được áp dụng:

$$\Delta = \text{Clip3}(-t_C, t_C, ((((q_0 - p_0) << 2) + p_1 - q_1 + 4) >> 3)) \quad (8\text{-}1192)$$

$$p_0' = \text{Clip1}(p_0 + \Delta) \quad (8\text{-}1193)$$

$$q_0' = \text{Clip1}(q_0 - \Delta) \quad (8\text{-}1194)$$

Khi pred_mode_plt_flag của đơn vị mã hóa mà bao gồm khối mã hóa chứa mẫu p_i là bằng 1, giá trị mẫu được lọc, p_i' được thay thế bằng giá trị mẫu đầu vào tương ứng p_i với i = 0..maxFilterLengthCbCr − 1.

Khi pred_mode_plt_flag của đơn vị mã hóa mà bao gồm khối mã hóa chứa mẫu q_i là bằng 1, giá trị mẫu được lọc, q_i' được thay thế bằng giá trị mẫu đầu vào tương ứng q_i với i = 0..maxFilterLengthCbCr − 1:

Khi loop_filter_across_subpic_enabled_flag[subPicIdxP] là bằng 0, trong đó subPicIdxP là chỉ số ảnh con của ảnh con chứa mẫu p_0, giá trị mẫu được lọc, p_i' được thay thế bằng giá trị mẫu đầu vào tương ứng p_i với i = 0..maxFilterLengthCbCr − 1.

Khi loop_filter_across_subpic_enabled_flag[subPicIdxQ] là bằng 0, trong đó subPicIdxQ là chỉ số ảnh con của ảnh con chứa mẫu q_0, giá trị mẫu được lọc, q_i' được thay thế bằng giá trị mẫu đầu vào tương ứng q_i với i = 0..maxFilterLengthCbCr − 1:

FIG. 3 là sơ đồ khối của thiết bị xử lý viđeo 300. Thiết bị 300 có thể được sử dụng để thực hiện một hoặc nhiều trong số các phương pháp được mô tả ở đây. Thiết bị 300 có thể được thực hiện trong điện thoại thông minh, máy tính bảng, máy tính, bộ thu internet vạn vật (Internet of Things - IoT), và v.v.. Thiết bị 300 có thể bao gồm một hoặc nhiều bộ xử lý 312, một hoặc nhiều bộ nhớ 314 và phần cứng xử lý viđeo 316. (Các) bộ xử lý 312 có thể được tạo cấu hình để thực hiện một hoặc nhiều phương pháp được mô tả trong phần mô tả. Bộ nhớ (các bộ nhớ) 314 có thể được sử dụng để lưu trữ dữ liệu và mã được sử dụng để thực hiện các phương pháp và các kỹ thuật được mô tả ở đây. Phần cứng xử lý viđeo 316 có thể được sử dụng để thực hiện, trong hệ mạch phần cứng, một số kỹ thuật được mô tả trong phần mô tả.

FIG. 4 là lưu đồ của phương pháp 400 của quá trình xử lý viđeo. Phương pháp 400 bao gồm bước xác định (402), đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó bộ dự báo vectơ chuyển động theo thời gian được xác định để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại bằng cách sử dụng chế độ afin có ở bên trong vùng viđeo thứ hai hay không, và thực hiện (404) biến đổi dựa trên xác định này.

Các giải pháp sau đây có thể được thực hiện như là các giải pháp ưu tiên theo một số phương án.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 1).

1. Phương pháp xử lý viđeo, bao gồm: xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó bộ dự báo vectơ chuyển động theo thời gian được xác định để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại bằng cách sử dụng chế độ afin có ở bên trong vùng viđeo thứ hai hay không; và thực hiện sự biến đổi dựa trên xác định này.

2. Phương pháp theo giải pháp 1, trong đó khối viđeo được bao phủ bởi vùng thứ nhất và vùng thứ hai.

3. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 2, trong đó, trong trường hợp mà vị trí của bộ dự báo vectơ chuyển động theo thời gian ở bên ngoài vùng viđeo thứ hai, thì bộ dự báo vectơ chuyển động theo thời gian được đánh dấu là không khả dụng và không được sử dụng trong việc biến đổi.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 2).

4. Phương pháp xử lý viđeo, bao gồm các bước: xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó mẫu nguyên trong ảnh tham chiếu được tìm nạp để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại có ở bên trong vùng viđeo thứ hai hay không, trong đó ảnh tham chiếu không được sử dụng trong quá trình nội suy trong khi biến đổi; và thực hiện sự biến đổi dựa trên xác định này.

5. Phương pháp theo giải pháp 4, trong đó khối viđeo được bao phủ bởi vùng thứ nhất và vùng thứ hai.

6. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 4 tới 5, trong đó, trong trường hợp mà vị trí của mẫu ở bên ngoài vùng viđeo thứ hai, thì mẫu được đánh dấu là không khả dụng và không được sử dụng trong việc biến đổi.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 3).

7. Phương pháp xử lý viđeo, bao gồm các bước: xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó giá trị mẫu độ sáng được tái tạo được tìm nạp để biến đổi giữa khối viđeo và biểu diễn dòng bit của khối

viđeo hiện tại có ở bên trong vùng viđeo thứ hai hay không; và thực hiện sự biến đổi dựa trên xác định này.

8. Phương pháp theo giải pháp 7, trong đó mẫu độ sáng được bao phủ bởi vùng thứ nhất và vùng thứ hai.

9. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 7 tới 8, trong đó, trong trường hợp mà vị trí của mẫu độ sáng ở bên ngoài vùng viđeo thứ hai, thì mẫu độ sáng được đánh dấu là không khả dụng và không được sử dụng trong việc biến đổi.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 4).

10. Phương pháp xử lý viđeo, bao gồm các bước: xác định, đối với khối viđeo trong vùng viđeo thứ nhất của viđeo, liệu vị trí mà ở đó việc kiểm tra liên quan đến tách, dẫn xuất sâu hoặc báo tín hiệu cờ tách đối với khối viđeo được thực hiện trong khi biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo hiện tại có ở bên trong vùng viđeo thứ hai hay không; và thực hiện sự biến đổi dựa trên xác định này.

11. Phương pháp theo giải pháp 10, trong đó vị trí được bao phủ bởi vùng thứ nhất và vùng thứ hai.

12. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 10 tới 11, trong đó, trong trường hợp mà vị trí ở bên ngoài vùng viđeo thứ hai, thì mẫu độ sáng được đánh dấu là không khả dụng và không được sử dụng trong việc biến đổi.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 8).

13. Phương pháp xử lý viđeo, bao gồm các bước: thực hiện sự biến đổi giữa viđeo bao gồm một hoặc nhiều ảnh viđeo bao gồm một hoặc nhiều khối viđeo, và sự biểu diễn được mã hóa của viđeo, trong đó sự biểu diễn được mã hóa phù hợp với yêu cầu cú pháp mã hóa rằng sự biến đổi không được sử dụng mã hóa/giải mã ảnh con và công cụ mã hóa/giải mã biến đổi độ phân giải động hoặc công cụ lấy mẫu lại ảnh tham chiếu bên trong đơn vị viđeo.

14. Phương pháp theo giải pháp 13, trong đó đơn vị viđeo tương ứng với dãy của một hoặc nhiều ảnh viđeo.

15. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 13 tới 14, trong đó công cụ mã hóa/giải mã biến đổi độ phân giải động bao gồm công cụ mã hóa/giải mã biến đổi độ phân giải thích ứng.

16. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 13 tới 14, trong đó công cụ mã hóa/giải mã biến đổi độ phân giải động bao gồm công cụ mã hóa/giải mã biến đổi độ phân giải động.

17. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 13 tới 16, trong đó sự biểu diễn được mã hóa chỉ báo rằng đơn vị viđeo phù hợp với yêu cầu cú pháp mã hóa.

18. Phương pháp theo giải pháp 17, trong đó sự biểu diễn được mã hóa chỉ báo rằng đơn vị viđeo sử dụng mã hóa ảnh con.

19. Phương pháp theo giải pháp 17, trong đó sự biểu diễn được mã hóa chỉ báo rằng đơn vị viđeo sử dụng công cụ mã hóa/giải mã biến đổi độ phân giải động hoặc công cụ lấy mẫu lại ảnh tham chiếu.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 10).

20. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 19, trong đó vùng viđeo thứ hai bao gồm ảnh con viđeo và trong đó các đường biên của vùng viđeo thứ hai và vùng viđeo khác cũng là biên giữa hai đơn vị cây mã hóa.

21. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 19, trong đó vùng viđeo thứ hai bao gồm ảnh con viđeo và trong đó các đường biên của vùng viđeo thứ hai và vùng viđeo khác cũng là biên giữa hai đơn vị cây mã hóa.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 11).

22. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 21, trong đó vùng viđeo thứ nhất và vùng viđeo thứ hai có các hình dạng hình chữ nhật.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 12).

23. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 22, trong đó vùng viđeo thứ nhất và vùng viđeo thứ hai là không xếp chồng.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 13).

24. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 23, trong đó ảnh viđeo được chia thành các vùng viđeo sao cho điểm ảnh trong ảnh viđeo được bao phủ bởi một và chỉ một vùng viđeo.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 15).

25. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 24, trong đó ảnh viđeo được tách thành vùng viđeo thứ nhất và vùng viđeo thứ hai do ảnh viđeo này ở trong lớp cụ thể của dãy viđeo.

Các giải pháp sau đây có thể được thực hiện cùng với các kỹ thuật bổ sung được mô tả trong các mục được liệt kê trong phần trước (ví dụ, mục 10).

26. Phương pháp xử lý viđeo, bao gồm các bước: thực hiện sự biến đổi giữa viđeo bao gồm một hoặc nhiều ảnh viđeo bao gồm một hoặc nhiều khối viđeo, và sự biểu diễn được mã hóa của viđeo, trong đó sự biểu diễn được mã hóa phù hợp với yêu cầu cú pháp mã hóa rằng phần tử cú pháp thứ nhất subpic_grid_idx[i][j] không lớn hơn phần tử cú pháp thứ hai max_subpics_minus1.

27. Phương pháp theo giải pháp 26, trong đó từ mã biểu diễn phần tử cú pháp thứ nhất không lớn hơn từ mã biểu diễn phần tử cú pháp thứ hai.

28. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 27, trong đó vùng viđeo thứ nhất bao gồm ảnh con viđeo.

29. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 28, trong đó vùng viđeo thứ hai bao gồm ảnh con viđeo.

30. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 tới 29, trong đó bước biến đổi bao gồm tạo mã viđeo thành sự biểu diễn được mã hóa.

31. Phương pháp theo giải pháp bất kỳ trong số các giải pháp từ 1 to 29, trong đó sự biến đổi bao gồm giải mã sự biểu diễn được mã hóa để tạo ra các giá trị điểm ảnh của viđeo.

32. Thiết bị giải mã viđeo bao gồm bộ xử lý được tạo cấu hình để thực hiện phương pháp được nêu trong một hoặc nhiều trong số các giải pháp từ 1 tới 31.

33. Thiết bị tạo mã viđeo bao gồm bộ xử lý được tạo cấu hình để thực hiện phương pháp được nêu trong một hoặc nhiều trong số các giải pháp từ 1 tới 31.

34. Sản phẩm chương trình máy tính có mã máy tính được lưu trong đó, mã này, khi được thực thi bởi bộ xử lý, khiến bộ xử lý thực hiện phương pháp được nêu trong giải pháp bất kỳ trong số các giải pháp từ 1 tới 31.

35. Phương pháp, thiết bị hoặc hệ thống được mô tả trong phần mô tả.

FIG. 8 là sơ đồ khối thể hiện hệ thống xử lý viđeo 800 được nêu làm ví dụ mà trong đó các kỹ thuật khác nhau được bộc lộ ở đây có thể được thực hiện. Các phương án khác nhau có thể bao gồm một số hoặc tất cả trong số các thành phần của hệ thống 800. Hệ thống 800 có thể bao gồm đầu vào 802 để nhận nội dung viđeo. Nội dung viđeo có thể được nhận trong định dạng thô hoặc không nén, ví dụ, các giá trị điểm ảnh đa thành phần 8 hoặc 10 bit, hoặc có thể trong định dạng được nén hoặc được tạo mã. Đầu vào 802 có thể biểu diễn giao diện mạng, giao diện buýt ngoại vi, hoặc giao diện lưu trữ. Các ví dụ về giao diện mạng bao gồm các giao diện dây dẫn chẳng hạn Ethernet, mạng quang thụ động (passive optical network - PON), v.v. và các giao diện không dây chẳng hạn Wi-Fi hoặc các giao diện tế bào.

Hệ thống 800 có thể bao gồm thành phần mã hóa 804 mà có thể thực hiện các phương pháp mã hóa hoặc tạo mã khác nhau được mô tả trong phần mô tả. Thành phần mã hóa 804 có thể giảm tốc độ bit trung bình của viđeo từ đầu vào 802 tới đầu ra của thành phần mã hóa 804 để đưa ra sự biểu diễn được mã hóa của viđeo. Do đó, đôi khi các kỹ thuật mã hóa này được gọi là nén viđeo hoặc các kỹ thuật chuyển mã viđeo. Đầu ra của thành phần mã hóa 804 có thể hoặc là được lưu trữ, hoặc được truyền qua đường truyền thông được kết nối, như được biểu diễn bởi thành phần 806. Sự biểu diễn dòng bit được lưu trữ hoặc được

truyền thông (hoặc được mã hóa) của viđeo được nhận ở đầu vào 802 có thể được sử dụng bởi thành phần 808 để tạo ra các giá trị điểm ảnh hoặc viđeo có thể phát mà được gửi tới giao diện hiển thị 810. Quá trình tạo ra viđeo người dùng xem được từ biểu diễn dòng bit đôi khi được gọi là giải nén viđeo. Hơn nữa, trong khi các hoạt động xử lý viđeo đã biết được gọi là các hoạt động hoặc các công cụ "mã hóa", cần phải thấy rằng các công cụ hoặc các hoạt động mã hóa được sử dụng ở bộ tạo mã và các công cụ hoặc các hoạt động giải mã tương ứng mà đảo ngược các kết quả mã hóa sẽ được thực hiện bởi bộ giải mã.

Các ví dụ về giao diện buýt ngoại vi hoặc giao diện hiển thị có thể bao gồm buýt nối tiếp vạn năng (universal serial bus - USB) hoặc giao diện đa phương tiện độ phân giải cao (high definition multimedia interface - HDMI) hoặc Displayport, và v.v.. Các ví dụ về giao diện lưu trữs bao gồm giao diện SATA (serial advanced technology attachment), PCI, IDE, và tương tự. Các kỹ thuật được mô tả trong phần mô tả có thể được thực hiện trong các thiết bị điện tử khác nhau chẳng hạn các điện thoại di động, các máy tính xách tay, các điện thoại thông minh hoặc các thiết bị khác mà có khả năng thực hiện xử lý dữ liệu kỹ thuật số và/hoặc hiển thị viđeo.

FIG. 9 là sơ đồ khối mà thể hiện hệ thống mã hóa viđeo 100 được nêu làm ví dụ mà có thể sử dụng các kỹ thuật theo sáng chế.

