(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) **1-0052729**

(51)[2021.01] **H04N 19/117**; H04N 19/186; H04N 19/593; H04N 19/136

(13) **B**

(21) 1-2022-04713
(22) 30/12/2020
(86) PCT/RU2020/050403 30/12/2020
(87) WO 2021/045654 A3 11/03/2021
(30) 62/955,405 30/12/2019 US; 62/958,293 07/01/2020 US
(45) 27/10/2025 451
(43) 25/10/2022 415A
(73) HUAWEI TECHNOLOGIES CO., LTD. (CN)
Huawei Administration Building, Bantian, Longgang District Shenzhen, Guangdong 518129, P. R. China
(72) FILIPPOV, Alexey Konstantinovich (RU); RUFITSKIY, Vasily Alexeevich (RU); ALSHINA, Elena Alexandrovna (RU).
(74) Công ty TNHH một thành viên Sở hữu trí tuệ VCCI (VCCI-IP CO.,LTD)

(54) PHƯƠNG PHÁP DỰ ĐOÁN KHỐI SẮC ĐỘ, BỘ GIẢI MÃ, THIẾT BỊ MÃ, PHƯƠNG TIỆN BẤT BIẾN ĐỌC ĐƯỢC BỞI MÁY TÍNH VÀ THIẾT BỊ GIẢI MÃ DỮ LIỆU VIĐEO

(21) 1-2022-04713

(57) Sáng chế đề cập đến các phương pháp và thiết bị mã để dự đoán nội bộ nhờ sử dụng mô hình tuyến tính. Phương pháp này bao gồm: xác định bộ lọc đối với thành phần độ chói của khối hiện tại dựa trên khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về, và áp dụng bộ lọc được xác định tới vùng của các mẫu độ chói được khôi phục của thành phần độ chói của khối hiện tại và/hoặc các mẫu độ chói trong vị trí được lựa chọn mà lân cận với khối hiện tại, để thu nhận các mẫu độ chói được khôi phục được lọc. Phương pháp này còn bao gồm: thu nhận, dựa trên các mẫu độ chói được khôi phục được lọc, các hệ số mô hình tuyến tính. Việc dự đoán thành phần chéo được thực hiện dựa trên các hệ số mô hình tuyến tính của thu nhận mô hình tuyến tính và mẫu độ chói được khôi phục được lọc.

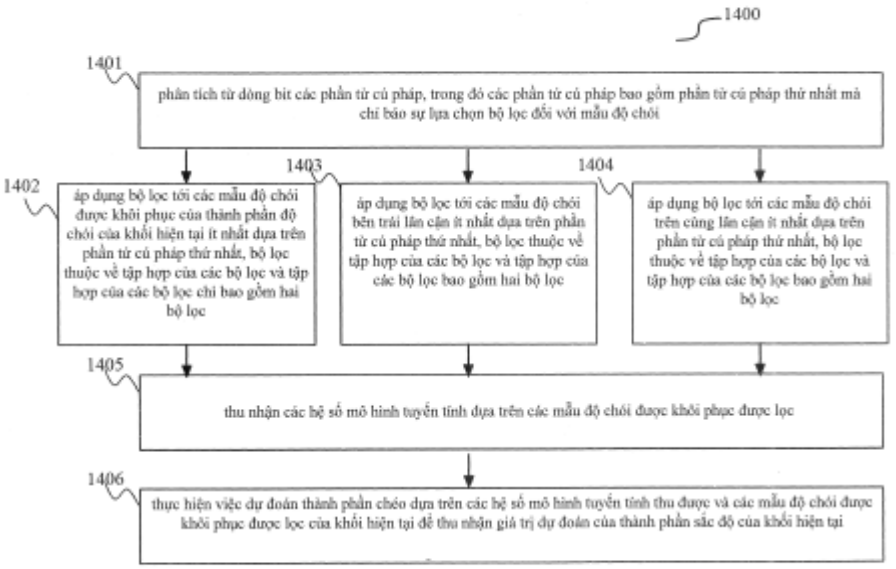


FIG. 14

Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập đến lĩnh vực xử lý ảnh và cụ thể hơn đề cập đến việc dự đoán nội bộ (như dự đoán nội bộ sắc độ) nhờ sử dụng mô hình tuyến tính thành phần chéo (CCLM-cross component linear model) và cụ thể hơn đề cập đến việc lọc không gian được sử dụng trong CCLM cho việc dự đoán nội bộ với các khuôn dạng sắc độ khác nhau.

Tình trạng kỹ thuật của sáng chế

Việc mã viđeo (mã hóa và giải mã video) được sử dụng trong phạm vi rộng của các ứng dụng viđeo số, chẳng hạn như TV quảng bá số, truyền viđeo qua internet và các mạng di động, các ứng dụng đàm thoại thời gian thực như trò chuyện viđeo, hội nghị truyền hình, DVD và các Blu-ray, hệ thống thu thập và chỉnh sửa nội dung video, và các máy quay của các ứng dụng bảo mật.

Lượng dữ liệu viđeo cần thiết để thể hiện ngay cả viđeo tương đối ngắn cũng có thể là đáng kể, điều này có thể dẫn đến khó khăn khi dữ liệu được truyền trực tuyến hoặc được truyền thông qua mạng truyền thông có dung lượng băng thông bị giới hạn. Do đó, dữ liệu viđeo thường được nén trước khi được truyền thông qua các mạng viễn thông hiện đại. Kích cỡ của viđeo cũng có thể là một vấn đề khi viđeo được lưu trữ trên thiết bị lưu trữ vì tài nguyên bộ nhớ có thể bị hạn chế. Các thiết bị nén video thường sử dụng phần mềm và/hoặc phần cứng tại nguồn để mã hóa dữ liệu viđeo trước khi truyền hoặc lưu trữ, do đó làm giảm lượng dữ liệu cần thiết để biểu diễn các ảnh viđeo số. Dữ liệu được nén sau đó sẽ được thu nhận tại đích bởi một thiết bị giải nén video mà giải mã dữ liệu viđeo. Với các tài nguyên mạng bị giới hạn và các nhu cầu ngày càng cao về

chất lượng viđeo cao hơn, các kỹ thuật nén và giải nén được cải tiến mà cải thiện tỷ lệ nén với ít hoặc không hy sinh chất lượng hình ảnh được mong muốn.

Bản chất kỹ thuật của sáng chế

Các phương án của sáng chế đề xuất các thiết bị và các phương pháp mã hóa và giải mã.

Theo khía cạnh thứ nhất của sáng chế, sáng chế đề cập đến phương pháp dự đoán nội bộ nhờ sử dụng mô hình tuyến tính, phương pháp này được thực hiện bởi thiết bị mã (cụ thể, thiết bị dự đoán nội bộ). Phương pháp này bao gồm:

- xác định bộ lọc đối với mẫu độ chói (như mỗi mẫu độ chói) thuộc về khối (tức là các mẫu bên trong của khối hiện tại), dựa trên khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về; cụ thể, các mẫu độ chói khác nhau có thể tương ứng với các bộ lọc khác nhau. Về cơ bản, phụ thuộc vào có nằm trên biên hay không.

- tại vị trí của mẫu độ chói (như mỗi mẫu độ chói) mà thuộc về khối hiện tại, áp dụng bộ lọc được xác định tới vùng của các mẫu độ chói được khôi phục, để thu nhận mẫu độ chói được khôi phục được lọc (như $Rec'_L[x, y]$);

- thu nhận, dựa trên mẫu độ chói được khôi phục được lọc, tập hợp của các mẫu độ chói được sử dụng như là đầu vào của thu nhận mô hình tuyến tính; và

thực hiện dự đoán thành phần chéo (như dự đoán sắc độ từ độ chói thành phần chéo hoặc dự đoán CCLM) dựa trên các hệ số mô hình tuyến tính của thu nhận mô hình tuyến tính và mẫu độ chói được khôi phục được lọc.

Các phương án của sáng chế đề cập đến bộ lọc sắc độ của CCLM. Sáng chế đề cập đến việc lọc các mẫu độ chói. Sáng chế đề cập đến việc lựa chọn bộ lọc mà được thực hiện bên trong CCLM.

CCLM liên quan đến dự đoán sắc độ, và sử dụng độ chói được khôi phục để dự đoán tín hiệu sắc độ,

CCLM == sắc độ từ độ chói.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó bước xác định bộ lọc, bao gồm:

Xác định bộ lọc dựa trên vị trí của mẫu độ chói trong khối hiện tại và khuôn dạng sắc độ; hoặc

xác định các bộ lọc tương ứng cho các mẫu độ chói mà thuộc về khối hiện tại, dựa trên các vị trí tương ứng của các mẫu độ chói trong khối hiện tại và khuôn dạng sắc độ. Có thể được hiểu rằng nếu các mẫu liền kề với khối hiện tại là khả dụng, bộ lọc có thể sử dụng chúng để lọc vùng biên của khối hiện tại.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó bước xác định bộ lọc, bao gồm: xác định bộ lọc dựa trên một hoặc nhiều trong số phần sau đây:

khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về,

vị trí của mẫu độ chói trong khối hiện tại,

số lượng mẫu độ chói mà thuộc về khối hiện tại,

độ rộng và độ cao của khối hiện tại, và

vị trí của mẫu sắc độ được lấy mẫu con so với mẫu độ chói trong khối hiện tại.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó khi mẫu sắc độ được lấy mẫu con không cùng vị trí với mẫu độ chói tương ứng, quan hệ thứ nhất (như bảng 4) giữa các bộ lọc và các giá trị của độ rộng và độ cao của khối hiện tại được sử dụng để xác định bộ lọc;

Khi mẫu sắc độ được lấy mẫu con có cùng vị trí với mẫu độ chói tương ứng, quan hệ thứ hai hoặc thứ ba (như các bảng 2 hoặc bảng 3) giữa các bộ lọc và các giá trị của độ rộng và độ cao của khối hiện tại được sử dụng cho việc xác định bộ lọc.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó quan hệ thứ hai hoặc thứ ba (như các bảng 2 hoặc bảng 3)

giữa các bộ lọc và các giá trị của độ rộng và độ cao của khối hiện tại được xác định trên cơ sở của số lượng các mẫu độ chói mà thuộc về khối hiện tại.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất theo sáng chế, trong đó bộ lọc bao gồm các hệ số không phải 0 tại các vị trí mà liền kề theo chiều ngang và dọc với vị trí của mẫu độ chói được khôi phục được lọc, khi thành phần sắc độ của khối hiện tại không được lấy mẫu con.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó vùng của các mẫu độ chói được khôi phục bao gồm các mẫu độ chói được khôi phục mà liên quan đến vị trí của mẫu được khôi phục được lọc, và vị trí của mẫu độ chói được khôi phục được lọc tương ứng với vị trí của mẫu độ chói mà thuộc về khối hiện tại, và vị trí của mẫu độ chói được khôi phục được lọc nằm bên trong khối độ chói của khối hiện tại.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó vùng của các mẫu độ chói được khôi phục bao gồm các mẫu độ chói được khôi phục tại các vị trí mà liền kề theo chiều ngang và dọc với vị trí của mẫu độ chói được khôi phục được lọc, và vị trí của mẫu độ chói được khôi phục được lọc tương ứng với vị trí của mẫu độ chói mà thuộc về khối hiện tại, và vị trí của mẫu độ chói được khôi phục được lọc nằm bên trong khối hiện tại (như khối độ chói hiện tại hoặc thành phần độ chói của khối hiện tại). Ví dụ, vị trí của mẫu độ chói được khôi phục được lọc nằm bên trong khối hiện tại (phần bên phải của Fig.8, có thể áp dụng bộ lọc tới các mẫu độ chói).

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó khuôn dạng sắc độ bao gồm khuôn dạng sắc độ YCbCr 4:4:4, khuôn dạng sắc độ YCbCr 4:2:0, khuôn dạng sắc độ YCbCr 4:2:2, hoặc đơn sắc.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó tập hợp của các mẫu độ chói được sử dụng như là đầu vào của thu nhận mô hình tuyến tính, bao gồm:

các mẫu được khôi phục độ chói biên mà được lấy mẫu con từ các mẫu độ chói được khôi phục được lọc (như $Rec'_L[x, y]$).

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó biến dự đoán đối với khối sắc độ hiện tại được thu nhận dựa trên:

$pred_C(i, j) = \alpha \cdot rec_L'(i, j) + \beta$

Trong đó $pred_C(i, j)$ biểu diễn mẫu sắc độ, và $rec_L(i, j)$ biểu diễn mẫu độ chói được khôi phục tương ứng.

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ nhất của sáng chế, trong đó mô hình tuyến tính là mô hình tuyến tính đa chiều (MDLM-multi-directional linear model), và các hệ số mô hình tuyến tính được sử dụng để thu nhận MDLM.

Theo khía cạnh thứ hai, sáng chế đề cập đến phương pháp mã hóa được thực hiện bởi thiết bị mã hóa, bao gồm:

thực hiện việc dự đoán nội bộ nhờ sử dụng mô hình tuyến tính (như mô hình tuyến tính thành phần chéo (CCLM), hoặc mô hình tuyến tính đa chiều (MDLM)); và

tạo ra dòng bit bao gồm các phần tử cú pháp, trong đó các phần tử cú pháp bao gồm phần tử cú pháp mà chỉ báo việc lựa chọn bộ lọc dùng cho mẫu độ chói mà thuộc về khối (như lựa chọn bộ lọc độ chói của CCLM, cụ thể, cờ SPS, như sps_cclm_colocated_chroma_flag).

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ hai của sáng chế, trong đó khi giá trị của phần tử cú pháp là 0 hoặc sai, bộ lọc được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán;

khi giá trị của phần tử cú pháp là 1 hoặc đúng, bộ lọc không được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán.

Theo khía cạnh thứ ba, sáng chế đề cập đến phương pháp giải mã được thực hiện bởi thiết bị giải mã, bao gồm:

phân tích từ dòng bit các phần tử cú pháp, trong đó các phần tử cú pháp bao gồm phần tử cú pháp mà chỉ báo việc lựa chọn bộ lọc dùng cho mẫu độ chói mà thuộc về khối (như lựa chọn bộ lọc độ chói của CCLM, cụ thể, cờ SPS, như sps_cclm_colocated_chroma_flag); và

thực hiện việc dự đoán nội bộ nhờ sử dụng mô hình tuyến tính được chỉ báo (như CCLM).

Theo cách thức có thể được thực hiện của phương pháp theo khía cạnh thứ ba của sáng chế, trong đó khi giá trị của phần tử cú pháp là 0 hoặc sai, bộ lọc được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán;

khi giá trị của phần tử cú pháp là 1 hoặc đúng, bộ lọc không được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán. Ví dụ, khi có cùng vị trí, không sử dụng bộ lọc sắc độ.

Theo khía cạnh thứ tư, sáng chế đề cập đến bộ giải mã, bao gồm:

một hoặc nhiều bộ xử lý; và

phương tiện lưu trữ bất biến đọc được bởi máy tính được ghép nối tới các bộ xử lý và lưu trữ chương trình để thực thi bởi các bộ xử lý, trong đó chương trình, khi được thực thi bởi các bộ xử lý, cấu hình bộ giải mã để thực hiện phương pháp theo khía cạnh thứ nhất hoặc thứ hai hoặc bất kỳ phương án có thể được thực hiện của khía cạnh thứ nhất, thứ hai hoặc thứ ba.

Theo khía cạnh thứ năm, sáng chế đề cập đến bộ mã hóa, bao gồm:

một hoặc nhiều bộ xử lý; và

phương tiện lưu trữ bất biến đọc được bởi máy tính được ghép nối tới các bộ xử lý và lưu trữ chương trình để thực thi bởi các bộ xử lý, trong đó chương trình, khi được thực thi bởi các bộ xử lý, cấu hình bộ mã hóa để thực hiện phương pháp theo khía cạnh thứ nhất hoặc thứ hai hoặc bất kỳ phương án có thể được thực hiện của khía cạnh thứ nhất, thứ hai hoặc thứ ba.

Theo khía cạnh thứ sáu, sáng chế đề cập đến thiết bị dự đoán nội bộ nhờ sử dụng mô hình tuyến tính, bao gồm:

- bộ xác định, có cấu trúc để xác định bộ lọc đối với mẫu độ chói (như mỗi mẫu độ chói) mà thuộc về khối, dựa trên khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về;

- bộ lọc, được tạo cấu hình tại vị trí của mẫu độ chói (như mỗi mẫu độ chói) mà thuộc về khối hiện tại, áp dụng bộ lọc được xác định tới vùng của các mẫu độ chói được khôi phục, để thu nhận mẫu độ chói được khôi phục được lọc (như $Rec'_L[x, y]$);

- bộ thu nhận, có cấu trúc để thu nhận, dựa trên mẫu độ chói được khôi phục được lọc, tập hợp của các mẫu độ chói được sử dụng như là đầu vào của thu nhận mô hình tuyến tính; và

bộ dự đoán, có cấu trúc để thực hiện dự đoán thành phần chéo (như dự đoán sắc độ từ độ chói thành phần chéo hoặc dự đoán CCLM) dựa trên các hệ số mô hình tuyến tính của thu nhận mô hình tuyến tính và mẫu độ chói được khôi phục được lọc.

Phương pháp theo khía cạnh thứ nhất của sáng chế có thể được thực hiện bởi thiết bị theo khía cạnh thứ sáu của sáng chế. Các đặc điểm và các cách thức thực hiện khác của phương pháp theo khía cạnh thứ sáu của sáng chế tương ứng với các đặc điểm và các cách thức thực hiện của thiết bị theo khía cạnh thứ nhất của sáng chế.

Theo khía cạnh khác của sáng chế, phương pháp dự đoán nội bộ nhờ sử dụng mô hình tuyến tính được đề xuất. Phương pháp này bao gồm các bước sau đây:

- xác định tập hợp của các bộ lọc lấy mẫu xuống dựa trên khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về;

- khi các mẫu độ chói trên cùng lân cận được khôi phục không khả dụng, thu nhận hai hàng trên cùng của các mẫu lân cận bằng cách tạo đệm từ khối độ chói được khôi phục;

- khi các mẫu độ chói bên trái lân cận được khôi phục không khả dụng, thu nhận ba cột bên trái của các mẫu lân cận bằng cách tạo đệm từ khối độ chói được khôi phục;

- thu nhận các mẫu độ chói được lấy mẫu xuống của các mẫu độ chói được khôi phục trong khối độ chói của khối hiện tại; và

- thu nhận các mẫu tham chiếu độ chói được lấy mẫu xuống trên cùng bằng cách áp dụng bộ lọc lấy mẫu xuống tương ứng trong số tập hợp của các bộ lọc lấy mẫu xuống tới các mẫu được lựa chọn của hai hàng trên cùng của các mẫu lân cận thu được, trong đó khi mẫu được lựa chọn nằm trong vị trí trên cùng-bên trái (1202), bộ lọc lấy mẫu xuống tương ứng được áp dụng tới kết hợp của các mẫu lân cận được khôi phục và được khôi phục; và

- thu nhận các mẫu tham chiếu độ chói được lấy mẫu xuống bên trái bằng cách áp dụng bộ lọc lấy mẫu xuống tương ứng trong số tập hợp của các bộ lọc lấy mẫu xuống tới các mẫu được lựa chọn của ba cột bên trái của các mẫu lân cận thu được;

- xác định một hoặc nhiều hệ số mô hình tuyến tính dựa trên mẫu tham chiếu độ chói được lấy mẫu xuống trên cùng hoặc bên trái hoặc kết hợp của các mẫu tham chiếu độ chói được lấy mẫu xuống trên cùng và bên trái và các mẫu tham chiếu sắc độ mà tương ứng với các mẫu tham chiếu độ chói được lấy mẫu xuống; và

thu nhận các mẫu dự đoán của khối sắc độ mà tương ứng với khối độ chói dựa trên các hệ số mô hình tuyến tính và các mẫu độ chói được lấy mẫu xuống của các mẫu độ chói được khôi phục trong khối độ chói.

Phương pháp theo khía cạnh thứ nhất của sáng chế có thể được thực hiện bởi thiết bị theo khía cạnh thứ sáu của sáng chế. Các đặc điểm và các cách thức thực hiện khác của phương pháp theo khía cạnh thứ thứ nhất của sáng chế tương ứng với các đặc điểm và các cách thức thực hiện của thiết bị theo khía cạnh thứ sáu của sáng chế.

Theo khía cạnh khác, sáng chế đề cập đến thiết bị giải mã dòng viđeo bao gồm bộ xử lý và bộ nhớ. Bộ nhớ lưu trữ các lệnh mà làm cho bộ xử lý thực hiện phương pháp theo khía cạnh thứ nhất hoặc thứ ba.

Theo khía cạnh khác, sáng chế đề cập đến thiết bị mã hóa dòng viđeo bao gồm bộ xử lý và bộ nhớ. Bộ nhớ lưu trữ các lệnh mà làm cho bộ xử lý thực hiện phương pháp theo khía cạnh thứ hai.

Theo khía cạnh khác, phương tiện lưu trữ đọc được bởi máy tính having lưu trữ trên đó các lệnh mà khi được thực thi, làm cho một hoặc nhiều bộ xử lý có cấu trúc để mã dữ liệu viđeo được đề xuất. Các lệnh làm cho một hoặc nhiều bộ xử lý thực hiện phương pháp theo khía cạnh thứ nhất hoặc thứ hai hoặc bất kỳ phương án có thể được thực hiện của khía cạnh thứ nhất, thứ hai hoặc thứ ba.

Theo khía cạnh thứ tư, sáng chế đề cập đến đến chương trình máy tính chứa mã chương trình để thực hiện phương pháp theo khía cạnh thứ nhất, thứ hai hoặc thứ ba hoặc bất kỳ phương án có thể được thực hiện của khía cạnh thứ nhất, thứ hai hoặc thứ ba khi được thực thi trên máy tính.

Theo các khía cạnh trước đó hoặc bất kỳ phương án có thể được thực hiện của các khía cạnh này, để áp dụng bộ lọc tới các mẫu độ chói được khôi phục trong khối độ chói của khối hiện tại, và/hoặc, tới các mẫu độ chói trong vị trí được lựa chọn mà lân cận với khối hiện tại, bộ lọc thuộc về tập hợp của các bộ lọc lấy mẫu xuống, và tập hợp của các bộ lọc lấy mẫu xuống chỉ bao gồm số lượng bộ lọc được làm giảm. Một cách tương ứng, có thể làm giảm số lượng nhánh đối với các bộ lọc được áp dụng tới các mẫu độ chói được khôi phục, và/hoặc, tới các mẫu độ chói trong vị trí được lựa chọn mà lân cận với khối hiện tại.

Các chi tiết của một hoặc nhiều phương án được mô tả trong các hình vẽ kèm theo và phần mô tả sau đây. Các đặc điểm, mục đích, và ưu điểm khác là rõ ràng từ bản mô tả, hình vẽ, và yêu cầu bảo hộ.

Mô tả vắn tắt các hình vẽ

Trong phần sau đây, các phương án của sáng chế được mô tả chi tiết hơn có tham chiếu đến các hình vẽ và hình vẽ đính kèm, trong đó:

FIG.1A là sơ đồ khối thể hiện ví dụ về hệ thống mã hóa viđeo có cấu trúc để thực hiện các phương án của sáng chế.

FIG.1B là sơ đồ khối thể hiện ví dụ khác của hệ thống mã hóa viđeo có cấu trúc để thực hiện các phương án của sáng chế.

FIG.2 là sơ đồ khối thể hiện ví dụ về bộ mã hóa viđeo có cấu trúc để thực hiện các phương án của sáng chế.

FIG.3 là sơ đồ khối thể hiện cấu trúc ví dụ của bộ giải mã viđeo có cấu trúc để thực hiện các phương án của sáng chế.

FIG.4 là sơ đồ khối minh họa ví dụ về thiết bị mã hóa hoặc thiết bị giải mã theo phương án của sáng chế;

FIG.5 là sơ đồ khối minh họa ví dụ khác về thiết bị mã hóa hoặc thiết bị giải mã theo phương án của sáng chế;

FIG.6 là hình vẽ minh họa khái niệm về mô hình tuyến tính thành phần chéo để dự đoán nội bộ sắc độ;

FIG.7 là hình vẽ minh họa phương pháp thu nhận tham số mô hình tuyến tính được đơn giản hóa;

FIG.8 là hình vẽ minh họa xử lý lấy mẫu uống các mẫu độ chói đối với khuôn dạng sắc độ YUV 4:2:0 và chúng tương ứng như thế nào với các mẫu sắc độ;

FIG.9 là hình vẽ minh họa các vị trí không gian của các mẫu độ chói mà được sử dụng để lọc lấy mẫu xuống trong trường hợp của khuôn dạng sắc độ YUV 4:2:0;

FIG.10A là hình vẽ minh họa các loại mẫu sắc độ khác nhau;

FIG.10B là hình vẽ minh họa các loại mẫu sắc độ khác nhau;

FIG.11 là hình vẽ minh họa phương pháp theo phương án ví dụ của sáng chế;

Fig.12A thể hiện phương án trong đó mẫu bên trái-trên cùng là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2 hoặc biên khối là biên dòng CTU;

Fig.12B thể hiện phương án trong đó biên khối không phải biên dòng CTU, mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:0 (hoặc bất kỳ khuôn dạng sắc độ khác mà sử dụng việc lấy mẫu con sắc độ dọc);

Fig.12C thể hiện phương án trong đó mẫu bên trái-trên cùng là không khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2 hoặc biên khối là biên dòng CTU;

Fig.12D thể hiện phương án trong đó biên khối không phải biên dòng CTU, mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:0 (hoặc bất kỳ khuôn dạng sắc độ khác mà sử dụng việc lấy mẫu con sắc độ dọc);

Fig.13 thể hiện hoạt động lọc đối với khối độ chói được khôi phục 1301 bởi bộ lọc ba nhánh ví dụ 1302;

FIG.14 minh họa phương pháp dự đoán khối sắc độ theo phương án của sáng chế;

FIG.15 minh họa thiết bị theo phương án của sáng chế;

FIG.16 là sơ đồ khối thể hiện cấu trúc ví dụ của hệ thống cung cấp nội dung 3100 mà thực hiện dịch vụ cung cấp nội dung; và

FIG.17 là sơ đồ khối thể hiện cấu trúc của ví dụ về thiết bị đầu cuối;

Trong phần sau đây, các dấu hiệu tham chiếu giống nhau liên quan đến các đặc tính giống nhau hoặc ít nhất là tương đương về mặt chức năng nếu không được quy định rõ ràng khác.

Mô tả chi tiết sáng chế

Trong phần mô tả sau đây, việc tham chiếu được thực hiện tới các hình vẽ kèm theo, mà tạo thành một phần của sáng chế và thể hiện, bằng cách minh họa, các khía cạnh cụ thể của các phương án của sáng chế hoặc các khía cạnh cụ thể mà trong đó các phương án của sáng chế có thể được sử dụng. Được hiểu rằng các phương án của sáng chế có thể được sử dụng trong các khía cạnh khác và bao gồm các thay đổi về cấu trúc hoặc lôgic không được thể hiện trong các hình vẽ.