Như được thể hiện trên FIG. 9, hệ thống mã hóa viđeo 100 có thể bao gồm thiết bị nguồn 110 và thiết bị đích 120. Thiết bị nguồn 110 tạo ra dữ liệu viđeo được tạo mã mà có thể được gọi là thiết bị tạo mã viđeo. Thiết bị đích 120 có thể giải mã dữ liệu viđeo được tạo mã được tạo ra bởi thiết bị nguồn 110 mà có thể được gọi là thiết bị giải mã viđeo.

Thiết bị nguồn 110 có thể bao gồm nguồn viđeo 112, bộ tạo mã viđeo 114, và giao diện đầu vào/đầu ra (I/O) 116.

Nguồn viđeo 112 có thể bao gồm nguồn chẳng hạn thiết bị quay viđeo, giao diện để nhận dữ liệu viđeo từ nhà cung cấp nội dung viđeo, và/hoặc hệ thống đồ họa máy tính để tạo ra dữ liệu viđeo, hoặc sự kết hợp của các nguồn này. Dữ liệu viđeo có thể bao gồm một hoặc nhiều ảnh. Bộ tạo mã viđeo 114 tạo mã dữ liệu viđeo từ nguồn viđeo 112 để tạo ra dòng bit. Dòng bit này có thể bao gồm dãy

của các bit mà tạo thành sự biểu diễn được mã hóa của dữ liệu viđeo. Dòng bit có thể bao gồm các ảnh được mã hóa và dữ liệu liên kết. Ảnh được mã hóa là sự biểu diễn được mã hóa của ảnh. Dữ liệu liên kết có thể bao gồm các tập thông số dãy, các tập thông số ảnh, và các cấu trúc cú pháp khác. Giao diện I/O 116 có thể bao gồm bộ điều biến/bộ giải điều biến (môđem) và/hoặc bộ truyền. Dữ liệu viđeo được tạo mã có thể được truyền trực tiếp tới thiết bị đích 120 qua giao diện I/O 116 thông qua mạng 130a. Dữ liệu viđeo được tạo mã cũng có thể được lưu trữ vào trong môi trường lưu trữ/máy chủ 130b để truy nhập bởi thiết bị đích 120.

Thiết bị đích 120 có thể bao gồm giao diện I/O 126, bộ giải mã viđeo 124, và bộ hiển thị 122.

Giao diện I/O 126 có thể bao gồm bộ thu và/hoặc môđem. Giao diện I/O 126 có thể thu nhận dữ liệu viđeo được tạo mã từ thiết bị nguồn 110 hoặc môi trường lưu trữ/ máy chủ 130b. Bộ giải mã viđeo 124 có thể giải mã dữ liệu viđeo được tạo mã. Bộ hiển thị 122 có thể hiển thị dữ liệu viđeo được giải mã cho người dùng. Bộ hiển thị 122 có thể được tích hợp với thiết bị đích 120, hoặc có thể bên ngoài thiết bị đích 120 mà được tạo cấu hình để giao diện với bộ hiển thị bên ngoài.

Bộ tạo mã viđeo 114 và bộ giải mã viđeo 124 có thể hoạt động theo chuẩn nén viđeo, chẳng hạn chuẩn mã hóa viđeo hiệu quả cao (High Efficiency Video Coding - HEVC), chuẩn mã hóa viđeo đa năng (Versatile Video Coding - VVC) và các chuẩn hiện tại và/hoặc tiếp sau khác.

FIG. 10 là sơ đồ khối thể hiện ví dụ của bộ tạo mã viđeo 200, mà có thể là bộ tạo mã viđeo 114 trong hệ thống 100 được thể hiện trên FIG. 9.

Bộ tạo mã viđeo 200 có thể được tạo cấu hình để thực hiện kỹ thuật bất kỳ hoặc tất cả các kỹ thuật theo sáng chế. Trong ví dụ trên FIG. 10, bộ tạo mã viđeo 200 bao gồm nhiều thành phần chức năng. Các kỹ thuật được mô tả theo sáng chế có thể được chia sẻ giữa các thành phần khác nhau của bộ tạo mã viđeo 200. Trong một số ví dụ, bộ xử lý có thể được tạo cấu hình để thực hiện kỹ thuật bất kỳ hoặc tất cả các kỹ thuật được mô tả theo sáng chế.

Các thành phần chức năng của bộ tạo mã viđeo 200 có thể bao gồm đơn vị

phân chia 201, đơn vị xác nhận 202 mà có thể bao gồm đơn vị chọn chế độ 203, đơn vị ước tính chuyển động 204, đơn vị bù chuyển động 205 và đơn vị dự báo trong ảnh 206, đơn vị tạo phần dư 207, đơn vị chuyển đổi 208, đơn vị lượng tử hóa 209, đơn vị lượng tử hóa nghịch đảo 210, đơn vị chuyển đổi nghịch đảo 211, đơn vị tái tạo 212, bộ đệm 213, và đơn vị tạo mã entropi 214.

Trong các ví dụ khác, bộ tạo mã viđeo 200 có thể bao gồm nhiều hơn, ít hơn các thành phần chức năng, hoặc các thành phần chức năng khác nhau. Trong một ví dụ, đơn vị xác nhận 202 có thể bao gồm đơn vị sao chép khối trong ảnh (intra block copy - IBC). Đơn vị IBC có thể thực hiện việc xác nhận trong chế độ IBC mà trong đó ít nhất một ảnh tham chiếu là ảnh mà khối viđeo hiện tại nằm trong đó.

Hơn nữa, một số thành phần, chẳng hạn đơn vị ước tính chuyển động 204 và đơn vị bù chuyển động 205 có thể được tích hợp cao, nhưng được biểu diễn một cách riêng rẽ trong ví dụ trên FIG. 5 nhằm mục đích giải thích.

Đơn vị phân chia 201 có thể phân chia ảnh thành một hoặc nhiều khối viđeo. Bộ tạo mã viđeo 200 và bộ giải mã viđeo 300 có thể hỗ trợ các kích thước khối viđeo khác nhau.

Đơn vị chọn chế độ 203 có thể chọn một trong số các chế độ mã hóa, trong ảnh hoặc ngoài ảnh, ví dụ, dựa trên các kết quả lỗi, và cấp khối được mã hóa trong ảnh hoặc ngoài ảnh thu được tới đơn vị tạo phần dư 207 để tạo ra dữ liệu khối phần dư và tới đơn vị tái tạo 212 để tái tạo khối được tạo mã để sử dụng dưới dạng ảnh tham chiếu. Trong một số ví dụ, đơn vị chọn chế độ 203 có thể chọn sự kết hợp của chế độ xác nhận trong ảnh và ngoài ảnh (combination of intra and inter predication - CIIP) mà trong đó sự xác nhận là dựa trên tín hiệu xác nhận ngoài ảnh và tín hiệu xác nhận trong ảnh. Đơn vị chọn chế độ 203 cũng có thể chọn độ phân giải đối với vectơ chuyển động (ví dụ, độ chính xác điểm ảnh con hoặc điểm ảnh nguyên) đối với khối trong trường hợp xác nhận ngoài ảnh.

Để thực hiện dự báo ngoài ảnh trên khối viđeo hiện tại, đơn vị ước tính chuyển động 204 có thể tạo ra thông tin chuyển động đối với khối viđeo hiện tại bằng cách so sánh một hoặc nhiều khung tham chiếu từ bộ đệm 213 với khối

viđeo hiện tại. Đơn vị bù chuyển động 205 có thể xác định khối viđeo được dự báo đối với khối viđeo hiện tại dựa trên thông tin chuyển động và các mẫu được giải mã của các ảnh từ bộ đệm 213 khác với ảnh được liên kết với khối viđeo hiện tại.

Đơn vị ước tính chuyển động 204 và đơn vị bù chuyển động 205 có thể thực hiện các hoạt động khác nhau đối với khối viđeo hiện tại, ví dụ, phụ thuộc vào việc liệu khối viđeo hiện tại đang ở trong lát I, lát P, hoặc lát B.

Trong một số ví dụ, đơn vị ước tính chuyển động 204 có thể thực hiện dự báo đơn hướng đối với khối viđeo hiện tại, và đơn vị ước tính chuyển động 204 có thể tìm kiếm các ảnh tham chiếu của danh sách 0 hoặc danh sách 1 đối với khối viđeo tham chiếu đối với khối viđeo hiện tại. Sau đó đơn vị ước tính chuyển động 204 có thể tạo ra chỉ số tham chiếu mà chỉ báo ảnh tham chiếu trong danh sách 0 hoặc danh sách 1 mà chứa khối viđeo tham chiếu và vectơ chuyển động mà chỉ báo sự chuyển vị không gian giữa khối viđeo hiện tại và khối viđeo tham chiếu. Đơn vị ước tính chuyển động 204 có thể xuất ra chỉ số tham chiếu, bộ chỉ báo hướng dự báo, và vectơ chuyển động dưới dạng thông tin chuyển động của khối viđeo hiện tại. Đơn vị bù chuyển động 205 có thể tạo ra khối viđeo được dự báo của khối hiện tại dựa trên khối viđeo tham chiếu được chỉ báo bởi thông tin chuyển động của khối viđeo hiện tại.

Trong các ví dụ khác, đơn vị ước tính chuyển động 204 có thể thực hiện dự báo hai hướng đối với khối viđeo hiện tại, đơn vị ước tính chuyển động 204 có thể tìm kiếm các ảnh tham chiếu trong danh sách 0 đối với khối viđeo tham chiếu đối với khối viđeo hiện tại và cũng có thể tìm kiếm các ảnh tham chiếu trong danh sách 1 đối với khối viđeo tham chiếu khác đối với khối viđeo hiện tại. Sau đó đơn vị ước tính chuyển động 204 có thể tạo ra các chỉ số tham chiếu mà chỉ báo các ảnh tham chiếu trong danh sách 0 và danh sách 1 chứa các khối viđeo tham chiếu và các vectơ chuyển động mà chỉ báo các chuyển vị không gian giữa các khối viđeo tham chiếu và khối viđeo hiện tại. Đơn vị ước tính chuyển động 204 có thể xuất ra các chỉ số tham chiếu và các vectơ chuyển động của khối viđeo hiện tại dưới dạng thông tin chuyển động của khối viđeo hiện tại. Đơn vị bù chuyển động 205 có thể tạo ra khối viđeo được dự báo của khối viđeo hiện tại

dựa trên các khối viđeo tham chiếu được chỉ báo bởi thông tin chuyển động của khối viđeo hiện tại.

Trong một số ví dụ, đơn vị ước tính chuyển động 204 có thể xuất ra tập đủ của thông tin chuyển động để xử lý giải mã của bộ giải mã.

Trong một số ví dụ, đơn vị ước tính chuyển động 204 có thể không xuất ra tập đủ của thông tin chuyển động đối với viđeo hiện tại. Thay vào đó, đơn vị ước tính chuyển động 204 có thể báo tín hiệu thông tin chuyển động của khối viđeo hiện tại nhờ tham chiếu tới thông tin chuyển động của khối viđeo khác. Ví dụ, đơn vị ước tính chuyển động 204 có thể xác định rằng thông tin chuyển động của khối viđeo hiện tại là đủ giống với thông tin chuyển động của khối viđeo lân cận.

Trong một ví dụ, đơn vị ước tính chuyển động 204 có thể chỉ báo, trong cấu trúc cú pháp được liên kết với khối viđeo hiện tại, giá trị mà chỉ báo tới bộ giải mã viđeo 300 rằng khối viđeo hiện tại có thông tin chuyển động giống như khối viđeo khác.

Trong ví dụ khác, đơn vị ước tính chuyển động 204 có thể nhận dạng, trong cấu trúc cú pháp được liên kết với khối viđeo hiện tại, khối viđeo khác và hiệu số vectơ chuyển động (motion vector difference - MVD). Hiệu số vectơ chuyển động chỉ báo hiệu số giữa vectơ chuyển động của khối viđeo hiện tại và vectơ chuyển động của khối viđeo được chỉ báo. Bộ giải mã viđeo 300 có thể sử dụng vectơ chuyển động của khối viđeo được chỉ báo và hiệu số vectơ chuyển động để xác định vectơ chuyển động của khối viđeo hiện tại.

Như đã nêu trên, bộ tạo mã viđeo 200 có thể báo tín hiệu theo cách dự báo vectơ chuyển động. Hai ví dụ về báo tín hiệu theo cách dự báo các kỹ thuật mà có thể được thực hiện bởi bộ tạo mã viđeo 200 bao gồm xác nhận vectơ chuyển động cao cấp (advanced motion vector predication - AMVP) và báo tín hiệu chế độ hợp nhất.

Đơn vị dự báo trong ảnh 206 có thể thực hiện dự báo trong ảnh trên khối viđeo hiện tại. Khi đơn vị dự báo trong ảnh 206 thực hiện dự báo trong ảnh trên khối viđeo hiện tại, đơn vị dự báo trong ảnh 206 có thể tạo ra dữ liệu dự báo đối với khối viđeo hiện tại dựa trên các mẫu được giải mã của các khối viđeo khác ở trong cùng ảnh. Dữ liệu dự báo đối với khối viđeo hiện tại có thể bao gồm khối

viđeo được dự báo và các phần tử cú pháp khác nhau.

Đơn vị tạo phần dư 207 có thể tạo ra dữ liệu phần dư đối với khối viđeo hiện tại bằng cách trừ đi (ví dụ, được chỉ báo bởi dấu trừ) (các) khối viđeo được dự báo của khối viđeo hiện tại từ khối viđeo hiện tại. Dữ liệu phần dư của khối viđeo hiện tại có thể bao gồm các khối viđeo dư mà tương ứng với các thành phần mẫu khác nhau của các mẫu trong khối viđeo hiện tại.

Trong các ví dụ khác, có thể không có dữ liệu phần dư đối với khối viđeo hiện tại đối với khối viđeo hiện tại, ví dụ trong chế độ bỏ qua, và đơn vị tạo phần dư 207 có thể không thực hiện hoạt động trừ.

Đơn vị xử lý chuyển đổi 208 có thể tạo ra một hoặc nhiều hệ số khối viđeo chuyển đổi đối với khối viđeo hiện tại bằng cách áp dụng một hoặc nhiều sự chuyển đổi lên khối viđeo dư được liên kết với khối viđeo hiện tại.

Sau khi đơn vị xử lý chuyển đổi 208 tạo ra hệ số khối viđeo chuyển đổi được liên kết với khối viđeo hiện tại, đơn vị lượng tử hóa 209 có thể lượng tử hóa hệ số khối viđeo chuyển đổi được liên kết với khối viđeo hiện tại dựa trên một hoặc nhiều giá trị thông số lượng tử hóa (quantization parameter - QP) được liên kết với khối viđeo hiện tại.

Đơn vị lượng tử hóa nghịch đảo 210 và đơn vị chuyển đổi nghịch đảo 211 có thể lần lượt áp dụng phép lượng tử hóa nghịch đảo và các chuyển đổi nghịch đảo vào hệ số khối viđeo chuyển đổi, để tái tạo khối viđeo dư từ hệ số khối viđeo chuyển đổi. Đơn vị tái tạo 212 có thể bổ sung khối viđeo dư được tái tạo vào các mẫu tương ứng từ một hoặc nhiều khối viđeo được dự báo được tạo ra bởi đơn vị xác nhận 202 để đưa ra khối viđeo được tái tạo được liên kết với khối hiện tại để lưu trữ trong bộ đệm 213.

Sau khi đơn vị tái tạo 212 tái tạo khối viđeo, hoạt động lọc vòng có thể được thực hiện để giảm các thông tin giả tạo khối viđeo trong khối viđeo.

Đơn vị tạo mã entropi 214 có thể nhận dữ liệu từ các thành phần chức năng khác của bộ tạo mã viđeo 200. Khi đơn vị tạo mã entropi 214 nhận dữ liệu, đơn vị tạo mã entropi 214 có thể thực hiện một hoặc nhiều hoạt động tạo mã entropi để tạo ra dữ liệu được tạo mã entropi và xuất ra dòng bit mà bao gồm dữ liệu được tạo mã entropi.

FIG. 11 là sơ đồ khối thể hiện ví dụ của bộ giải mã viđeo 300 mà có thể là bộ giải mã viđeo 114 trong hệ thống 100 được thể hiện trên FIG. 9.

Bộ giải mã viđeo 300 có thể được tạo cấu hình để thực hiện kỹ thuật bất kỳ hoặc tất cả các kỹ thuật theo sáng chế. Trong ví dụ trên FIG. 11, bộ giải mã viđeo 300 bao gồm nhiều thành phần chức năng. Các kỹ thuật được mô tả theo sáng chế có thể được chia sẻ giữa các thành phần khác nhau của bộ giải mã viđeo 300. Trong một số ví dụ, bộ xử lý có thể được tạo cấu hình để thực hiện kỹ thuật bất kỳ hoặc tất cả các kỹ thuật được mô tả theo sáng chế.

Trong ví dụ trên FIG. 11, bộ giải mã viđeo 300 bao gồm đơn vị giải mã entropi 301, đơn vị bù chuyển động 302, đơn vị dự báo trong ảnh 303, đơn vị lượng tử hóa nghịch đảo 304, đơn vị chuyển đổi nghịch đảo 305, và đơn vị tái tạo 306 và bộ đệm 307. Bộ giải mã viđeo 300 có thể, trong một số ví dụ, thực hiện giải mã bước chuyển thường là thuận nghịch để tạo mã bước chuyển được mô tả liên quan đến bộ tạo mã viđeo 200 (ví dụ, FIG. 10).

Đơn vị giải mã entropi 301 có thể lấy ra dòng bit được tạo mã. Dòng bit được tạo mã có thể bao gồm dữ liệu viđeo được mã hóa entropi (ví dụ, các khối được tạo mã của dữ liệu viđeo). Đơn vị giải mã entropi 301 có thể giải mã dữ liệu viđeo được mã hóa entropi, và từ dữ liệu viđeo được giải mã entropi, đơn vị bù chuyển động 302 có thể xác định thông tin chuyển động bao gồm các vectơ chuyển động, độ chính xác vectơ chuyển động, các chỉ số danh sách ảnh tham chiếu, và thông tin chuyển động khác. Đơn vị bù chuyển động 302 có thể, ví dụ, xác định thông tin này bằng cách thực hiện AMVP và chế độ hợp nhất.

Đơn vị bù chuyển động 302 có thể tạo nên các khối được bù chuyển động, có thể bằng cách thực hiện nội suy dựa trên các bộ lọc nội suy. Các bộ nhận dạng đối với các bộ lọc nội suy cần được sử dụng với độ chính xác điểm ảnh con có thể được bao gồm trong các phần tử cú pháp.

Đơn vị bù chuyển động 302 có thể sử dụng các bộ lọc nội suy như được sử dụng bởi bộ tạo mã viđeo 20 trong khi tạo mã của khối viđeo để tính toán các giá trị được nội suy đối với các điểm ảnh nguyên con của khối tham chiếu. Đơn vị bù chuyển động 302 có thể xác định các bộ lọc nội suy được sử dụng bởi bộ tạo mã viđeo 200 theo thông tin cú pháp nhận được và sử dụng các bộ lọc nội suy

để đưa ra các khối dự báo.

Đơn vị bù chuyển động 302 có thể sử dụng một số cú pháp thông tin để xác định các kích thước của các khối được sử dụng để tạo mã (các) khung và/hoặc (các) lát của dãy viđeo được tạo mã, thông tin phân chia mà mô tả cách thức mỗi khối macro của ảnh của dãy viđeo được tạo mã được phân chia, các chế độ chỉ báo cách thức mỗi phân chia được tạo mã, một hoặc nhiều khung tham chiếu (và các danh sách khung tham chiếu) đối với mỗi khối được tạo mã ngoài ảnh, và thông tin khác để giải mã dãy viđeo được tạo mã.

Đơn vị dự báo trong ảnh 303 có thể sử dụng các chế độ dự báo trong ảnh ví dụ nhận được trong dòng bit để tạo thành khối dự báo từ các khối liền kề trong không gian. Đơn vị lượng tử hóa nghịch đảo 303 nghịch đảo lượng tử hóa, tức là, giải lượng tử hóa, các hệ số khối viđeo được lượng tử hóa được cấp trong dòng bit và được giải mã bởi đơn vị giải mã entropi 301. Đơn vị chuyển đổi nghịch đảo 303 áp dụng phép chuyển đổi nghịch đảo.

Đơn vị tái tạo 306 có thể lấy tổng các khối dư với các khối dự báo tương ứng được tạo ra bởi đơn vị bù chuyển động 202 hoặc đơn vị dự báo trong ảnh 303 để tạo thành các khối được giải mã. Nếu mong muốn, bộ lọc giải khối cũng có thể được áp dụng để lọc các khối được giải mã nhằm loại bỏ các thông tin giả tạo khối. Sau đó các khối viđeo được giải mã được lưu trữ trong bộ đệm 307, mà cấp các khối tham chiếu cho việc xác nhận bù chuyển động/trong ảnh tiếp sau và cũng đưa ra viđeo được giải mã để trình diễn trên bộ hiển thị.

FIG. 12 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1200 bao gồm, ở hoạt động 1210, bước thực hiện sự biến đổi giữa ảnh của viđeo và biểu diễn dòng bit của viđeo. Ảnh bao gồm một hoặc nhiều ảnh con, và biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra rằng chiều dài của phần tử cú pháp là bằng Ceil(Log2(SS)) bit. SS là lớn hơn 0, và phần tử cú pháp chỉ báo vị trí ngang hoặc dọc của góc trên cùng bên trái của đơn vị cây mã hóa của ảnh con của ảnh.

Theo một số phương án, quy tắc định dạng còn chỉ ra rằng giá trị mặc định của chiều dài của phần tử cú pháp thứ hai là bằng Ceil(Log2(SS)) - P, trong đó SS là lớn hơn 0. Phần tử cú pháp thứ hai chỉ báo chiều rộng mặc định hoặc chiều

cao mặc định của ảnh con của ảnh. Theo một số phương án, chiều rộng ảnh tối đa trong các mẫu độ sáng được biểu diễn dưới dạng pic_width_max_in_luma_samples và kích cỡ của khối cây mã hóa được biểu diễn dưới dạng CtbSizeY. SS bằng (pic_width_max_in_luma_samples + RR) / CtbSizeY trong trường hợp phần tử cú pháp chỉ ra vị trí ngang của góc trên cùng bên trái của đơn vị cây mã hóa hoặc chiều rộng mặc định của ảnh con, RR là số nguyên khác không. Theo một số phương án, chiều cao ảnh tối đa trong các mẫu độ sáng được biểu diễn dưới dạng pic_height_max_in_luma_samples và kích cỡ của khối cây mã hóa được biểu diễn dưới dạng CtbSizeY. SS bằng (pic_height_max_in_luma_samples + RR) / CtbSizeY trong trường hợp phần tử cú pháp chỉ ra vị trí dọc của góc trên cùng bên trái của đơn vị cây mã hóa hoặc chiều cao mặc định của ảnh con, RR là số nguyên khác không. Theo một số phương án, RR = CtbSizeY-1.

FIG. 13 là hình vẽ thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. phương pháp 1300 bao gồm, ở hoạt động 1310, thực hiện sự biến đổi giữa ảnh của viđeo và biểu diễn dòng bit của viđeo, trong đó ảnh bao gồm một hoặc nhiều ảnh con. Biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra rằng các ảnh con khác nhau có các bộ nhận dạng khác nhau.

Theo một số phương án, quy tắc định dạng chỉ ra rằng các bộ nhận dạng của một hoặc nhiều ảnh con được bao gồm ở bậc đơn vị viđeo, đơn vị viđeo bao gồm tập thông số dãy, tập thông số ảnh, hoặc tiêu đề ảnh. Theo một số phương án, cờ cú pháp thứ nhất ở tập thông số dãy chỉ báo liệu việc báo tín hiệu các bộ nhận dạng của một hoặc nhiều ảnh con có mặt ở bậc tập thông số dãy hay không, cờ cú pháp thứ hai ở tập thông số ảnh chỉ báo liệu việc báo tín hiệu các bộ nhận dạng của một hoặc nhiều ảnh con có mặt ở bậc tập thông số ảnh hay không, và cờ cú pháp thứ ba ở tiêu đề ảnh chỉ báo liệu việc báo tín hiệu các bộ nhận dạng của một hoặc nhiều ảnh con có mặt ở bậc tiêu đề ảnh hay không. Ít nhất một trong số cờ cú pháp thứ nhất, cờ cú pháp thứ hai, hoặc cờ cú pháp thứ ba là bằng 1. Theo một số phương án, việc báo tín hiệu các bộ nhận dạng của một hoặc nhiều ảnh con được bỏ qua ở bậc tiêu đề ảnh trong trường hợp cờ cú pháp thứ

nhất chỉ báo rằng việc báo tín hiệu các bộ nhận dạng của một hoặc nhiều ảnh con là có mặt ở bậc tập thông số dãy.

Theo một số phương án, quy tắc định dạng chỉ ra rằng các bộ nhận dạng của một hoặc nhiều ảnh con được bao gồm trong danh sách các bộ nhận dạng ảnh con. Theo một số phương án, bộ nhận dạng của ảnh con thứ nhất trong danh sách được ký hiệu là SubpicIdList[i] và bộ nhận dạng của ảnh con thứ hai được ký hiệu là SubpicIdList[j], trong đó j = i - P. Quy tắc định dạng chỉ ra rằng hiệu số D[i] giữa SubpicIdList[i] và SubpicIdList[j] được chỉ báo trong biểu diễn dòng bit. Theo một số phương án, P là bằng 1. Theo một số phương án, i > P. Theo một số phương án, D[i] là lớn hơn 0. Theo một số phương án, D[i]-1 được bao gồm trong biểu diễn dòng bit.

Theo một số phương án, danh sách các bộ nhận dạng ảnh con được xác định dựa trên thứ tự mà tập thông số dãy được xem xét đầu tiên, tập thông số ảnh được xem xét tiếp theo, và tiêu đề ảnh được xem xét cuối cùng. Theo một số phương án, danh sách các bộ nhận dạng ảnh con được xác định dựa trên thứ tự mà tiêu đề ảnh được xem xét đầu tiên, tập thông số ảnh được xem xét tiếp theo, và tập thông số dãy được xem xét cuối cùng.

Theo một số phương án, quy tắc định dạng chỉ ra rằng, trong trường hợp các bộ nhận dạng của một hoặc nhiều ảnh con được bỏ qua ở một hoặc nhiều bậc đơn vị viđeo, các giá trị mặc định được gán cho các bộ nhận dạng trong danh sách các bộ nhận dạng ảnh con. Một hoặc nhiều đơn vị viđeo bao gồm ít nhất tập thông số dãy, tập thông số ảnh, hoặc tiêu đề ảnh. Theo một số phương án, bộ nhận dạng của ảnh con trong danh sách được ký hiệu là SubpicIdList[i], và giá trị mặc định đối với SubpicIdList[i] là i + P, P là giá trị dịch vị.

FIG. 14 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1400 bao gồm, ở hoạt động 1410, bước xác định, để biến đổi giữa khối của ảnh con thứ nhất của viđeo và biểu diễn dòng bit của viđeo, liệu có áp dụng quá trình giải khối ngang qua mép giữa khối và khối lân cận của ảnh con thứ hai dựa trên việc liệu quá trình lọc vòng có được cho phép ngang qua các đường biên ảnh con hay không. Phương pháp 1400 cũng bao gồm, ở hoạt động 1420, bước thực hiện sự biến đổi dựa trên xác định này.

Theo một số phương án, cờ cú pháp thứ nhất chỉ báo liệu quá trình lọc vòng có được cho phép truy cập các mẫu ngang qua các đường biên đối với phía thứ nhất của mép và cờ cú pháp thứ hai chỉ báo liệu quá trình lọc vòng có được cho phép truy cập các mẫu ngang qua các đường biên đối với phía thứ hai của mép hay không. Mép không được lọc trong trường hợp ít nhất một trong số cờ cú pháp thứ nhất hoặc cờ cú pháp thứ hai chỉ báo rằng quá trình lọc vòng là không được cho phép. Theo một số phương án, cờ cú pháp thứ nhất chỉ báo rằng quá trình lọc vòng được cho phép truy cập các mẫu ngang qua các đường biên đối với phía thứ nhất của mép, và mép không được lọc do cờ cú pháp thứ hai chỉ báo rằng quá trình lọc vòng không được cho phép truy cập các mẫu ngang qua các đường biên đối với phía thứ hai của mép.

Theo một số phương án, phía thứ nhất của mép là được lọc và việc liệu phía thứ hai của mép có được lọc hay không được xác định riêng rẽ với nhau. Theo một số phương án, các mẫu ở phía thứ nhất của mép được lọc do cờ cú pháp thứ nhất bằng 1. Theo một số phương án, các mẫu ở phía thứ hai của mép được lọc do cờ cú pháp thứ hai bằng 1. Theo một số phương án, phía thứ nhất của mép là ở trong ảnh con thứ nhất, và phía thứ hai của mép là ở trong ảnh con thứ hai.

FIG. 15 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1500 bao gồm, ở hoạt động 1510, bước thực hiện sự biến đổi giữa viđeo và biểu diễn dòng bit của viđeo. Biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra phần tử cú pháp chỉ báo kích thước khối tối đa được sử dụng đối với công cụ mã hóa bỏ qua chuyển đổi trong tập thông số ảnh được báo tín hiệu một cách độc lập với các cờ cú pháp bất kỳ trong tập thông số dãy.

FIG. 16 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1600 bao gồm, ở hoạt động 1610, bước thực hiện sự biến đổi giữa viđeo và biểu diễn dòng bit của viđeo. Biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ ra phần tử cú pháp chỉ báo kích thước khối tối đa được sử dụng đối với công cụ mã hóa bỏ qua chuyển đổi được báo tín hiệu dựa trên cờ cú pháp trong tập thông số dãy.

Theo một số phương án, phần tử cú pháp được bao gồm trong tập thông số dãy hoặc tiêu đề ảnh. Theo một số phương án, phần tử cú pháp bao gồm log2_transform_skip_max_size_minus2. Theo một số phương án, cờ cú pháp trong tập thông số dãy bao gồm sps_transform_skip_enabled_flag.