Các từ viết tắt sau đây được áp dụng:

ABT: BT bất đối xứng (asymmetric BT)

AMVP: dự đoán vectơ chuyển động cải tiến (advanced motion vector prediction)

ASIC: Mạch tích hợp ứng dụng riêng (application-specific integrated circuit)

AVC: Mã hóa viđeo cải tiến (Advanced Video Coding)

B: dự đoán hai chiều (bidirectional prediction)

BT: cây nhị phân (binary tree)

CABAC: mã hóa số học nhị phân thích nghi ngữ cảnh (context-adaptive binary arithmetic coding)

CAVLC: mã hóa độ dài biến thiên thích nghi ngữ cảnh (context-adaptive variable-length coding)

CD: đĩa nén (compact disc)

CD-ROM: bộ nhớ chỉ đọc dạng đĩa nén (compact disc read-only memory)

CPU: bộ xử lý trung tâm (central processing unit)

CRT: ống tia âm cực (cathode-ray tube)

CTU: đơn vị cây mã hóa (coding tree unit)

CU: đơn vị mã hóa (coding unit)

DASH: Tạo dòng thích nghi động trên HTTP (Dynamic Adaptive Streaming over HTTP)

DCT: biến đổi côsin rời rạc (discrete cosine transform)

DMM: chế độ mô hình hóa độ sâu (depth modeling mode)

DRAM: bộ nhớ truy nhập ngẫu nhiên động (dynamic random-access memory)

DSL: đường dây thuê bao số (digital subscriber line)

DSP: bộ xử lý tín hiệu số (digital signal processor)

DVD: đĩa viđeo số (digital video disc)

EEPROM: bộ nhớ chỉ đọc khả trình có thể xóa bằng điện (electrically-erasable programmable read-only memory)

EO: điện thành quang (electrical-to-optical)

FPGA: mảng cổng khả trình dạng trường (field-programmable gate array)

FTP: Giao thức truyền tệp (File Transfer Protocol)

GOP: nhóm của các ảnh

GPB: P/B tổng quát hóa

GPU: bộ xử lý đồ họa (graphics processing unit)

HD: phân giải cao (high-definition)

HEVC High Efficiency Video Coding-Mã hóa viđeo hiệu quả cao

HM: Mô hình kiểm tra HEVC (HEVC Test Model)

I: chế độ nội bộ (intra-mode)

IC: mạch tích hợp (integrated circuit)

ISO/IEC: Tổ chức quốc tế về chuẩn hóa/Ủy ban điện kỹ thuật quốc tế

ITU-T: Ban chuẩn hóa viễn thông ủy ban viễn thông quốc tế

JVET: Đội khai thác video liên kết (Joint Video Exploration Team)

LCD: màn hình tinh thể lỏng (liquid-crystal display)

LCU: đơn vị mã hóa lớn nhất (largest coding unit)

LED: điôt phát quang (light-emitting diode)

MPEG: Nhóm chuyên gia ảnh động (Motion Picture Expert Group)

MPEG-2: Nhóm chuyên gia ảnh động 2 (Motion Picture Expert Group 2)

MPEG-4: Nhóm chuyên gia ảnh động 4 (Motion Picture Expert Group 2)

MTT: cây kiểu đa mức (multi-type tree)

mux-demux: bộ ghép kênh-bộ giải ghép kênh (multiplexer-demultiplexer)

MV: vectơ chuyển động (motion vector)

NAS: bộ lưu trữ gắn theo mạng (network-attached storage)

OE: quang thành điện (optical-to-electrical)

OLED: điôt phát quang hữu cơ (organic light-emitting diode)

PIPE: entropy phân chia khoảng cách xác suất (probability interval portioning entropy)

P: dự đoán đơn hướng (unidirectional prediction)

PPS: tập hợp tham số ảnh (picture parameter set)

PU: đơn vị dự đoán (prediction unit)

QT: cây tứ phân (quadtree, quaternary tree)

QTBT: cây nhị phân cộng cây tứ phân (quadtree plus binary tree)

RAM: bộ nhớ truy nhập ngẫu nhiên (random-access memory)

RDO: tối ưu hóa méo tỷ lệ (rate-distortion optimization)

RF: tần số vô tuyến (radio frequency)

ROM: bộ nhớ chỉ đọc (read-only memory)

Rx: bộ thu

SAD: tổng của các độ chênh lệch tuyệt đối (sum of absolute differences)

SBAC: mã hóa số học dựa trên cú pháp (syntax-based arithmetic coding)

SH: thông tin tiêu đề lát (slice header)

SPS: sequence parameter set - Tập hợp tham số chuỗi

SRAM: bộ nhớ truy nhập ngẫu nhiên tĩnh (static random-access memory)

SSD: tổng của độ chênh lệch bình phương (sum of squared differences)

SubCE: Số mũ lõi con (SubCore Experiment)

TCAM: bộ nhớ khả lập địa chỉ nội dung bậc ba (ternary content-addressable memory)

TT: cây tam phân (ternary tree)

Tx: bộ truyền

TU: đơn vị biến đổi (transform unit)

UDP: Giao thức gói dữ liệu người dùng (User Datagram Protocol)

VCEG: Nhóm chuyên gia mã hóa viđeo (Video Coding Experts Group)

VTM: Mô hình kiểm tra VVC (VVC Test Model)

VVC: Mã hóa viđeo đa năng (Versatile Video Coding).

Ví dụ, được hiểu rằng sáng chế liên quan đến phương pháp được mô tả cũng có thể đúng với thiết bị hoặc hệ thống tương ứng được tạo cấu hình để thực hiện phương pháp đó và ngược lại. Ví dụ, nếu một hoặc nhiều bước của phương pháp cụ thể được mô tả, thiết bị tương ứng có thể bao gồm một hoặc nhiều bộ phận, ví dụ các bộ phận chức năng, để thực hiện một hoặc nhiều các bước của phương pháp được mô tả (ví dụ, một bộ phận mà thực hiện một hoặc nhiều bước, hoặc mỗi trong số nhiều bộ phận mà thực hiện một hoặc nhiều bước), ngay cả nếu một hoặc nhiều bộ phận này không được mô tả rõ ràng hoặc được minh họa trong các hình vẽ. Mặt khác, ví dụ, nếu thiết bị cụ thể được mô tả dựa trên một hoặc nhiều bộ phận, ví dụ, các bộ phận chức năng, phương pháp tương ứng có thể bao gồm một bước để thực hiện chức năng của một hoặc nhiều bộ phận (ví dụ, một bước để thực hiện chức năng của một hoặc nhiều bộ phận, hoặc mỗi

trong số nhiều bước được sử dụng để thực hiện chức năng của một hoặc nhiều bộ phận), ngay cả nếu một hoặc nhiều bước này không được mô tả rõ ràng hoặc được minh họa trong các hình vẽ. Ngoài ra, sẽ được hiểu rằng các đặc điểm của các phương án ví dụ khác nhau và/hoặc các khía cạnh được mô tả ở đây có thể được kết hợp với nhau, trừ khi được mô tả cụ thể khác.

Việc mã hóa video thường liên quan đến xử lý của chuỗi ảnh, mà tạo thành viđeo hoặc chuỗi viđeo. Thay vì thuật ngữ "ảnh", thuật ngữ "khung" hoặc "hình ảnh" có thể được sử dụng là từ đồng nghĩa trong lĩnh vực mã hóa viđeo. Việc mã hóa viđeo (hoặc mã hóa nói chung) bao gồm hai phần, mã hóa viđeo và giải mã viđeo. Việc mã hóa viđeo được thực hiện tại phía nguồn, và thường bao gồm xử lý (ví dụ, bằng cách nén) ảnh video gốc để làm giảm lượng dữ liệu được yêu cầu để biểu diễn các ảnh video (để lưu trữ và/hoặc truyền hiệu quả hơn). Việc giải mã video được thực hiện tại phía đích, và thường bao gồm xử lý ngược so với bộ mã hóa để khôi phục ảnh viđeo. Các phương án đề cập đến việc "mã hóa" các ảnh video (hoặc các ảnh nói chung) sẽ được hiểu là liên quan đến việc "mã hóa" hoặc "giải mã" các ảnh video hoặc các chuỗi viđeo tương ứng. Sự kết hợp giữa phần mã hóa và phần giải mã cũng được gọi là mã hóa-giải mã (Codec-Coding và Decoding).

Trong trường hợp mã hóa viđeo không tổn hao, các ảnh viđeo gốc có thể được khôi phục, tức là các ảnh video được khôi phục có chất lượng tương tự như các ảnh viđeo gốc (giả sử không bị mất khi truyền hoặc mất dữ liệu khác trong quá trình lưu trữ hoặc truyền). Trong trường hợp mã hóa viđeo tổn hao, việc nén thêm, chẳng hạn như bằng cách lượng tử hóa, được thực hiện, để làm giảm lượng dữ liệu mà biểu diễn các ảnh viđeo, mà không thể được khôi phục hoàn toàn tại bộ giải mã, tức là chất lượng của các ảnh viđeo được khôi phục thấp hơn hoặc kém hơn so với chất lượng của các ảnh viđeo gốc.

Một số tiêu chuẩn mã hóa viđeo thuộc nhóm "mã hóa-giải mã viđeo lai tổn hao" (tức là, kết hợp dự đoán thời gian và không gian trong miền mẫu và mã hóa

biến đổi 2D để áp dụng lượng tử hóa trong miền biến đổi). Mỗi ảnh trong chuỗi viđeo thường được phân chia thành tập hợp của các khối không chồng lấn, và việc mã hóa thường được thực hiện tại cấp độ khối. Nói cách khác, tại bộ mã hóa, viđeo thường được xử lý, tức là được mã hóa, ở cấp độ khối (khối video), ví dụ, bằng cách sử dụng dự đoán không gian (nội ảnh) và/hoặc dự đoán thời gian (liên ảnh) để tạo ra khối dự đoán, trừ đi khối dự đoán từ khối hiện tại (khối hiện đang được xử lý/sẽ được xử lý) để thu được khối dư, biến đổi khối dư này và lượng tử hóa khối dư này trong miền biến đổi để giảm lượng dữ liệu cần được truyền (nén), trong khi bộ giải mã, xử lý ngược với với bộ mã hóa được áp dụng tới khối được mã hóa hoặc nén để khôi phục khối hiện tại cho việc biểu diễn. Hơn nữa, bộ mã hóa sao chép vòng xử lý của bộ giải mã sao cho cả hai sẽ tạo ra các dự đoán giống đồng nhất (ví dụ, các dự đoán nội bộ và dự đoán liên đới) và/hoặc cấu trúc lại để xử lý, tức là mã hóa, các khối tiếp theo.

Trong phần sau đây, các phương án của hệ thống mã hóa viđeo 10, bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 được mô tả dựa trên các Fig.1 đến Fig.3.

Fig.1A là sơ đồ khối giản lược minh họa hệ thống mã hóa ví dụ 10, tức là hệ thống mã hóa viđeo 10 (hoặc một cách ngắn gọn là hệ thống mã hóa 10) mà có thể sử dụng các kỹ thuật của sáng chế. Bộ mã hóa viđeo 20 (hoặc một cách ngắn gọn là bộ mã hóa 20) và bộ giải mã viđeo 30 (hoặc một cách ngắn gọn là bộ giải mã 30) của hệ thống mã hóa viđeo 10 đại diện cho các ví dụ của thiết bị có thể có cấu trúc để thực hiện các kỹ thuật theo các ví dụ khác nhau được mô tả trong sáng chế.

Như được thể hiện trong FIG.1A, hệ thống mã hóa 10 bao gồm thiết bị nguồn 12 có cấu trúc để cung cấp dữ liệu ảnh được mã hóa 21, ví dụ, tới thiết bị đích 14 để giải mã dữ liệu ảnh được mã hóa 13.

Thiết bị nguồn 12 bao gồm bộ mã hóa 20, và có thể ngoài ra, tức là, một cách tùy chọn, bao gồm nguồn ảnh 16, bộ tiền xử lý (hoặc bộ xử lý trước) 18, tức là bộ tiền xử lý ảnh 18, và giao diện truyền thông hoặc bộ truyền thông 22.

Nguồn ảnh 16 có thể bao gồm hoặc là bất kỳ loại thiết bị chụp ảnh, ví dụ, camera để chụp ảnh thế giới thực và/hoặc bất kỳ loại thiết bị tạo ảnh, ví dụ, bộ xử lý đồ họa máy tính để tạo ra hoạt ảnh máy tính, hoặc bất kỳ loại thiết bị khác để thu nhận và/hoặc cung cấp ảnh thế giới thực, ảnh do máy tính tạo ra (ví dụ, nội dung màn hình, ảnh thực tế ảo (VR-virtual reality)) và/hoặc bất kỳ sự kết hợp của chúng (ví dụ, ảnh thực tế tăng cường (AR-augmented reality)). Nguồn ảnh có thể là bất kỳ loại bộ nhớ hoặc bộ lưu trữ mà lưu trữ bất kỳ các ảnh nêu trên .

Để phân biệt với bộ tiền xử lý 18 và xử lý được thực hiện bởi bộ xử lý trước 18, ảnh hoặc dữ liệu ảnh 17 có thể được gọi là ảnh thô hoặc dữ liệu ảnh thô 17.

Bộ tiền xử lý 18 có cấu trúc để thu dữ liệu ảnh (thô) 17 và thực hiện việc tiền xử lý trên dữ liệu ảnh 17 để thu nhận ảnh được tiền xử lý 19 hoặc dữ liệu ảnh được tiền xử lý 19. Việc tiền xử lý được thực hiện bởi bộ tiền xử lý 18 có thể, ví dụ, bao gồm cắt, chuyển đổi khuôn dạng màu (tức là từ RGB thành YCbCr), hiệu chỉnh màu, hoặc khử tạp âm. Có thể được hiểu rằng bộ tiền xử lý 18 có thể là bộ phận tùy chọn.

Bộ mã hóa viđeo 20 có cấu trúc để thu dữ liệu ảnh được tiền xử lý 19 và cung cấp dữ liệu ảnh được mã hóa 21 (các chi tiết hơn nữa sẽ được mô tả dưới đây, ví dụ, dựa trên Fig.2).

Giao diện truyền thông 22 của thiết bị nguồn 12 có thể có cấu trúc để thu dữ liệu ảnh được mã hóa 21 và để truyền dữ liệu ảnh được mã hóa 21 (hoặc bất kỳ phiên bản được xử lý tiếp) trên kênh truyền thông 13 tới thiết bị khác, tức là thiết bị đích 14 hoặc bất kỳ thiết bị khác, để lưu trữ hoặc khôi phục trực tiếp.

Thiết bị đích 14 bao gồm bộ giải mã 30 (tức là bộ giải mã viđeo 30), và có thể, tức là một cách tùy chọn, bao gồm thêm giao diện truyền thông hoặc bộ truyền thông 28, bộ hậu xử lý 32 (hoặc bộ phận hậu xử lý 32) và thiết bị hiển thị 34.

Giao diện truyền thông 28 của thiết bị đích 14 có cấu trúc để thu dữ liệu ảnh được mã hóa 21 (hoặc bất kỳ phiên bản được xử lý tiếp), tức là một cách trực tiếp từ thiết bị nguồn 12 hoặc từ bất kỳ nguồn khác, tức là thiết bị lưu trữ, tức là

thiết bị lưu trữ dữ liệu ảnh được mã hóa, và cung cấp dữ liệu ảnh được mã hóa 21 tới bộ giải mã 30.

Giao diện truyền thông 22 và giao diện truyền thông 28 có thể có cấu trúc để truyền hoặc thu dữ liệu ảnh được mã hóa 21 hoặc dữ liệu được mã hóa 13 thông qua liên kết truyền thông trực tiếp giữa thiết bị nguồn 12 và thiết bị đích 14, tức là kết nối có dây hoặc không dây trực tiếp, hoặc thông qua loại mạng bất kỳ, tức là mạng có dây hoặc không dây hoặc bất kỳ kết hợp của chúng, hoặc bất kỳ loại mạng công cộng hoặc riêng tư, hoặc bất kỳ loại kết hợp của chúng.

Ví dụ, giao diện truyền thông 22 có thể có cấu trúc để đóng gói dữ liệu ảnh được mã hóa 21 thành khuôn dạng thích hợp, ví dụ như các gói tin và/hoặc xử lý dữ liệu ảnh được mã hóa bằng cách sử dụng bất kỳ loại mã hóa truyền hoặc xử lý truyền để truyền qua liên kết truyền thông hoặc mạng truyền thông.

Ví dụ, giao diện truyền thông 28, mà tạo thành phần đối xứng của giao diện truyền thông 22, có thể có cấu trúc để nhận dữ liệu được truyền và xử lý dữ liệu truyền bằng cách sử dụng bất kỳ loại giải mã hoặc xử lý truyền tương ứng và/hoặc giải gói để thu được dữ liệu ảnh được mã hóa 21.

Cả hai giao diện truyền thông 22 và giao diện truyền thông 28 có thể có cấu trúc như là giao diện truyền thông một chiều như được chỉ báo bởi mũi tên đối với kênh truyền thông 13 trong Fig.1A mà trỏ từ thiết bị nguồn 12 đến thiết bị đích 14, hoặc các giao diện truyền thông hai hướng, và có thể có cấu trúc, ví dụ để gửi và nhận các bản tin, ví dụ để thiết lập kết nối, xác nhận và trao đổi bất kỳ thông tin nào khác liên quan đến liên kết truyền thông và/hoặc truyền dữ liệu, ví dụ như truyền dữ liệu ảnh được mã hóa.

Bộ giải mã 30 có cấu trúc để thu dữ liệu ảnh được mã hóa 21 và cung cấp dữ liệu ảnh được giải mã 31 hoặc ảnh được giải mã 31 (các chi tiết khác sẽ được mô tả dưới đây, ví dụ, dựa trên Fig.3 hoặc Fig.5).

Bộ hậu xử lý 32 của thiết bị đích 14 có cấu trúc để hậu xử lý dữ liệu ảnh được giải mã 31 (còn được gọi là dữ liệu ảnh được khôi phục), ví dụ, ảnh được giải mã 31, để thu nhận dữ liệu ảnh được hậu xử lý 33, ví dụ, ảnh được hậu xử lý

33 . Việc hậu xử lý được thực hiện bởi bộ hậu xử lý 32 có thể bao gồm, ví dụ, việc chuyển đổi khuôn dạng màu (ví dụ, từ YCbCr thành RGB), hiệu chỉnh màu, biên tập, lấy mẫu lại, hoặc bất kỳ xử lý khác, ví dụ, để chuẩn bị dữ liệu ảnh được giải mã 31 cho việc hiển thị, ví dụ, bởi thiết bị hiển thị 34.

Thiết bị hiển thị 34 của thiết bị đích 14 có cấu trúc để thu dữ liệu ảnh được hậu xử lý 33 để hiển thị ảnh, ví dụ, tới người dùng, người xem. Thiết bị hiển thị 34 có thể là hoặc có thể bao gồm loại bất kỳ của màn hình để trình diễn ảnh được khôi phục, ví dụ, màn hình hoặc máy hiển thị được tích hợp hoặc phía ngoài. Các màn hình có thể, ví dụ bao gồm màn hình tinh thể lỏng (LCD-liquid crystal display), màn hình điốt phát quang hữu cơ (OLED-organic light emitting diode), màn hình plasma, máy chiếu, màn hình micro LED, tinh thể lỏng trên silicon (LCoS), bộ xử lý ánh sáng số (DLP-digital light processor) hoặc bất kỳ loại màn hình khác.

Mặc dù Fig.1A thể hiện thiết bị nguồn 12 và thiết bị đích 14 là các thiết bị riêng biệt, nhưng các phương án của các thiết bị cũng có thể bao gồm cả hai hoặc cả hai chức năng, thiết bị nguồn 12 hoặc chức năng tương ứng và thiết bị đích 14 hoặc chức năng tương ứng. Theo các phương án này, thiết bị nguồn 12 hoặc chức năng tương ứng và thiết bị đích 14 hoặc chức năng tương ứng có thể được thực hiện bằng cách sử dụng cùng phần cứng và/hoặc phần mềm hoặc bởi phần cứng và/hoặc phần mềm riêng biệt hoặc bất kỳ sự kết hợp của chúng.

Sẽ là rõ ràng đối với người có hiểu biết trung bình dựa trên phần mô tả, sự tồn tại và sự phân chia (chính xác) các chức năng của các bộ phận hoặc chức năng khác nhau trong thiết bị nguồn 12 và/hoặc thiết bị đích 14 như được thể hiện trong Fig.1A có thể khác nhau tùy thuộc vào thiết bị và ứng dụng thực tế.

Bộ mã hóa **20** (ví dụ, bộ mã hóa viđeo 20) hoặc bộ giải mã **30** (ví dụ, bộ giải mã viđeo 30) hoặc cả bộ mã hóa 20 và bộ giải mã 30 có thể được thực hiện thông qua mạch xử lý như trong Fig.1B, chẳng hạn như một hoặc nhiều bộ vi xử lý, bộ xử lý tín hiệu số (DSP-digital signal processor), mạch tích hợp ứng dụng riêng (ASIC-application-specific integrated circuit), mảng cổng lập trình dạng trường

(FPGA-field-programmable gate array), logic rời rạc, phần cứng, mã hóa viđeo chuyên dụng hoặc bất kỳ kết hợp của chúng. Bộ mã hóa 20 có thể được thực hiện thông qua mạch xử lý 46 để thể hiện các môđun khác nhau như được mô tả liên quan đến bộ mã hóa 20 của FIG.2 và/hoặc bất kỳ hệ thống mã hóa hoặc hệ thống con nào khác được mô tả ở đây. Bộ giải mã 30 có thể được thực hiện thông qua mạch xử lý 46 để thể hiện các môđun khác nhau như được mô tả liên quan đến bộ giải mã 30 của FIG.3 và/hoặc bất kỳ hệ thống giải mã hoặc hệ thống con khác được mô tả ở đây. Mạch xử lý có thể có cấu trúc để thực hiện các hoạt động khác nhau như được mô tả sau đây. Như được thể hiện trên Fig.5, nếu các kỹ thuật được thực hiện một phần trong phần mềm, thiết bị có thể lưu trữ các lệnh đối với phần mềm trong phương tiện lưu trữ bất biến đọc được bởi máy tính thích hợp, và có thể thực thi các lệnh trong phần cứng bằng cách sử dụng một hoặc nhiều bộ xử lý để thực hiện các kỹ thuật trong sáng chế. Bộ mã hóa video **20** hoặc bộ giải mã video **30** có thể được tích hợp như là một phần của bộ mã hóa/giải mã kết hợp (CODEC) trong một thiết bị, ví dụ, như thể hiện trong Fig.1B.

Thiết bị nguồn 12 và thiết bị đích 14 có thể bao gồm bất kỳ một trong số các thiết bị khác nhau, bao gồm loại bất kỳ của thiết bị cầm tay hoặc cố định, ví dụ, máy tính xách tay cỡ nhỏ hoặc máy tính laptop, điện thoại di động, điện thoại thông minh, thiết bị dạng bảng hoặc máy tính bảng, camera, máy tính để bản, hộp giải mã, tivi, thiết bị hiển thị, máy chơi nội dung đa phương tiện số, máy chơi trò chơi video, thiết bị truyền dòng viđeo (như máy chủ cung cấp nội dung hoặc máy chủ phân phát nội dung), thiết bị máy thu quảng bá, hoặc thiết bị máy truyền quảng bá, hoặc loại tương tự và có thể sử dụng hoặc có thể không sử dụng loại hệ điều hành bất kỳ. Trong một vài trường hợp, thiết bị nguồn 12 và thiết bị đích 14 có thể được trang bị cho việc truyền thông không dây. Do đó, thiết bị nguồn 12 và thiết bị đích 14 có thể là các thiết bị truyền thông không dây.

Trong một vài trường hợp, hệ thống mã hóa viđeo 10 được thể hiện trên FIG.1A chỉ là ví dụ, và các kỹ thuật trong sáng chế có thể được áp dụng tới thiết lập mã

hóa video (ví dụ, mã hóa video hoặc giải mã video) mà không cần bao gồm bất kỳ truyền thông dữ liệu giữa các thiết bị mã hóa và thiết bị giải mã. Trong các ví dụ khác, dữ liệu được truy xuất từ bộ nhớ cục bộ, được truyền trực tuyến qua mạng hoặc loại tương tự. Thiết bị mã hóa viđeo có thể mã hóa dữ liệu và lưu trữ dữ liệu tới bộ nhớ, và/hoặc thiết bị giải mã viđeo có thể gọi ra và giải mã dữ liệu từ bộ nhớ. Trong một vài ví dụ, việc mã hóa và giải mã được thực hiện bởi các thiết bị mà không truyền thông với nhau mà chỉ đơn giản mã hóa dữ liệu tới bộ nhớ và/hoặc gọi ra và giải mã dữ liệu từ bộ nhớ.

Để thuận tiện cho việc mô tả, các phương án của sáng chế được mô tả ở đây, ví dụ, bằng cách tham chiếu đến Mã hóa video hiệu quả cao (HEVC-High-Efficiency Video Coding) hoặc tham chiếu đến phần mềm tham chiếu của mã hóa video đa năng (VVC-Versatile Video coding), tiêu chuẩn mã hóa video thế hệ tiếp theo được phát triển bởi Nhóm cộng tác về mã hóa video (JCT-VC) của Nhóm chuyên gia mã hóa video ITU-T (VCEG) và Nhóm chuyên gia hình ảnh ISO/IEC (MPEG). Người có hiểu biết trung bình trong lĩnh vực kỹ thuật sẽ hiểu rằng các phương án của sáng chế không bị giới hạn ở HEVC hoặc VVC.

Bộ mã hóa và phương pháp mã hóa

Fig.2 cho thấy sơ đồ khối giản lược của bộ mã hóa viđeo ví dụ 20 mà có cấu trúc để thực hiện các kỹ thuật của sáng chế. Trong ví dụ của Fig.2, bộ mã hóa viđeo 20 bao gồm bộ đầu vào 201 (hoặc giao diện đầu vào 201), bộ tính toán phần dư 204, bộ xử lý biến đổi 206, bộ lượng tử hóa 208, bộ lượng tử hóa ngược 210, và bộ xử lý biến đổi ngược 212, bộ khôi phục 214, bộ lọc vòng 220, bộ đệm ảnh được giải mã (DPB-decoded picture buffer) 230, bộ lựa chọn chế độ 260, bộ mã hóa entropy 270 và bộ đầu ra 272 (hoặc giao diện đầu ra 272). Bộ lựa chọn chế độ (abcdef//////) 260 may include an inter prediction unit 244, an intra prediction unit 254 and a partitioning unit 262. Bộ dự đoán liên đới 244 có thể bao gồm bộ ước lượng chuyển động và bộ bù chuyển động (không được hiển thị). Bộ mã hóa viđeo 20 như được thể hiện trên FIG.2 có thể cũng được gọi là bộ mã hóa viđeo lai hoặc bộ mã hóa viđeo theo bộ mã hóa-giải mã viđeo lai.

Bộ tính toán phần dư 204, bộ xử lý biến đổi 206, bộ lượng tử hóa 208, bộ lựa chọn chế độ 260 có thể tạo thành đường truyền tín hiệu tiến của bộ mã hóa 20, trong khi đó bộ lượng tử hóa ngược 210, bộ xử lý biến đổi ngược 212, bộ khôi phục 214, bộ đệm 216, bộ lọc vòng 220, bộ đệm ảnh được giải mã (DPB) 230, bộ dự đoán liên đới 244 và bộ dự đoán nội bộ 254 có thể tạo thành đường truyền tín hiệu lùi của bộ mã hóa viđeo 20, trong đó đường truyền tín hiệu lùi của bộ mã hóa viđeo 20 tương ứng với đường truyền tín hiệu của bộ giải mã (xem bộ giải mã viđeo 30 trong Fig.3). Bộ lượng tử hóa ngược 210, bộ xử lý biến đổi ngược 212, bộ khôi phục 214, bộ lọc vòng 220, bộ đệm ảnh được giải mã (DPB) 230, bộ dự đoán liên đới 244 và bộ dự đoán nội bộ 254 cũng tạo thành "bộ giải mã lắp trong" của bộ mã hóa viđeo 20.

Ảnh & Phân chia ảnh (Các Ảnh & Khối)

Bộ mã hóa 20 có thể có cấu trúc để thu, ví dụ, thông qua bộ đầu vào 201, ảnh 17 (hoặc dữ liệu ảnh 17), tức là ảnh của các chuỗi ảnh tạo thành viđeo hoặc chuỗi viđeo. Ảnh hoặc dữ liệu ảnh thu được cũng có thể là ảnh được tiền xử lý 19 (hoặc dữ liệu ảnh được tiền xử lý 19). Để đơn giản, phần mô tả sau đây đề cập đến Fig.17. Ảnh 17 cũng có thể được gọi là ảnh hiện tại hoặc ảnh cần được mã hóa (cụ thể trong mã hóa viđeo để phân biệt ảnh hiện tại với các hình ảnh khác, ví dụ như các ảnh được mã hóa và/hoặc giải mã trước đó của cùng chuỗi viđeo, tức là chuỗi video cũng bao gồm ảnh hiện tại).

Ảnh (kỹ thuật số) là hoặc có thể được xem là mảng hoặc ma trận hai chiều của các mẫu với các giá trị cường độ. Mẫu trong mảng cũng có thể được gọi là điểm ảnh (dạng ngắn của phần tử ảnh) hoặc viết tắt là "pel". Số lượng mẫu trong các chiều (hoặc các trục) ngang và dọc của mảng hoặc ảnh xác định kích cỡ và/hoặc độ phân giải của ảnh. Đối với trình diễn màu, ba thành phần màu thường được sử dụng, tức là, ảnh có thể được biểu diễn hoặc bao gồm ba mảng mẫu. Trong khuôn dạng RBG hoặc không gian màu, ảnh bao gồm các mảng mẫu đỏ, xanh lá cây, và xanh nước biển tương ứng. Tuy nhiên, trong mã hóa video, mỗi điểm ảnh thường được biểu diễn trong khuôn dạng sắc độ và độ chói hoặc không gian

màu, ví dụ, YCbCr mà bao gồm thành phần độ chói được chỉ báo bởi Y (đôi lúc được chỉ báo bởi L) và hai thành phần sắc độ được chỉ báo bởi Cb và Cr. Thành phần độ chói (hoặc viết tắt là luma) Y biểu diễn độ sáng hoặc cường độ mức xám (ví dụ tương tự trong ảnh thang màu xám), trong khi hai thành phần sắc độ (hoặc viết tắt là chroma) Cb và Cr biểu diễn sắc độ hoặc các thành phần thông tin màu. Do đó, ảnh trong khuôn dạng YCbCr bao gồm mảng mẫu độ chói của các giá trị mẫu độ chói (Y), và hai mảng mẫu sắc độ của các giá trị sắc độ (Cb và Cr). Ảnh trong khuôn dạng RGB có thể được chuyển đổi hoặc biến đổi thành khuôn dạng YCbCr và ngược lại, xử lý này còn được gọi là chuyển đổi hoặc biến đổi màu. Nếu ảnh là đơn sắc, ảnh có thể bao gồm chỉ mảng mẫu độ chói. Do đó, ảnh có thể, ví dụ, là mảng của các mẫu độ chói trong khuôn dạng đơn sắc hoặc mảng của các mẫu đội chói và hai mảng của các mẫu sắc độ tương ứng trong khuôn dạng màu 4:2:0, 4:2:2, và 4:4:4.