FIG. 17 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1700 bao gồm, ở hoạt động 1710, bước thực hiện sự biến đổi giữa khối hiện tại của viđeo và biểu diễn dòng bit của viđeo. Phương pháp 1700 cũng bao gồm, ở hoạt động 1720, bước xác định, sau khi biến đổi, cách thức cập nhật bảng dự báo vectơ chuyển động dựa trên tiến trình dựa trên việc liệu khối hiện tại đã được mã hóa bằng cách sử dụng chế độ phân chia hình học mà trong đó khối được dự báo bởi tổng có trọng số của ít nhất hai dự báo. Các trọng số dành cho tổng có trọng số được tạo ra cho ít nhất hai phân chia. Ít nhất một số phân chia có mép góc, và trong đó bảng dự báo vectơ chuyển động dựa trên tiến trình bao gồm các ứng viên chuyển động dựa trên các khối được mã hóa trước đó của viđeo.

Theo một số phương án, bảng dự báo vectơ chuyển động dựa trên tiến trình không được cập nhật trong trường hợp khối hiện tại được mã hóa bằng cách sử dụng chế độ phân chia hình học. Theo một số phương án, bảng dự báo vectơ chuyển động dựa trên tiến trình được cập nhật trong trường hợp khối hiện tại được mã hóa bằng cách sử dụng chế độ phân chia hình học. Theo một số phương án, bảng dự báo vectơ chuyển động dựa trên tiến trình được cập nhật bằng cách sử dụng thông tin chuyển động của một dự báo của khối hiện tại. Theo một số phương án, bảng dự báo vectơ chuyển động dựa trên tiến trình được cập nhật bằng cách sử dụng thông tin chuyển động của nhiều dự báo của khối hiện tại.

FIG. 18 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1800 bao gồm, ở hoạt động 1810, bước thực hiện sự biến đổi giữa đơn vị xử lý hiện tại của viđeo và biểu diễn dòng bit của viđeo bằng cách sử dụng quá trình lọc vòng thích ứng giao thành phần mà trong đó các mẫu sắc độ được lọc dựa trên các mẫu độ sáng tương ứng. Đơn vị xử lý hiện tại bao gồm đơn vị xử lý lọc vòng thích ứng được xác định bởi hai đường biên ảo. Các mẫu độ sáng mà nằm bên ngoài đơn vị xử lý hiện tại được

loại trừ khỏi việc lọc các mẫu sắc độ của đơn vị xử lý hiện tại.

Theo một số phương án, các mẫu độ sáng mà nằm bên ngoài đơn vị xử lý hiện tại được đệm, và các mẫu độ sáng được đệm được sử dụng để lọc các mẫu sắc độ của đơn vị xử lý hiện tại.

FIG. 19 thể hiện biểu diễn dưới dạng lưu đồ của phương pháp xử lý viđeo theo công nghệ theo sáng chế. Phương pháp 1900 bao gồm, ở hoạt động 1910, bước thực hiện sự biến đổi giữa đơn vị xử lý hiện tại của viđeo và biểu diễn dòng bit của viđeo bằng cách sử dụng quá trình lọc vòng thích ứng giao thành phần mà trong đó các mẫu sắc độ được lọc dựa trên các mẫu độ sáng tương ứng. Đơn vị xử lý hiện tại bao gồm đơn vị xử lý lọc vòng thích ứng được giới hạn bởi hai đường biên ảo. Các mẫu độ sáng mà nằm bên ngoài đơn vị xử lý hiện tại được sử dụng để lọc các mẫu sắc độ của đơn vị xử lý hiện tại.

Theo một số phương án, việc biến đổi tạo ra viđeo từ biểu diễn dòng bit. Theo một số phương án, việc biến đổi tạo ra biểu diễn dòng bit từ viđeo.

Một số phương án của công nghệ theo sáng chế bao gồm việc ra quyết định hoặc việc xác định để cho phép công cụ hoặc chế độ xử lý viđeo. Trong một ví dụ, khi công cụ hoặc chế độ xử lý viđeo là được phép, bộ tạo mã sẽ sử dụng hoặc thực hiện công cụ hoặc chế độ trong việc xử lý khối của viđeo, nhưng có thể không nhất thiết phải cải biến dòng bit thu được dựa trên việc sử dụng công cụ hoặc chế độ này. Nghĩa là, việc biến đổi từ khối của viđeo sang biểu diễn dòng bit của viđeo sẽ sử dụng công cụ hoặc chế độ xử lý viđeo khi được phép dựa trên quyết định hoặc việc xác định này. Trong ví dụ khác, khi công cụ hoặc chế độ xử lý viđeo là được phép, bộ giải mã sẽ xử lý dòng bit trong khi biết rằng dòng bit đã được cải biến dựa trên công cụ hoặc chế độ xử lý viđeo này. Nghĩa là, việc biến đổi từ biểu diễn dòng bit của viđeo thành khối của viđeo sẽ được thực hiện bằng cách sử dụng công cụ hoặc chế độ xử lý viđeo mà đã được phép dựa trên quyết định hoặc việc xác định này.

Một số phương án của công nghệ theo sáng chế bao gồm ra quyết định hoặc việc xác định để không cho phép công cụ hoặc chế độ xử lý viđeo. Trong một ví dụ, khi công cụ hoặc chế độ xử lý viđeo là không được phép, bộ tạo mã sẽ không sử dụng công cụ hoặc chế độ này trong việc biến đổi của khối của viđeo thành

biểu diễn dòng bit của viđeo. Trong ví dụ khác, khi công cụ hoặc chế độ xử lý viđeo là không được phép, bộ giải mã sẽ xử lý dòng bit trong khi biết rằng dòng bit đã không được cải biến bằng cách sử dụng công cụ hoặc chế độ xử lý viđeo mà đã được phép dựa trên quyết định hoặc việc xác định này.

Giải pháp được mô tả và các giải pháp khác, các ví dụ, các phương án, các môđun và các hoạt động chức năng được mô tả trong phần mô tả này có thể được thực hiện trong hệ mạch điện tử kỹ thuật số, hoặc trong phần mềm, phần sụn, hoặc phần cứng máy tính, bao gồm các kết cấu được bộc lộ trong phần mô tả này và các kết cấu tương đương của chúng, hoặc trong những sự kết hợp của một hoặc nhiều trong số chúng. Các phương án được bộc lộ và các phương án khác có thể được thực hiện dưới dạng một hoặc nhiều sản phẩm chương trình máy tính, ví dụ, một hoặc nhiều môđun của các lệnh chương trình máy tính được tạo mã trong vật ghi máy tính đọc được để được thực thi bởi, hoặc để điều khiển hoạt động của, thiết bị xử lý dữ liệu. Vật ghi máy tính đọc được có thể là thiết bị lưu trữ máy đọc được, nền lưu trữ máy đọc được, thiết bị nhớ, vật thể thực hiện tín hiệu lan truyền máy đọc được, hoặc sự kết hợp của một hoặc nhiều thiết bị này. Thuật ngữ "thiết bị xử lý dữ liệu" bao gồm tất cả thiết bị, các bộ phận, và các máy để xử lý dữ liệu, bao gồm, ví dụ, bộ xử lý lập trình được, máy tính, hoặc nhiều bộ xử lý hoặc nhiều máy tính. Ngoài phần cứng ra thì thiết bị có thể bao gồm mã mà tạo ra môi trường thực thi đối với chương trình máy tính đang được xem xét, ví dụ, mã mà cấu thành phần cứng bộ xử lý, ngăn xếp giao thức, hệ thống quản lý cơ sở dữ liệu, hệ điều hành, hoặc sự kết hợp của một hoặc nhiều trong số chúng. Tín hiệu lan truyền là tín hiệu được tạo ra một cách nhân tạo, ví dụ, tín hiệu điện, quang hoặc điện từ do máy tạo ra, nghĩa là được tạo ra để tạo mã thông tin để truyền tới thiết bị thu thích hợp.

Chương trình máy tính (còn được biết đến là chương trình, phần mềm, ứng dụng phần mềm, tập lệnh, hoặc mã) có thể được viết dưới dạng bất kỳ của ngôn ngữ lập trình, bao gồm các ngôn ngữ biên dịch hoặc diễn dịch, và có thể được khai triển dưới dạng bất kỳ, bao gồm dưới dạng chương trình độc lập hoặc dưới dạng môđun, thành phần, thủ tục con, hoặc đơn vị khác thích hợp để sử dụng trong môi trường tính toán. Chương trình máy tính không nhất thiết phải tương

ứng với tệp tin trong hệ thống tệp tin. Chương trình có thể được lưu trữ trong một phần của tệp tin mà giữ các chương trình hoặc dữ liệu khác (ví dụ, một hoặc nhiều tập lệnh được lưu trữ trong tài liệu ngôn ngữ được đánh dấu), trong một tệp tin được dành riêng cho chương trình đang được xem xét, hoặc trong nhiều tệp tin được phối hợp (ví dụ, các tệp tin mà lưu trữ một hoặc nhiều môđun, các chương trình con, hoặc các phần của mã). Chương trình máy tính có thể được khai triển để được thực thi trên một máy tính hoặc trên nhiều máy tính mà nằm ở một địa điểm hoặc được phân bố trên nhiều địa điểm và được kết nối bởi mạng truyền thông.

Các quá trình và các luồng lôgic được mô tả trong phần mô tả này có thể được thực hiện bởi một hoặc nhiều bộ xử lý lập trình được thực thi một hoặc nhiều chương trình máy tính để thực hiện các chức năng bằng cách thao tác trên dữ liệu đầu vào và tạo ra đầu ra. Các quá trình và các luồng lôgic này cũng có thể được thực hiện bởi, và thiết bị cũng có thể được thực hiện dưới dạng, hệ mạch lôgic chuyên dụng, ví dụ, mảng cổng trường lập trình được FPGA (field programmable gate array) hoặc mạch tích hợp chuyên dụng ASIC (application specific intergrated circuit).

Các bộ xử lý thích hợp để được thực thi chương trình máy tính bao gồm, ví dụ, các bộ vi xử lý cả vạn năng lẫn chuyên dụng, và một hoặc nhiều bộ xử lý bất kỳ thuộc loại bất kỳ của máy tính kỹ thuật số. Nói chung, bộ xử lý sẽ nhận các lệnh và dữ liệu từ bộ nhớ chỉ đọc hoặc bộ nhớ truy nhập ngẫu nhiên hoặc từ cả hai. Các phần tử thiết yếu của máy tính là bộ xử lý để thực hiện các lệnh và một hoặc nhiều thiết bị nhớ để lưu trữ các lệnh và dữ liệu. Nói chung, máy tính cũng sẽ bao gồm, hoặc được kết nối theo cách hoạt động được để nhận dữ liệu từ hoặc chuyển dữ liệu tới, hoặc cả hai, một hoặc nhiều bộ phận lưu trữ dung lượng lớn để lưu trữ dữ liệu, ví dụ, các đĩa từ, đĩa quang từ, hoặc các đĩa quang. Tuy nhiên, máy tính không cần có các bộ phận này. Môi trường máy tính đọc được thích hợp để lưu trữ các lệnh chương trình máy tính và dữ liệu bao gồm tất cả các dạng của bộ nhớ bất khả biến, môi trường và các thiết bị nhớ, bao gồm, ví dụ, các thiết bị nhớ bán dẫn, ví dụ, EPROM, EEPROM, và các thiết bị nhớ nhanh; các đĩa từ, ví dụ, các ổ cứng trong hoặc các ổ lưu động; các đĩa quang từ; và các đĩa CD

ROM và DVD-ROM. Bộ xử lý và bộ nhớ có thể được bổ sung bởi, hoặc được tích hợp trong, hệ mạch lôgic chuyên dụng.

Trong khi bản mô tả sáng chế này chứa nhiều chi tiết cụ thể, các chi tiết này không được diễn giải như các giới hạn đối với phạm vi của đối tượng bất kỳ hoặc của những gì có thể được yêu cầu bảo hộ, mà thay vào đó là sự mô tả các dấu hiệu mà có thể là cụ thể đối với các phương án cụ thể của các kỹ thuật cụ thể. Các dấu hiệu đã biết mà được mô tả trong bản mô tả này trong ngữ cảnh của các phương án riêng rẽ cũng có thể được thực hiện khi kết hợp trong một phương án. Ngược lại, các dấu hiệu khác nhau mà được mô tả trong ngữ cảnh của một phương án cũng có thể được thực hiện trong nhiều phương án riêng rẽ hoặc trong sự kết hợp nhỏ thích hợp bất kỳ. Hơn nữa, mặc dù các dấu hiệu có thể được mô tả trên đây dưới dạng hoạt động trong các sự kết hợp đã biết và thậm chí ban đầu được nêu như vậy, một hoặc nhiều dấu hiệu từ sự kết hợp được nêu có thể trong một số trường hợp được loại bỏ khỏi sự kết hợp này, và sự kết hợp được nêu có thể hướng đến sự kết hợp nhỏ hoặc biến thể của sự kết hợp nhỏ.

Tương tự, trong khi các hoạt động được ký hiệu trong các hình vẽ theo thứ tự cụ thể, không được hiểu rằng điều này yêu cầu rằng các hoạt động này được thực hiện theo thứ tự cụ thể đã được thể hiện hoặc theo thứ tự tuần tự, hoặc rằng tất cả các hoạt động được minh họa phải được thực hiện, để đạt được các kết quả mong muốn. Hơn nữa, việc tách rời các thành phần hệ thống khác nhau trong các phương án được mô tả trong bản mô tả này không được hiểu là yêu cầu sự tách rời như vậy trong tất cả các phương án.

Chỉ một vài phương án và các ví dụ là được mô tả và các phương án, các cải tiến và các biến thể khác có thể được thực hiện dựa trên những gì được mô tả và được thể hiện trong bản mô tả này.

YÊU CẦU BẢO HỘ

1. Phương pháp xử lý dữ liệu viđeo bao gồm các bước:

xác định, để biến đổi giữa viđeo bao gồm ảnh được phân chia thành một hoặc nhiều ảnh phụ và dòng bit của viđeo, vị trí của ảnh phụ của một hoặc nhiều ảnh phụ dựa trên phần tử cú pháp thứ nhất và phần tử cú pháp thứ hai được bao gồm trong dòng bit; và

thực hiện biến đổi dựa trên việc xác định,

trong đó phần tử cú pháp thứ nhất chỉ báo vị trí nằm ngang của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ nhất được dẫn xuất là Ceil(Log 2(S1)) bit,

trong đó phần tử cú pháp thứ hai chỉ báo vị trí dọc của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ hai được dẫn xuất là Ceil(Log 2(S2)) bit,

trong đó S1 và S2 lớn hơn 0, và CtbSizeY ký hiệu kích thước của khối cây tạo mã, và

trong đó S1 bằng (S3+CtbSizeY−1)/CtbSizeY, và S2 bằng (S4+CtbSizeY−1)/CtbSizeY, trong đó S3 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều rộng lớn nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng, và trong đó S4 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều cao cao nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng.

2. Phương pháp theo điểm 1, trong đó phần tử cú pháp thứ ba được bao gồm trong dòng bit để xác định chiều rộng của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ ba được dẫn xuất là Ceil(Log 2(S1)) bits, và

trong đó phần tử cú pháp thứ tư được bao gồm trong dòng bit để xác định chiều cao của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ tư được dẫn xuất là Ceil(Log 2(S2)) bit.

3. Phương pháp theo điểm 1, trong đó S3 và S4 lần lượt là bội số nguyên của Max(8, MinCbSizeY).

4. Phương pháp theo điểm 1, trong đó dòng bit tuân theo quy tác định dạng mà xác định rằng các ảnh phụ khác nhau có các giá trị ID ảnh phụ khác nhau.