Các phương án của bộ mã hóa 20 có thể bao gồm bộ phân chia ảnh (không được thể hiện trên FIG.2), có cấu trúc để phân chia ảnh 17 thành các khối ảnh (thường không chồng lấn) 203. Các khối này cũng có thể được gọi là các khối gốc, khối macrô (H.264/AVC) hoặc khối cây mã hóa (CTB) hoặc đơn vị cây mã hóa (CTU) (H.265/HEVC và VVC). Bộ phân chia có thể có cấu trúc để sử dụng kích cỡ khối giống nhau đối với tất cả ảnh của chuỗi viđeo và lưới tương ứng mà xác định kích cỡ khối hiện tại, hoặc để thay đổi kích cỡ khối hiện tại giữa các ảnh, hoặc các tập con, hoặc các nhóm ảnh, và phân chia mỗi ảnh thành các khối tương ứng.

Trong các phương án khác, bộ mã hóa viđeo có thể có cấu trúc để thu một cách trực tiếp khối 203 của ảnh 17, ví dụ, một, một vài hoặc tất cả khối tạo thành ảnh 17. Khối ảnh 203 cũng có thể được gọi là khối ảnh hiện tại hoặc khối ảnh cần được mã hóa.

Tương tự ảnh 17, khối ảnh 203 một lần nữa hoặc có thể được xem là mảng hoặc ma trận hai chiều của các mẫu với các giá trị cường độ (các giá trị mẫu), mặc dù có kích thước nhỏ hơn so với ảnh 17. Nói cách khác, khối hiện tại 203 có thể

bao gồm, ví dụ, một mảng mẫu (ví dụ một mảng độ chói trong trường hợp của ảnh đơn sắc 17, hoặc mảng độ chói hoặc mảng sắc độ trong trường hợp của ảnh màu) hoặc ba mảng mẫu (ví dụ một mảng độ chói và hai mảng sắc độ trong trường hợp của ảnh màu 17) hoặc bất kỳ số lượng và/hoặc loại mảng khác phụ thuộc vào khuôn dạng màu được áp dụng. Số lượng mẫu theo chiều (hoặc trục) ngang và dọc của khối 203 xác định kích cỡ của khối hiện tại 203. Do đó, khối có thể, ví dụ, là mảng MxN (M-cột x hàng N) của các mẫu hoặc mảng MxN của các hệ số biến đổi.

Các phương án của bộ mã hóa viđeo 20 như được thể hiện trên FIG.2 có thể có cấu trúc để mã hóa ảnh 17 theo từng khối, tức là việc mã hóa và dự đoán được thực hiện trên mỗi khối 203.

Các phương án của bộ mã hóa viđeo 20 như được thể hiện trên FIG.2 có thể còn có cấu trúc để phân chia và/hoặc mã hóa ảnh bằng cách sử dụng các lát (cũng được gọi là các lát viđeo), trong đó ảnh có thể được phân chia hoặc được mã hóa nhờ sử dụng một hoặc nhiều lát (thường không chồng lấn), và mỗi lát có thể bao gồm một hoặc nhiều khối (ví dụ các CTU).

Các phương án của bộ mã hóa viđeo 20 như được thể hiện trên FIG.2 có thể còn có cấu trúc để phân chia và/hoặc mã hóa ảnh bằng cách sử dụng các nhóm ô (cũng được gọi là các nhóm ô viđeo) và/hoặc các ô (cũng được gọi là các ô viđeo), trong đó ảnh có thể được phân chia hoặc được mã hóa nhờ sử dụng một hoặc nhiều nhóm ô (thường không xếp chồng), và mỗi nhóm ô có thể bao gồm, ví dụ một hoặc nhiều khối (ví dụ các CTU) hoặc một hoặc nhiều ô, trong đó mỗi ô, , ví dụ, có thể có dạng chữ nhật và có thể bao gồm một hoặc nhiều khối (ví dụ, các CTU), ví dụ các khối hoàn chỉnh hoặc riêng phần.

Tính toán phần dư

Bộ tính toán phần dư 204 có thể có cấu trúc để tính toán khối dư 205 (cũng được gọi là phần dư 205) dựa trên khối ảnh 203 và khối dự đoán 265 (chi tiết hơn về khối dự đoán 265 được mô tả sau đây), ví dụ bằng cách trừ các giá trị mẫu của

khối dự đoán 265 từ các giá trị mẫu của khối ảnh 203, theo từng mẫu (theo từng điểm ảnh) để thu nhận khối dư 205 trong miền mẫu.

Biến đổi

Bộ xử lý biến đổi 206 có thể có cấu trúc để áp dụng biến đổi, tức là biến đổi côsin rời rạc (DCT-discrete cosine transform) hoặc biến đổi sin rời rạc (DST-discrete sine transform), trên các giá trị mẫu của khối dư 205 để thu nhận các hệ số biến đổi 207 trong miền biến đổi. Các hệ số biến đổi 207 cũng có thể được gọi là các hệ số dư biến đổi và biểu diễn khối dư 205 trong miền biến đổi.

Bộ xử lý biến đổi 206 có thể có cấu trúc để áp dụng phép xấp xỉ nguyên của DCT/DST, như các biến đổi được chỉ rõ đối với HEVC/H.265. So với phép biến đổi DCT trực giao, các phép xấp xỉ nguyên này thường được chia tỷ lệ theo hệ số nhất định. Để duy trì định chuẩn của khối dư mà được xử lý bởi các biến đổi tiến và biến đổi ngược, hệ số tỷ lệ bổ sung được áp dụng như là một phần của xử lý biến đổi. Các hệ số biến đổi tỷ lệ thường được lựa chọn dựa trên các điều kiện định trước như các hệ số biến đổi tỷ lệ là lũy thừa của hai đối với các toán tử dịch, độ sâu bit của các hệ số biến đổi, sự cân bằng giữa độ chính xác và chi phí thực hiện, v.v. Các hệ số biến đổi tỷ lệ cụ thể, ví dụ, được chỉ rõ đối với việc biến đổi ngược, ví dụ bởi bộ xử lý biến đổi ngược 212 (và biến đổi ngược tương ứng, ví dụ bởi bộ xử lý biến đổi ngược 312 tại bộ giải mã viđeo 30) và các hệ số biến đổi tỷ lệ tương ứng đối với biến đổi tiến, ví dụ bởi bộ xử lý biến đổi 206, tại bộ mã hóa 20 có thể được chỉ rõ tương ứng.

Các phương án của bộ mã hóa viđeo 20 (tương ứng với bộ xử lý biến đổi 206) có thể có cấu trúc để xuất ra các tham số biến đổi, tức là loại biến đổi hoặc các biến đổi, ví dụ một cách trực tiếp hoặc được mã hóa hoặc được nén thông qua bộ mã hóa entropy 270, sao cho, ví dụ, bộ giải mã viđeo 30 có thể thu và sử dụng các tham số biến đổi để giải mã.

Lượng tử hóa

Bộ lượng tử hóa 208 có thể có cấu trúc để lượng tử hóa các hệ số biến đổi 207 để thu nhận các hệ số được lượng tử hóa 209, ví dụ bằng cách áp dụng lượng tử

hóa tỷ lệ hoặc lượng tử hóa vectơ. Các hệ số được lượng tử hóa 209 cũng có thể được gọi là hệ số biến đổi được lượng tử hóa 209 hoặc hệ số dư được lượng tử hóa 209.

Xử lý lượng tử hóa có thể làm giảm độ sâu bit được kết hợp với một số hoặc tất cả các hệ số biến đổi 207. Ví dụ, hệ số biến đổi n-bit có thể được làm tròn xuống thành hệ số biến đổi m-bit trong khi lượng tử hóa, trong đó n lớn hơn m. Độ lượng tử hóa có thể được cải biến bằng cách điều chỉnh tham số lượng tử hóa (quantization parameter, QP). Ví dụ, đối với lượng tử hóa tỷ lệ, các tỷ lệ khác nhau có thể được áp dụng để thu được việc lượng tử hóa tinh hơn hoặc thô hơn. Các kích cỡ bước lượng tử hóa nhỏ hơn tương ứng với việc lượng tử hóa tinh hơn, trong khi các kích cỡ bước lượng tử hóa lớn hơn tương ứng với việc lượng tử hóa thô hơn. Kích cỡ bước lượng tử hóa có thể được áp dụng có thể được chỉ báo bởi tham số lượng tử hóa (QP). Tham số lượng tử hóa có thể, ví dụ là chỉ số đối với tập hợp được xác định trước của các kích cỡ bước lượng tử hóa có thể được áp dụng. Ví dụ, các tham số lượng tử hóa nhỏ có thể tương ứng với lượng tử hóa tinh (kích cỡ bước lượng tử hóa nhỏ) và các tham số lượng tử hóa lớn có thể tương ứng với lượng tử hóa thô (bước lượng tử hóa lớn), hoặc ngược lại. Việc lượng tử hóa có thể bao gồm phân chia bởi kích cỡ bước lượng tử hóa và giải lượng tử ngược và/hoặc tương ứng, tức là bởi bộ lượng tử hóa ngược 210, có thể bao gồm việc nhân bởi kích cỡ bước lượng tử hóa. Các phương án theo một vài tiêu chuẩn, tức là HEVC, có thể có cấu trúc để sử dụng tham số lượng tử hóa để xác định kích cỡ bước lượng tử hóa. Nói chung, kích cỡ bước lượng tử hóa có thể được tính toán dựa trên tham số lượng tử hóa nhờ sử dụng phép xấp xỉ điểm cố định của phương trình bao gồm phép chia. Các hệ số tỷ lệ bổ sung có thể được đưa vào cho việc lượng tử hóa và giải lượng tử để khôi phục dạng chuẩn của khối dư, mà có thể được cải biến do việc chia tỷ lệ được sử dụng trong phép xấp xỉ điểm cố định của phương trình được sử dụng cho kích cỡ bước lượng tử hóa và tham số lượng tử hóa. Trong một cách thức thực hiện ví dụ, tỷ lệ của phép biến đổi ngược và giải lượng tử có thể được kết hợp. Ngoài ra, các bảng lượng tử hóa tùy chỉnh có thể được sử dụng và được báo hiệu từ bộ mã

hóa đến bộ giải mã, ví dụ như trong dòng bit. Lượng tử hóa là hoạt động tổn hao, trong đó tổn hao tăng lên khi kích cỡ bước lượng tử hóa tăng lên.

Các phương án của bộ mã hóa viđeo 20 (tương ứng với bộ lượng tử hóa 208) có thể có cấu trúc để xuất ra các tham số lượng tử hóa (QP), tức là một cách trực tiếp hoặc được mã hóa thông qua bộ mã hóa entropy 270, sao cho, ví dụ, bộ giải mã viđeo 30 có thể thu và áp dụng các tham số lượng tử hóa để giải mã.

Lượng tử hóa ngược

Bộ lượng tử hóa ngược 210 có cấu trúc để áp dụng lượng tử hóa ngược của bộ lượng tử hóa 208 trên các hệ số được lượng tử hóa để thu nhận các hệ số được giải lượng tử 211, ví dụ bằng cách áp dụng nghịch đảo của phương pháp lượng tử hóa mà bộ lượng tử hóa 208 áp dụng dựa trên hoặc sử dụng cùng kích cỡ bước lượng tử hóa như bộ lượng tử hóa 208. Các hệ số được giải lượng tử 211 cũng có thể được gọi là hệ số dư được giải lượng tử 211 và tương ứng - mặc dù thường không đồng nhất với các hệ số biến đổi do sự tổn hao khi lượng tử hóa - với các hệ số biến đổi 207 .

Biến đổi ngược

Bộ xử lý biến đổi ngược 212 có cấu trúc để áp dụng việc biến đổi ngược của biến đổi được áp dụng bởi bộ xử lý biến đổi 206, tức là biến đổi côsin rời rạc (DCT-discrete cosine transform) ngược hoặc biến đổi sin rời rạc (DST-discrete sine transform) ngược hoặc các biến đổi ngược khác, để thu nhận khối dư được khôi phục 213 (hoặc các hệ số được giải lượng tử tương ứng 213) trong miền mẫu. Khối dư được khôi phục 213 cũng có thể được gọi là khối biến đổi 213.

Khôi phục

Bộ khôi phục 214 (tức là bộ cộng hoặc bộ cộng tổng 214) có cấu trúc để cộng khối biến đổi 213 (tức là khối dư được khôi phục 213) tới khối dự đoán 265 để thu nhận khối được khôi phục 215 trong miền mẫu, tức là bằng cách cộng – theo từng mẫu - các giá trị mẫu của khối dư được khôi phục 213 và các giá trị mẫu của khối dự đoán 265.

Lọc

Bộ phận lọc vòng 220 (hoặc một cách ngắn gọn là "bộ lọc vòng" 220), có cấu trúc để lọc khối được khôi phục 215 để thu nhận khối được lọc 221, hoặc nói chung, lọc các mẫu được khôi phục để thu nhận các mẫu được lọc. Ví dụ, bộ lọc vòng lặp có cấu trúc để làm phẳng các chuyển tiếp điểm ảnh hoặc cải thiện chất lượng viđeo. Bộ lọc vòng 220 có thể bao gồm một hoặc nhiều bộ lọc vòng như bộ lọc giải khối, bộ lọc bù thích ứng mẫu (SAO) hoặc một hoặc nhiều bộ lọc khác, ví dụ, bộ lọc song phương, bộ lọc vòng thích ứng (ALF), các bộ lọc san bằng, làm nhạy hoặc bộ lọc cộng tác, hoặc bất kỳ sự kết hợp của chúng. Mặc dù bộ lọc vòng 220 được hiển thị trong Fig.2 là bộ lọc vòng nội bộ, trong các cấu trúc khác, bộ lọc vòng 220 có thể được thực hiện như bộ lọc vòng sau. Khối được lọc 221 cũng có thể được gọi là khối được khôi phục được lọc 221.

Các phương án của bộ mã hóa viđeo 20 (một cách tương ứng là bộ lọc vòng 220) có thể có cấu trúc để xuất ra các tham số lọc vòng (như thông tin độ dịch thích nghi mẫu), tức là một cách trực tiếp hoặc được mã hóa thông qua bộ mã hóa entropy 270, sao cho, ví dụ, bộ giải mã 30 có thể thu và áp dụng cùng tham số lọc vòng hoặc bộ lọc vòng tương ứng để giải mã.

Bộ đệm ảnh được giải mã

Bộ đệm ảnh được giải mã (DPB) 230 có thể là bộ nhớ mà lưu trữ các ảnh tham chiếu, hoặc nói chung là dữ liệu ảnh tham chiếu, để mã hóa dữ liệu viđeo bởi bộ mã hóa viđeo 20. DPB 230 có thể được tạo thành bởi bất kỳ thiết bị bộ nhớ, chẳng hạn như bộ nhớ truy nhập ngẫu nhiên động (DRAM-dynamic random access memory), bao gồm DRAM đồng bộ (SDRAM), RAM từ tính (MRAM), RAM điện trở (RRAM) hoặc các loại thiết bị bộ nhớ khác . Bộ đệm ảnh được giải mã (DPB) 230 có thể được có cấu trúc để lưu trữ một hoặc nhiều khối được lọc 221. Bộ đệm ảnh được giải mã 230 có thể còn có cấu trúc để lưu trữ các khối được lọc trước đó, ví dụ như các khối được khôi phục và được lọc trước đó 221, của cùng ảnh hiện tại hoặc của các ảnh khác nhau, ví dụ như các ảnh được khôi phục trước đó và có thể cung cấp các ảnh được khôi phục hoàn chỉnh trước đó,

tức là các ảnh được giải mã (và các khối và mẫu tham chiếu tương ứng) và/hoặc ảnh hiện tại được khôi phục một phần (và các khối và mẫu tham chiếu tương ứng), ví dụ cho việc dự đoán liên đới. Bộ đệm ảnh được giải mã (DPB) 230 cũng có thể có cấu trúc để lưu trữ một hoặc nhiều khối được khôi phục không được lọc 215, hoặc nói chung là các mẫu được khôi phục không được lọc, ví dụ, nếu khối được khôi phục 215 không được lọc bởi bộ lọc vòng 220 hoặc bất kỳ phiên bản khác được xử lý thêm của các khối hoặc mẫu được khôi phục.

Lựa chọn chế độ (Phân chia & Dự đoán)

Bộ lựa chọn chế độ 260 bao gồm bộ phân chia 262, bộ dự đoán liên đới 244 và bộ dự đoán nội bộ 254, và có cấu trúc để thu hoặc thu nhận dữ liệu ảnh gốc, tức là khối gốc 203 (khối hiện tại 203 của ảnh hiện tại 17), và dữ liệu ảnh được khôi phục, tức là các mẫu hoặc các khôi được khôi phục được lọc và/hoặc không được lọc của cùng ảnh (hiện tại) và/hoặc từ một hoặc nhiều ảnh được giải mã trước đó, tức là từ bộ đệm ảnh được giải mã 230 hoặc các bộ đệm khác (tức là bộ đệm dòng, không được thể hiện).. Dữ liệu ảnh được khôi phục được sử dụng như là dữ liệu ảnh tham chiếu cho việc dự đoán, tức là dự đoán liên đới hoặc dự đoán nội bộ, để thu nhận khối dự đoán 265 hoặc biến dự đoán 265.

Bộ lựa chọn chế độ 260 có thể có cấu trúc để xác định hoặc lựa chọn phân vùng đối với chế độ dự đoán khối hiện tại (bao gồm không phân chia) và chế độ dự đoán (tức là chế độ dự đoán liên đới hoặc nội bộ) và tạo ra khối dự đoán tương ứng 265, mà được sử dụng cho việc tính toán của khối dư 205 và cho việc khôi phục của khối được khôi phục 215.

Các phương án của bộ lựa chọn chế độ 260 có thể có cấu trúc để lựa chọn chế độ phân chia và dự đoán (ví dụ, từ những phương án được hỗ trợ hoặc khả dụng đối với bộ lựa chọn chế độ 260), cung cấp kết quả phù hợp nhất hoặc nói cách khác là phần dư nhỏ nhất (phần dư nhỏ nhất có nghĩa là việc nén tốt hơn để truyền hoặc lưu trữ), hoặc thông tin tiêu đề báo hiệu nhỏ nhất (thông tin tiêu đề báo hiệu nhỏ nhất có nghĩa là việc nén tốt hơn để truyền hoặc lưu trữ), mà xem

xét hoặc cân bằng cả hai. Bộ lựa chọn chế độ 260 có thể có cấu trúc để xác định chế độ phân chia và dự đoán dựa trên tối ưu hóa méo tỷ lệ (RDO-rate distortion optimization), tức là lựa chọn chế độ dự đoán mà cung cấp méo tỷ lệ nhỏ nhất. Các thuật ngữ như "tốt nhất", "nhỏ nhất", "tối ưu", v.v. trong ngữ cảnh này không nhất thiết đề cập đến "tốt nhất", "nhỏ nhất", "tối ưu" chung v.v. nhưng cũng có thể đề cập đến việc hoàn thành tiêu chuẩn chấm dứt hoặc lựa chọn như giá trị vượt quá hoặc nằm dưới ngưỡng hoặc các điều kiện khác có khả năng dẫn đến "lựa chọn dưới mức tối ưu" nhưng giảm độ phức tạp và thời gian xử lý.

Nói cách khác, bộ phân chia 262 có thể có cấu trúc để phân chia khối hiện tại 203 thành các phân vùng khối nhỏ hơn hoặc các khối con (mà tạo thành khối lần nữa), tức là sử dụng lặp lại việc phân chia cây tứ phân (QT-quad-tree), phân chia cây nhị phân (Bt-binary tree) hoặc phân chia cây tam phân (TT-triple-tree) hoặc bất kỳ kết hợp của chúng, và để thực hiện, ví dụ, việc dự đoán đối với mỗi phân vùng khối hiện tại hoặc các khối con, trong đó việc lựa chọn chế độ bao gồm việc lựa chọn của cấu trúc cây của khối được phân chia 203 và các chế độ dự đoán được áp dụng tới mỗi phân vùng khối hiện tại hoặc các khối con.

Trong phần sau đây, xử lý phân chia (ví dụ, bởi bộ phân chia 260) và xử lý dự đoán (bởi bộ dự đoán liên đới 244 và bộ dự đoán nội bộ 254) được thực hiện bởi bộ mã hóa viđeo ví dụ 20 sẽ được giải thích trong chi tiết hơn.

Phân chia

Bộ phân chia 262 có thể phân chia (hoặc tách) khối hiện tại 203 thành các phân vùng nhỏ hơn, ví dụ các khối nhỏ hơn có kích cỡ hình vuông hoặc hình chữ nhật. Các khối nhỏ hơn này (mà cũng có thể được gọi là các khối con) có thể được phân chia tiếp thành các phân vùng nhỏ hơn nữa. Điều này cũng được gọi là phân chia dạng cây hoặc phân chia dạng cây phân cấp, trong đó khối gốc, tức là tại mức cây gốc 0 (mức phân cấp 0, độ sâu 0), có thể được phân chia đệ quy, tức là được phân chia thành hai khối hoặc nhiều hơn của mức cây thấp hơn tiếp theo, tức là các nút tại mức cây 1 (mức phân cấp 1, độ sâu 1), trong đó các khối này có thể được phân chia tiếp thành hai khối hoặc nhiều hơn của mức thấp hơn

tiếp theo, tức là mức cây 2 (mức phân cấp 2, độ sâu 2), v.v. cho đến khi việc phân chia được kết thúc, ví dụ do tiêu chuẩn kết thúc được thỏa mãn, ví dụ, đạt tới độ sâu cây lớn nhất hoặc kích cỡ khối nhỏ nhất. Các khối mà không được phân chia tiếp cũng được gọi là các khối nhánh hoặc nút nhánh của cây. Cây mà sử dụng phân chia thành hai phân vùng được gọi là cây nhị phân (BT-binary-tree), cây mà sử dụng phân chia thành ba phân vùng được gọi là cây tam phân (TT-ternary-tree) và cây mà sử dụng phân chia thành bốn phân vùng được gọi là cây tứ phân (QT-quad-tree).

Như đã đề cập trước đó, thuật ngữ "khối" như được sử dụng ở đây có thể là một phần, cụ thể là phần hình vuông hoặc hình chữ nhật, của ảnh. Ví dụ, tham chiếu tới HEVC và VVC, khối hiện tại có thể là hoặc tương ứng với đơn vị cây mã hóa (CTU-coding tree unit), đơn vị mã hóa (CU-coding unit), đơn vị dự đoán (PU-prediction unit) và đơn vị biến đổi (TU-transform unit) và/hoặc các khối tương ứng, ví dụ khối cây mã hóa (CTB-coding tree block), khối mã hóa (CB-coding block), khối biến đổi (TB-transform block) hoặc khối dự đoán (PB-prediction block).

Ví dụ, đơn vị cây mã hóa (CTU) có thể là hoặc bao gồm CTB của các mẫu độ chói, hai CTB tương ứng của các mẫu sắc độ của ảnh mà có ba mảng mẫu hoặc CTB của các mẫu của ảnh đơn sắc hoặc ảnh mà được mã hóa sử dụng ba mặt phẳng màu riêng biệt và các cấu trúc cú pháp được sử dụng để mã hóa các mẫu. Một cách tương ứng, khối cây mã hóa (CTB) có thể là khối NxN của các mẫu đối với một vài giá trị của N sao cho việc phân chia của thành phần thành các CTB là sự phân chia. Đơn vị mã hóa (CU-coding unit) có thể là hoặc bao gồm khối mã hóa của các mẫu độ chói, hai khối mã hóa tương ứng của các mẫu sắc độ của ảnh mà có ba mảng mẫu, hoặc khối mã hóa của các mẫu của ảnh đơn sắc hoặc ảnh mà được mã hóa nhờ sử dụng ba mặt phẳng màu riêng biệt và các cấu trúc cú pháp được sử dụng để mã hóa các mẫu. Một cách tương ứng, khối mã hóa (CB) có thể là khối MxN của các mẫu đối với một vài giá trị của M và N sao cho việc phân chia CTB thành các khối mã hóa là sự phân chia.

Trong các phương án, ví dụ, theo HEVC, đơn vị cây mã hóa (CTU-coding tree unit) có thể được phân chia thành các CU bằng cách sử dụng cấu trúc cây tứ phân được ký hiệu là cây mã hóa. Quyết định về mã vùng ảnh nhờ sử dụng dự đoán ảnh liên đới (theo thời gian) hay dự đoán nội ảnh (theo không gian) có thể được thực hiện tại mức CU. Mỗi CU có thể còn được phân chia thành một, hai, hoặc bốn PU theo loại phân chia PU. Bên trong một PU, xử lý dự đoán giống nhau được áp dụng và thông tin liên quan có thể được truyền tới bộ giải mã trên cơ sở PU. Sau khi thu nhận khối dư bằng cách áp dụng xử lý dự đoán dựa trên loại phân chia PU, CU có thể được phân chia thành các đơn vị biến đổi (TU) theo cấu trúc cây tứ phân khác tương tự cây mã hóa đối với CU.

Trong các phương án, ví dụ, theo tiêu chuẩn mã hóa viđeo mới nhất hiện đang phát triển, mà được gọi là mã hóa viđeo đa năng (VVC-Versatile Video Coding), việc phân chia cây nhị phân và cây tứ phân (QTBT) kết hợp là ví dụ được sử dụng để phân chia khối mã hóa. Trong cấu trúc khối QTBT, CU có thể có dạng hình vuông hoặc hình chữ nhật. Ví dụ, đơn vị cây mã hóa (CTU-coding tree unit) được phân chia đầu tiên bởi cấu trúc cây tứ phân. Các nút nhánh cây tứ phân được phân chia tiếp bởi cấu trúc cây nhị phân hoặc cây tam phân (hoặc bậc ba). Các nút nhánh cây phân chia được gọi là các đơn vị mã hóa (CU-coding unit), và phân đoạn này được sử dụng để xử lý dự đoán và biến đổi mà không cần bất kỳ phân chia tiếp. Điều này có nghĩa rằng CU, PU và TU có cùng kích cỡ khối trong cấu trúc khối mã hóa QTBT. Một cách song song, nhiều phân vùng, ví dụ, phân vùng cây tam phân có thể được sử dụng cùng với cấu trúc khối QTBT.

Trong một ví dụ, bộ lựa chọn chế độ 260 của bộ mã hóa viđeo 20 có thể có cấu trúc để thực hiện bất kỳ kết hợp của các kỹ thuật phân chia được mô tả ở đây.

Như được mô tả nêu trên, bộ mã hóa viđeo 20 có cấu trúc để xác định hoặc lựa chọn chế độ dự đoán tốt nhất hoặc tối ưu từ tập hợp chế độ dự đoán (ví dụ,

được xác định trước). Tập hợp các chế độ dự đoán có thể bao gồm, ví dụ, các chế độ dự đoán nội bộ và/hoặc các chế độ dự đoán liên đới.

Dự đoán nội bộ

Tập hợp chế độ dự đoán nội bộ có thể bao gồm 35 chế độ dự đoán nội bộ khác nhau, ví dụ, các chế độ vô hướng như chế độ DC (hoặc trung bình) và chế độ phẳng, hoặc các chế độ có hướng, ví dụ như được định nghĩa trong HEVC, hoặc có thể bao gồm 67 chế độ dự đoán nội bộ khác nhau, ví dụ, các chế độ vô hướng như chế độ DC (hoặc trung bình) và chế độ phẳng, hoặc các chế độ có hướng như được định nghĩa trong VVC.

Bộ dự đoán nội bộ 254 có cấu trúc để sử dụng các mẫu được khôi phục của các khối lân cận của cùng ảnh hiện tại để tạo ra khối dự đoán nội bộ 265 theo chế độ dự đoán nội bộ của tập hợp các chế độ dự đoán nội bộ.

Bộ dự đoán nội bộ 254 (hoặc nói chung là bộ lựa chọn chế độ 260) còn có cấu trúc để xuất ra các tham số dự đoán nội bộ (hoặc nói chung là thông tin chỉ báo chế độ dự đoán nội bộ được lựa chọn cho khối hiện tại) tới bộ mã hóa entropy 270 dưới dạng các phần tử cú pháp 266 để đưa vào dữ liệu ảnh được mã hóa 21, sao cho, ví dụ, bộ giải mã viđeo 30 có thể thu và sử dụng các tham số dự đoán để giải mã.

Dự đoán liên đới

Tập hợp của các chế độ dự đoán liên đới phụ thuộc vào các ảnh tham chiếu khả dụng (tức là các ảnh trước đó ít nhất đã được giải mã một phần, ví dụ được lưu trữ trong DBP 230) và các tham số dự đoán liên đới khác, ví dụ như toàn bộ ảnh tham chiếu hay chỉ một phần, ví dụ, vùng cửa sổ tìm kiếm xung quanh vùng của khối hiện tại, của ảnh tham chiếu được sử dụng để tìm kiếm khối tham chiếu phù hợp nhất và/hoặc ví dụ, nội suy điểm ảnh có được áp dụng hay không, ví dụ, nửa/nửa điểm ảnh và/hoặc nội suy một phần tư điểm hay, hay không.

Ngoài các chế độ dự đoán nêu trên, chế độ bỏ qua và/hoặc chế độ trực tiếp có thể được áp dụng.

Bộ dự đoán liên đới 244 có thể bao gồm bộ ước lượng chuyển động (ME-motion estimation) và bộ bù chuyển động (MC-motion compensation) (cả hai không được thể hiện trên FIG.2). Bộ ước lượng chuyển động có thể có cấu trúc để thu hoặc thu nhận khối ảnh 203 (khối ảnh hiện tại 203 của ảnh hiện tại 17) và ảnh được giải mã 231, hoặc ít nhất một hoặc nhiều khối được khôi phục trước đó, tức là các khối được khôi phục của một hoặc nhiều các ảnh được giải mã trước đó khác/khác nhau 231, để ước lượng chuyển động. Ví dụ, chuỗi viđeo có thể bao gồm ảnh hiện tại và các ảnh được giải mã trước đó 231, hay nói cách khác, ảnh hiện tại và các ảnh được giải mã trước đó 231 có thể là một phần của hoặc tạo thành chuỗi ảnh mà tạo thành chuỗi viđeo.