5. Phương pháp theo điểm 4, trong đó khi giá trị của một hoặc nhiều phần tử cú pháp bằng 0, giá trị mặc định được gán cho giá trị ID ảnh phụ của ảnh phụ,

trong đó, khi giá trị của một hoặc nhiều phần tử cú pháp bằng 0, ánh xạ ID ảnh phụ không được báo hiệu tường minh, không trong tập hợp tham số chuỗi cũng không trong tập hợp tham số ảnh.

6. Phương pháp theo điểm 5, trong đó khi giá trị của một hoặc nhiều phần tử cú pháp bằng 0, phần tử cú pháp thứ năm xác định ID của ảnh phụ không được bao gồm trong tập hợp tham số ảnh và phần tử cú pháp thứ sáu xác định ID của ảnh phụ không được bao gồm trong tập hợp tham số chuỗi.

7. Phương pháp theo điểm 5, trong đó ảnh phụ là ảnh phụ thứ i trong ảnh và giá trị mặc định bằng i, và trong đó i là số nguyên nhỏ hơn số lượng ảnh phụ trong ảnh.

8. Phương pháp theo điểm 1, trong đó phép biến đổi bao gồm bước mã hóa ảnh thành dòng bit.

9. Phương pháp theo điểm 1, trong đó phép biến đổi bao gồm bước giải mã ảnh từ dòng bit.

10. Thiết bị xử lý dữ liệu viđeo bao gồm bộ xử lý và bộ nhớ bất biến có các lệnh trên đó, trong đó các lên khi thực thi bằng bộ xử lý, khiến bộ xử lý:

xác định, để biến đổi giữa viđeo bao gồm ảnh được phân chia thành một hoặc nhiều ảnh phụ và dòng bit của viđeo, vị trí của ảnh phụ của một hoặc nhiều ảnh phụ dựa trên phần tử cú pháp thứ nhất và phần tử cú pháp thứ hai được bao gồm trong dòng bit; và

thực hiện phép biến đổi dựa trên the xác định,

trong đó phần tử cú pháp thứ nhất chỉ báo vị trí nằm ngang của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ nhất được dẫn xuất là Ceil(Log 2(S1)) bit,

trong đó phần tử cú pháp thứ hai chỉ báo vị trí dọc của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ hai được dẫn xuất là Ceil(Log 2(S2)) bit,

trong đó S1 và S2 lớn hơn 0, và CtbSizeY ký hiệu kích thước của khối cây tạo mã, và

trong đó S1 bằng (S3+CtbSizeY−1)/CtbSizeY, và S2 bằng (S4+CtbSizeY−1)/CtbSizeY, trong đó S3 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều rộng lớn nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng, và trong đó S4 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều cao cao nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng.

11. Thiết bị theo điểm 10, trong đó phần tử cú pháp thứ ba được bao gồm trong dòng bit để xác định chiều rộng của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ ba được dẫn xuất là Ceil(Log 2(S1)) bit, và

trong đó phần tử cú pháp thứ tư được bao gồm trong dòng bit để xác định chiều cao của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ tư được dẫn xuất là Ceil(Log 2(S2)) bit.

12. Thiết bị theo điểm 10, trong đó S3 và S4 lần lượt là bội số nguyên của Max(8, MinCbSizeY).

13. Thiết bị theo điểm 10, trong đó dòng bit tuân theo quy tác định dạng mà xác định rằng các ảnh phụ khác nhau có các giá trị ID ảnh phụ khác nhau.

14. Thiết bị theo điểm 13, trong đó khi giá trị của một hoặc nhiều phần tử cú pháp bằng 0, giá trị mặc định được gán cho giá trị ID ảnh phụ của ảnh phụ,

trong đó khi giá trị của một hoặc nhiều phần tử cú pháp bằng 0, ánh xạ ID ảnh phụ không được báo hiệu tường minh, không nằm trong tập hợp tham số chuỗi hoặc trong tập hợp tham số ảnh.

15. Vật ghi máy tính đọc được bất biến lưu trữ các lệnh khiến bộ xử lý:

xác định, để biến đổi giữa viđeo bao gồm ảnh được phân chia thành một hoặc nhiều ảnh phụ và dòng bit của viđeo, vị trí của ảnh phụ của một hoặc nhiều ảnh phụ dựa trên phần tử cú pháp thứ nhất và phần tử cú pháp thứ hai được bao gồm trong dòng bit; và

thực hiện phép biến đổi dựa trên bước xác định,

trong đó phần tử cú pháp thứ nhất chỉ báo vị trí nằm ngang của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ nhất được dẫn xuất là Ceil(Log 2(S1)) bit,

trong đó phần tử cú pháp thứ hai chỉ báo vị trí dọc của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ hai được dẫn xuất là Ceil(Log 2(S2)) bit,

trong đó S1 và S2 lớn hơn 0, và CtbSizeY ký hiệu kích thước của khối cây tạo mã, và

trong đó S1 bằng (S3+CtbSizeY−1)/CtbSizeY, và S2 bằng (S4+CtbSizeY−1)/CtbSizeY, trong đó S3 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều rộng lớn nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng, và trong đó S4 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều cao cao nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng.

16. Vật ghi máy tính đọc được bất biến theo điểm 15, trong đó phần tử cú pháp thứ ba được bao gồm trong dòng bit để xác định chiều rộng của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ ba được dẫn xuất là Ceil(Log 2(S1)) bit, và trong đó phần tử cú pháp thứ tư được bao gồm trong dòng bit để xác định chiều cao của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ tư được dẫn xuất là Ceil(Log 2(S2)) bit.

17. Vật ghi máy tính đọc được bất biến lưu trữ dòng bit của viđeo được tạo bởi phương pháp được thực hiện bởi thiết bị xử lý viđeo, trong đó phương pháp bao gồm các bước:

xác định, đối với viđeo bao gồm ảnh được phân chia thành một hoặc nhiều ảnh phụ, vị trí của ảnh phụ của một hoặc nhiều ảnh phụ dựa trên phần tử cú pháp thứ nhất và phần tử cú pháp thứ hai được bao gồm trong dòng bit; và

tạo dòng bit dựa trên bước xác định,

trong đó phần tử cú pháp thứ nhất chỉ báo vị trí nằm ngang của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ nhất được dẫn xuất là Ceil(Log 2(S1)) bit,

trong đó phần tử cú pháp thứ hai chỉ báo vị trí dọc của đơn vị cây tạo mã trên bên trái của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ hai được dẫn xuất là Ceil(Log 2(S2)) bit,

trong đó S1 và S2 lớn hơn 0, và CtbSizeY ký hiệu kích thước của khối cây tạo mã, và

trong đó S1 bằng (S3+CtbSizeY−1)/CtbSizeY, và S2 bằng (S4+CtbSizeY−1)/CtbSizeY, trong đó S3 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều rộng lớn nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng, và trong đó S4 là giá trị của phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi và xác định chiều cao cao nhất của ảnh được giải mã viện dẫn đến tập hợp tham số chuỗi theo các đơn vị mẫu độ sáng.

18. Vật ghi máy tính đọc được bất biến theo điểm 17, trong đó phần tử cú pháp thứ ba được bao gồm trong dòng bit để xác định chiều rộng của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ ba được dẫn xuất là Ceil(Log 2(S1)) bit, và trong đó phần tử cú pháp thứ tư được bao gồm trong dòng bit để xác định chiều cao của ảnh phụ theo đơn vị CtbSizeY và độ dài của phần tử cú pháp thứ tư được dẫn xuất là Ceil(Log 2(S2)) bit.

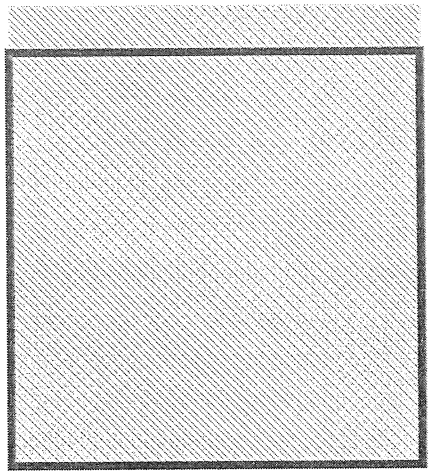

FIG. 1

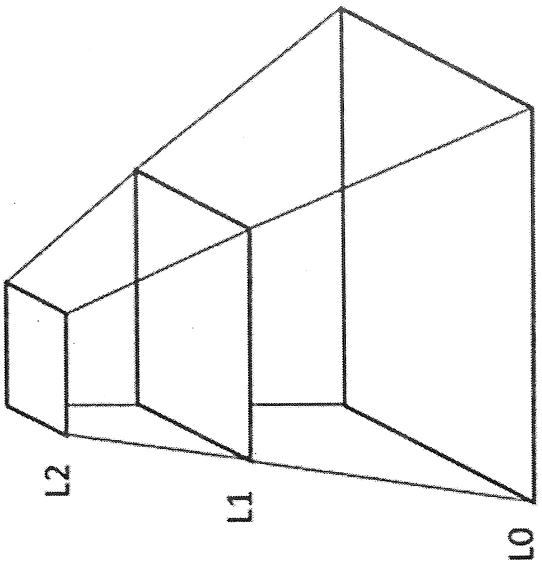

FIG. 2

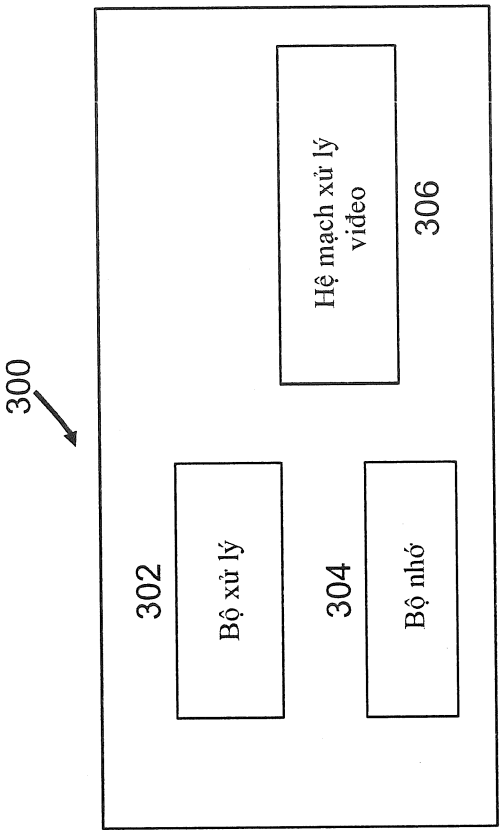

FIG. 3

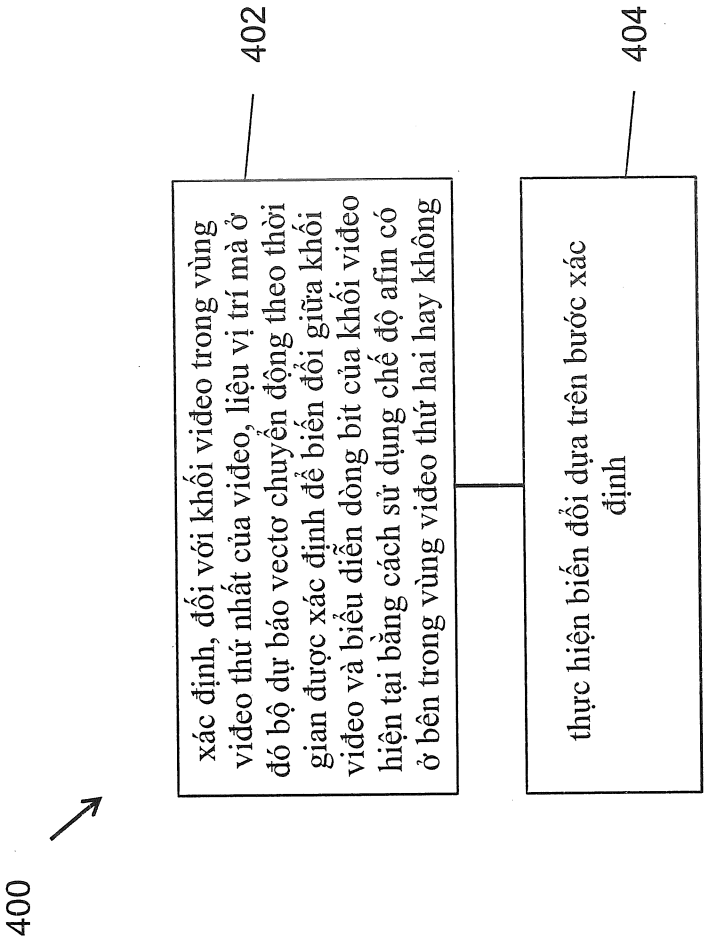
400
402
xác định, đối với khối video trong vùng video thứ nhất của video, liệu vị trí mà ở đó bộ dự báo vectơ chuyển động theo thời gian được xác định để biến đổi giữa khối video và biểu diễn dòng bit của khối video hiện tại bằng cách sử dụng chế độ affin có ở bên trong vùng video thứ hai hay không
404
thực hiện biến đổi dựa trên bước xác định