Ví dụ, bộ mã hóa 20 có thể có cấu trúc để: lựa chọn khối tham chiếu từ các khối tham chiếu của cùng ảnh hoặc các ảnh khác nhau trong các ảnh khác, và cung cấp ảnh tham chiếu (hoặc chỉ số ảnh tham chiếu) và/hoặc độ dịch (độ dịch không gian) giữa vị trí (các tọa độ x-y) của khối tham chiếu và vị trí của khối hiện tại như là các tham số dự đoán liên đới tới bộ ước lượng chuyển động. Độ dịch này cũng được gọi là vectơ chuyển động (MV-motion vector).

Bộ bù chuyển động có cấu trúc để: thu nhận, ví dụ, thu tham số dự đoán liên đới, và thực hiện việc dự đoán liên đới dựa trên hoặc bằng cách sử dụng tham số dự đoán liên đới để thu nhận khố dự đoán liên đới 265. Việc bù chuyển động được thực hiện bởi bộ bù chuyển động có thể bao gồm truy xuất hoặc tạo ra khối dự đoán dựa trên vectơ chuyển động/khối được xác định bằng cách ước lượng chuyển động, có thể thực hiện các nội suy trên độ chính xác điểm ảnh con. Việc lọc nội suy có thể tạo ra các mẫu điểm ảnh bổ sung từ các mẫu điểm ảnh đã biết, nhờ đó tăng khả năng số lượng khối dự đoán ứng viên mà có thể được sử dụng để mã hóa khối ảnh. Sau khi thu vectơ chuyển động đối với PU của khối ảnh hiện tại, bộ bù chuyển động có thể xác định vị trí khối dự đoán mà các vectơ chuyển động trỏ tới trong một trong số các danh sách ảnh tham chiếu.

Bộ bù chuyển động cũng có thể tạo ra các phần tử cú pháp được kết hợp với các khối hiện tại và các lát viđeo để bộ giải mã viđeo 30 sử dụng trong việc giải mã

các khối ảnh của lát viđeo. Ngoài các lát và các phần tử cú pháp tương ứng, các nhóm ô và /hoặc các ô và các phần tử cú pháp tương ứng có thể được tạo ra hoặc được sử dụng.

Mã hóa Entropy

Bộ mã hóa entropy 270 có cấu trúc để áp dụng, ví dụ, phương pháp hoặc thuật toán mã hóa entropy (tức là phương pháp mã hóa độ dài biến thiên (VLC-variable length coding), phương pháp VLC thích nghi ngữ cảnh (CAVLC), phương pháp mã hóa số học, nhị phân hóa, mã hóa số học nhị phân thích nghi ngữ cảnh (CABAC), mã hóa số học nhị phân thích nghi ngữ cảnh dựa trên cú pháp (SBAC), mã hóa entropy phân chia khoảng cách xác suất (PIPE-probability interval partitioning entropy) hoặc phương pháp hoặc kỹ thuật mã hóa entropy khác) hoặc kỹ thuật đường vòng (không nén) trên các hệ số được lượng tử hóa 209, các tham số dự đoán liên đới, các tham số nội dự đoán, tham số lọc vòng và/hoặc các phần tử cú pháp khác để thu nhận dữ liệu ảnh được mã hóa 21 mà có thể được xuất ra thông qua bộ đầu ra 272, ví dụ dưới dạng của dòng bit được mã hóa 21, sao cho, ví dụ bộ giải mã viđeo 30 có thể thu và sử dụng các tham số để giải mã. Dòng bit được mã hóa 21 có thể được truyền tới bộ giải mã viđeo 30, hoặc có thể được lưu trữ trong bộ nhớ cho việc truyền tiếp theo hoặc được gọi ra bởi bộ giải mã viđeo 30.

Các biến thể cấu trúc khác của bộ mã hóa viđeo 20 có thể được sử dụng để mã hóa dòng viđeo. Ví dụ, bộ mã hóa dựa trên không biến đổi 20 có thể lượng tử hóa trực tiếp tín hiệu dư mà không có bộ xử lý biến đổi 206 đối với một vài khối hoặc khung. Trong cách thức thực hiện khác, bộ mã hóa 20 có thể có bộ lượng tử hóa 208 và bộ lượng tử hóa ngược 210 mà được kết hợp thành một bộ phận.

Bộ giải mã và phương pháp giải mã

Fig.3 thể hiện ví dụ về bộ giải mã viđeo 30 mà có cấu trúc để thực hiện các kỹ thuật của sáng chế. Bộ giải mã viđeo 30 có cấu trúc để thu dữ liệu ảnh được mã hóa 21 (tức là dòng bit được mã hóa 21), tức là được mã hóa bởi bộ mã hóa 20, để thu nhận ảnh được giải mã 331. Dữ liệu ảnh được mã hóa hoặc dòng bit bao

gồm thông tin để giải mã dữ liệu ảnh được mã hóa, ví dụ dữ liệu mà biểu diễn các khối ảnh của lát viđeo được mã hóa (và/hoặc các nhóm ô hoặc các ô) và các phần tử cú pháp được kết hợp.

Trong ví dụ của Fig.3, bộ giải mã 30 bao gồm bộ giải mã entropy 304, bộ lượng tử hóa ngược 310, bộ xử lý biến đổi ngược 312, bộ khôi phục 314 (ví dụ bộ cộng tổng 314), bộ lọc vòng 320, bộ đệm ảnh được giải mã (DBP) 330, bộ áp dụng chế độ 360, bộ dự đoán liên đới 344 và bộ dự đoán nội bộ 354. Bộ dự đoán liên đới 344 có thể là hoặc bao gồm bộ bù chuyển động. Trong một số ví dụ, bộ giải mã viđeo 30 có thể thực hiện đường giải mã thường tương hỗ với đường mã hóa được mô tả liên quan đến bộ mã hóa viđeo 100 từ Fig.2.

Như được giải thích đối với bộ mã hóa 20, bộ lượng tử hóa ngược 210, bộ xử lý biến đổi ngược 212, bộ khôi phục 214, bộ lọc vòng 220, bộ đệm ảnh được giải mã (DPB) 230, bộ dự đoán liên đới 344 và bộ dự đoán nội bộ 354 cũng tạo thành "bộ giải mã lắp trong" của bộ mã hóa viđeo 20. Do đó, bộ lượng tử hóa ngược 310 có thể đồng nhất về chức năng với bộ lượng tử hóa ngược 110, bộ xử lý biến đổi ngược 312 có thể đồng nhất về chức năng với bộ xử lý biến đổi ngược 212, bộ khôi phục 314 có thể đồng nhất về chức năng với bộ khôi phục 214, bộ lọc vòng 320 có thể đồng nhất về chức năng với bộ lọc vòng 220, và bộ đệm ảnh được giải mã 330 có thể đồng nhất về chức năng với bộ đệm ảnh được giải mã 230. Do đó, các giải thích được mô tả cho các bộ phận và chức năng tương ứng của bộ mã hóa viđeo 20 áp dụng tương ứng cho các bộ phận và chức năng tương ứng của bộ giải mã viđeo 30.

Giải mã Entropy

Bộ giải mã entropy 304 có cấu trúc để phân tích dòng bit 21 (hoặc nói chung là dữ liệu ảnh được mã hóa 21) và thực hiện, ví dụ, việc giải mã entropy đối với dữ liệu ảnh được mã hóa 21 để thu nhận, ví dụ, các hệ số được lượng tử hóa 309 và/hoặc các tham số mã hóa được giải mã (không được thể hiện trên FIG.3), tức là bất kỳ hoặc tất cả tham số dự đoán liên đới (tức là chỉ số ảnh tham chiếu và vectơ chuyển động), tham chiếu dự đoán nội bộ (tức là chế độ hoặc chỉ số dự

đoán nội bộ), các tham số biến đổi, các tham số lượng tử hóa, các tham số lọc vòng, và/hoặc các phần tử cú pháp khác. Bộ giải mã entropy 304 có thể có cấu trúc để áp dụng các thuật toán hoặc sơ đồ giải mã tương ứng với các sơ đồ mã hóa như được mô tả liên quan đến bộ mã hóa entropy 270 của bộ mã hóa 20. Bộ giải mã entropy 304 có thể còn có cấu trúc để cung cấp các tham số dự đoán liên đới, tham số dự đoán nội bộ và/hoặc các phần tử cú pháp khác tới bộ áp dụng chế độ 360 và các tham số khác tới các bộ phận khác của bộ giải mã 30. Bộ giải mã viđeo 30 có thể thu các phần tử cú pháp tại cấp lát viđeo và/hoặc cấp khối viđeo. Ngoài các lát và các phần tử cú pháp tương ứng, các nhóm ô và/hoặc các ô và các phần tử cú pháp tương ứng có thể được thu và/hoặc được sử dụng.

Lượng tử hóa ngược

Bộ lượng tử hóa ngược 310 có thể có cấu trúc để thu các tham số lượng tử hóa (QP) (hoặc nói chung là thông tin liên quan đến lượng tử hóa ngược) và các hệ số được lượng tử từ dữ liệu ảnh được mã hóa 21 (ví dụ bằng cách phân tích và/hoặc giải lượng tử, ví dụ bởi bộ giải mã entropy 304) và để áp dụng, dựa trên các tham số lượng tử hóa, việc lượng tử hóa ngược trên các hệ số được lượng tử hóa được giải mã 309 để thu nhận các hệ số được lượng tử hóa 311, mà có thể cũng được gọi là các hệ số biến đổi 311. Xử lý lượng tử hóa ngược có thể bao gồm việc sử dụng tham số lượng tử hóa được xác định bởi bộ mã hóa viđeo 20 đối với mỗi khối viđeo trong lát viđeo (hoặc ô hoặc nhóm ô) để xác định mức độ lượng tử hóa và loại tương tự, mức độ lượng tử hóa ngược mà sẽ được áp dụng.

Biến đổi ngược

Bộ xử lý biến đổi ngược 312 có thể có cấu trúc để thu các hệ số được giải lượng tử 311, cũng được gọi là các hệ số biến đổi 311, và để áp dụng biến đổi tới các hệ số được giải lượng tử 311 để thu nhận các khối dư được khôi phục 213 trong miền mẫu. Các khối dư được khôi phục 213 cũng có thể được gọi là các khối biến đổi 313. Việc biến đổi có thể là biến đổi ngược, ví dụ, DCT ngược, DST ngược, biến đổi nguyên ngược, hoặc xử lý biến đổi nghịch đảo tương tự về mặt

khái niệm. Bộ xử lý biến đổi ngược 312 có thể còn có cấu trúc để thu các tham số biến đổi hoặc thông tin tương ứng từ dữ liệu ảnh được mã hóa 21 (ví dụ bằng cách phân tích và/hoặc giải mã, ví dụ bởi bộ giải mã entropy 304) để xác định biến đổi cần được áp dụng tới các hệ số được giải lượng tử 311.

Khôi phục

Bộ khôi phục 314 (tức là bộ cộng hoặc bộ cộng tổng 314) có thể có cấu trúc để cộng khối dư được khôi phục 313, tới khối dự đoán 365 để thu nhận khối được khôi phục 315 trong miền mẫu, tức là bằng cách cộng các giá trị mẫu của khối dư được khôi phục 313 và các giá trị mẫu của khối dự đoán 365.

Lọc

Bộ lọc vòng 320 (hoặc trong vòng lặp mã hóa hoặc sau vòng lặp mã hóa) có cấu trúc để lọc khối được khôi phục 315 để thu nhận khối được lọc 321, ví dụ để làm mượt sự chuyển tiếp điểm ảnh, hoặc cải thiện chất lượng viđeo. Bộ lọc vòng 320 có thể bao gồm một hoặc nhiều bộ lọc vòng như bộ lọc giải khối, bộ lọc bù thích ứng mẫu (SAO) hoặc một hoặc nhiều bộ lọc khác, ví dụ, bộ lọc song phương, bộ lọc vòng thích ứng (ALF), các bộ lọc san bằng, làm nhạy hoặc bộ lọc cộng tác, hoặc bất kỳ sự kết hợp của chúng. Mặc dù bộ lọc vòng 320 được hiển thị trong Fig.3 là bộ lọc vòng nội bộ, trong các cấu trúc khác, bộ lọc vòng 320 có thể được thực hiện như bộ lọc vòng sau.

Bộ đệm ảnh được giải mã

Sau đó, các khối viđeo được giải mã 321 của ảnh được lưu trữ trong bộ đệm ảnh được giải mã 330, mà lưu trữ các ảnh được giải mã 331 như là các ảnh tham chiếu cho việc bù chuyển động tiếp theo đối với các ảnh khác và/hoặc cho việc xuất ra màn hình tương ứng.

Bộ giải mã 30 có cấu trúc để xuất ra ảnh được giải mã 311, tức là thông qua bộ đầu ra 312, để trình diễn hoặc quan sát tới người dùng.

Dự đoán

Bộ dự đoán liên đới 344 có thể đồng nhất với bộ dự đoán liên đới 244 (cụ thể, đồng nhất với bộ bù chuyển động) và bộ dự đoán nội bộ 354 có thể đồng nhất với bộ dự đoán liên đới 254 về chức năng, và thực hiện các quyết định phân chia hoặc phân vùng và dự đoán dựa trên các tham số phân chia và/hoặc dự đoán hoặc thông tin tương ứng thu được từ dữ liệu ảnh được mã hóa 21 (ví dụ bằng cách phân tích và/hoặc giải mã, ví dụ bởi bộ giải mã entropy 304). Bộ áp dụng chế độ 360 có thể có cấu trúc để thực hiện việc dự đoán (dự đoán nội bộ hoặc liên đới) theo khối dựa trên các ảnh được khôi phục, các khối hoặc các mẫu tương ứng (được lọc hoặc không được lọc) để thu nhận khối dự đoán 365.

Khi lát viđeo được mã hóa như là lát được mã hóa nội bộ (I), bộ dự đoán nội bộ 354 của bộ áp dụng chế độ 360 có cấu trúc để tạo khối dự đoán 365 đối với khối ảnh của lát viđeo hiện tại dựa trên chế độ dự đoán nội bộ được báo hiệu và dữ liệu từ các khối đã giải mã trước đó của ảnh hiện tại. Khi ảnh viđeo được mã hóa như là lát được mã hóa liên đới (tức là, B, hoặc P), bộ dự đoán liên đới 344 (tức là bộ bù chuyển động) của bộ áp dụng chế độ 360 có cấu trúc để tạo ra các khối dự đoán 365 đối với khối viđeo của lát viđeo hiện tại dựa trên các vectơ chuyển động và các phần tử cú pháp khác thu được từ bộ giải mã entropy 304. Đối với việc dự đoán liên đới, các khối dự đoán có thể được tạo ra từ một trong số các ảnh tham chiếu trong một trong số các danh sách ảnh tham chiếu. Bộ giải mã viđeo 30 có thể xây dựng các danh sách khung tham chiếu: danh sách 0 và danh sách 1 bằng cách sử dụng kỹ thuật xây dựng mặc định dựa trên các ảnh tham chiếu được lưu trữ trong DPB 330. Điều giống hoặc tương tự có thể được áp dụng cho hoặc bởi các phương án mà sử dụng các nhóm ô (ví dụ, các nhóm ô viđeo) và/hoặc các ô (ví dụ, các ô viđeo) ngoài các lát (ví dụ, các lát viđeo), ví dụ viđeo có thể được mã hóa nhờ sử dụng các nhóm ô và/hoặc các ô I, P hoặc B.

Bộ áp dụng chế độ 360 có cấu trúc để xác định thông tin dự đoán đối với khối viđeo của lát video hiện tại bằng cách phân tích vectơ chuyển động hoặc thông tin liên quan và các phần tử cú pháp khác, và sử dụng thông tin dự đoán để tạo

ra các khối dự đoán đối với khối viđeo hiện tại đang được giải mã. Ví dụ, bộ áp dụng chế độ 360 sử dụng một vài phần tử cú pháp thu được, để xác định chế độ dự đoán (ví dụ, dự đoán nội bộ hoặc liên đới) được sử dụng để mã hóa các khối viđeo của lát viđeo, loại lớp dự đoán liên đới (ví dụ, lát B, lát P, hoặc lát GPB), thông tin xây dựng đối với một hoặc nhiều danh sách ảnh tham chiếu đối với lát, các vectơ chuyển động đối với mỗi khối viđeo được mã hóa liên đới của lát, trạng thái dự đoán liên đới đối với mỗi khối viđeo được mã hóa liên đới của lát, và thông tin khác, để giải mã các khối viđeo trong lát viđeo hiện tại. Điều giống hoặc tương tự có thể được áp dụng cho hoặc bởi các phương án mà sử dụng các nhóm ô (ví dụ các nhóm ô viđeo) và/hoặc các ô (ví dụ các ô viđeo) ngoài các lát (ví dụ các lát viđeo), ví dụ viđeo có thể được mã hóa nhờ sử dụng các nhóm ô và/hoặc các ô I, P hoặc B.

Các phương án của bộ giải mã viđeo 30 như được thể hiện trên FIG.3 có thể có cấu trúc để phân chia và/hoặc giải mã ảnh bằng cách sử dụng các lát (cũng được gọi là các lát viđeo), trong đó ảnh có thể được phân chia hoặc được giải mã nhờ sử dụng một hoặc nhiều lát (thường không chồng lấn), và mỗi lát có thể bao gồm một hoặc nhiều khối (ví dụ các CTU).

Các phương án của bộ giải mã viđeo 30 như được thể hiện trên FIG.3 có thể có cấu trúc để phân chia và/hoặc giải mã ảnh bằng cách sử dụng các nhóm ô (cũng được gọi là các nhóm ô viđeo) và/hoặc các ô (cũng được gọi là các ô viđeo), trong đó ảnh có thể được phân chia hoặc được giải mã nhờ sử dụng một hoặc nhiều nhóm ô (thường không xếp chồng), và mỗi nhóm ô có thể bao gồm, ví dụ một hoặc nhiều khối (tức là các CTU) hoặc một hoặc nhiều ô, trong đó mỗi ô, , ví dụ, có thể có dạng chữ nhật và có thể bao gồm một hoặc nhiều khối (ví dụ, các CTU), ví dụ các khối hoàn chỉnh hoặc riêng phần.

Các biến thể khác của bộ giải mã viđeo 30 có thể được sử dụng để giải mã dữ liệu ảnh được mã hóa 21. Ví dụ, bộ giải mã 30 có thể tạo ra dòng viđeo đầu ra mà không cần bộ lọc vòng 320. Ví dụ, bộ giải mã dựa trên không biến đổi 30 có

thể lượng tử hóa ngược trực tiếp tín hiệu dư mà không có bộ xử lý biến đổi ngược 312 đối với một vài khối hoặc khung. Trong cách thức thực hiện khác, bộ giải mã video 30 có thể có bộ lượng tử hóa ngược 310 và bộ xử lý biến đổi ngược 312 được kết hợp thành một bộ phận.

Cần hiểu rằng, trong bộ mã hóa 20 và bộ giải mã 30, kết quả xử lý của bước hiện tại có thể được xử lý thêm và sau đó được xuất tới bước tiếp theo. Ví dụ, sau khi lọc nội suy, việc thu nhận vectơ chuyển động hoặc lọc vòng, hoạt động khác, chẳng hạn như cắt hoặc dịch chuyển, có thể được thực hiện trên kết quả xử lý của việc lọc nội suy, thu nhận vectơ chuyển động hoặc lọc vòng.

Cần lưu ý rằng các hoạt động khác có thể được áp dụng tới các vectơ chuyển động thu được của khối hiện tại (bao gồm nhưng không giới hạn ở vectơ chuyển động điểm điều khiển trong chế độ affin, vectơ chuyển động khối con trong chế độ affin, phẳng, ATMVP, các vectơ chuyển động thời gian, v.v). Ví dụ, giá trị của vectơ chuyển động được giới hạn trong một phạm vi được xác định trước theo bit đại diện của nó. Nếu bit đại diện của vectơ chuyển động là bitDepth , thì phạm vi là -2 ^ (bitDepth-1) ~ 2 ^ (bitDepth-1) -1, trong đó "^" có nghĩa là lũy thừa. Ví dụ, nếu bitDepth được thiết lập bằng 16, phạm vi là -32768 ~ 32767; nếu bitDepth được thiết lập bằng 18, phạm vi là -131072~131071. Ví dụ, giá trị của vectơ chuyển động thu được (ví dụ, các MV của bốn khối con 4x4 trong một khối 8x8) bị giới hạn sao cho độ chênh lệch lớn nhất giữa các phân nguyên của bốn MV khối con 4x4 không lớn hơn N điểm ảnh, như không quá 1 điểm ảnh. Ở đây cung cấp hai phương pháp để giới hạn vectơ chuyển động theo bitDepth .

Phương pháp 1: Loại bỏ MSB (bit quan trọng nhất-most significant bit) tràn dòng bằng các phép toán dòng

$$ux = (mvx + 2^{bitDepth}) \% 2^{bitDepth} \quad (1)$$

$$mvx = (ux >= 2^{bitDepth-1}) ? (ux - 2^{bitDepth}) : ux \quad (2)$$

$$uy = (mvy + 2^{bitDepth}) \% 2^{bitDepth} \quad (3)$$

$$mvy = (uy >= 2^{bitDepth-1}) ? (uy - 2^{bitDepth}) : uy \quad (4)$$

trong đó mvx là thành phần chiều ngang của vectơ chuyển động của khối ảnh hoặc khối con, mvy là thành phần chiều dọc của vectơ chuyển động của khối ảnh hoặc khối con và ux và uy chỉ báo giá trị trung gian ;

Ví dụ, nếu giá trị của mvx là -32769, sau khi áp dụng công thức (1) và (2), giá trị kết quả là 32767. Trong hệ thống máy tính, số thập phân được lưu trữ như là dạng phần bù của hai. Phần bù của hai -32769 là 1,0111,1111,1111,1111 (17 bit), sau đó MSB bị loại bỏ, do đó phần bù của hai thu được là 0111,1111,1111,1111 (số thập phân là 32767), mà tương tự như đầu ra bằng cách áp dụng công thức (1) và (2).

$$ux= (mvpx + mvdx +2^{bitDepth}) \% 2^{bitDepth} \quad (5)$$

$$mvx = (ux >= 2^{bitDepth-1}) ? (ux - 2^{bitDepth}) : ux \quad (6)$$

$$uy= (mvpy + mvdy +2^{bitDepth}) \% 2^{bitDepth} \quad (7)$$

$$mvy = (uy >= 2^{bitDepth-1}) ? (uy - 2^{bitDepth}) : uy \quad (8)$$

Các phép toán có thể được áp dụng trong tổng mvp và mvd , như được thể hiện trong công thức (5) đến (8).

Phương pháp 2: Loại bỏ MSB tràn dòng bằng cách cắt giá trị

$$vx = Clip3(-2^{bitDepth-1}, 2^{bitDepth-1} -1, vx)$$

$$vy = Clip3(-2^{bitDepth-1}, 2^{bitDepth-1} -1, vy)$$

trong đó vx là thành phần chiều ngang của vectơ chuyển động của khối ảnh hoặc khối con, vy là thành phần chiều dọc của vectơ chuyển động của khối ảnh hoặc khối con; x, y và z tương ứng với ba giá trị đầu vào của xử lý cắt MV và định nghĩa của hàm Clip3 là như sau:

$$Clip3(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{trường hợp khác} \end{cases}$$

FIG.4 là sơ đồ giản lược của thiết bị mã hóa viđeo 400 theo phương án của sáng chế. Thiết bị mã viđeo 400 là thích hợp để thực hiện các phương án được bộc lộ như được mô tả ở đây. Theo phương án của sáng chế, thiết bị mã hóa vieo 400 có thể là bộ giải mã như bộ giải mã viđeo 30 của FIG.1A hoặc bộ mã hóa như bộ mã hóa viđeo 20 của FIG.1A.

Thiết bị mã hóa viđeo 400 bao gồm các cổng vào 410 (hoặc các cổng đầu vào 410) và các bộ thu (Rx) 420 để thu dữ liệu; bộ xử lý, bộ lôgic, hoặc bộ xử lý trung tâm (CPU-central processing unit) 430 để xử lý dữ liệu; các bộ truyền (Tx) 440 và các cổng ra 450 (hoặc các cổng đầu ra 450) để truyền dữ liệu; và bộ nhớ 460 để lưu trữ dữ liệu. Thiết bị mã viđeo 400 có thể cũng bao gồm các thành phần quang-điện (OE) và các thành phần điện-quang (EO) được ghép nối tới các cổng vào 410, các bộ thu 420, các bộ truyền 440, và các cổng ra 450 để nhập hoặc xuất các tín hiệu điện hoặc quang.

Bộ xử lý 430 được thực hiện bởi phần cứng và phần mềm. Bộ xử lý 430 có thể được thực hiện dưới dạng một hoặc nhiều chip CPU, lõi (ví dụ, như bộ xử lý đa lõi), các FPGA, ASIC và DSP. Bộ xử lý 430 truyền thông với các cổng vào 410, bộ thu 420, bộ phát 440, cổng ra 450 và bộ nhớ 460. Bộ xử lý 430 bao gồm môđun mã 470. Môđun mã hóa 470 thực hiện các phương án được bộc lộ như được mô tả nêu trên. Ví dụ, môđun mã hóa 470 thực hiện, xử lý, chuẩn bị hoặc cung cấp các hoạt động mã hóa khác nhau. Do đó, việc bao gồm môđun mã hóa 470 cung cấp cải tiến đáng kể đối với chức năng của thiết bị mã viđeo 400 và tạo hiệu ứng biến đổi thiết bị mã viđeo 400 thành trạng thái khác. Ngoài ra, môđun mã hóa 470 được thực hiện như các lệnh được lưu trữ trong bộ nhớ 460 và được thực thi bởi bộ xử lý 430.

Bộ nhớ 460 có thể bao gồm một hoặc nhiều đĩa, ổ băng và ổ bán dẫn và có thể được sử dụng như một thiết bị lưu trữ dữ liệu tràn dòng, để lưu trữ các chương trình khi các chương trình này được lựa chọn để thực thi và để lưu trữ các lệnh và dữ liệu được đọc trong quá trình thực thi chương trình. Ví dụ, bộ nhớ 460 có thể là bộ nhớ bất biến và/hoặc không bất biến và có thể là bộ nhớ chỉ đọc

(ROM-read-only memory), bộ nhớ truy cập ngẫu nhiên (RAM-random access memory), bộ nhớ định địa chỉ nội dung bậc ba (TCAM-ternary content-addressable memory) và/hoặc bộ nhớ truy nhập ngẫu nhiên tĩnh (SRAM-static random-access memory).

Fig.5 là sơ đồ khối giản lược của thiết bị 500 mà có thể được sử dụng như là thiết bị nguồn 12 hoặc thiết bị đích 14 hoặc cả hai thiết bị từ Fig.1 theo phương án ví dụ.

Bộ xử lý 502 trong thiết bị 500 có thể là bộ xử lý trung tâm. Ngoài ra, bộ xử lý 502 có thể là bất kỳ loại thiết bị khác hoặc nhiều thiết bị, có khả năng thao tác hoặc xử lý thông tin hiện có hoặc được phát triển sau này. Mặc dù các các cách thức thực hiện được bộc lộ có thể được thực hiện với một bộ xử lý như được minh họa, ví dụ, bộ xử lý 502, các ưu điểm về tốc độ và hiệu quả có thể đạt được khi sử dụng nhiều hơn một bộ xử lý.

Bộ nhớ 504 trong thiết bị 500 có thể là thiết bị bộ nhớ chỉ đọc (ROM) hoặc thiết bị bộ nhớ truy cập ngẫu nhiên (RAM) trong quá trình thực hiện. Bất kỳ loại thiết bị lưu trữ thích hợp khác đều có thể được sử dụng làm bộ nhớ 504. Bộ nhớ 504 có thể bao gồm mã và dữ liệu 506 mà được được truy nhập bởi bộ xử lý 502 bằng cách sử dụng kênh truyền 512. Bộ nhớ 504 có thể bao gồm thêm hệ điều hành 508 và các chương trình ứng dụng 510 , các chương trình ứng dụng 510 bao gồm ít nhất một chương trình mà cho phép bộ xử lý 502 thực hiện các phương pháp được mô tả ở đây. Ví dụ, các chương trình ứng dụng 510 có thể bao gồm các ứng dụng từ 1 đến N, các chương trình này bao gồm thêm ứng dụng mã hóa viđeo mà thực hiện các phương pháp được mô tả ở đây.

Thiết bị 500 cũng có thể bao gồm một hoặc nhiều thiết bị đầu ra, chẳng hạn như màn hình 518. Trong một ví dụ, màn hình 518 có thể là màn hình cảm ứng chạm mà kết hợp màn hình với phần tử cảm ứng chạm mà có thể hoạt động để cảm nhận đầu vào chạm. Màn hình 518 có thể được ghép nối với bộ xử lý 502 thông qua kênh truyền 512.

Mặc dù được mô tả ở đây là một kênh truyền, kênh truyền 512 của thiết bị 500 có thể bao gồm nhiều kênh truyền. Ngoài ra, bộ lưu trữ thứ cấp 514 có thể được ghép nối trực tiếp với các thành phần khác của thiết bị 500 hoặc có thể được truy nhập qua mạng và có thể bao gồm một bộ tích hợp như thẻ nhớ hoặc nhiều bộ phận như nhiều thẻ nhớ. Do đó, thiết bị 500 có thể được thực hiện với nhiều cấu hình khác nhau.

Việc dự đoán nội bộ của các mẫu sắc độ có thể được thực hiện nhờ sử dụng các mẫu của khối độ chói được khôi phục.

Trong phát triển HEVC, dự đoán nội bộ sắc độ mô hình tuyến tính thành phần chéo (CCLM-Cross-component Linear Model) được đề xuất [J. Kim, S.-W. Park, J.-Y. Park, và B.-M. Jeon, Dự đoán sắc độ nội bộ nhờ sử dụng tương quan liên kênh, tài liệu JCTVC-B021, tháng 7 năm 2010]. CCLM sử dụng tương quan tuyến tính giữa mẫu sắc độ và mẫu độ chói tại vị trí tương ứng trong khối mã hóa. Khi khối sắc độ được mã hóa nhờ sử dụng CCLM, mô hình tuyến tính được thu nhận từ các mẫu độ chói và sắc độ lân cận được khôi phục bởi hồi quy tuyến tính. Các mẫu sắc độ trong khối hiện tại có thể sau đó được dự đoán bởi các mẫu độ chói được khôi phục trong khối hiện tại với mô hình tuyến tính thu được (như được thể hiện trên Fig.6):

$$C(x,y) = \alpha \times L(x,y) + \beta ,$$

trong đó C và L chỉ báo các giá trị độ chói và sắc độ, một cách lần lượt. Các tham số α và β được thu nhận bởi phương pháp bình phương tối thiểu như sau:

$$\alpha = \frac{R(L,C)}{R(L,L)}$$
$$\beta = M(C) - \alpha \times M(L) ,$$

trong đó $M(A)$ biểu diễn giá trị trung bình của A và $R(A,B)$ được xác định như sau:

$$R(A,B) = M\big((A - M(A)) \times (B - M(B))\big) .$$

Nếu ảnh được mã hóa hoặc giải mã có khuôn dạng mà chỉ rõ số lượng mẫu khác nhau đối với các thành phần độ chói và sắc độ (ví dụ khuôn dạng 4:2:0 YCbCr), các mẫu độ chói được lấy mẫu xuống trước khi mô hình hóa và dự đoán.

Phương pháp này được áp dụng để sử dụng trong VTM2.0. Cụ thể, việc thu nhận tham số được thực hiện như sau:

$$\alpha = \frac{N \cdot \sum(L(n) \cdot C(n)) - \sum L(n) \cdot \sum C(n)}{N \cdot \sum(L(n) \cdot L(n)) - \sum L(n) \cdot \sum L(n)},$$

$$\beta = \frac{\sum C(n) - \alpha \cdot \sum L(n)}{N},$$

trong đó $L(n)$ biểu diễn các mẫu độ chói được khôi phục lân cận trên cùng và bên trái được lấy mẫu xuống, $C(n)$ biểu diễn các mẫu sắc độ được khôi phục lân cận trên cùng và bên trái.

Trong [G. Laroche, J. Taquet, C. Gisquet, P. Onno (Canon), "CE3: Đơn giản hóa mô hình tuyến tính thành phần chéo (Thử nghiệm 5.1)", Tài liệu đưa vào Hội thảo lần thứ 12 JVET tại Macao, Trung Quốc, tháng 10 năm 2018] phương pháp khác nhau để thu nhận α và β được đề xuất (xem Fig.7). Cụ thể, các tham số mô hình tuyến tính α và β được thu nhận theo các công thức sau đây:

$$\alpha = \frac{C(B) - C(A)}{L(B) - L(A)}$$

$\beta = L(A) - \alpha C(A)$, trong đó

trong đó $B = \text{argmax}(L(n))$ và $A = \text{argmin}(L(n))$ là các vị trí của các giá trị lớn nhất và nhỏ nhất trong các mẫu độ chói.

Fig.8 thể hiện vị trí của các mẫu kết quả bên trái và phía trên và mẫu của khối hiện tại liên quan trong chế độ CCLM nếu khuôn dạng sắc độ YCbCr 4:4:4 đang được sử dụng.

Để thực hiện dự đoán thành phần chéo, đối với khuôn dạng sắc độ 4:2:0, khối độ chói được khôi phục cần được lấy mẫu xuống để khớp với kích cỡ của tín hiệu sắc độ hoặc các mẫu sắc độ hoặc khối sắc độ. Bộ lọc lấy mẫu xuống mặc định được sử dụng trong chế độ CCLM mode là như sau.

$$Rec'_L[x,y] = (2 \times Rec_L[2x,2y] + 2 \times Rec_L[2x,2y+1] + Rec_L[2x-1,2y] + Rec_L[2x+1,2y] + Rec_L[2x-1,2y+1] + Rec_L[2x+1,2y+1] + 4) >> 3$$

Lưu ý rằng việc lấy mẫu xuống này giả định quan hệ pha "loại 0" đối với các vị trí của các mẫu sắc độ so với các vị trí của các mẫu độ chói, tức là lấy mẫu cùng vị trí theo chiều ngang và lấy mẫu xen kẽ theo chiều dọc. Bộ lọc lấy mẫu xuống 6 nhánh nêu trên được thể hiện trên Fig.9 được sử dụng là bộ lọc mặc định cho cả chế độ CCLM đơn mô hình và chế độ CCLM đa mô hình. Các vị trí không gian của các mẫu được sử dụng bởi bộ lọc lấy mẫu xuống 6 nhánh được biểu diễn trên Fig.9. Các mẫu 901, 902, và 903 có các trọng số bằng 2, 1, và 0, một cách lần lượt.

Nếu các mẫu độ chói có vị trí trên biên khối và các khối trên cùng và bên trái liền kề không khả dụng, các công thức sau đây được sử dụng:

$Rec'_L[x,y] = Rec_L[2x,2y]$, nếu hàng với $y = 0$ là hàng thứ nhất nhất của CTU, $x = 0$ cũng như các khối liền kề bên trái và trên cùng là không khả dụng;

$Rec'_L[x,y] = (2 \times Rec_L[2x,2y] + Rec_L[2x-1,2y] + Rec_L[2x+1,2y] + 2) >> 2$, nếu hàng với $y = 0$ là hàng thứ nhất 1 của CTU và khối liền kề trên cùng là không khả dụng.

$Rec'_L[x,y] = (Rec_L[2x,2y] + Rec_L[2x,2y+1] + 1) >> 1$, nếu $x = 0$ cũng như các khối liền kề bên trái và trên cùng là không khả dụng.

Các Fig.10A và Fig.10B minh họa vị trí thành phần sắc độ trong trường hợp của phương thức lấy mẫu 4:2:0. Rõ ràng, điều tương tự có thể áp dụng tới các phương thức lấy mẫu khác.

Đã biết rằng, khi xem xét việc lấy mẫu của các thành phần độ chói và sắc độ trong phương thức lấy mẫu 4:2:0, có thể có độ dịch giữa các lưới thành phần độ chói và sắc độ. Trong khối của 2×2 điểm ảnh, các thành phần sắc độ được dịch chuyển thực tế bởi nửa điểm ảnh theo chiều dọc so với thành phần độ chói (được minh họa trên Fig.10A). Việc dịch chuyển này có thể có ảnh hưởng lên các bộ lọc nội suy khi việc lấy mẫu xuống từ 4:4:4, hoặc khi lấy mẫu lên. Trên Fig.10B,

các mẫu lấy mẫu khác nhau được biểu diễn, trong trường hợp của ảnh được đan xen. Điều này có nghĩa rằng tính chẵn lẻ, tức là việc các điểm ảnh có nằm trên các trường trên cùng hoặc dưới cùng của ảnh được đan xen hay không, được tính đến.

Như được đề xuất trong [P. Hanhart, Y. He, "CE3: Bộ lọc lấy mẫu xuống CCLM cải biến đối với nội dung "loại-2" (Thử nghiệm 2.4)", Tài liệu đưa vào JVET-M0142 tới Hội thảo thứ 13 JVET tại Marrakech, Ma-Rốc, tháng 1 năm 2019] và được đưa vào dự thảo tiêu chuẩn kỹ thuật VVC (phiên bản 4), để tránh việc căn sai giữa các mẫu sắc độ và các mẫu độ chói được lấy mẫu xuống trong CCLM đối với nội dung "loại-2", các bộ lọc lấy mẫu xuống sau đây được áp dụng tới độ chói để xác định mô hình tuyến tính và dự đoán:

3-nhánh: $$\mathrm{Rec_L}'(i,j) = [\mathrm{Rec_L}(2i-1,2j) + 2\cdot \mathrm{rec_L}(2i,2j) + \mathrm{Rec_L}(2i+1,2j) + 2] \gg 2$$

5-nhánh: $$\mathrm{Rec_L}'(i,j) = [\mathrm{Rec_L}(2i,2j-1) + \mathrm{Rec_L}(2i-1,2j) + 4\cdot \mathrm{Rec_L}(2i,2j) + \mathrm{Rec_L}(2i+1,2j) + \mathrm{Rec_L}(2i,2j+1) + 4] \gg 3$$

Để tránh việc tăng số lượng bộ đệm dòng, các cải biến này không được áp dụng tại biên CTU trên cùng. Việc lựa chọn bộ lọc lấy mẫu xuống được quản lý bởi cờ SPS sps_cclm_colocated_chroma_flag. Khi giá trị của sps_cclm_colocated_chroma_flag là 0 hoặc sai, bộ lọc lấy mẫu xuống được áp dụng tới độ chói để xác định mô hình tuyến tính và dự đoán; khi giá trị của sps_cclm_colocated_chroma_flag là 1 hoặc đúng, bộ lọc lấy mẫu xuống không được áp dụng tới độ chói để xác định mô hình tuyến tính và dự đoán.

Các mẫu được khôi phục độ chói biên $^{L()}$ mà được sử dụng để thu nhận các tham số mô hình tuyến tính như được mô tả nêu trên được lấy mẫu con từ các mẫu độ chói được lọc $^{Rec'_L[x,y]}$.

Bảng 1. Các khuôn dạng sắc độ được mô tả trong tiêu chuẩn kỹ thuật VVC

chroma_format_idc	**separate_colour_plane_flag**	**Khuôn dạng sắc độ**	**SubWidthC**	**SubHeightC**
0	0	Đơn sắc	1	1
1	0	4:2:0	2	2
2	0	4:2:2	2	1
3	0	4:4:4	1	1
3	1	4:4:4	1	1

Xử lý lọc và lấy mẫu con các mẫu độ chói được mô tả trong mục 8.3.4.2.8 của tiêu chuẩn kỹ thuật VVC

8.3.4.2.8. Mô tả kỹ thuật của chế độ dự đoán nội bộ INTRA_LT_CCLM, INTRA_L_CCLM và INTRA_T_CCLM

Các đầu vào đối với xử lý này là:

- chế độ dự đoán nội bộ predModeIntra,
- vị trí mẫu (xTbC, yTbC) của mẫu trên cùng-bên trái của khối biến đổi hiện tại so với mẫu trên cùng-bên trái của ảnh hiện tại,
- biến nTbW chỉ rõ độ rộng khối biến đổi,
- biến nTbH chỉ rõ độ cao khối biến đổi,
- các mẫu lân cận sắc độ p[x][y], với x = −1, y = 0..2 * nTbH − 1 và x = 0.. 2 * nTbW − 1, y = − 1.

Đầu ra của xử lý này là các mẫu được dự đoán predSamples[x][y], với x = 0..nTbW − 1, y = 0..nTbH − 1.

Vị trí sắc độ hiện tại (xTbY, yTbY) được thu nhận như sau:

$$(xTbY, yTbY) = (xTbC << 1, yTbC << 1) \quad (8\text{-}155)$$

Các biến availL, availT và availTL được thu nhận như sau:

- Tính khả dụng của xử lý thu nhận các mẫu lân cận bên trái đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC) như là các đầu vào, và đầu ra được gán là availL.
- Tính khả dụng của xử lý thu nhận các mẫu lân cận trên cùng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC, yTbC − 1) như là các đầu vào, và đầu ra được gán là availT.
- Tính khả dụng của xử lý thu nhận các mẫu lân cận trên cùng-bên trái đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC − 1) như là các đầu vào, và đầu ra được gán là availTL.
- Số lượng của các mẫu sắc độ lân cận trên cùng-bên trái khả dụng numTopRight được thu nhận như sau:
  - Biến numTopRight được thiết lập bằng 0 và availTR được thiết lập bằng đúng (TRUE).
  - Khi predModeIntra bằng INTRA_T_CCLM, phần sau đây được áp dụng đối với x = nTbW..2 * nTbW − 1 cho đến khi availTR bằng sai (FALSE) hoặc x bằng 2 * nTbW − 1:
    - Xử lý thu nhận khả dụng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) và vị trí sắc độ lân cận (xTbC + x, yTbC − 1) như là các đầu vào, và đầu ra được gán là availableTR
    - Khi availableTR bằng đúng (TRUE), numTopRight được gia tăng thêm 1.
- Số lượng các mẫu sắc độ lân cận bên trái-phía dưới khả dụng numLeftBelow được thu nhận như sau:

- Biến numLeftBelow được thiết lập bằng 0 và availLB được thiết lập bằng đúng (TRUE).
- Khi predModeIntra bằng INTRA_L_CCLM, phần sau đây được áp dụng đối với y = nTbH..2 * nTbH − 1 cho đến khi availLB bằng sai (FALSE) hoặc y bằng 2 * nTbH − 1:
  - Xử lý thu nhận khả dụng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC + y) như là các đầu vào, và đầu ra được gán là availableLB
  - Khi availableLB bằng đúng (TRUE), numLeftBelow được gia tăng thêm 1.

Số lượng các mẫu sắc độ lân cận khả dụng trên phía trên cùng và trên cùng-bên trái numTopSamp và số lượng mẫu sắc độ lân cận khả dụng trên phía bên trái và bên trái-phía dưới nLeftSamp được thu nhận như sau:

- Nếu predModeIntra bằng INTRA_LT_CCLM, phần sau đây được áp dụng:

  numSampT = availT ? nTbW : 0(8-156)

  numSampL = availL ? nTbH : 0(8-157)

- Nếu không phải, phần sau đây được áp dụng:

  numSampT = (availT && predModeIntra = = INTRA_T_CCLM) ? (nTbW + numTopRight) : 0 (8-158)

  numSampL = (availL && predModeIntra = = INTRA_L_CCLM) ? (nTbH + numLeftBelow) : 0 (8-159)

Biến bCTUboundary được thu nhận như sau:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) = = 0) ? Đúng (TRUE): Sai (FALSE). (8-160)

Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu cả numSampL và numSampT bằng 0, phần sau đây được áp dụng:

predSamples[x][y] = 1 << ($BitDepth_C$ − 1) (8-161)

– Nếu không phải, các bước có thứ tự sau đây được áp dụng:

1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với x = 0..nTbW * 2 − 1, y= 0..nTbH * 2 − 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).

2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:

   – Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = −1..−3, y = 0..2 * numSampL − 1, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

   – Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0..2 * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

   – Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

   – Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

   – pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (pY[2 * x][2 * y − 1] +
pY[2 * x − 1][2 * y] + 4 * pY[2 * x][2 * y] + pY[2 * x + 1][2 * y] + (8-162)
pY[2 * x][2 * y + 1] + 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (pY[0][2 * y − 1] +
pY[−1][2 * y] + 4 * pY[0][2 * y] + pY[1][2 * y] + (8-163)
pY[0][2 * y + 1] + 4) >> 3

– Nếu không phải, pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (pY[0][2 * y − 1] + 2 * pY[0][2 * y] + pY[0][2 * y + 1] + 2) >> 2 (8-164)

– Nếu availT bằng đúng (TRUE), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] = (pY[2 * x][−1] +
pY[2 * x − 1][0] + 4 * pY[2 * x][0] + pY[2 * x + 1][0] + (8-165)
pY[2 * x][1] + 4) >> 3

– Nếu không phải, pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] = (pY[2 * x − 1][0] + 2 * pY[2 * x][0] + pY[2 * x + 1][0] + 2) >> 2

(8-166)

– Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = (pY[0][−1] +
pY[−1][0] + 4 * pY[0][0] + pY[1][0] + (8-167)
pY[0][1] + 4) >> 3

– Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = (pY[−1][0] + 2 * pY[0][0] + pY[1][0] + 2) >> 2 (8-168)

– Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = (pY[0][−1] + 2 * pY[0][0] + pY[0][1] + 2) >> 2 (8-169)

– Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = pY[0][0] (8-170)

– Nếu không phải, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (pY[2 * x − 1][2 * y] +
pY[2 * x − 1][2 * y + 1] +
2* pY[2 * x][2 * y] + 2*pY[2 * x][2 * y + 1] +
(8-171)
pY[2 * x + 1][2 * y] +
pY[2 * x + 1][2 * y + 1] + 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (pY[−1][2 * y] + pY[−1][2 * y + 1] +
2* pY[0][2 * y] + 2*pY[0][2*y + 1] + (8-172)
pY[1][2 * y] + pY[1][2 * y + 1] + 4) >> 3

-- Nếu không phải, pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (pY[0][2 * y] + pY[0][2 * y + 1] + 1) >> 1 (8-173)

Tình trạng kỹ thuật đã biết không chỉ rõ xử lý lọc độ chói đối với trường hợp khi thành phần sắc độ không được lấy mẫu con. Thành phần độ chói được xử lý như thế nào đối với trường hợp khi khuôn dạng sắc độ khác với 4:2:0 sẽ được mô tả sau đây.

Phần mô tả này bộc lộ phương pháp xử lý các mẫu độ chói mà được sử dụng như là dữ liệu đầu vào để xác định các tham số của mô hình tuyến tính. Phương pháp này bao gồm xác định hai bộ lọc mà được áp dụng có điều kiện theo chiều ngang và dọc.

Các phương án của sáng chế mô tả việc đưa ra tập hợp của các điều kiện, bao gồm nhưng không bị giới hạn ở các hệ số lấy mẫu sắc độ, mà được kiểm tra để xác định các hệ số của bộ lọc mà được áp dụng tới các mẫu độ chói được khôi phục.

Như được thể hiện trên Fig.11, phương pháp này được mô tả như sau.

Khối hiện tại 1101 là để xác định hoặc thu hoặc nhận các giá trị của SubWidthC (tức là độ rộng của khối ảnh) và SubHeightC (tức là, độ cao của khối ảnh) dựa trên khuôn dạng sắc độ của ảnh đang được mã hóa.

Khối hiện tại 1102 là để định nghĩa hoặc xác định bộ lọc "F" được sử dụng cho các giá trị SubWidthC và SubHeightC.

Các phương án ví dụ, về làm thế nào các bộ lọc có thể được kết hợp với các giá trị tương ứng của SubWidthC và SubHeightC như được thể hiện trong các bảng 2-5. Bộ lọc không gian "F" được xác định dưới dạng của ma trận hệ số. Các vị

trí tương ứng mà các hệ số này được áp dụng tới, được xác định so với vị trí (x,y) của mẫu độ chói được lọc như sau:

$$\begin{bmatrix} (x-1,y-1) & (x,y-1) & (x+1,y-1) \\ (x-1,y) & (x,y) & (x+1,y) \\ (x-1,y+1) & (x,y+1) & (x+1,y+1) \end{bmatrix}.$$

Khi vị trí của các mẫu được khôi phục được lọc đầu ra nằm trên biên khối, một vài vị trí lân cận có thể trở nên không khả dụng, do các khối lân cận không khả dụng. Trong trường hợp này, các vị trí của các mẫu đầu vào được cải biến để lựa chọn các vị trí giống nhau. Việc cải biến lấy mẫu có thể được thực hiện như là bộ lọc tương đương có kích thước nhỏ hơn mà có các hệ số khác nhau.

Cụ thể, khi vị trí của mẫu đầu ra nằm trên biên bên trái của khối hiện tại và các mẫu liền kề bên trái của khối độ chói không khả dụng, các vị trí để lọc được xác định như sau:

$$\begin{bmatrix} (x,y-1) & (x,y-1) & (x+1,y-1) \\ (x,y) & (x,y) & (x+1,y) \\ (x,y+1) & (x,y+1) & (x+1,y+1) \end{bmatrix}.$$

Khi vị trí của mẫu đầu ra nằm trên biên trên cùng của khối hiện tại và các mẫu liền kề phía trên cùng của khối độ chói không khả dụng, các vị trí để lọc được xác định như sau:

$$\begin{bmatrix} (x-1,y) & (x,y) & (x+1,y) \\ (x-1,y) & (x,y) & (x+1,y) \\ (x-1,y+1) & (x,y+1) & (x+1,y+1) \end{bmatrix}.$$

Khi vị trí của mẫu đầu ra nằm trên biên bên phải của khối hiện tại, các vị trí để lọc được xác định như sau:

$$\begin{bmatrix} (x-1,y-1) & (x,y-1) & (x,y-1) \\ (x-1,y) & (x,y) & (x,y) \\ (x-1,y+1) & (x,y+1) & (x,y+1) \end{bmatrix}.$$

Khi vị trí của mẫu đầu ra nằm trên biên dưới cùng của khối hiện tại, các vị trí để lọc được xác định như sau:

$$\begin{bmatrix} (x-1,y-1) & (x,y-1) & (x+1,y-1) \\ (x-1,y) & (x,y) & (x+1,y) \\ (x-1,y) & (x,y) & (x+1,y) \end{bmatrix}$$

Bảng 2. Kết hợp của bộ lọc không gian với các giá trị của SubWidthC và SubHeightC

SubWidthC	**SubHeightC**	**Bộ lọc không gian F**
1	1	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
1	2	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
2	1	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$
2	2	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

Bảng 3. Kết hợp của bộ lọc không gian với các giá trị của SubWidthC và SubHeightC

SubWidthC	**SubHeightC**	**Bộ lọc không gian**

1	1	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
1	2	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
2	1	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$
2	2	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

Bảng 4. Kết hợp của bộ lọc không gian với các giá trị của SubWidthC và SubHeightC

SubWidthC	**SubHeightC**	**Bộ lọc không gian F**

1	1	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
1	2	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
2	1	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$
2	2	$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$

Bảng 5. Kết hợp của bộ lọc không gian với các giá trị của SubWidthC và SubHeightC

SubWidthC	**SubHeightC**	**Bộ lọc không gian F**

1	1	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
1	2	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$
2	1	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$
2	2	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$

Khối hiện tại 1103 là để thực hiện việc lọc của mẫu độ chói được khôi phục để thu nhận các giá trị mẫu độ chói được lọc $Rec'_L[x,y]$. Cụ thể, điều này được thực hiện bằng cách áp dụng bộ lọc được lựa chọn "F" tới các mẫu được khôi phục $Rec_L[x,y]$:

$$Rec'_L[x,y] = (\sum_{i=-1}^{1}\sum_{j=-1}^{1} Rec'_L[x+i,y+j]\cdot F[i+1,j+1] + \frac{N}{2}) >> \log_2(N)$$

,

trong đó F biểu diễn bộ lọc, N là tổng hệ số của bộ lọc F,. (*x*,*y*) biểu diễn vị trí của mẫu được khôi phục.

Phương án bổ sung để chuyển đổi giữa các loại bộ lọc (tức là các kết hợp bộ lọc được xác định trong các bảng 2-5) phụ thuộc vào vị trí của các mẫu sắc độ được lấy mẫu con so với các mẫu độ chói. Theo ví dụ của sáng chế, khi các mẫu sắc độ được lấy mẫu con không được sắp xếp cùng vị trí với các mẫu độ chói tương ứng (như được báo hiệu bởi cờ trong dòng bit), bảng 4 được sử dụng. Nếu không phải , các bảng 2 hoặc bảng 3 được sử dụng cho khối hiện tại.

Việc phân biệt giữa bảng 2 hoặc bảng 3 có thể được thực hiện trên cơ sở của số lượng các mẫu độ chói mà thuộc về khối hiện tại. Ví dụ, đối với các khối bao gồm 64 mẫu hoặc ít hơn, không có việc lọc sắc độ được áp dụng trong trường hợp khi không có việc lấy mẫu con sắc độ được thực hiện (bảng 2). Nếu không phải, khi kích cỡ khối lớn hơn 64 mẫu, bảng 3 đang được sử dụng để xác định bộ lọc "F". Giá trị 64 chỉ là ví dụ, các giá trị ngưỡng khác có thể áp dụng.

Theo phương án khác, bộ lọc F được lựa chọn theo khuôn dạng sắc độ và loại sắc độ như được thể hiện trong các bảng 6-10. Loại sắc độ chỉ rõ sự chuyển dịch của thành phần sắc độ và được thể hiện trên các Fig.10A và Fig.10B. Trong các bảng 6-10, bộ lọc được chỉ rõ trong cột "YUV 4:2:0" được sử dụng trong tình trạng kỹ thuật Dự thảo VVC. Các cột "YUV 4:2:2" và "YUV 4:4:4" xác định các bộ lọc mà thay thế các bộ lọc được xác định trong cột "YUV 4:2:0" khi khuôn dạng sắc độ tương ứng được xác định.

Bảng 6. Kết hợp của bộ lọc không gian F tới các giá trị của khuôn dạng sắc độ và loại sắc độ

Loại sắc độ	**YUV 4:2:0**	**YUV 4:2:2**	**YUV 4:4:4**
Loại-0	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

	$\begin{bmatrix}0&0&0\\0&1&0\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$
Loại-2	$\begin{bmatrix}0&0&0\\1&2&1\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\1&2&1\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$
	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\1&2&1\\1&2&1\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$

Bảng 7. Kết hợp của bộ lọc không gian F tới các giá trị của khuôn dạng sắc độ và loại sắc độ

Loại sắc độ	**YUV 4:2:0**	**YUV 4:2:2**	**YUV 4:4:4**
Loại-0	$\begin{bmatrix}0&0&0\\1&2&1\\1&2&1\end{bmatrix}$	$\begin{bmatrix}0&0&0\\1&2&1\\1&2&1\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$
	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$
	$\begin{bmatrix}0&0&0\\1&2&1\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\1&2&1\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$
	$\begin{bmatrix}0&0&0\\0&1&0\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$

Loại-2	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

Bảng 8. Kết hợp của bộ lọc không gian F tới các giá trị của khuôn dạng sắc độ và loại sắc độ

Loại sắc độ	**YUV 4:2:0**	**YUV 4:2:2**	**YUV 4:4:4**
Loại-0	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Loại-2	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

Bảng 9. Kết hợp của bộ lọc không gian F tới các giá trị của khuôn dạng sắc độ và loại sắc độ

Loại sắc độ	**YUV 4:2:0**	**YUV 4:2:2**	**YUV 4:4:4**
Loại-0	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Loại-2	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

Bảng 10. Kết hợp của bộ lọc không gian F tới các giá trị của khuôn dạng sắc độ và loại sắc độ

Loại sắc độ	**YUV 4:2:0**	**YUV 4:2:2**	**YUV 4:4:4**
Loại-0	$\begin{bmatrix}0&0&0\\1&2&1\\1&2&1\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$
	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$
	$\begin{bmatrix}0&0&0\\1&2&1\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$
	$\begin{bmatrix}0&0&0\\0&1&0\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$
Loại-2	$\begin{bmatrix}0&0&0\\1&2&1\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$
	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&4&1\\0&1&0\end{bmatrix}$

Bộ lọc $\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$ có thể được thực hiện trong các cách khác nhau, bao gồm hoạt động đường vòng bộ lọc (tức là bằng các thiết lập giá trị đầu ra thành giá trị

đầu vào). Ngoài ra, có thể được thực hiện nhờ sử dụng các phép toán cộng hoặc dịch chuyển tương tự, tức là:

$$Rec'_L[x,y] = (\sum_{i=-1}^{1}\sum_{j=-1}^{1} Rec'_L[x+i, y+j]\cdot F[i+1, j+1] + \frac{N}{2}) >> \log_2(N) = (N + \frac{N}{2}) >> \log_2(N)$$

Theo các thay đổi được đề xuất, phương pháp này có thể được thực hiện như là văn bản mô tả:

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

pDsY[x][y] =

(F[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

+ F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC *

y + 1] + 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] =

(F[1][0] * pY[0][SubHeightC * y − 1] +

+ F[0][1] * pY[− 1][SubHeightC * y] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ 2) >> 2

– Nếu không phải, pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] =

(2 * F[1][0] * pY[0][SubHeightC * y − 1] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ 2) >> 2

– Nếu availT bằng đúng (TRUE), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] =

(F[1][0] * pY[SubWidthC * x][− 1] +

+ F[0][1] * pY[SubWidthC * x − 1][0] +

+ F[1][1] * pY[SubWidthC * x][0] +

+ F[2][1] * pY[SubWidthC * x + 1][0] +

+ F[1][2] * pY[SubWidthC * x][1] + 4) >> 3

– Nếu không phải, pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] =

(F[1][0] * pY[SubWidthC * x][− 1] +

+ F[0][1] * pY[SubWidthC * x − 1][0] +

+ F[1][1] * pY[SubWidthC * x][0] +

+ F[2][1] * pY[SubWidthC * x + 1][0] +

+ F[1][2] * pY[SubWidthC * x][1] + 4) >> 3

– Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] =

(F[1][0] * pY[0][− 1] +

+ F[0][1] * pY[− 1][0] +

+ F[1][1] * pY[0][0] +

+ F[2][1] * pY[1][0] +

+ F[1][2] * pY[0][1] + 4) >> 3

– Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] =

(F[0][1] * pY[− 1][0] +

+ F[1][1] * pY[0][0] +

+ F[2][1] * pY[1][0] +

+ 2) >> 2

– Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = (pY[0][−1] + 2 * pY[0][0] + pY[0][1] + 2) >> 2 (8-169)

– Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = pY[0][0] (8-170)

– Nếu không phải, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] =

(F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[2][2] * pY[SubWidthC * x +1][SubHeightC *

y + 1] + 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] =

(F[0][1] * pY[− 1][SubHeightC * y] +

+ F[0][2] * pY[− 1][SubHeightC * y + 1] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] +

+ F[2][1] * pY[1][SubHeightC * y] +

+ F[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải, pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] =

(F[1][1] * pY[0][SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] + 1) >> 1

Bộ lọc F[i][j] được đề cập trong phần mô tả nêu trên được chỉ rõ theo các phương án của sáng chế.