FIG. 4

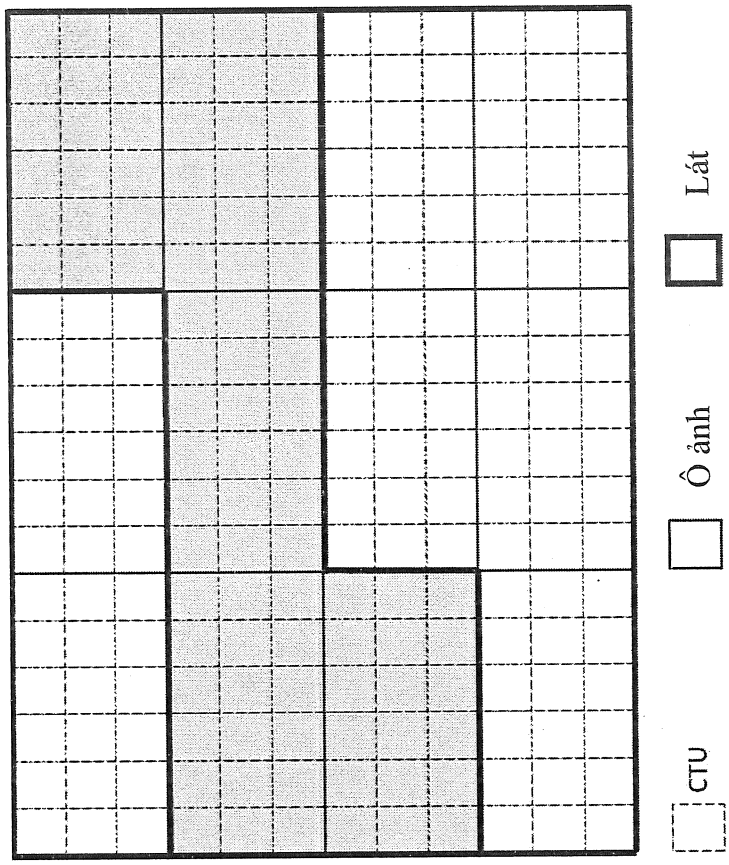

FIG. 5

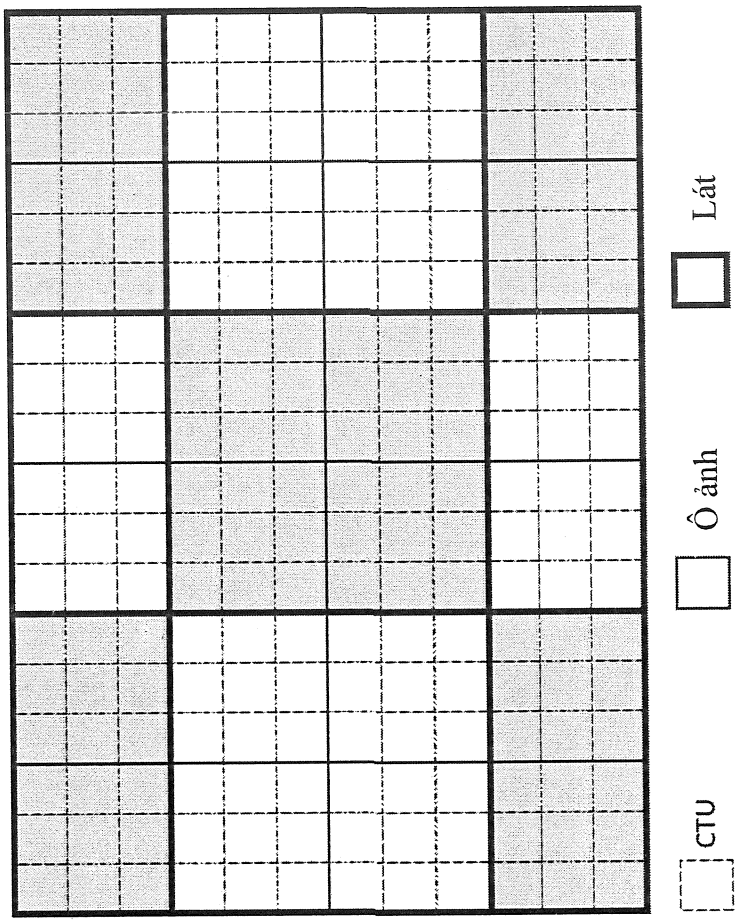

FIG. 6

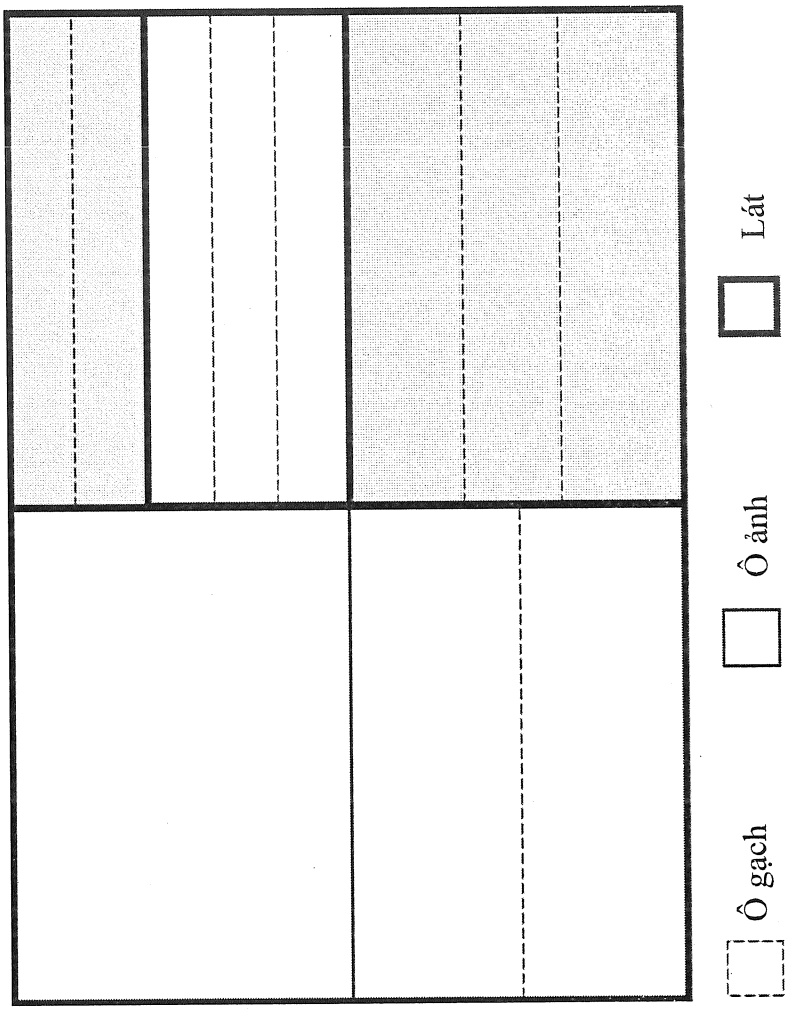

FIG. 7

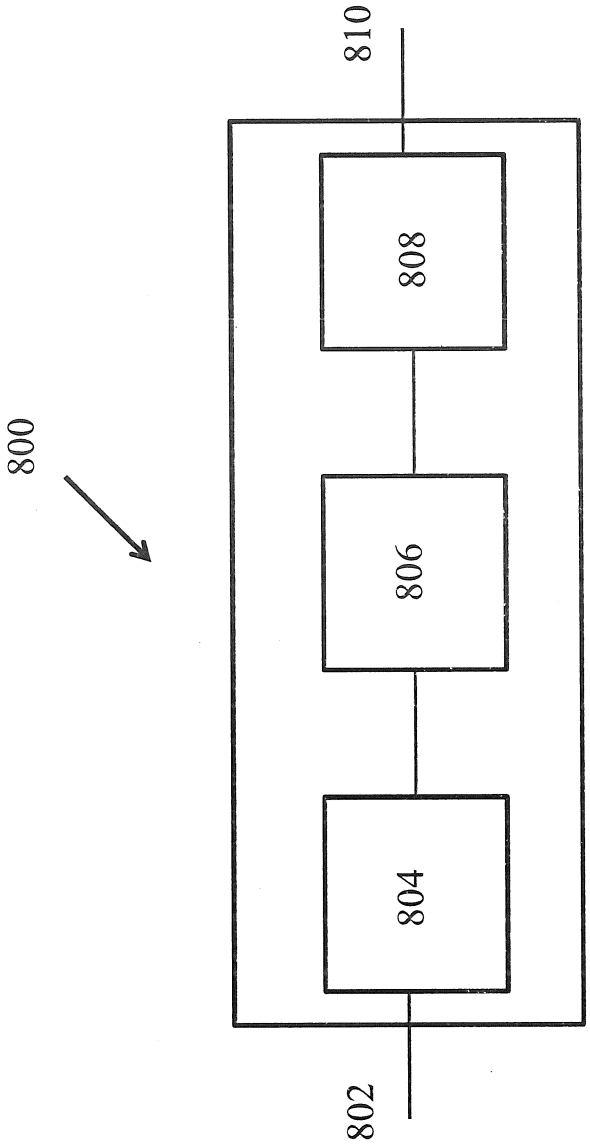

FIG. 8

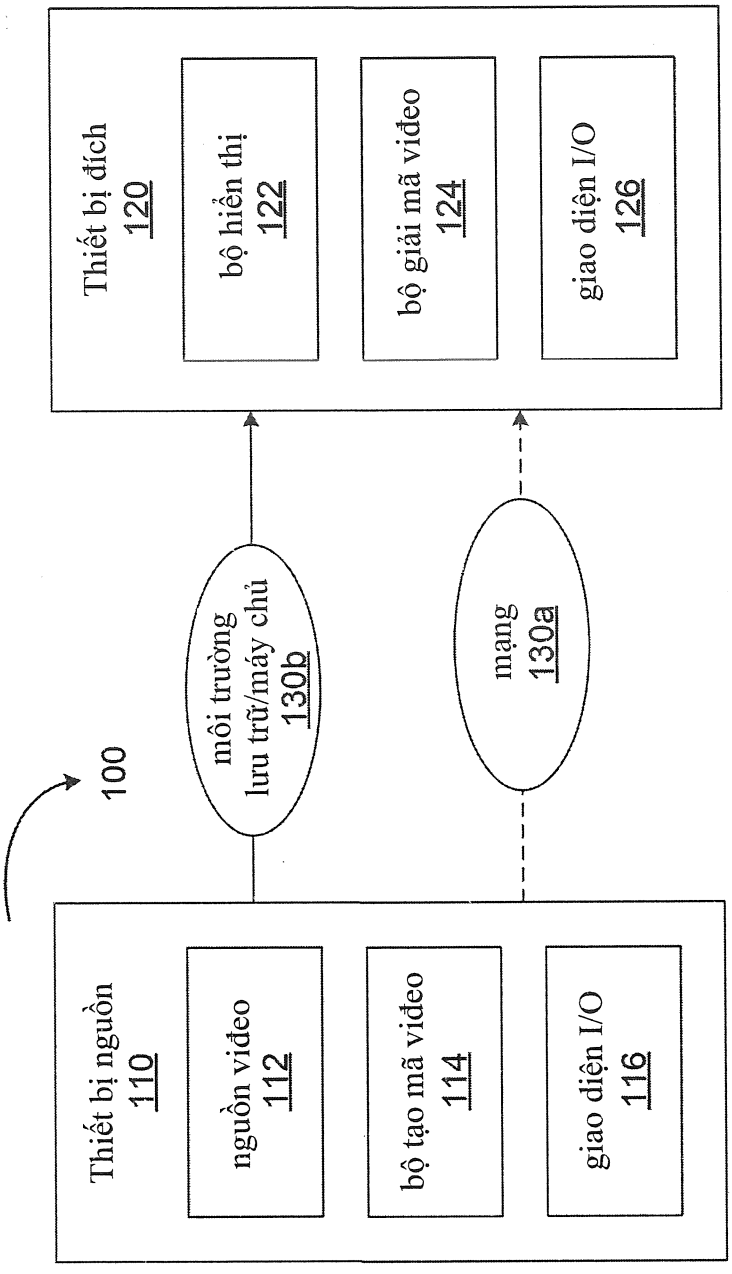
Thiết bị đích
120
bộ hiển thị
122
bộ giải mã viđeo
124
giao diện I/O
126
môi trường
lưu trữ/máy chủ
130b
mạng
130a
100
Thiết bị nguồn
110
nguồn viđeo
112
bộ tạo mã viđeo
114
giao diện I/O
116

FIG. 9

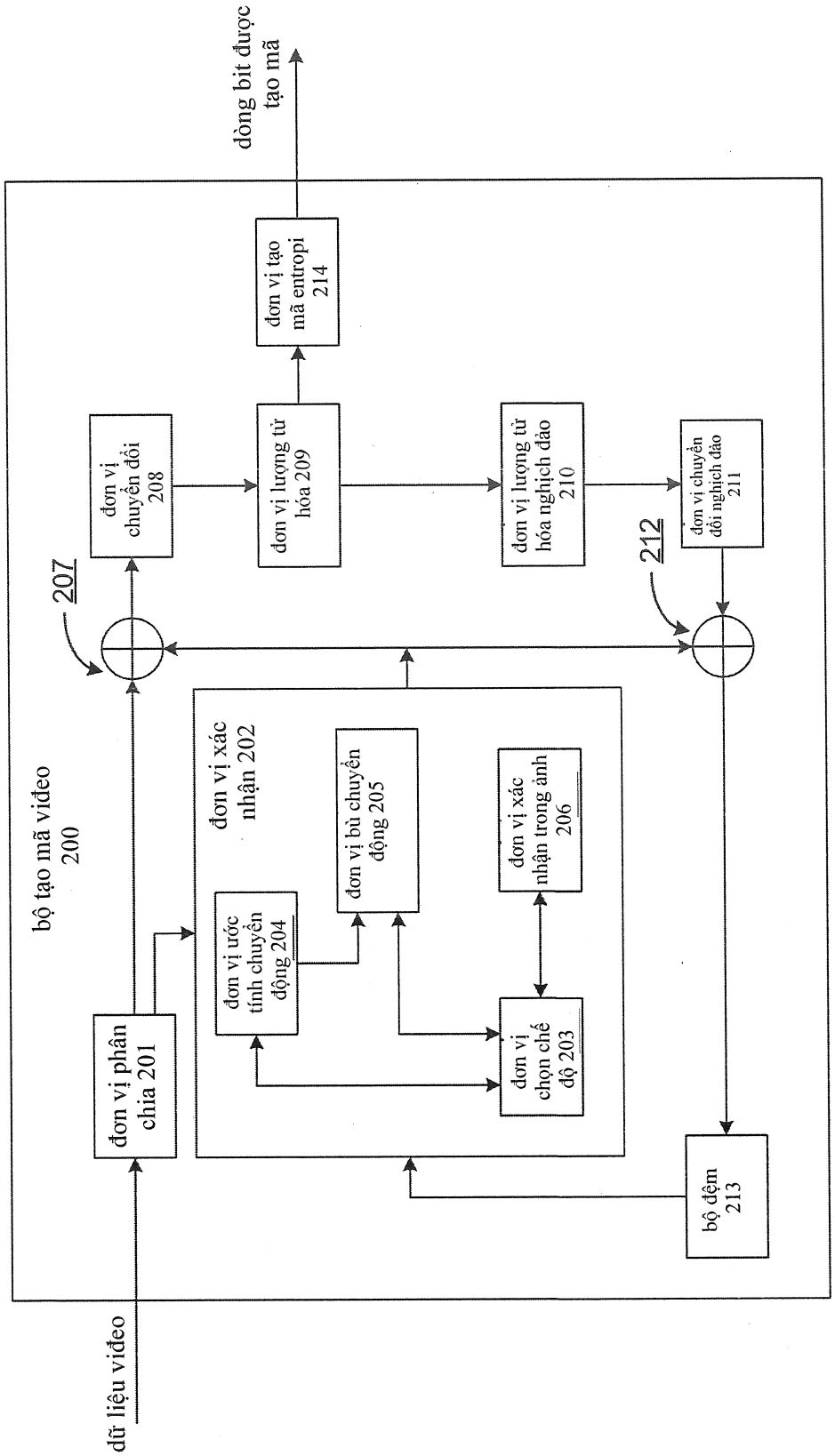
dòng bit được tạo mã
đơn vị tạo mã entropi 214
đơn vị chuyển đổi 208
đơn vị lượng tử hóa 209
đơn vị lượng tử hóa nghịch đảo 210
đơn vị chuyển đổi nghịch đảo 211
207
212
bộ tạo mã video 200
đơn vị xác nhận 202
đơn vị ước tính chuyển động 204
đơn vị bù chuyển động 205
đơn vị xác nhận trong ảnh 206
đơn vị chọn chế độ 203
đơn vị phân chia 201
bộ đệm 213
dữ liệu video