Phương án ví dụ khác có thể được mô tả dưới dạng của phần dự thảo tiêu chuẩn kỹ thuật VVC như sau:

8.4.4.2.8 Mô tả kỹ thuật của chế độ dự đoán nội bộ INTRA_LT_CCLM, INTRA_L_CCLM và INTRA_T_CCLM

Các đầu vào đối với xử lý này là:

- chế độ dự đoán nội bộ predModeIntra,
- vị trí mẫu (xTbC, yTbC) của mẫu trên cùng-bên trái của khối biến đổi hiện tại so với mẫu trên cùng-bên trái của ảnh hiện tại,
- biến nTbW chỉ rõ độ rộng khối biến đổi,
- biến nTbH chỉ rõ độ cao khối biến đổi,
- các mẫu lân cận sắc độ p[x][y], với x = −1, y = 0..2 * nTbH − 1 và x = 0.. 2 * nTbW − 1, y = − 1.

Đầu ra của xử lý này là các mẫu được dự đoán predSamples[x][y], với x = 0..nTbW − 1, y = 0..nTbH − 1.

Vị trí sắc độ hiện tại (xTbY, yTbY) được thu nhận như sau:

(xTbY, yTbY) = (xTbC << (SubWidthC − 1), yTbC << (SubHeightC − 1))(8-156)

Các biến availL, availT và availTL được thu nhận như sau:

- Tính khả dụng của xử lý thu nhận các mẫu lân cận bên trái đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC) như là các đầu vào, và đầu ra được gán là availL.
- Tính khả dụng của xử lý thu nhận các mẫu lân cận trên cùng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC, yTbC − 1) như là các đầu vào, và đầu ra được gán là availT.
- Tính khả dụng của xử lý thu nhận các mẫu lân cận trên cùng-bên trái đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC − 1) như là các đầu vào, và đầu ra được gán là availTL.

- Số lượng của các mẫu sắc độ lân cận trên cùng-bên trái khả dụng numTopRight được thu nhận như sau:
  - Biến numTopRight được thiết lập bằng 0 và availTR được thiết lập bằng đúng (TRUE).
  - Khi predModeIntra bằng INTRA_T_CCLM, phần sau đây được áp dụng đối với x = nTbW..2 * nTbW − 1 cho đến khi availTR bằng sai (FALSE) hoặc x bằng 2 * nTbW − 1:
    - Xử lý thu nhận khả dụng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) và vị trí sắc độ lân cận (xTbC + x, yTbC − 1) như là các đầu vào, và đầu ra được gán là availableTR
    - Khi availableTR bằng đúng (TRUE), numTopRight được gia tăng thêm 1.
- Số lượng các mẫu sắc độ lân cận bên trái-phía dưới khả dụng numLeftBelow được thu nhận như sau:
  - Biến numLeftBelow được thiết lập bằng 0 và availLB được thiết lập bằng đúng (TRUE).
  - Khi predModeIntra bằng INTRA_L_CCLM, phần sau đây được áp dụng đối với y = nTbH..2 * nTbH − 1 cho đến khi availLB bằng sai (FALSE) hoặc y bằng 2 * nTbH − 1:
    - Xử lý thu nhận khả dụng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC + y) như là các đầu vào, và đầu ra được gán là availableLB
    - Khi availableLB bằng đúng (TRUE), numLeftBelow được gia tăng thêm 1.

Số lượng các mẫu sắc độ lân cận khả dụng trên phía trên cùng và trên cùng-bên trái numTopSamp và số lượng mẫu sắc độ lân cận khả dụng trên phía bên trái và bên trái-phía dưới nLeftSamp được thu nhận như sau:

– Nếu predModeIntra bằng INTRA_LT_CCLM, phần sau đây được áp dụng:

numSampT = availT ? nTbW : 0

numSampL = availL ? nTbH : 0

– Nếu không phải, phần sau đây được áp dụng:

numSampT = (availT && predModeIntra = = INTRA_T_CCLM) ? (nTbW + numTopRight) : 0

numSampL = (availL && predModeIntra = = INTRA_L_CCLM) ? (nTbH + numLeftBelow) : 0

Biến bCTUboundary được thu nhận như sau:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) = = 0) ?
Đúng (TRUE): Sai (FALSE).

Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu cả numSampL và numSampT bằng 0, phần sau đây được áp dụng:

predSamples[x][y] = 1 << ($BitDepth_C$ − 1)

– Nếu không phải, các bước có thứ tự sau đây được áp dụng:

1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với x = 0..nTbW * SubWidthC − 1, y= 0..nTbH * SubHeightC − 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).

2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:

   – Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = −1..−3, y = 0.. SubHeightC * numSampL − 1,

được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

– Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0.. SubWidthC * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

– Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pDsY[x][y] với x=1..nTbW-1, y=1..nTbH – 1 được thu nhận như sau:

pDstY[x][y] = pY[x][y] // chỉ để giải thích: Không có bộ lọc đối với YUV 4:4:4//

Nếu không phải, phần sau đây được áp dụng cho tập hợp của các bộ lọc {F3, F5, F6}. // Ở đây xác định các hệ số//

F3[0] = 1, F3[1] = 2, F3[2] = 1

–Nếu SubWidthC==2 và SubHeightC==2

F5[0][1] = 1, F5[1][1] = 4, F3[2][1] = 1, F5[1][0] = 1, F5[1][2] = 1

F6[0][1] = 1, F6[1][1] = 2, F6[2][1] = 1,

F6[0][2] = 1, F6[1][2] = 2, F6[2][2] = 1,

F2[0] = 1, F2[1] = 1

–Nếu không phải

F5[0][1] = 0, F5[1][1] = 8, F3[2][1] = 0, F5[1][0] = 0, F5[1][2] = 0

F6[0][1] = 2, F6[1][1] = 4, F6[2][1] = 2,

F6[0][2] = 0, F6[1][2] = 0, F6[2][2] = 0,

F2[0] = 2, F2[1] = 0

–// xem phần in đậm của các phương án của sáng chế //

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

–pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[x][y] =

(F[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

+ F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC *

y + 1] + 4) >> 3

//Chỉ để giải thích ở đây: áp dụng bộ lọc được xác định

và tất cả sự xuất hiện khác của bộ lọc "F" //

–Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[0][y] =

(F[1][0] * pY[0][SubHeightC * y − 1] +

+ F[0][1] * pY[− 1][SubHeightC * y] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[2][1] * pY[1] [SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] + 4) >> 3

–Nếu không phải, pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][y] =

(F[0] * pY[0][SubHeightC * y − 1] +

+ F[1] * pY[0][SubHeightC * y] +

+ F[2] * pY[0][SubHeightC * y + 1] +

+ 2) >> 2

–Nếu availT bằng đúng (TRUE), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[x][0] =

(F[1][0] * pY[SubWidthC * x][− 1] +

+ F[0][1] * pY[SubWidthC * x − 1][0] +

+ F[1][1] * pY[SubWidthC * x][0] +

+ F[2][1] * pY[SubWidthC * x + 1][0] +

+ F[1][2] * pY[SubWidthC * x][1] + 4) >> 3

–Nếu không phải, pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[x][0] =

= (F[0] * pY[SubWidthC * x − 1][0] +

+ F[1] * pY[SubWidthC * x][0] +

+ F[2] * pY[SubWidthC * x + 1][0] + 2) >> 2

–Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[0][0] =

(F[1][0] * pY[0][− 1] +

+ F[0][1] * pY[− 1][0] +

+ F[1][1] * pY[0][0] +

+ F[2][1] * pY[1][0] +

+ F[1][2] * pY[0][1] + 4) >> 3

–Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][0] =

(F[0] * pY[− 1][0] +

+ F[1] * pY[0][0] +

+ F[2] * pY[1][0] +

+ 2) >> 2

–Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][0] =

(F[0] * pY[0][-1] +

+ F[1] * pY[0][0] ++ F[2] * pY[0][1] +

+ 2) >> 2

–Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = pY[0][0]

– Nếu không phải, phần sau đây được áp dụng:

–pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F6:

pDsY[x][y] =

(F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[2][2] * pY[SubWidthC * x +1][SubHeightC *
y + 1] + 4) >> 3

–Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F6:

pDsY[0][y] =

(F[0][1] * pY[− 1][SubHeightC * y] +

+ F[0][2] * pY[− 1][SubHeightC * y + 1] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] +

+ F[2][1] * pY[1][SubHeightC * y] +

+ F[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3

–Nếu không phải, pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F2:

pDsY[0][y] =

(F[0] * pY[0][SubHeightC * y] +

+ F[1] * pY[0][SubHeightC * y + 1] + 1) >> 1

4. Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận được lấy mẫu xuống pLeftDsY[y] với y = 0..numSampL − 1 được thu nhận như sau:

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pLeftDsY[y] với y=0..nTbH – 1 được thu nhận như sau: pLeftDsY[y] = pY[-1][y]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– pLeftDsY[y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pLeftDsY[y] =

= F[1][0] * pY[− SubWidthC][SubHeightC * y − 1] +

+ F[0][1] * pY[−1 − SubWidthC][SubHeightC * y] +

+ F[1][1] * pY[− SubWidthC][SubHeightC * y] +

+ F[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

+ F[1][2] * pY[− SubWidthC][SubHeightC * y + 1] + 4) >> 3

– Nếu availTL bằng đúng (TRUE), pLeftDsY[0] được thu nhận như sau đối với F được thiết lập thành F5:

pLeftDsY[0] =

= F[1][0] * pY[− SubWidthC][− 1] +

+ F[0][1] * pY[−1 − SubWidthC][0] +

+ F[1][1] * pY[− SubWidthC][0] +

+ F[2][1] * pY[1 − SubWidthC][0] +

+ F[1][2] * pY[− SubWidthC][1] + 4) >> 3

– Nếu không phải, pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pLeftDsY [0] =

(F[0] * pY[−1 − SubWidthC][0] +

+ F[1] * pY[− SubWidthC][0] +

+ F[2] * pY[1 − SubWidthC][0] +

+ 2) >> 2

– Nếu không phải, phần sau đây được áp dụng đối với F được thiết lập thành F6:

pLeftDsY[y] =

= (F[0][1] * pY[−1− SubWidthC][SubHeightC * y] +

+ F[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

+ F[1][1] * pY[-- SubWidthC][SubHeightC * y] +

+ F[1][2] * pY[− SubWidthC][SubHeightC * y + 1] +

+ F[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

+ F[2][2] * pY[1 − SubWidthC][SubHeightC *

y + 1] + 4) >> 3

5. Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pTopDsY[x] với x = 0..numSampT − 1 được chỉ rõ như sau:

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pTopDsY[x] = pY[x][-1] đối với x=0..numSampT-1

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– pTopDsY[x] với x = 1..numSampT − 1 được thu nhận như sau:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F5:

pTopDsY[x] =

= (F[1][0] * pY[SubWidthC * x][− 1 − SubHeightC] +

+ F[0][1] * pY[SubWidthC * x − 1][− SubHeightC] +

+ F[1][1] * pY[SubWidthC * x][− SubHeightC] +

+ F[2][1] * pY[SubWidthC * x + 1][− SubHeightC] +

+ F[1][2] * pY[SubWidthC * x][1 − SubHeightC] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pTopDsY[x] =

= (F[0] * pY[SubWidthC * x − 1][−1] +

+ F[1] * pY[SubWidthC * x][−1] +

+ F[2] * pY[SubWidthC * x + 1][−1] +

+ 2) >> 2

– pTopDsY[0] được thu nhận như sau:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F5:

pTopDsY[0] =

= F[1][0] * pY[− 1] [− 1− SubHeightC] +

+ F[0][1] * pY[− 1] [− SubHeightC] +

+ F[1][1] * pY[0] [− SubHeightC] +

+ F[2][1] * pY[1] [− SubHeightC] + +

+ F[1][2] pY[− 1] [1 − SubHeightC] + + 4) >> 3

– Nếu không phải if availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pTopDsY[0] =

= (F[0] * pY[−1][−1] +

+ F[1] * pY[0][−1] +

+ F[2] * pY[1][-1] +

+ 2) >> 2

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pTopDsY[0] =

= (F[0] * pY[0][−1] +

+ F[1] * pY[0][-2] +

+ F[2] * pY[0][-1] +

+ 2) >> 2

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pTopDsY[0] = pY[0][−1]

– Nếu không phải, phần sau đây được áp dụng:

– pTopDsY[x] với x = 1..numSampT − 1 được thu nhận như sau:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pTopDsY[x] =

= (F[0][1] * pY[SubWidthC * x − 1][−2] +

+ F[0][2] * pY[SubWidthC * x − 1][−1] +

+ F[1][1] * pY[SubWidthC * x][−2] +

+ F[1][2] * pY[SubWidthC * x][−1] +

+ F[2][1] * pY[SubWidthC * x + 1][−2] +

+ F[2][2] * pY[SubWidthC * x + 1][−1] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pTopDsY[x] =

= (F[0] * pY [SubWidthC * y − 1] [− 1] +

+ F[1] * pY[SubWidthC * y] [− 1] +

+ F[2] * pY[SubWidthC * y + 1] [− 1] +

+ 2) >> 2

– pTopDsY[0] được thu nhận như sau:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pTopDsY[0] =

= (F[0][1] * pY[− 1][−2] +

+ F[0][2] * pY[− 1][−1] +

+ F[1][1] * pY[0][−2] +

+ F[1][2] * pY[0][−1] +

+ F[2][1] * pY[1][−2] +

+ F[2][2] * pY[1][−1] + 4) >> 3

– Nếu không phải if availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pTopDsY[0] =

= (F[0] * pY [− 1] [− 1] +

+ F[1] * pY[0] [− 1] +

+ F[2] * pY[1] [− 1] +

+ 2) >> 2

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F2:

pTopDsY[0] = (F[1] * pY[0][−2] + F[0] * pY[0][−1] + 1) >> 1

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pTopDsY[0] = pY[0][−1]

6. Các biến nS, xS, yS được thu nhận như sau:

– Nếu predModeIntra bằng INTRA_LT_CCLM, phần sau đây được áp dụng:

nS = ((availL && availT) ? Min(nTbW,
nTbH) : (availL ? nTbH : nTbW))

xS = 1 << (((nTbW > nTbH) && availL && availT) ? (Log2(nT bW) − Log2(nTbH)) : 0) (8-192)

yS = 1 << (((nTbH > nTbW) && availL && availT) ? (Log2(nT bH) − Log2(nTbW)) : 0) (8-193)

– Nếu không phải, nếu predModeIntra bằng INTRA_L_CCLM, phần sau đây được áp dụng:

nS = numSampL

xS = 1

yS = 1

– Nếu không phải (predModeIntra bằng INTRA_T_CCLM), phần sau đây được áp dụng:

nS = numSampT

xS = 1

yS = 1

7. Các biến minY, maxY, minC và maxC được thu nhận như sau:

– Biến minY được thiết lập bằng 1 << ($BitDepth_Y$) + 1 và biến maxY được thiết lập bằng −1.

– Nếu availT bằng đúng (TRUE), các biến minY, maxY, minC và maxC với x = 0..nS − 1 được thu nhận như sau:

– Nếu minY lớn hơn pTopDsY[x * xS], phần sau đây được áp dụng:

minY = pTopDsY[x * xS]

minC = p[x * xS][−1]

– Nếu maxY nhỏ hơn pTopDsY[x * xS], phần sau đây được áp dụng:

maxY = pTopDsY[x * xS]

maxC = p[x * xS][−1]

– Nếu availL bằng đúng (TRUE), các biến minY, maxY, minC và maxC với y = 0..nS − 1 được thu nhận như sau:

– Nếu minY lớn hơn pLeftDsY[y * yS], phần sau đây được áp dụng:

minY = pLeftDsY[y * yS]

minC = p[−1][y * yS]

– Nếu maxY nhỏ hơn pLeftDsY[y * yS], phần sau đây được áp dụng:

maxY = pLeftDsY[y * yS]

maxC = p[−1][y * yS]

8. Các biến a, b, và k được thu nhận như sau:

– Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

k = 0

a = 0

b = 1 << ($BitDepth_C$ − 1)

– Nếu không phải, phần sau đây được áp dụng:

diff = maxY − minY

– Nếu diff không bằng 0, phần sau đây được áp dụng:

diffC = maxC − minC

x = Floor(Log2(diff))

normDiff = ((diff << 4) >> x) & 15

x += (normDiff != 0) ? 1 : 0

y = Floor(Log2(Abs (diffC))) + 1

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y$$

$$k = ((3 + x - y) < 1) ? 1 : 3 + x - y$$

$$a = ((3 + x - y) < 1) ? Sign(a) * 15 : a$$

$$b = minC - ((a * minY) >> k)$$

Trong đó divSigTable[ ] được chỉ rõ như sau:

divSigTable[ ] = { 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 }

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

$$k = 0$$

$$a = 0$$

$$b = minC$$

9. Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

$$predSamples[x][y] = Clip1C(((pDsY[x][y] * a) >> k) + b)$$

Phương án khác mô tả phương pháp để thu nhận các tham số CCLM với nhiều nhất bốn mẫu sắc độ lân cận và các mẫu độ chói được lấy mẫu xuống tương ứng của chúng.

Giả định các kích thước khối sắc độ hiện tại là W×H, thì W' và H' được thiết lập là

- W'=W, H'=H khi chế độ LM được áp dụng;
- W'=W+H khi chế độ LM-A được áp dụng;
- H'=H+W khi chế độ LM-L được áp dụng;

Các vị trí lân cận phía trên được ký là S[0, -1]…S[W'-1, -1] và các vị trí lân cận bên trái được ký hiệu là S[-1, 0]…S[-1, H'-1]. Sau đó bốn mẫu được lựa chọn là

- S[W'/4, -1], S[3W'/4, -1], S[-1, H'/4], S[-1, 3H'/4] khi chế độ LM được áp dụng và cả mẫu lân cận bên trái và phía trên là khả dụng;

- S[W'/8, -1], S[3W'/8, -1], S[5W'/8, -1], S[7W'/8, -1] khi chế độ LM-A được áp dụng hoặc chỉ các mẫu lân cận phía trên là khả dụng;
- S[-1, H'/8], S[-1, 3H'/8], S[-1, 5H'/8], S[-1, 7H'/8] khi chế độ LM-L được áp dụng hoặc chỉ các mẫu lân cận bên trái là khả dụng;

Bốn mẫu độ chói lân cận tại vị trí được lựa chọn được lấy mẫu xuống và được so sánh bốn lần để tìm ra hai giá trị nhỏ hơn: x^0_A và x^1_A, và hai giá trị lớn hơn: x^0_B và x^1_B. Các giá trị mẫu sắc độ tương ứng được ký hiệu là y^0_A, y^1_A, y^0_B và y^1_B. Sau đó x_A, x_B, y_A và y_B được thu nhận là:

$x_A=(x^0_A + x^1_A +1)>>1$; $x_B=(x^0_B + x^1_B +1)>>1$; $y_A=(y^0_A + y^1_A +1)>>1$; $y_B=(y^0_B + y^1_B +1)>>1$.

Phần mô tả dưới dạng của phần dự thảo tiêu chuẩn kỹ thuật VVC như sau:

8.4.4.2.8 Mô tả kỹ thuật của chế độ dự đoán nội bộ INTRA_LT_CCLM, INTRA_L_CCLM và INTRA_T_CCLM

Các đầu vào đối với xử lý này là:

- chế độ dự đoán nội bộ predModeIntra,
- vị trí mẫu (xTbC, yTbC) của mẫu trên cùng-bên trái của khối biến đổi hiện tại so với mẫu trên cùng-bên trái của ảnh hiện tại,
- biến nTbW chỉ rõ độ rộng khối biến đổi,
- biến nTbH chỉ rõ độ cao khối biến đổi,
- các mẫu lân cận sắc độ p[x][y], với x = −1, y = 0..2 * nTbH − 1 và x = 0.. 2 * nTbW − 1, y = − 1.

Đầu ra của xử lý này là các mẫu được dự đoán predSamples[x][y], với x = 0..nTbW − 1, y = 0..nTbH − 1.

Vị trí sắc độ hiện tại (xTbY, yTbY) được thu nhận như sau:

(xTbY, yTbY) = (xTbC << (SubWidthC − 1), yTbC << (SubHeightC − 1))

Các biến availL, availT và availTL được thu nhận như sau:

- Tính khả dụng của xử lý thu nhận các mẫu lân cận bên trái đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC) như là các đầu vào, và đầu ra được gán là availL.
- Tính khả dụng của xử lý thu nhận các mẫu lân cận trên cùng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC, yTbC − 1) như là các đầu vào, và đầu ra được gán là availT.
- Tính khả dụng của xử lý thu nhận các mẫu lân cận trên cùng-bên trái đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC − 1) như là các đầu vào, và đầu ra được gán là availTL.
- Số lượng của các mẫu sắc độ lân cận trên cùng-bên trái khả dụng numTopRight được thu nhận như sau:
  - Biến numTopRight được thiết lập bằng 0 và availTR được thiết lập bằng đúng (TRUE).
  - Khi predModeIntra bằng INTRA_T_CCLM, phần sau đây được áp dụng đối với x = nTbW..2 * nTbW − 1 cho đến khi availTR bằng sai (FALSE) hoặc x bằng 2 * nTbW − 1:
    - Xử lý thu nhận khả dụng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) và vị trí sắc độ lân cận (xTbC + x, yTbC − 1) như là các đầu vào, và đầu ra được gán là availableTR
    - Khi availableTR bằng đúng (TRUE), numTopRight được gia tăng thêm 1.
- Số lượng các mẫu sắc độ lân cận bên trái-phía dưới khả dụng numLeftBelow được thu nhận như sau:

- Biến numLeftBelow được thiết lập bằng 0 và availLB được thiết lập bằng đúng (TRUE).
- Khi predModeIntra bằng INTRA_L_CCLM, phần sau đây được áp dụng đối với y = nTbH..2 * nTbH − 1 cho đến khi availLB bằng sai (FALSE) hoặc y bằng 2 * nTbH − 1:
  - Xử lý thu nhận khả dụng đối với khối được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) và vị trí sắc độ lân cận (xTbC − 1, yTbC + y) như là các đầu vào, và đầu ra được gán là availableLB
  - Khi availableLB bằng đúng (TRUE), numLeftBelow được gia tăng thêm 1.

Số lượng các mẫu sắc độ lân cận khả dụng trên phía trên cùng và trên cùng-bên trái numTopSamp và số lượng mẫu sắc độ lân cận khả dụng trên phía bên trái và bên trái-phía dưới nLeftSamp được thu nhận như sau:

- Nếu predModeIntra bằng INTRA_LT_CCLM, phần sau đây được áp dụng:

  numSampT = availT ? nTbW : 0

  numSampL = availL ? nTbH : 0

- Nếu không phải, phần sau đây được áp dụng:

  numSampT = (availT && predModeIntra = = INTRA_T_CCLM) ?

  (nTbW + Min(numTopRight, nTbH)) : 0

  numSampL = (availL && predModeIntra = = INTRA_L_CCLM) ?

  (nTbH + Min(numLeftBelow, nTbW)) : 0

Biến bCTUboundary được thu nhận như sau:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) = = 0) ?

Đúng (TRUE): Sai (FALSE).

Biến cntN và mảng pickPosN[] với N được thay thế bởi L và T, được thu nhận như sau:

- Biến numIs4N được thiết lập bằng ((availT && availL && predModeIntra == INTRA_LT_CCLM) ? 0 : 1).
- Biến startPosN được thiết lập bằng numSampN >> (2 + numIs4N).
- Biến pickStepN được thiết lập bằng Max(1, numSampN >> (1 + numIs4N)).
- Nếu availN bằng đúng (TRUE) và predModeIntra bằng INTRA_LT_CCLM hoặc INTRA_N_CCLM, cntN được thiết lập bằng Min(numSampN, (1 + numIs4N) << 1), và pickPosN[pos] được thiết lập bằng (startPosN + pos * pickStepN), với pos = 0..(cntN – 1).
- Nếu không phải, cntN được thiết lập bằng 0.

Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW – 1, y = 0..nTbH – 1 được thu nhận như sau:

- Nếu cả numSampL và numSampT bằng 0, phần sau đây được áp dụng:

  predSamples[x][y] = 1 << ($BitDepth_C$ – 1)

- Nếu không phải, các bước có thứ tự sau đây được áp dụng:

  1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với x = 0..nTbW * SubWidthC – 1, y= 0..nTbH * SubHeightC – 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).
  2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:
     - Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = –1..–3, y = 0.. SubHeightC * numSampL – 1, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

– Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0.. SubWidthC * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

– Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pDsY[x][y] với x=1..nTbW-1, y=1..nTbH – 1 được thu nhận như sau:

pDstY[x][y] = pY[x][y]

– Nếu không phải, phần sau đây được áp dụng cho tập hợp của các bộ lọc {F3, F5, F6}.

F3[0] = 1, F3[1] = 2, F3[2] = 1

– Nếu SubWidthC==2 và SubHeightC==2

F5[0][1] = 1, F5[1][1] = 4, F3[2][1] = 1, F5[1][0] = 1, F5[1][2] = 1

F6[0][1] = 1, F6[1][1] = 2, F6[2][1] = 1,

F6[0][2] = 1, F6[1][2] = 2, F6[2][2] = 1,

F2[0] = 1, F2[1] = 1

– Nếu không phải

F5[0][1] = 0, F5[1][1] = 8, F3[2][1] = 0, F5[1][0] = 0, F5[1][2] = 0

F6[0][1] = 2, F6[1][1] = 4, F6[2][1] = 2,

F6[0][2] = 0, F6[1][2] = 0, F6[2][2] = 0,

F2[0] = 2, F2[1] = 0

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

–pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[x][y] =

(F[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

+ F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC *

y + 1] + 4) >> 3

–Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[0][y] =

(F[1][0] * pY[0][SubHeightC * y − 1] +

+ F[0][1] * pY[− 1][SubHeightC * y] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[2][1] * pY[1] [SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] + 4) >> 3

–Nếu không phải, pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][y] =

(F[0] * pY[0][SubHeightC * y − 1] +

+ F[1] * pY[0][SubHeightC * y] +

+ F[2] * pY[0][SubHeightC * y + 1] +

+ 2) >> 2

–Nếu availT bằng đúng (TRUE), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[x][0] =

(F[1][0] * pY[SubWidthC * x][− 1] +

+ F[0][1] * pY[SubWidthC * x − 1][0] +

+ F[1][1] * pY[SubWidthC * x][0] +

+ F[2][1] * pY[SubWidthC * x + 1][0] +

+ F[1][2] * pY[SubWidthC * x][1] + 4) >> 3

–Nếu không phải, pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[x][0] =

= (F[0] * pY[SubWidthC * x − 1][0] +

+ F[1] * pY[SubWidthC * x][0] +

+ F[2] * pY[SubWidthC * x + 1][0] + 2) >> 2

–Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[0][0] =

(F[1][0] * pY[0][− 1] +

+ F[0][1] * pY[− 1][0] +

+ F[1][1] * pY[0][0] +

+ F[2][1] * pY[1][0] +

+ F[1][2] * pY[0][1] + 4) >> 3

–Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][0] =

(F[0] * pY[− 1][0] +

+ F[1] * pY[0][0] +

+ F[2] * pY[1][0] +

+ 2) >> 2

–Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][0] =

(F[0] * pY[0][-1] +

+ F[1] * pY[0][0] +

+ F[2] * pY[0][1] +

+ 2) >> 2

–Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = pY[0][0]

– Nếu không phải, phần sau đây được áp dụng:

–pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F6:

pDsY[x][y] =

(F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[2][2] * pY[SubWidthC * x +1][SubHeightC *

y + 1] + 4) >> 3

–Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F6:

pDsY[0][y] =

(F[0][1] * pY[− 1][SubHeightC * y] +

+ F[0][2] * pY[− 1][SubHeightC * y + 1] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] +

+ F[2][1] * pY[1][SubHeightC * y] +

+ F[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3

–Nếu không phải, pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F2:

pDsY[0][y] =

(F[0] * pY[0][SubHeightC * y] +

+ F[1] * pY[0][SubHeightC * y + 1] + 1) >> 1

4. Khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[–1][pickPosL[idx]] với idx = 0..(cntL

– 1), và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY [idx] với idx = 0..(cntL-1) được thu nhận như sau:

– Biến y được thiết lập bằng pickPosL[idx].

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pSelDsY [i] = pY[-1][y]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– Nếu y > 0 || availTL == đúng (TRUE), đối với F được thiết lập thành F5:

pSelDsY [idx] =

= F[1][0] * pY[− SubWidthC][SubHeightC * y − 1] +

+ F[0][1] * pY[−1 − SubWidthC][SubHeightC * y] +

+ F[1][1] * pY[− SubWidthC][SubHeightC * y] +

+ F[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

+ F[1][2] * pY[− SubWidthC][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải, đối với F được thiết lập thành F3:

pSelDsY [idx] =

(F[0] * pY[−1 − SubWidthC][0] +

+ F[1] * pY[− SubWidthC][0] +

+ F[2] * pY[1 − SubWidthC][0] +

+ 2) >> 2

– Nếu không phải, phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[−1− SubWidthC][SubHeightC * y] +

+ F[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

+ F[1][1] * pY[− SubWidthC][SubHeightC * y] +

+ F[1][2] * pY[− SubWidthC][SubHeightC * y + 1] +

+ F[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

+ F[2][2] * pY[1 − SubWidthC][SubHeightC *

y + 1] + 4) >> 3

5. Khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx – cntL]][-1] với idx = cntL..(cntL + cntT – 1), và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY [idx] với idx = cntL.. cntL+ cntT − 1 được chỉ rõ như sau:

– Biến x được thiết lập bằng pickPosT[idx – cntL].

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pSelDsY [idx] = pY[x][-1]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– Nếu x > 0:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F5:

pSelDsY [idx] =

= (F[1][0] * pY[SubWidthC * x][− 1 − SubHeightC] +

+ F[0][1] * pY[SubWidthC * x − 1][− SubHeightC] +

+ F[1][1] * pY[SubWidthC * x][− SubHeightC] +

+ F[2][1] * pY[SubWidthC * x + 1][− SubHeightC] +

+ F[1][2] * pY[SubWidthC * x][1 − SubHeightC] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY[SubWidthC * x − 1][−1] +

+ F[1] * pY[SubWidthC * x][−1] +

+ F[2] * pY[SubWidthC * x + 1][−1] +

+ 2) >> 2

– Nếu không phải:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F5:

pSelDsY [idx] =

= F[1][0] * pY[− 1] [− 1− SubHeightC] +

+ F[0][1] * pY[− 1] [− SubHeightC] +

+ F[1][1] * pY[0] [− SubHeightC] +

+ F[2][1] * pY[1] [− SubHeightC] +

+ F[1][2] pY[− 1] [1 − SubHeightC] + 4) >> 3

– Nếu không phải if availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY[−1][−1] +

+ F[1] * pY[0][−1] +(8-182)

+ F[2] * pY[1][-1] +

+ 2) >> 2

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY[0][−1] +

+ F[1] * pY[0][−2] +

+ F[2] * pY[0][-1] +

+ 2) >> 2

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY [idx] = pY[0][−1]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu x > 0:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[SubWidthC * x − 1][−2] +

+ F[0][2] * pY[SubWidthC * x − 1][−1] +

+ F[1][1] * pY[SubWidthC * x][−2] +

+ F[1][2] * pY[SubWidthC * x][−1] +

+ F[2][1] * pY[SubWidthC * x + 1][−2] +

+ F[2][2] * pY[SubWidthC * x + 1][−1] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY [SubWidthC * y − 1] [− 1] +

+ F[1] * pY[SubWidthC * y] [− 1] +

+ F[2] * pY[SubWidthC * y + 1] [− 1] +

+ 2) >> 2

– Nếu không phải:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[− 1][−2] +

+ F[0][2] * pY[− 1][−1] +

+ F[1][1] * pY[0][−2] +

+ F[1][2] * pY[0][−1] +

+ F[2][1] * pY[1][−2] +

+ F[2][2] * pY[1][−1] + 4) >> 3

– Nếu không phải if availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY [− 1] [− 1] +

+ F[1] * pY[0] [− 1] +

+ F[2] * pY[1] [− 1] +

+ 2) >> 2

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F2:

pSelDsY [idx] = (F[1] * pY[0][−2] + F[0] * pY[0][−1] + 1) >> 1

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY [idx] = pY[0][−1]

6. Khi cntT+ cntL không bằng 0, các biến minY, maxY, minC và maxC được thu nhận như sau:

– Khi cntT+cntL bằng 2, thiết lập pSelComp[3] bằng pSelComp[0], pSelComp[2] bằng pSelComp[1], pSelComp[0] bằng pSelComp[1], và pSelComp[1] bằng pSelComp[3], với Comp được thay thế bởi DsY và C.

– Các mảng minGrpIdx[] và maxGrpIdx[] được thiết lập là: minGrpIdx[0] = 0, minGrpIdx[1] = 2, maxGrpIdx[0] = 1, maxGrpIdx[1] = 3.

– Nếu pSelDsY[minGrpIdx[0]] > pSelDsY[minGrpIdx[1]], Swap(minGrpIdx[0], minGrpIdx[1]).

– Nếu pSelDsY[maxGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(maxGrpIdx[0], maxGrpIdx[1]).

– Nếu pSelDsY[minGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(minGrpIdx, maxGrpIdx).

– Nếu pSelDsY[minGrpIdx[1]] > pSelDsY[maxGrpIdx[0]], Swap(minGrpIdx[1], maxGrpIdx[0]).

- maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1.
- maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1.
- minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1.
- minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1.

7. Các biến a, b, và k được thu nhận như sau:

- Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

  k = 0

  a = 0

  b = 1 << ($BitDepth_C$ − 1)

- Nếu không phải, phần sau đây được áp dụng:

  diff = maxY − minY

  - Nếu diff không bằng 0, phần sau đây được áp dụng:

    diffC = maxC − minC

    x = Floor(Log2(diff))

    normDiff = ((diff << 4) >> x) & 15

    x += (normDiff != 0) ? 1 : 0

    y = Floor(Log2(Abs (diffC))) + 1

    a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y

    k = ((3 + x − y) < 1) ? 1 : 3 + x − y

    a = ((3 + x − y) < 1) ? Sign(a) * 15 : a

    b = minC − ((a * minY) >> k)

Trong đó divSigTable[ ] được chỉ rõ như sau:

divSigTable[ ] = { 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 }

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

k = 0

a = 0

b = minC

8. Các mẫu dự đoán predSamples[x][y] với
x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

predSamples[x][y] = Clip1C(((pDsY[x][y] * a) >> k) + b)

Phương án khác có sự khác biệt về truy xuất các mẫu tham chiếu độ chói so với vị trí của khối được dự đoán liên quan đến biên khối. Cụ thể, khi khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2 và các thành phần sắc độ và độ chói được sắp xếp cùng vị trí (được thể hiện là "Loại mẫu sắc độ 2" và "Loại mẫu sắc độ 4" trên Fig.10B), việc lấy mẫu từ thành phần độ chói có thể được thực hiện với các độ dịch khác nhau liên quan đến phía trên cùng của khối độ chói (xem Fig.12A và Fig 12C).

Để thu nhận mẫu tham chiếu độ chói được lọc tại vị trí 1202, bộ lọc với các hệ số [1 2 1]/4 được áp dụng tới tập hợp của các mẫu tham chiếu mà bao gồm các mẫu tại các vị trí 1202. Fig.12A thể hiện phương án trong đó mẫu bên trái-trên cùng là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2 hoặc biên khối 1201 là biên dòng CTU. Trong trường hợp này, bộ lọc [1 2 1]/4 được áp dụng tới các mẫu tham chiếu 1202 (được thể hiện bởi các hình chữ nhật có nét gạch), sao cho hệ số 2/4 được áp dụng tới hệ số trong vị trí trung tâm.

Fig.12B thể hiện phương án trong đó biên khối 1201 không phải biên dòng CTU, mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:0 (hoặc bất kỳ khuôn dạng sắc độ khác mà sử dụng việc lấy mẫu con sắc độ dọc). Trong trường hợp này, bộ lọc [1 2 1; 1 2 1] / 8 được áp dụng tới các

mẫu tham chiếu 1202 (được thể hiện bởi các hình chữ nhật có nét gạch), sao cho các hệ số "2/4" được áp dụng tới các hệ số trong hai vị trí trung tâm.

Fig.12C thể hiện phương án trong đó mẫu bên trái-trên cùng là không khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2 hoặc biên khối 1201 là biên dòng CTU.

Trong trường hợp này, bộ lọc đường vòng [1]/1 được áp dụng tới mẫu tham chiếu 1202 (được thể hiện bởi hình chữ nhật có nét gạch).

Fig.12D thể hiện phương án trong đó biên khối 1201 không phải biên dòng CTU, mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:0 (hoặc bất kỳ khuôn dạng sắc độ khác mà sử dụng việc lấy mẫu con sắc độ dọc). Trong trường hợp này, bộ lọc [1 1] / 2 được áp dụng tới các mẫu tham chiếu 1202 (được thể hiện bởi các hình chữ nhật có nét gạch).

Một phần của dự thảo tiêu chuẩn kỹ thuật VVC được thể hiện dưới đây tương ứng với các trường hợp được thể hiện trên các FIG.12A-12D:

Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

- Nếu cả numSampL và numSampT bằng 0, phần sau đây được áp dụng:

  predSamples[x][y] = 1 << ($BitDepth_C$ − 1)

- Nếu không phải, các bước có thứ tự sau đây được áp dụng:

  1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với x = 0..nTbW * SubWidthC − 1, y= 0..nTbH * SubHeightC − 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).

  2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:

     - Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = −1..−3, y = 0.. SubHeightC * numSampL − 1,

được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

– Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0.. SubWidthC * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

– Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pDsY[x][y] với x=1..nTbW-1, y=1..nTbH – 1 được thu nhận như sau:

pDstY[x][y] = pY[x][y]

– Nếu không phải, phần sau đây được áp dụng cho tập hợp của các bộ lọc {F3, F5, F6}.

F3[0] = 1, F3[1] = 2, F3[2] = 1

– Nếu SubWidthC==2 và SubHeightC==2

F5[0][1] = 1, F5[1][1] = 4, F3[2][1] = 1, F5[1][0] = 1, F5[1][2] = 1

F6[0][1] = 1, F6[1][1] = 2, F6[2][1] = 1,

F6[0][2] = 1, F6[1][2] = 2, F6[2][2] = 1,

F2[0] = 1, F2[1] = 1

– Nếu không phải

F5[0][1] = 0, F5[1][1] = 8, F3[2][1] = 0, F5[1][0] = 0, F5[1][2] = 0

F6[0][1] = 2, F6[1][1] = 4, F6[2][1] = 2,

F6[0][2] = 0, F6[1][2] = 0, F6[2][2] = 0,

F2[0] = 2, F2[1] = 0

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

–pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[x][y] =

(F[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

+ F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC *

y + 1] + 4) >> 3

–Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[0][y] =

(F[1][0] * pY[0][SubHeightC * y − 1] +

+ F[0][1] * pY[− 1][SubHeightC * y] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[2][1] * pY[1] [SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] + 4) >> 3

–Nếu không phải, pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][y] =

(F[0] * pY[0][SubHeightC * y − 1] +

+ F[1] * pY[0][SubHeightC * y] +

+ F[2] * pY[0][SubHeightC * y + 1] +

+ 2) >> 2

–Nếu availT bằng đúng (TRUE), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[x][0] =

(F[1][0] * pY[SubWidthC * x][− 1] +

+ F[0][1] * pY[SubWidthC * x − 1][0] +

+ F[1][1] * pY[SubWidthC * x][0] +

+ F[2][1] * pY[SubWidthC * x + 1][0] +

+ F[1][2] * pY[SubWidthC * x][1] + 4) >> 3

–Nếu không phải, pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[x][0] =

= (F[0] * pY[SubWidthC * x − 1][0] +

+ F[1] * pY[SubWidthC * x][0] +

+ F[2] * pY[SubWidthC * x + 1][0] + 2) >> 2

–Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F5:

pDsY[0][0] =

(F[1][0] * pY[0][− 1] +

+ F[0][1] * pY[− 1][0] +

+ F[1][1] * pY[0][0] +

+ F[2][1] * pY[1][0] +

+ F[1][2] * pY[0][1] + 4) >> 3

–Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][0] =

(F[0] * pY[− 1][0] +

+ F[1] * pY[0][0] +

+ F[2] * pY[1][0] +

+ 2) >> 2

–Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau đối với F được thiết lập thành F3:

pDsY[0][0] =

(F[0] * pY[0][-1] +

+ F[1] * pY[0][0] +

+ F[2] * pY[0][1] +

+ 2) >> 2

–Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = pY[0][0]

– Nếu không phải, phần sau đây được áp dụng:

–pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F6:

pDsY[x][y] =

(F[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

+ F[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

+ F[1][1] * pY[SubWidthC * x][SubHeightC * y] +

+ F[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

+ F[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

+ F[2][2] * pY[SubWidthC * x +1][SubHeightC *

y + 1] + 4) >> 3

–Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F6:

pDsY[0][y] =

(F[0][1] * pY[− 1][SubHeightC * y] +

+ F[0][2] * pY[− 1][SubHeightC * y + 1] +

+ F[1][1] * pY[0][SubHeightC * y] +

+ F[1][2] * pY[0][SubHeightC * y + 1] +

+ F[2][1] * pY[1][SubHeightC * y] +

+ F[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3

–Nếu không phải, pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau đối với F được thiết lập thành F2:

pDsY[0][y] =

(F[0] * pY[0][SubHeightC * y] +

+ F[1] * pY[0][SubHeightC * y + 1] + 1) >> 1

4. Khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[−1][pickPosL[idx]] với idx = 0..(cntL − 1), và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY [idx] với idx = 0..(cntL-1) được thu nhận như sau:

– Biến y được thiết lập bằng pickPosL[idx].

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pSelDsY [i] = pY[-1][y]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– Nếu y > 0 || availTL == đúng (TRUE), đối với F được thiết lập thành F5:

pSelDsY [idx] =

= F[1][0] * pY[− SubWidthC][SubHeightC * y − 1] +

+ F[0][1] * pY[−1 − SubWidthC][SubHeightC * y] +

+ F[1][1] * pY[− SubWidthC][SubHeightC * y] +

+ F[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

+ F[1][2] * pY[− SubWidthC][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải, đối với F được thiết lập thành F3:

pSelDsY [idx] =

(F[0] * pY[−1 − SubWidthC][0] +

+ F[1] * pY[− SubWidthC][0] +

+ F[2] * pY[1 − SubWidthC][0] +

+ 2) >> 2

– Nếu không phải, phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[−1− SubWidthC][SubHeightC * y] +

+ F[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

+ F[1][1] * pY[− SubWidthC][SubHeightC * y] +

+ F[1][2] * pY[− SubWidthC][SubHeightC * y + 1] +

+ F[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

+ F[2][2] * pY[1 − SubWidthC][SubHeightC *

y + 1] + 4) >> 3

5. Khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx – cntL]][-1] với idx = cntL..(cntL + cntT – 1), và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY [idx] với idx = cntL.. cntL+ cntT − 1 được chỉ rõ như sau:

– Biến x được thiết lập bằng pickPosT[idx – cntL].

– Nếu SubWidthC==1 và SubHeightC==1, phần sau đây được áp dụng:

– pSelDsY [idx] = pY[x][-1]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_cclm_colocated_chroma_flag bằng 1, phần sau đây được áp dụng:

– Nếu x > 0:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F5:

pSelDsY [idx] =

= (F[1][0] * pY[SubWidthC * x][− 1 − SubHeightC] +

+ F[0][1] * pY[SubWidthC * x − 1][− SubHeightC] +

+ F[1][1] * pY[SubWidthC * x][− SubHeightC] +

+ F[2][1] * pY[SubWidthC * x + 1][− SubHeightC] +

+ F[1][2] * pY[SubWidthC * x][1 − SubHeightC] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY[SubWidthC * x − 1][−1] +

+ F[1] * pY[SubWidthC * x][−1] +

+ F[2] * pY[SubWidthC * x + 1][−1] +

+ 2) >> 2

– Nếu không phải:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F5:

pSelDsY [idx] =

= F[1][0] * pY[− 1] [− 1− SubHeightC] +

+ F[0][1] * pY[− 1] [− SubHeightC] +

+ F[1][1] * pY[0] [− SubHeightC] +

+ F[2][1] * pY[1] [− SubHeightC] +

+ F[1][2] pY[− 1] [1 − SubHeightC] + 4) >> 3

– Nếu không phải if availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY[−1][−1] +

+ F[1] * pY[0][−1] +(8-182)

+ F[2] * pY[1][-1] +

+ 2) >> 2

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY[0][−1] +

+ F[1] * pY[0][−2] +

+ F[2] * pY[0][-1] +

+ 2) >> 2

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY [idx] = pY[0][−1]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu x > 0:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[SubWidthC * x − 1][-1] +

+ F[0][2] * pY[SubWidthC * x − 1][-2] +

+ F[1][1] * pY[SubWidthC * x][-1] +

+ F[1][2] * pY[SubWidthC * x][-2] +

+ F[2][1] * pY[SubWidthC * x + 1][-1] +

+ F[2][2] * pY[SubWidthC * x + 1][-2] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY [SubWidthC * y − 1] [− 1] +

+ F[1] * pY[SubWidthC * y] [− 1] +

+ F[2] * pY[SubWidthC * y + 1] [− 1] +

+ 2) >> 2

– Nếu không phải:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[− 1][-1] +

+ F[0][2] * pY[− 1][-2] +

+ F[1][1] * pY[0][-1] +

+ F[1][2] * pY[0][-2] +

+ F[2][1] * pY[1][-1] +

+ F[2][2] * pY[1][-2] + 4) >> 3

– Nếu không phải if availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY [− 1] [− 1] +

$$+ F[1] * pY[\,0\,][\,-1\,] +$$

$$+ F[2] * pY[\,1\,][\,-1\,] +$$

$$+ 2\,) >> 2$$

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F2:

$$pSelDsY\,[\,idx\,] = (\,F[1] * pY[\,0\,][\,-2\,] + F[0] * pY[\,0\,][\,-1\,] + 1\,) >> 1$$

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

$$pSelDsY\,[\,idx\,] = pY[\,0\,][\,-1\,]$$

6. Khi cntT+ cntL không bằng 0, các biến minY, maxY, minC và maxC được thu nhận như sau:

– Khi cntT+cntL bằng 2, thiết lập pSelComp[3] bằng pSelComp[0], pSelComp[2] bằng pSelComp[1], pSelComp[0] bằng pSelComp[1], và pSelComp[1] bằng pSelComp[3], với Comp được thay thế bởi DsY và C.

– Các mảng minGrpIdx[] và maxGrpIdx[] được thiết lập là: minGrpIdx[0] = 0, minGrpIdx[1] = 2, maxGrpIdx[0] = 1, maxGrpIdx[1] = 3.

– Nếu pSelDsY[minGrpIdx[0]] > pSelDsY[minGrpIdx[1]], Swap(minGrpIdx[0], minGrpIdx[1]).

– Nếu pSelDsY[maxGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(maxGrpIdx[0], maxGrpIdx[1]).

– Nếu pSelDsY[minGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(minGrpIdx, maxGrpIdx).

- Nếu pSelDsY[minGrpIdx[1]] > pSelDsY[maxGrpIdx[0]], Swap(minGrpIdx[1], maxGrpIdx[0]).
- maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1.
- maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1.
- minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1.
- minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1.

7. Các biến a, b, và k được thu nhận như sau:
   - Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

     k = 0

     a = 0

     b = 1 << ($BitDepth_C$ − 1)
   - Nếu không phải, phần sau đây được áp dụng:

     diff = maxY − minY

     - Nếu diff không bằng 0, phần sau đây được áp dụng:

       diffC = maxC − minC

       x = Floor(Log2(diff))

       normDiff = ((diff << 4) >> x) & 15

       x += (normDiff != 0) ? 1 : 0

       y = Floor(Log2(Abs (diffC))) + 1

       a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y

       k = ((3 + x − y) < 1) ? 1 : 3 + x − y

       a = ((3 + x − y) < 1) ? Sign(a) * 15 : a

b = minC − ((a * minY) >> k)

Trong đó divSigTable[ ] được chỉ rõ như sau:

divSigTable[ ] = { 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 }

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

k = 0

a = 0

b = minC

8. Các mẫu dự đoán predSamples[x][y] với
x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

predSamples[x][y] = Clip1C(((pDsY[x][y] * a) >> k) + b)

Công thức khác của bản mô tả này sử dụng ký hiệu tuần tự của các bộ lọc (F1, F2, F3 và F4). Sáng chế sử dụng ký hiệu của các bộ lọc tuần tự là như sau:

Các đầu vào đối với xử lý này là:

– chế độ dự đoán nội bộ predModeIntra,

– vị trí mẫu (xTbC, yTbC) của mẫu trên cùng-bên trái của khối biến đổi hiện tại so với mẫu trên cùng-bên trái của ảnh hiện tại,

– biến nTbW chỉ rõ độ rộng khối biến đổi,

– biến nTbH chỉ rõ độ cao khối biến đổi,

– biến cIdx chỉ rõ thành phần màu của khối hiện tại,

– các mẫu lân cận sắc độ p[x][y], với x = −1, y = 0..2 * nTbH − 1 và x = 0.. 2 * nTbW − 1, y = − 1.

Đầu ra của xử lý này là các mẫu được dự đoán predSamples[x][y], với x = 0..nTbW − 1, y = 0..nTbH − 1.

Vị trí sắc độ hiện tại (xTbY, yTbY) được thu nhận như sau:

$$(xTbY, yTbY) = (xTbC << (SubWidthC - 1), yTbC << (SubHeightC - 1)) \quad (346)$$

Các biến availL, availT và availTL được thu nhận như sau:

- Xử lý thu nhận tính khả dụng khối lân cận được chỉ rõ trong mục 6.4.4 được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC), vị trí sắc độ lân cận (xTbC − 1, yTbC), checkPredModeY được thiết lập bằng sai (FALSE), và cIdx như là các đầu vào, và đầu ra được gán là availL.
- Xử lý thu nhận tính khả dụng khối lân cận được chỉ rõ trong mục 6.4.4 được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC, yTbC), vị trí sắc độ lân cận (xTbC, yTbC − 1), checkPredModeY được thiết lập bằng sai (FALSE), và cIdx như là các đầu vào, và đầu ra được gán là availT.
- Biến availTL được thu nhận như sau:

$$availTL = availL \;\&\&\; availT \quad (347)$$

- Số lượng của các mẫu sắc độ lân cận trên cùng-bên trái khả dụng numTopRight được thu nhận như sau:
  - Biến numTopRight được thiết lập bằng 0 và availTR được thiết lập bằng đúng (TRUE).
  - Khi predModeIntra bằng INTRA_T_CCLM, phần sau đây được áp dụng đối với x = nTbW..2 * nTbW − 1 cho đến khi availTR bằng sai (FALSE) hoặc x bằng 2 * nTbW − 1:
    - Xử lý thu nhận tính khả dụng khối lân cận được chỉ rõ trong mục 6.4.4 được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC) vị trí sắc độ lân cận (xTbC + x, yTbC − 1), checkPredModeY được thiết lập bằng sai (FALSE), và cIdx như là các đầu vào, và đầu ra được gán là availTR
    - Khi availTR bằng đúng (TRUE), numTopRight được gia tăng thêm 1.

- Số lượng các mẫu sắc độ lân cận bên trái-phía dưới khả dụng numLeftBelow được thu nhận như sau:
  - Biến numLeftBelow được thiết lập bằng 0 và availLB được thiết lập bằng đúng (TRUE).
  - Khi predModeIntra bằng INTRA_L_CCLM, phần sau đây được áp dụng đối với y = nTbH..2 * nTbH − 1 cho đến khi availLB bằng sai (FALSE) hoặc y bằng 2 * nTbH − 1:
    - Xử lý thu nhận tính khả dụng khối lân cận được chỉ rõ trong mục 6.4.4 được gọi ra với vị trí sắc độ hiện tại (xCurr, yCurr) được thiết lập bằng (xTbC , yTbC), vị trí sắc độ lân cận (xTbC − 1, yTbC + y), checkPredModeY được thiết lập bằng sai (FALSE), và cIdx như là các đầu vào, và đầu ra được gán là availLB
    - khi availLB bằng đúng (TRUE), numLeftBelow được gia tăng thêm 1.

Số lượng các mẫu sắc độ lân cận khả dụng trên phía trên cùng và trên cùng-bên trái numSampT và số lượng mẫu sắc độ lân cận khả dụng trên phía bên trái và bên trái-phía dưới numSampL được thu nhận như sau:

- Nếu predModeIntra bằng INTRA_LT_CCLM, phần sau đây được áp dụng:

$$numSampT = availT\ ?\ nTbW : 0 \quad (348)$$

$$numSampL = availL\ ?\ nTbH : 0 \quad (349)$$

- Nếu không phải, phần sau đây được áp dụng:

$$numSampT = (availT \&\& predModeIntra == INTRA_T_CCLM)\ ?\ (nTbW + Min(numTopRight, nTbH)) : 0 \quad (350)$$

$$numSampL = (availL \&\& predModeIntra == INTRA_L_CCLM)\ ?\ (nTbH + Min(numLeftBelow, nTbW)) : 0 \quad (351)$$

Biến bCTUboundary được thu nhận như sau:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) = = 0) ?
Đúng (TRUE): Sai (FALSE). (352)

Biến cntN và mảng pickPosN với N được thay thế bởi L và T, được thu nhận như sau:

– Biến numIs4N được thu nhận như sau:

numIs4N = ((availT && availL && predModeIntra = =
INTRA_LT_CCLM) ? 0 : 1) (353)

– Biến startPosN được thiết lập bằng numSampN >> (2 + numIs4N).

– Biến pickStepN được thiết lập bằng Max(1, numSampN >>
(1 + numIs4N)).

– Nếu availN bằng đúng (TRUE) và predModeIntra bằng INTRA_LT_CCLM hoặc INTRA_N_CCLM, phép gán sau đây được thực hiện:

  – cntN được thiết lập bằng Min(numSampN, (1 + numIs4N) << 1).

  – pickPosN[pos] được thiết lập bằng (startPosN + pos * pickStepN), với pos = 0.. cntN − 1.

– Nếu không phải, cntN được thiết lập bằng 0.

Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu cả numSampL và numSampT bằng 0, phần sau đây được áp dụng:

predSamples[x][y] = 1 << (BitDepth − 1)
(354)

– Nếu không phải, các bước có thứ tự sau đây được áp dụng:

1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với x = 0..nTbW * SubWidthC − 1, y= 0..nTbH * SubHeightC − 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).

2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:

– Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = −1..−3, y = 0..SubHeightC * numSampL − 1, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

– Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0..SubWidthC * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

– Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

$$pDstY[x][y] = pY[x][y] \quad (355)$$

– Nếu không phải, phần sau đây được áp dụng:

– Mảng các hệ số bộ lọc một chiều F1 và F2, và các mảng hệ số bộ lọc hai chiều F3 và F4 được chỉ rõ như sau.

$$F1[0] = 2, F1[1] = 0 \quad (356)$$

$$F2[0] = 1, F2[1] = 2, F2[2] = 1 \quad (357)$$

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2 (358)

– Nếu cả SubWidthC và SubHeightC bằng 2, phần sau đây được áp dụng:

F1[0] = 1, F1[1] = 1

(359)

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1 (360)

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1 (361)

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1 (362)

– Nếu không phải, phần sau đây được áp dụng:

F3[1][1] = 8

(363)

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2, (364)

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F3[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

F3[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F3[1][1] * pY[SubWidthC * x][SubHeightC * y] + (365)

F3[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F3[1][2] * pY[SubWidthC * x][SubHeightC * y + 1]
+ 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y]
với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F3[1][0] * pY[0][SubHeightC * y − 1
] +
F3[0][1] * pY[− 1][SubHeightC * y] +
F3[1][1] * pY[0][SubHeightC * y] + (366)
F3[2][1] * pY[1][SubHeightC * y] +

F3[1][2] * pY[0][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải (availL bằng sai (FALSE)), pDsY[0][y] với
y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F2[0] * pY[0][SubHeightC * y − 1] +
F2[1] * pY[0][SubHeightC * y] + (367)

F2[2] * pY[0][SubHeightC * y + 1] + 2) >> 2

– Nếu availT bằng đúng (TRUE), pDsY[x][0]
với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] = (F3[1][0] * pY[SubWidthC * x][− 1] +
F3[0][1] * pY[SubWidthC * x − 1][0] +
F3[1][1] * pY[SubWidthC * x][0] + (368)
F3[2][1] * pY[SubWidthC * x + 1][0] +

F3[1][2] * pY[SubWidthC * x][1] + 4) >> 3

– Nếu không phải (availT bằng sai (FALSE)), pDsY[x][0] với
x = 1..nTbW − 1 được thu nhận:

$$pDsY[x][0] = (F2[0] * pY[SubWidthC * x - 1][0] + F2[1] * pY[SubWidthC * x][0] + F2[2] * pY[SubWidthC * x + 1][0] + 2) >> 2 \quad (369)$$

– Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = (F3[1][0] * pY[0][-1] + F3[0][1] * pY[-1][0] + F3[1][1] * pY[0][0] + F3[2][1] * pY[1][0] + F3[1][2] * pY[0][1] + 4) >> 3 \quad (370)$$

– Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = (F2[0] * pY[-1][0] + F2[1] * pY[0][0] + F2[2] * pY[1][0] + 2) >> 2 \quad (371)$$

– Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = (F2[0] * pY[0][-1] + F2[1] * pY[0][0] + F2[2] * pY[0][1] + 2) >> 2 \quad (372)$$

– Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = pY[0][0] \quad (373)$$

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F4[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F4[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

F4[1][1] * pY[SubWidthC * x][SubHeightC * y] + (374)

F4[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

F4[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F4[2][2] * pY[SubWidthC * x + 1][SubHeightC * y + 1] + 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F4[0][1] * pY[− 1][SubHeightC * y] +

F4[0][2] * pY[− 1][SubHeightC * y + 1] +
F4[1][1] * pY[0][SubHeightC * y] + (375)
F4[1][2] * pY[0][SubHeightC * y + 1] +
F4[2][1] * pY[1][SubHeightC * y] +

F4[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải (availL bằng sai (FALSE)), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F1[0] * pY[0][SubHeightC * y] +

F1[1] * pY[0][SubHeightC * y + 1] + 1) >> 1 (376)

4. Khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[−1][pickPosL[idx]] với idx = 0..cntL − 1, và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY[idx] với idx = 0..cntL − 1 được thu nhận như sau:

– Biến y được thiết lập bằng pickPosL[idx].