FIG. 10

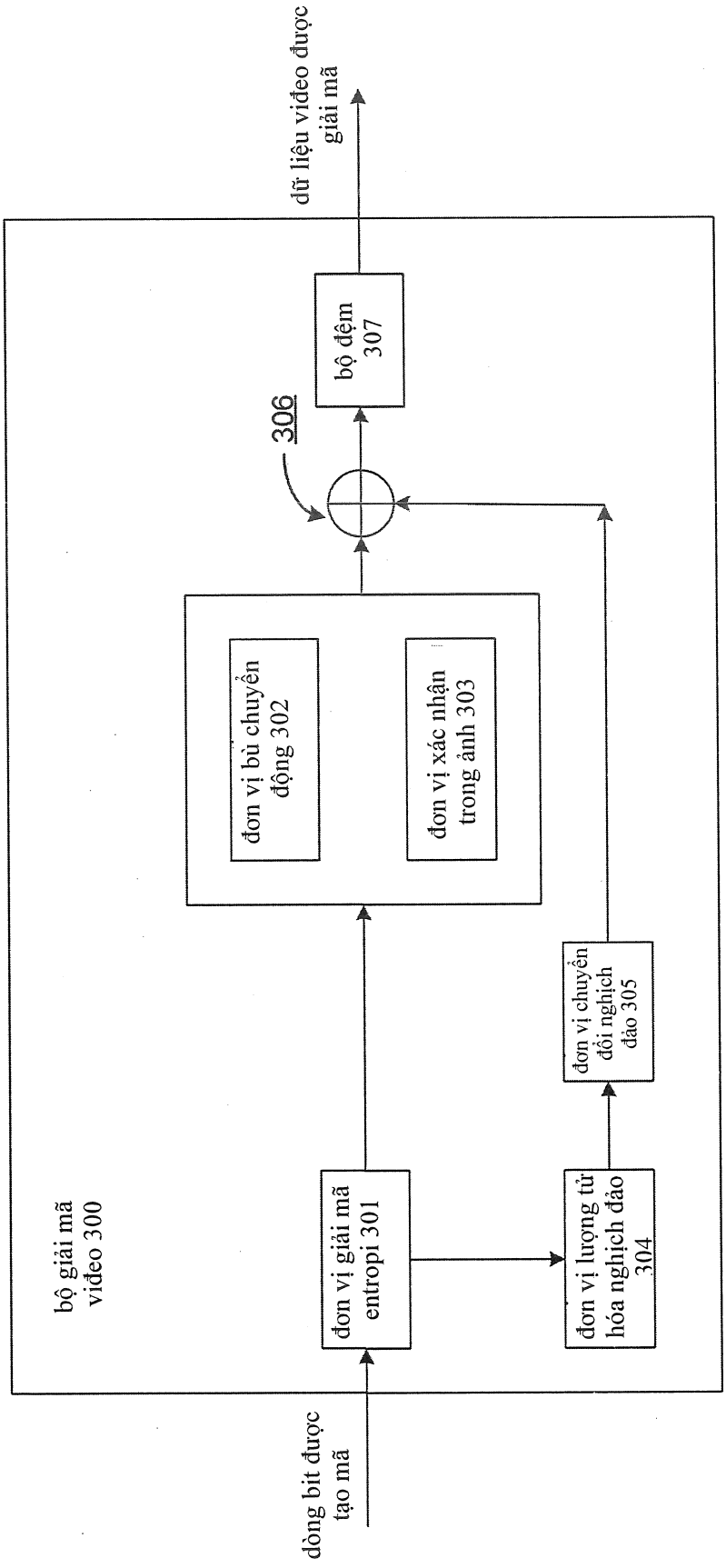
dòng bit được tạo mã
bộ giải mã video 300
đơn vị giải mã entropi 301
đơn vị lượng tử hóa nghịch đảo 304
đơn vị chuyển đổi nghịch đảo 305
đơn vị bù chuyển động 302
đơn vị xác nhận trong ảnh 303
306
bộ đệm 307
dữ liệu video được giải mã

FIG. 11

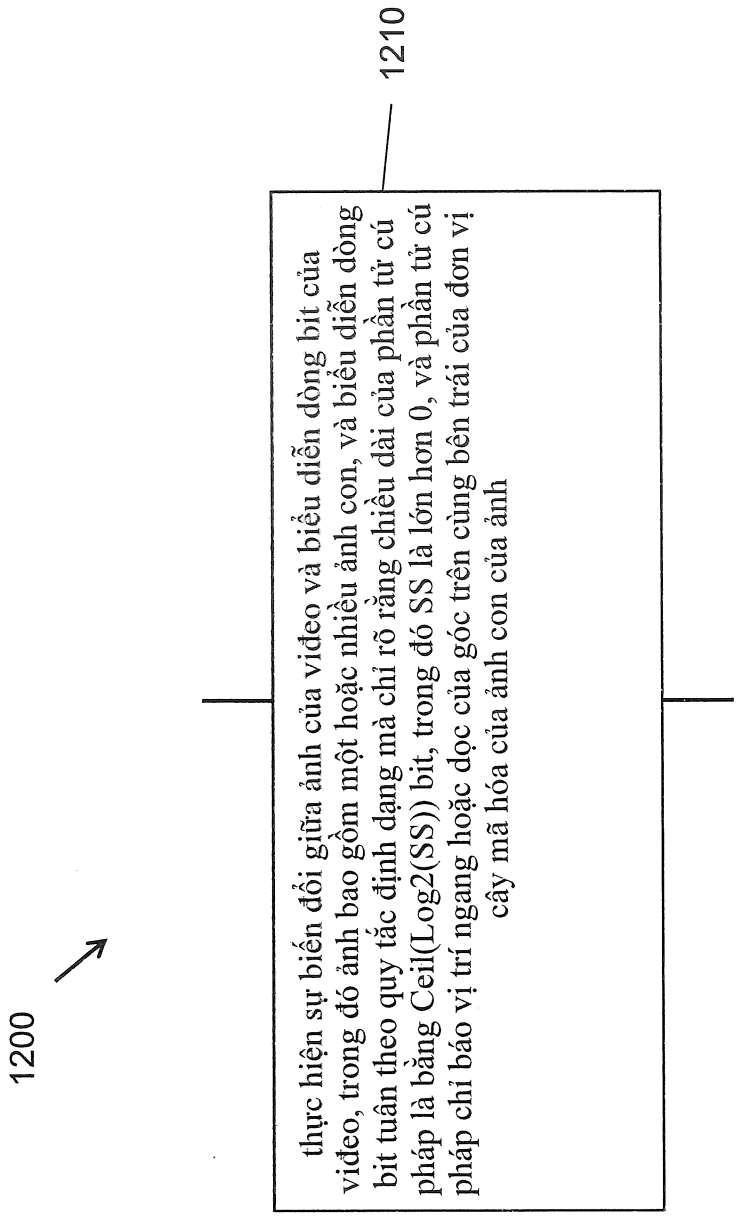
1200
1210
thực hiện sự biến đổi giữa ảnh của video và biểu diễn dòng bit của video, trong đó ảnh bao gồm một hoặc nhiều ảnh con, và biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ rõ rằng chiều dài của phần tử cú pháp là bằng Ceil(Log2(SS)) bit, trong đó SS là lớn hơn 0, và phần tử cú pháp chỉ báo vị trí ngang hoặc dọc của góc trên cùng bên trái của đơn vị cây mã hóa của ảnh con của ảnh

FIG. 12

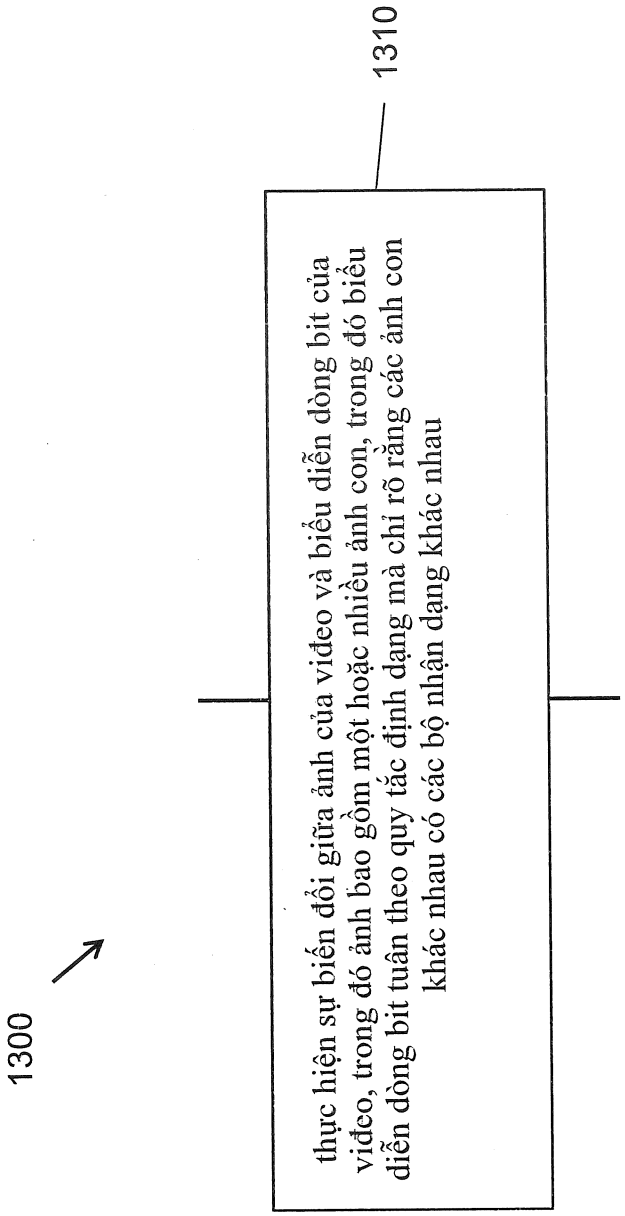
1300
1310
thực hiện sự biến đổi giữa ảnh của video và biểu diễn dòng bit của video, trong đó ảnh bao gồm một hoặc nhiều ảnh con, trong đó biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ rõ rằng các ảnh con khác nhau có các bộ nhận dạng khác nhau

FIG. 13

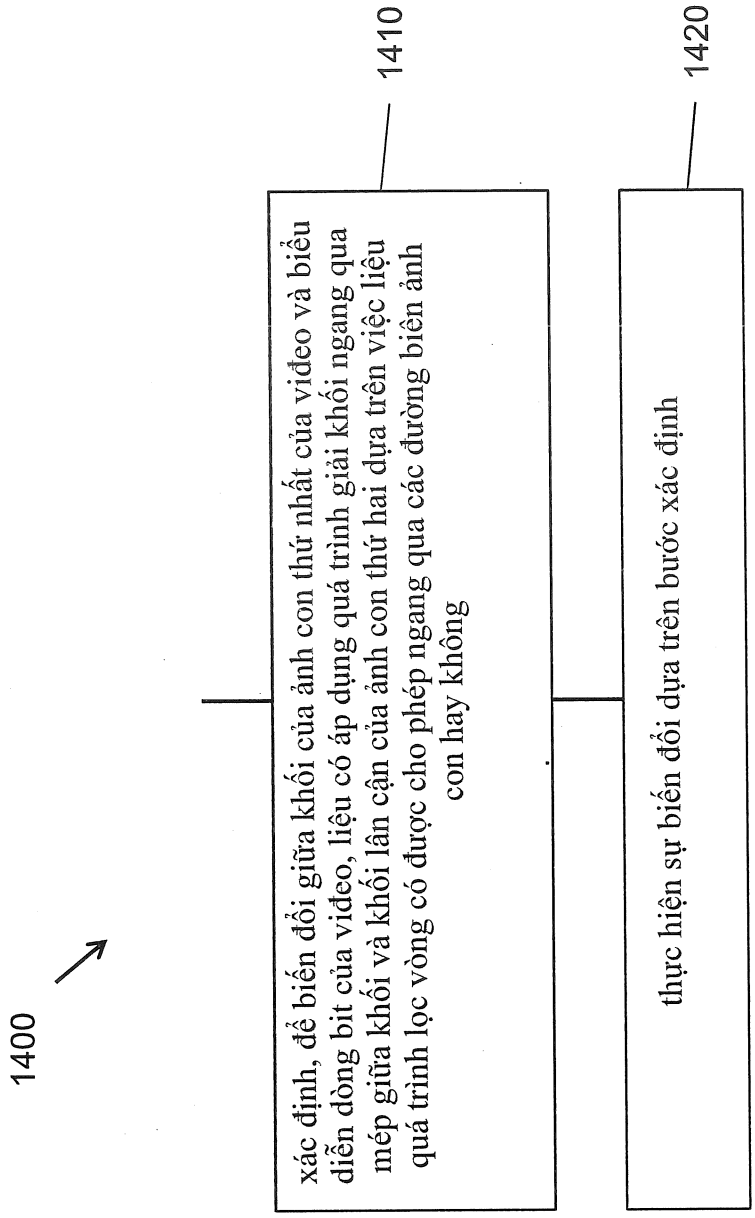
1400
1410
xác định, để biến đổi giữa khối của ảnh con thứ nhất của video và biểu diễn dòng bit của video, liệu có áp dụng quá trình giải khối ngang qua mép giữa khối và khối lân cận của ảnh con thứ hai dựa trên việc liệu quá trình lọc vòng có được cho phép ngang qua các đường biên ảnh con hay không
1420
thực hiện sự biến đổi dựa trên bước xác định

FIG. 14

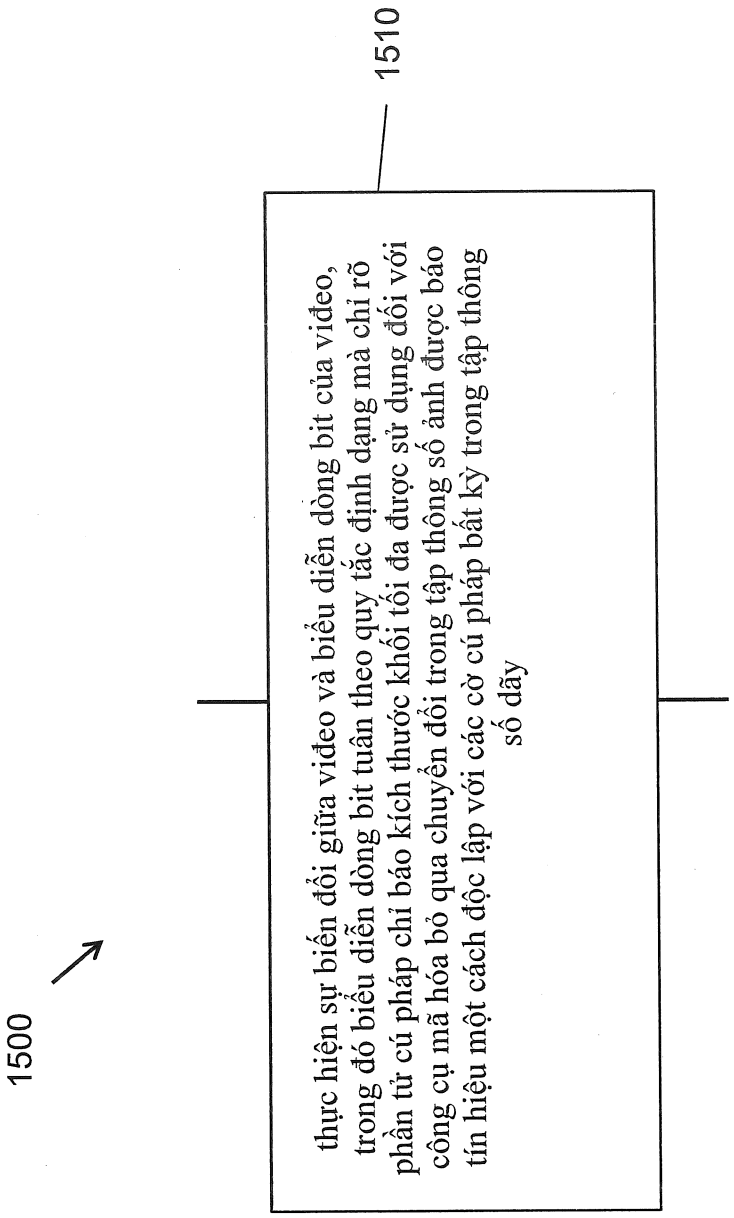
1500
1510
thực hiện sự biến đổi giữa video và biểu diễn dòng bit của video, trong đó biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ rõ phần tử cú pháp chỉ báo kích thước khối tối đa được sử dụng đối với công cụ mã hóa bỏ qua chuyển đổi trong tập thông số ảnh được báo tín hiệu một cách độc lập với các cờ cú pháp bất kỳ trong tập thông số dãy

FIG. 15

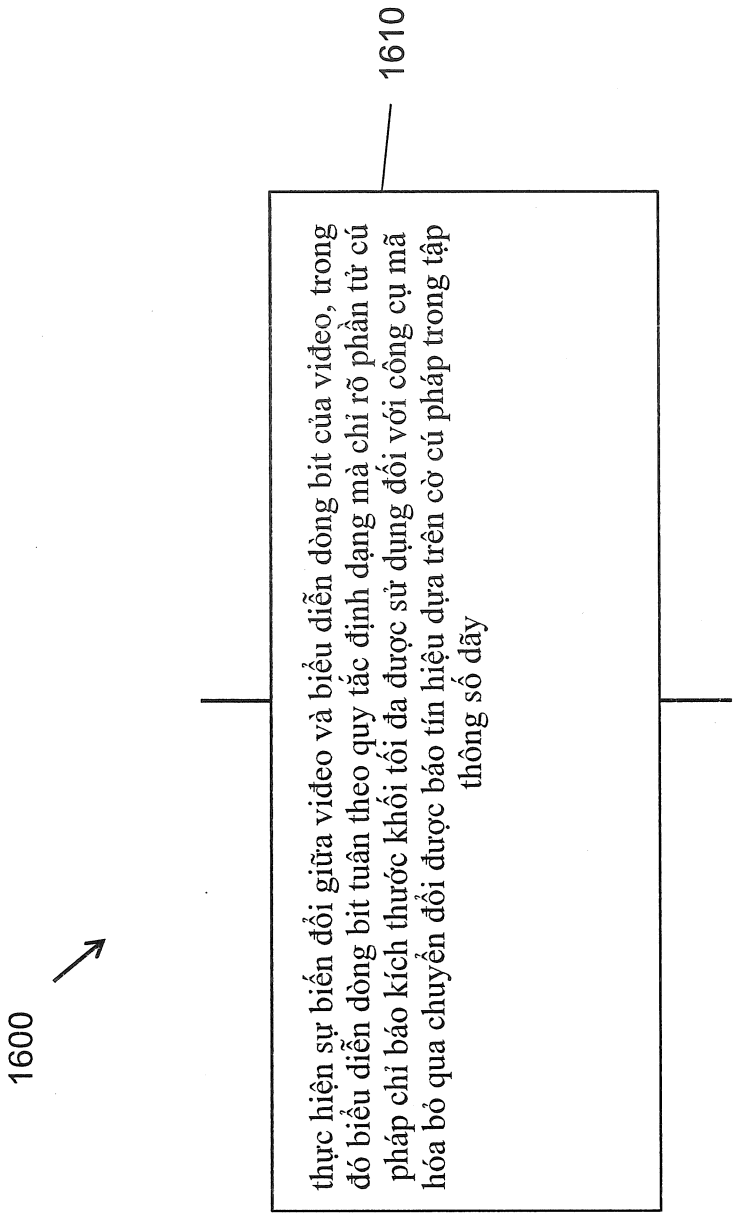
1600
1610
thực hiện sự biến đổi giữa video và biểu diễn dòng bit của video, trong đó biểu diễn dòng bit tuân theo quy tắc định dạng mà chỉ rõ phần tử cú pháp chỉ báo kích thước khối tối đa được sử dụng đối với công cụ mã hóa bỏ qua chuyển đổi được báo tín hiệu dựa trên cờ cú pháp trong tập thông số dãy

FIG. 16

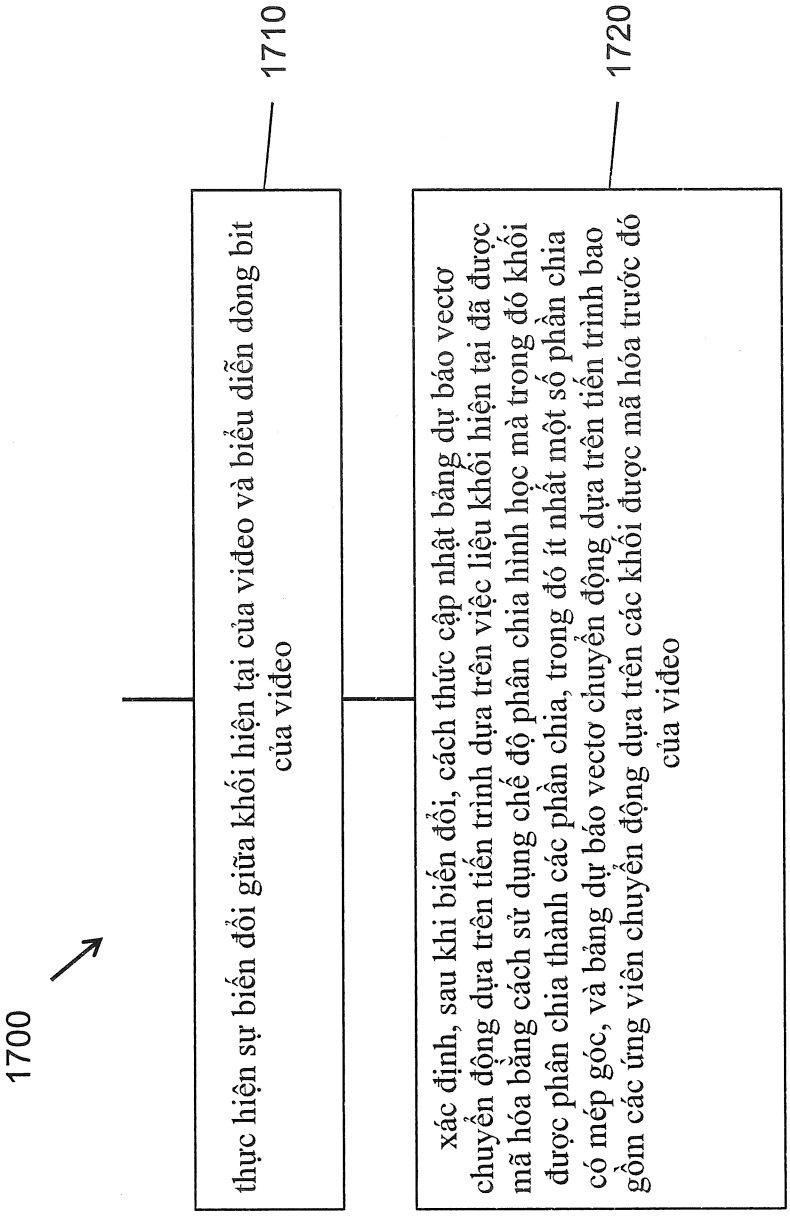
1700
1710
thực hiện sự biến đổi giữa khối hiện tại của video và biểu diễn dòng bit của video
1720
xác định, sau khi biến đổi, cách thức cập nhật bảng dự báo vectơ chuyển động dựa trên tiến trình dựa trên việc liệu khối hiện tại đã được mã hóa bằng cách sử dụng chế độ phân chia hình học mà trong đó khối được phân chia thành các phần chia, trong đó ít nhất một số phần chia có mép góc, và bảng dự báo vectơ chuyển động dựa trên tiến trình bao gồm các ứng viên chuyển động dựa trên các khối được mã hóa trước đó của video

FIG. 17

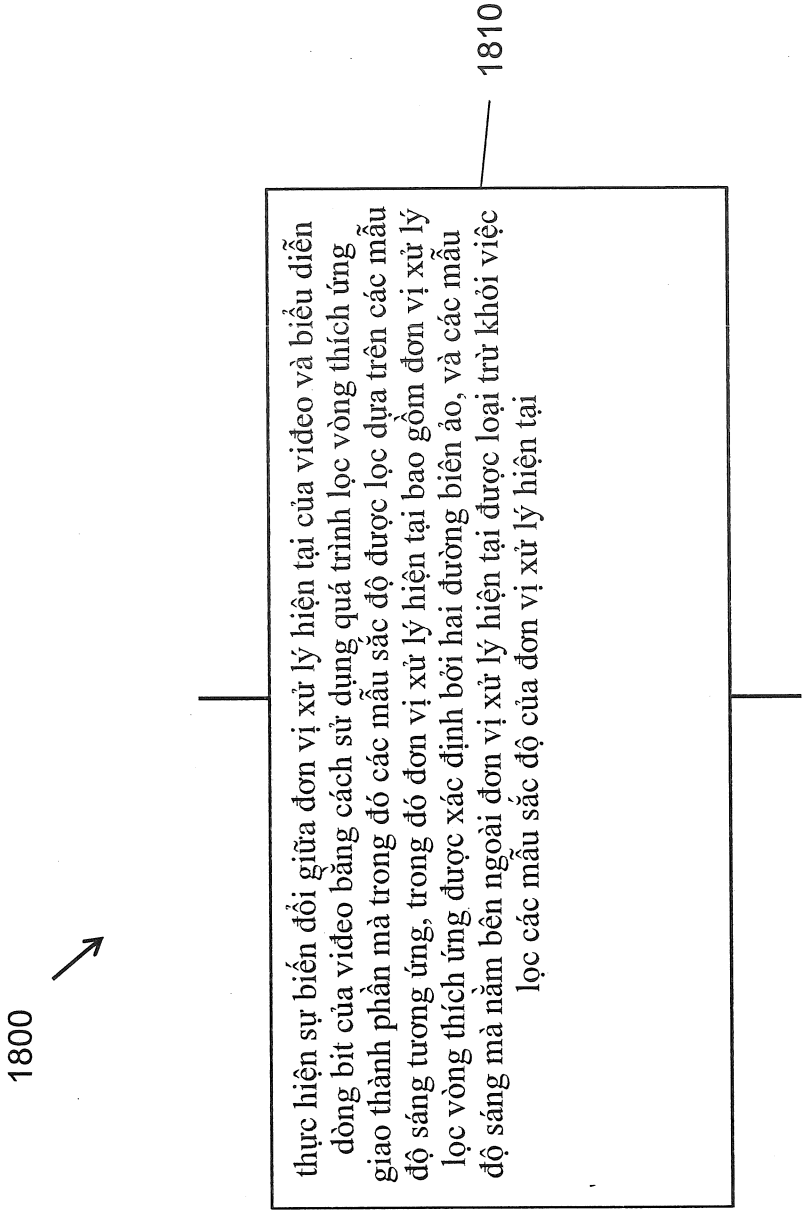
1800
1810
thực hiện sự biến đổi giữa đơn vị xử lý hiện tại của video và biểu diễn dòng bit của video bằng cách sử dụng quá trình lọc vòng thích ứng giao thành phần mà trong đó các mẫu sắc độ được lọc dựa trên các mẫu độ sáng tương ứng, trong đó đơn vị xử lý hiện tại bao gồm đơn vị xử lý lọc vòng thích ứng được xác định bởi hai đường biên ảo, và các mẫu độ sáng mà nằm bên ngoài đơn vị xử lý hiện tại được loại trừ khỏi việc lọc các mẫu sắc độ của đơn vị xử lý hiện tại

FIG. 18

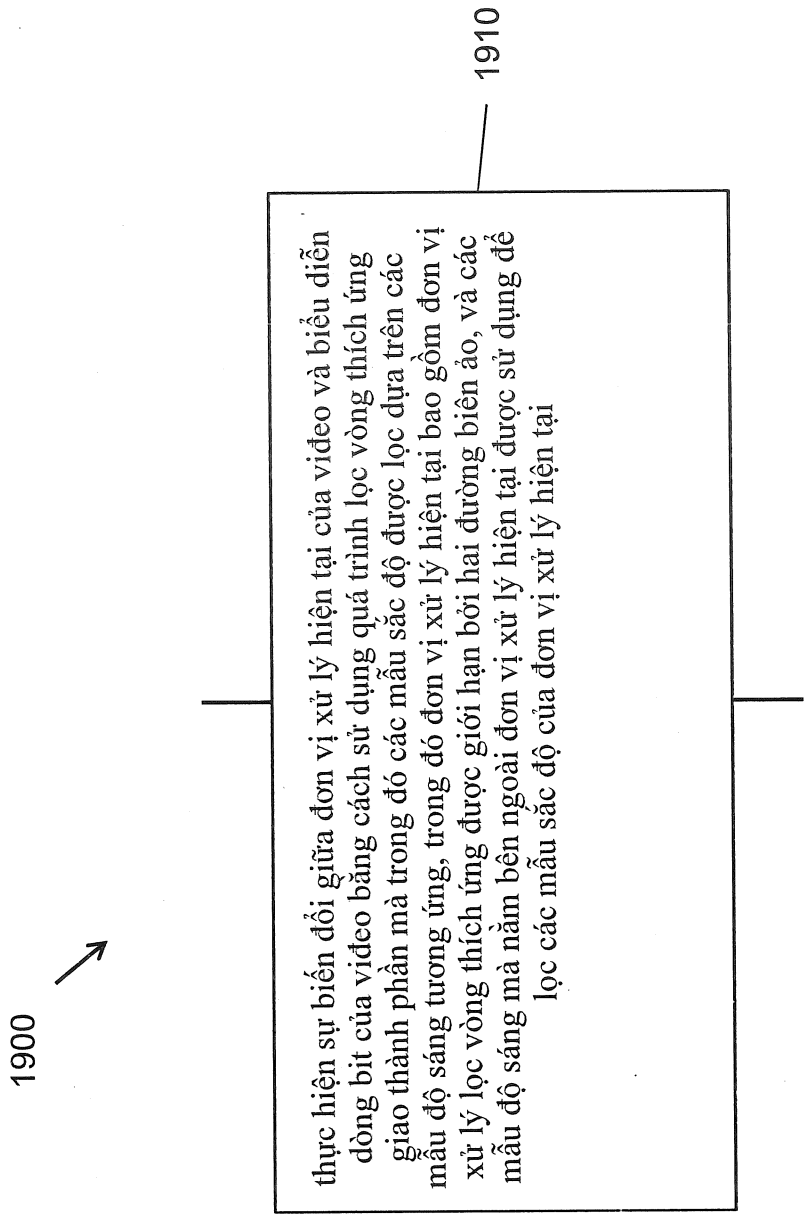
1900
1910
thực hiện sự biến đổi giữa đơn vị xử lý hiện tại của video và biểu diễn dòng bit của video bằng cách sử dụng quá trình lọc vòng thích ứng giao thành phần mà trong đó các mẫu sắc độ được lọc dựa trên các mẫu độ sáng tương ứng, trong đó đơn vị xử lý hiện tại bao gồm đơn vị xử lý lọc vòng thích ứng được giới hạn bởi hai đường biên ảo, và các mẫu độ sáng mà nằm bên ngoài đơn vị xử lý hiện tại được sử dụng để lọc các mẫu sắc độ của đơn vị xử lý hiện tại

FIG. 19