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

pSelDsY[idx] = pY[− 1][y]
(377)

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– Nếu y lớn hơn 0 hoặc availTL bằng đúng (TRUE), pSelDsY[idx] được thu nhận như sau:

pSelDsY[idx] = (F3[1][0] *
pY[− SubWidthC][SubHeightC * y − 1] +
F3[0][1] *
pY[−1 − SubWidthC][SubHeightC * y] +

F3[1][1] * pY[−SubWidthC][SubHeightC * y] +
(378)

F3[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F3[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +
4) >> 3

– Nếu không phải (y bằng 0), pSelDsY[idx] được thu nhận như sau:

pSelDsY[idx] = (F2[0] * pY[−1 − SubWidthC][0] +
F2[1] * pY[−SubWidthC][0] + (379)
F2[2] * pY[1 − SubWidthC][0] + 2) >> 2

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1 − SubWidthC][Sub HeightC * y] +

F4[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

F4[1][1] * pY[−SubWidthC][SubHeightC * y] + (380)

F4[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +

F4[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F4[2][2] * pY[1 − SubWidthC][SubHeightC * y + 1] + 4) >> 3

5. Khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx − cntL]][-1] với idx = cntL..cntL + cntT − 1, và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY[idx] với idx = 0..cntL + cntT − 1 được chỉ rõ như sau:

- Biến x được thiết lập bằng pickPosT[idx − cntL].
- Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

$$pSelDsY[idx] = pY[x][-1] \quad (381)$$

- Nếu không phải, phần sau đây được áp dụng:
  - Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:
    - Nếu x lớn hơn 0, phần sau đây được áp dụng:
      - Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

$$\begin{aligned} pSelDsY[idx] = (& F3[1][0] * pY[SubWidthC * x][-1 - SubHeightC] + \\ & F3[0][1] * pY[SubWidthC * x - 1][-SubHeightC] + \\ & F3[1][1] * pY[SubWidthC * x][-SubHeightC] + \\ & F3[2][1] * pY[SubWidthC * x + 1][-SubHeightC] + \\ & F3[1][2] * pY[SubWidthC * x][1 - SubHeightC] + 4) >> 3 \end{aligned} \quad (382)$$

      - Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

$$\begin{aligned} pSelDsY[idx] = (& F2[0] * pY[SubWidthC * x - 1][-1] + \\ & F2[1] * pY[SubWidthC * x][-1] + \\ & F2[2] * pY[SubWidthC * x + 1][-1] + 2) >> 2 \end{aligned} \quad (383)$$

– Nếu không phải (x bằng 0), phần sau đây được áp dụng:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

$$pSelDsY[idx] = (F3[1][0] * pY[-1][-1 - SubHeightC] + F3[0][1] * pY[-1][-SubHeightC] + F3[1][1] * pY[0][-SubHeightC] + F3[2][1] * pY[1][-SubHeightC] + F3[1][2] * pY[-1][1 - SubHeightC] + 4) >> 3 \quad (384)$$

– Nếu không phải, nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:

$$pSelDsY[idx] = (F2[0] * pY[-1][-1] + F2[1] * pY[0][-1] + F2[2] * pY[1][-1] + 2) >> 2 \quad (385)$$

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

$$pSelDsY[idx] = (F2[0] * pY[0][-1] + F2[1] * pY[0][-2] + F2[2] * pY[0][-1] + 2) >> 2 \quad (386)$$

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

$$pSelDsY[idx] = pY[0][-1] \quad (387)$$

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– Nếu x lớn hơn 0, phần sau đây được áp dụng:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[SubWidthC x − 1][−1] +
F4[0][2] * pY[SubWidthC * x − 1][−2] +
F4[1][1] * pY[SubWidthC * x][−1] +(388)
F4[1][2] * pY[SubWidthC * x][−2] +
F4[2][1] * pY[SubWidthC * x + 1][−1] +
F4[2][2] * pY[SubWidthC * x + 1][−2] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] + (389)
F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2

– Nếu không phải (x bằng 0), phần sau đây được áp dụng:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1][−1] + F4[0][2] * pY[−1][−2] +
F4[1][1] * pY[0][−1] + F4[1][2] * pY[0][−2] + (390)

F4[2][1] * pY[1][−1] + F4[2][2] * pY[1][−2] + 4) >> 3

– Nếu không phải, nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[−1][−1] + F2[1] * pY[0][−1] + F2[2] * pY[1][− 1] + 2) >> 2 (391)

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F1[1] * pY[0][−2] + F1[0] * pY[0][−1] + 1) >> 1 (392)

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = pY[0][−1] (393)

6. Khi cntT+ cntL không bằng 0, các biến minY, maxY, minC và maxC được thu nhận như sau:

– Khi cntT + cntL bằng 2, pSelComp[3] được thiết lập bằng pSelComp[0], pSelComp[2] được thiết lập bằng pSelComp[1], pSelComp[0] được thiết lập bằng pSelComp[1], và pSelComp[1] được thiết lập bằng pSelComp[3], với Comp được thay thế bởi DsY và C.

– Các mảng minGrpIdx và maxGrpIdx được thu nhận như sau:

minGrpIdx[0] = 0 (394)

minGrpIdx[1] = 2 (395)

maxGrpIdx[0] = 1 (396)

maxGrpIdx[1] = 3 (397)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[minGrpIdx[1]], minGrpIdx[0] và minGrpIdx[1] được hoán đổi như sau:

(minGrpIdx[0], minGrpIdx[1]) = Swap(minGrpIdx[0], minGrpIdx[1]) (398)

– Khi pSelDsY[maxGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], maxGrpIdx[0] và maxGrpIdx[1] được hoán đổi như sau:

(maxGrpIdx[0], maxGrpIdx[1]) = Swap(maxGrpIdx[0], maxGrpIdx[1]) (399)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], các mảng minGrpIdx và maxGrpIdx được hoán đổi như sau:

(minGrpIdx, maxGrpIdx) = Swap(minGrpIdx, maxGrpIdx) (400)

– Khi pSelDsY[minGrpIdx[1]] lớn hơn pSelDsY[maxGrpIdx[0]], minGrpIdx[1] và maxGrpIdx[0] được hoán đổi như sau:

(minGrpIdx[1], maxGrpIdx[0]) = Swap(minGrpIdx[1], maxGrpIdx[0]) (401)

– Các biến maxY, maxC, minY và minC được thu nhận như sau:

maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1 (402)

maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1 (403)

minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1 (404)

minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1 (405)

7. Các biến a, b, và k được thu nhận như sau:

– Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

k = 0 (406)

a = 0 (407)

b = 1 << (BitDepth − 1) (408)

– Nếu không phải, phần sau đây được áp dụng:

diff = maxY − minY (409)

– Nếu diff không bằng 0, phần sau đây được áp dụng:

diffC = maxC − minC (410)

x = Floor(Log2(diff)) (411)

normDiff = ((diff << 4) >> x) & 15 (412)

x += (normDiff != 0) ? 1 : 0 (413)

y = Floor(Log2(Abs (diffC))) + 1 (414)

$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y$ (415)

k = ((3 + x − y) < 1) ? 1 : 3 + x − y (416)

a = ((3 + x − y) < 1) ? Sign(a) * 15 : a (417)

b = minC − ((a * minY) >> k) (418)

Trong đó divSigTable[ ] được chỉ rõ như sau:

divSigTable[ ] = { 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 } (419)

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

k = 0 (420)

$$a = 0 \quad (421)$$

$$b = \text{minC} \quad (422)$$

8. Các mẫu dự đoán predSamples[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

$$\text{predSamples}[x][y] = \text{Clip1}(((\text{pDsY}[x][y] * a) >> k) + b) \quad (423)$$

Lưu ý – Xử lý này sử dụng sps_chroma_vertical_collocated_flag. Tuy nhiên, để đơn giản hóa cách thức thực hiện, xử lý này không sử dụng sps_chroma_horizontal_collocated_flag.

Theo phương án khác, hoạt động tạo đệm có thể được sử dụng để thực hiện việc lựa chọn lọc trên biên của khối độ chói được khôi phục. Khi bộ lọc được áp dụng tới vùng của các mẫu được khôi phục, xử lý lọc cần được điều chỉnh khi vị trí của vùng được lọc nằm trên biên khối.

Fig.13 thể hiện hoạt động lọc đối với khối độ chói được khôi phục 1301 bởi bộ lọc ba nhánh ví dụ 1302. Khi lọc các mẫu biên, một trong số các hệ số bộ lọc được sắp xếp cùng vị trí với mẫu được khôi phục mà nằm phía ngoài của các trường hợp các mẫu độ chói được khôi phục có thể không khả dụng đối với phía bên trái 1303, và do đó, vị trí phía ngoài có thể không được sử dụng trong các hoạt động lọc.

Trong các phương án trước đó, đối với cột bên trái và hàng trên cùng được đề xuất để làm giảm số lượng nhánh đối với các bộ lọc được áp dụng tới các mẫu độ chói được khôi phục. Thực tế, các hệ số lọc được xác định là phụ thuộc vị trí.

Theo phương án khác, sự phụ thuộc vị trí này được điều khiển bằng cách tạo đệm bên trái và trên cùng đối với các mẫu được khôi phục biên. Khi phía trái không khả dụng, các mẫu của phía không khả dụng được thu nhận bằng cách sao chép theo chiều ngang các mẫu được khôi phục ngoài cùng bên trái của khối được khôi phục 1301. Hoạt động này thực tế là sự sao chép của cột thứ nhất của các mẫu được khôi phục sang bên trái. Khi phía trên cùng không khả dụng, các

mẫu của phía không khả dụng được thu nhận bằng cách sao chép theo chiều dọc các mẫu được khôi phục ngoài cùng trên cùng của khối được khôi phục 1301. Hoạt động này thực tế là sự sao chép của hàng thứ nhất của các mẫu được khôi phục phía trên khối 1301.

Có thể được nhận thấy rằng việc lọc tuyến tính của vùng của các mẫu được khôi phục mà bao gồm các mẫu được tạo đệm có thể được thực hiện mà không sử dụng hoạt động đệm. Thực tế, đối với vị trí của vùng này, bộ lọc nhánh dài hơn có thể được thay thế bởi bộ lọc tương đương có bậc nhỏ hơn. Sự có mặt của các mẫu được tạo đệm trong vùng được lọc làm cho có thể đơn giản hóa hoạt động lọc bằng cách nhóm các hệ số.

Phương án này có thể được biểu diễn như là phần sau đây của tiêu chuẩn kỹ thuật VVC:

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:
   - Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:
     - pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

$$pDstY[x][y] = pY[x][y] \quad (355)$$

   - Nếu không phải, phần sau đây được áp dụng:
     - Mảng các hệ số bộ lọc một chiều F1 và F2, và các mảng hệ số bộ lọc hai chiều F3 và F4 được chỉ rõ như sau.

$$F1[0] = 4, F1[1] = 0 \quad (356)$$

$$F2[0] = 1, F2[1] = 2, F2[2] = 1 \quad (357)$$

$$F3[i][j] = F4[i][j] = 0, \text{ với } i = 0..2, j = 0..2 \quad (358)$$

– Nếu cả SubWidthC và SubHeightC bằng 2, phần sau đây được áp dụng:

F1[0] = 3, F1[1] = 1 (359)

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1 (360)

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1 (361)

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1 (362)

– Nếu không phải, phần sau đây được áp dụng:

F3[1][1] = 8 (363)

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2, (364)

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F3[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

F3[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F3[1][1] * pY[SubWidthC * x][SubHeightC * y] + (365)

F3[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F3[1][2] * pY[SubWidthC * x][SubHeightC * y + 1]
+ 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y]
với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F3[1][0] * pY[0][SubHeightC * y − 1
] +
F3[0][1] * pY[− 1][SubHeightC * y] +
F3[1][1] * pY[0][SubHeightC * y] + (366)
F3[2][1] * pY[1][SubHeightC * y] +

F3[1][2] * pY[0][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải (availL bằng sai (FALSE)), pDsY[0][y] với
y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F2[0] * pY[0][SubHeightC * y − 1] +
F2[1] * pY[0][SubHeightC * y] + (367)

F2[2] * pY[0][SubHeightC * y + 1] + 2) >> 2

– Nếu availT bằng đúng (TRUE), pDsY[x][0]
với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] = (F3[1][0] * pY[SubWidthC * x][− 1] +
F3[0][1] * pY[SubWidthC * x − 1][0] +
F3[1][1] * pY[SubWidthC * x][0] + (368)
F3[2][1] * pY[SubWidthC * x + 1][0] +

F3[1][2] * pY[SubWidthC * x][1] + 4) >> 3

– Nếu không phải (availT bằng sai (FALSE)), pDsY[x][0] với
x = 1..nTbW − 1 được thu nhận:

$$pDsY[x][0] = (F2[0] * pY[SubWidthC * x - 1][0] + F2[1] * pY[SubWidthC * x][0] + F2[2] * pY[SubWidthC * x + 1][0] + 2) >> 2 \quad (369)$$

– Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = (F3[1][0] * pY[0][-1] + F3[0][1] * pY[-1][0] + F3[1][1] * pY[0][0] + F3[2][1] * pY[1][0] + F3[1][2] * pY[0][1] + 4) >> 3 \quad (370)$$

– Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = (F2[0] * pY[-1][0] + F2[1] * pY[0][0] + F2[2] * pY[1][0] + 2) >> 2 \quad (371)$$

– Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = (F2[0] * pY[0][-1] + F2[1] * pY[0][0] + F2[2] * pY[0][1] + 2) >> 2 \quad (372)$$

– Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:

$$pDsY[0][0] = pY[0][0] \quad (373)$$

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F4[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F4[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

F4[1][1] * pY[SubWidthC * x][SubHeightC * y] + (374)

F4[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

F4[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F4[2][2] * pY[SubWidthC * x + 1][SubHeightC * y + 1] + 4) >> 3

– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F4[0][1] * pY[− 1][SubHeightC * y] +

F4[0][2] * pY[− 1][SubHeightC * y + 1] +
F4[1][1] * pY[0][SubHeightC * y] + (375)
F4[1][2] * pY[0][SubHeightC * y + 1] +
F4[2][1] * pY[1][SubHeightC * y] +

F4[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải (availL bằng sai (FALSE)), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F1[0] * pY[0][SubHeightC * y] +

F1[1] * pY[0][SubHeightC * y + 1] + 2) >> 2 (376)

4. Khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[−1][pickPosL[idx]] với idx = 0..cntL − 1, và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY[idx] với idx = 0..cntL − 1 được thu nhận như sau:

– Biến y được thiết lập bằng pickPosL[idx].

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

pSelDsY[idx] = pY[− 1][y]
(377)

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– Nếu y lớn hơn 0 hoặc availTL bằng đúng (TRUE), pSelDsY[idx] được thu nhận như sau:

pSelDsY[idx] = (F3[1][0] *
pY[− SubWidthC][SubHeightC * y − 1] +
F3[0][1] *
pY[−1 − SubWidthC][SubHeightC * y] +

F3[1][1] * pY[−SubWidthC][SubHeightC * y] +
(378)

F3[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F3[1][2] * pY[−SubWidthC][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải (y bằng 0), pSelDsY[idx] được thu nhận như sau:

pSelDsY[idx] = (F2[0] * pY[−1 − SubWidthC][0] + F2[1] * pY[−SubWidthC][0] + F2[2] * pY[1 − SubWidthC][0] + 2) >> 2 (379)

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1 − SubWidthC][SubHeightC * y] + F4[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] + F4[1][1] * pY[−SubWidthC][SubHeightC * y] + F4[1][2] * pY[−SubWidthC][SubHeightC * y + 1] + F4[2][1] * pY[1 − SubWidthC][SubHeightC * y] + F4[2][2] * pY[1 − SubWidthC][SubHeightC * y + 1] + 4) >> 3 (380)

5. Khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx − cntL]][-1] với idx = cntL..cntL + cntT − 1, và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY[idx] với idx = 0..cntL + cntT − 1 được chỉ rõ như sau:

– Biến x được thiết lập bằng pickPosT[idx − cntL].

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

$$pSelDsY[idx] = pY[x][-1] \quad (381)$$

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– Nếu x lớn hơn 0, phần sau đây được áp dụng:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

$$\begin{aligned} pSelDsY[idx] = (& F3[1][0] * pY[SubWidthC * x][-1 - SubHeightC] + \\ & F3[0][1] * pY[SubWidthC * x - 1][-SubHeightC] + \\ & F3[1][1] * pY[SubWidthC * x][-SubHeightC] + \\ & F3[2][1] * pY[SubWidthC * x + 1][-SubHeightC] + \\ & F3[1][2] * pY[SubWidthC * x][1 - SubHeightC] + 4) >> 3 \end{aligned} \quad (382)$$

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

$$\begin{aligned} pSelDsY[idx] = (& F2[0] * pY[SubWidthC * x - 1][-1] + \\ & F2[1] * pY[SubWidthC * x][-1] + \\ & F2[2] * pY[SubWidthC * x + 1][-1] + 2) >> 2 \end{aligned} \quad (383)$$

– Nếu không phải (x bằng 0), phần sau đây được áp dụng:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F3[1][0] * pY[−1][− 1 − SubHeightC] +
F3[0][1] * pY[−1][−SubHeightC] +
F3[1][1] * pY[0][−SubHeightC] + (384)
F3[2][1] * pY[1][−SubHeightC] +
F3[1][2] * pY[−1][1 − SubHeightC] + 4) >> 3

– Nếu không phải, nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[−1][−1] + F2[1] * pY[0][−1] +
F2[2] * pY[1][−1] + 2) >> 2 (385)

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[0][−1] + F2[1] * pY[0][−2] +
F2[2] * pY[0][−1] + 2) >> 2 (386)

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = pY[0][−1]
(387)

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– Nếu x lớn hơn 0, phần sau đây được áp dụng:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[SubWidthC x − 1][−1] +
F4[0][2] * pY[SubWidthC * x − 1][−2] +
F4[1][1] * pY[SubWidthC * x][−1] +
F4[1][2] * pY[SubWidthC * x][−2] +
F4[2][1] * pY[SubWidthC * x + 1][−1] +
F4[2][2] * pY[SubWidthC * x + 1][−2] + 4) >> 3 (388)

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] +
F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2 (389)

– Nếu không phải (x bằng 0), phần sau đây được áp dụng:

– Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1][−1] + F4[0][2] * pY[−1][−2] +
F4[1][1] * pY[0][−1] + F4[1][2] * pY[0][−2] +
F4[2][1] * pY[1][−1] + F4[2][2] * pY[1][−2] + 4) >> 3 (390)

– Nếu không phải, nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[−1][−1] + F2[1] * pY[0][−1] +
F2[2] * pY[1][− 1] + 2) >> 2 (391)

– Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F1[1] * pY[0][−2] + F1[0] * pY[0][−1] + 2) >> 2 (392)

– Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = pY[0][−1]
(393)

6. Khi cntT+ cntL không bằng 0, các biến minY, maxY, minC và maxC được thu nhận như sau:

– Khi cntT + cntL bằng 2, pSelComp[3] được thiết lập bằng pSelComp[0], pSelComp[2] được thiết lập bằng pSelComp[1], pSelComp[0] được thiết lập bằng pSelComp[1], và pSelComp[1] được thiết lập bằng pSelComp[3], với Comp được thay thế bởi DsY và C.

– Các mảng minGrpIdx và maxGrpIdx được thu nhận như sau:

minGrpIdx[0] = 0 (394)

minGrpIdx[1] = 2 (395)

maxGrpIdx[0] = 1 (396)

maxGrpIdx[1] = 3 (397)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[minGrpIdx[1]], minGrpIdx[0] và minGrpIdx[1] được hoán đổi như sau:

(minGrpIdx[0], minGrpIdx[1]) = Swap(minGrpIdx[0], minGrpIdx[1]) (398)

– Khi pSelDsY[maxGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], maxGrpIdx[0] và maxGrpIdx[1] được hoán đổi như sau:

(maxGrpIdx[0], maxGrpIdx[1]) = Swap(maxGrpIdx[0], maxGrpIdx[1]) (399)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], các mảng minGrpIdx và maxGrpIdx được hoán đổi như sau:

(minGrpIdx, maxGrpIdx) = Swap(minGrpIdx, maxGrpIdx) (400)

– Khi pSelDsY[minGrpIdx[1]] lớn hơn pSelDsY[maxGrpIdx[0]], minGrpIdx[1] và maxGrpIdx[0] được hoán đổi như sau:

(minGrpIdx[1], maxGrpIdx[0]) = Swap(minGrpIdx[1], maxGrpIdx[0]) (401)

– Các biến maxY, maxC, minY và minC được thu nhận như sau:

maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1 (402)

maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1 (403)

minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1 (404)

minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1 (405)

7. Các biến a, b, và k được thu nhận như sau:

– Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

$$k = 0 \quad (406)$$

$$a = 0 \quad (407)$$

$$b = 1 << (BitDepth - 1) \quad (408)$$

– Nếu không phải, phần sau đây được áp dụng:

$$diff = maxY - minY \quad (409)$$

– Nếu diff không bằng 0, phần sau đây được áp dụng:

$$diffC = maxC - minC \quad (410)$$

$$x = Floor(Log2(diff)) \quad (411)$$

$$normDiff = ((diff << 4) >> x) \& 15 \quad (412)$$

$$x += (normDiff != 0) ? 1 : 0 \quad (413)$$

$$y = Floor(Log2(Abs (diffC))) + 1 \quad (414)$$

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y \quad (415)$$

$$k = ((3 + x - y) < 1) ? 1 : 3 + x - y \quad (416)$$

$$a = ((3 + x - y) < 1) ? Sign(a) * 15 : a \quad (417)$$

$$b = minC - ((a * minY) >> k) \quad (418)$$

Trong đó divSigTable[ ] được chỉ rõ như sau:

$$divSigTable[ ] = \{ 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 \} \quad (419)$$

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

$$k = 0 \quad (420)$$

$$a = 0 \quad (421)$$

$$b = minC \quad (422)$$

8. Các mẫu dự đoán predSamples[x][y] với
   x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

$$predSamples[x][y] = Clip1(((pDsY[x][y] * a) >> k) + b) \quad (423)$$

   Lưu ý – Xử lý này sử dụng sps_chroma_vertical_collocated_flag. Tuy nhiên, để đơn giản hóa cách thức thực hiện, xử lý này không sử dụng sps_chroma_horizontal_collocated_flag.

Biến thể khác của phương án bao gồm hoạt động tạo đệm. Trong trường hợp này, bộ lọc "F1" không được định nghĩa rõ ràng, nhưng thay vì đó được thực hiện như là kết hợp của việc tạo đệm và lọc như sử dụng bộ lọc "F2". Mô tả kỹ thuật đối với biến thể này được đưa ra dưới đây:

1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với
   x = 0..nTbW * SubWidthC − 1, y= 0..nTbH * SubHeightC − 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).

2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:

   – Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = −1..−3, y = 0..SubHeightC * numSampL − 1, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

   – Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0..SubWidthC * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

   – Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

– Khi availT bằng sai (FALSE), các mẫu độ chói trên cùng lân cận pY[x][y] với x = -1..SubWidthC * numSampT − 1, y = −1, -2, được thiết lập bằng các mẫu độ chói pY[x][0].

– Khi availL bằng sai (FALSE), các mẫu độ chói bên trái lân cận pY[x][y] với x = −1, -2, -3, y = -1..SubHeightC * numSampL − 1, được thiết lập bằng các mẫu độ chói pY[0][y].

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

$$pDstY[x][y] = pY[x][y] \quad (355)$$

– Nếu không phải, phần sau đây được áp dụng:

– Mảng các hệ số bộ lọc một chiều F1 và F2, và các mảng hệ số bộ lọc hai chiều F3 và F4 được chỉ rõ như sau.

~~F1[0] = 2, F1[1] = 0~~ ~~(356)~~

$$F2[0] = 1, F2[1] = 2, F2[2] = 1 \quad (357)$$

$$F3[i][j] = F4[i][j] = 0, \text{ với } i = 0..2, j = 0..2 \quad (358)$$

– Nếu cả SubWidthC và SubHeightC bằng 2, phần sau đây được áp dụng:

~~F1[0] = 1, F1[1] = 1~~ ~~(359)~~

$$F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1 \quad (360)$$

$$F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1 \quad (361)$$

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1 (362)

– Nếu không phải, phần sau đây được áp dụng:

F3[1][1] = 8

(363)

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2, (364)

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F3[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

F3[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F3[1][1] * pY[SubWidthC * x][SubHeightC * y] +

(365)

F3[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F3[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] + 4) >> 3

~~– Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau:~~

~~pDsY[0][y] = (F3[1][0] * pY[0][SubHeightC * y − 1] +~~

~~F3[0][1] * pY[− 1][SubHeightC * y] +~~

~~F3[1][1] * pY[0][SubHeightC * y] + (366)~~

F3[2][1] * pY[1][SubHeightC * y] +

F3[1][2] * pY[0][SubHeightC * y + 1] + 4) >> 3

Nếu không phải (availL bằng sai (FALSE)), pDsY[0][y] với y = 1..nTbH − 1 được thu nhận như sau:

pDsY[0][y] = (F2[0] * pY[0][SubHeightC * y − 1] +
F2[1] * pY[0][SubHeightC * y] + (367)
F2[2] * pY[0][SubHeightC * y + 1] + 2) >> 2

Nếu availT bằng đúng (TRUE), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận như sau:

pDsY[x][0] = (F3[1][0] * pY[SubWidthC * x][− 1] +
F3[0][1] * pY[SubWidthC * x − 1][0] +
F3[1][1] * pY[SubWidthC * x][0] + (368)
F3[2][1] * pY[SubWidthC * x + 1][0] +
F3[1][2] * pY[SubWidthC * x][1] + 4) >> 3

Nếu không phải (availT bằng sai (FALSE)), pDsY[x][0] với x = 1..nTbW − 1 được thu nhận:

pDsY[x][0] = (F2[0] * pY[SubWidthC * x − 1][0] +
F2[1] * pY[SubWidthC * x][0] + (369)
F2[2] * pY[SubWidthC * x + 1][0] + 2) >> 2

Nếu availL bằng đúng (TRUE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:

pDsY[0][0] = (F3[1][0] * pY[0][− 1] +
F3[0][1] * pY[− 1][0] +
F3[1][1] * pY[0][0] + (370)

~~F3[2][1] * pY[1][0] +~~

~~F3[1][2] * pY[0][1] + 4) >> 3~~

~~– Nếu không phải, nếu availL bằng đúng (TRUE) và availT bằng sai (FALSE), pDsY[0][0] được thu nhận như sau:~~

~~pDsY[0][0] = (F2[0] * pY[− 1][0] + F2[1] * pY[0][0] +~~

~~F2[2] * pY[1][0] + 2) >> 2 (371)~~

~~– Nếu không phải, nếu availL bằng sai (FALSE) và availT bằng đúng (TRUE), pDsY[0][0] được thu nhận như sau:~~

~~pDsY[0][0] = (F2[0] * pY[0][− 1] + F2[1] * pY[0][0] +~~

~~F2[2] * pY[0][1] + 2) >> 2 (372)~~

~~– Nếu không phải (availL bằng sai (FALSE) và availT bằng sai (FALSE)), pDsY[0][0] được thu nhận như sau:~~

~~pDsY[0][0] = pY[0][0]~~

~~(373)~~

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F4[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F4[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

F4[1][1] * pY[SubWidthC * x][SubHeightC * y] + (374)

F4[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

F4[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F4[2][2] * pY[SubWidthC * x + 1][SubHeightC * y + 1] + 4) >> 3

~~— Nếu availL bằng đúng (TRUE), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:~~

~~pDsY[0][y] = (F4[0][1] * pY[−1][SubHeightC * y] +~~

~~F4[0][2] * pY[−1][SubHeightC * y + 1] +~~

~~F4[1][1] * pY[0][SubHeightC * y] +~~

~~F4[1][2] * pY[0][SubHeightC * y + 1] +~~

~~F4[2][1] * pY[1][SubHeightC * y] +~~

~~F4[2][2] * pY[1][SubHeightC * y + 1] + 4) >> 3~~

~~— Nếu không phải (availL bằng sai (FALSE)), pDsY[0][y] với y = 0..nTbH − 1 được thu nhận như sau:~~

~~pDsY[0][y] = (F1[0] * pY[0][SubHeightC * y] +~~

~~F1[1] * pY[0][SubHeightC * y + 1] + 1) >> 1 (376)~~

4. Khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[−1][pickPosL[idx]] với idx = 0..cntL − 1, và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY[idx] với idx = 0..cntL − 1 được thu nhận như sau:

   – Biến y được thiết lập bằng pickPosL[idx].

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

pSelDsY[idx] = pY[− 1][y]

(377)

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

~~– Nếu y lớn hơn 0 hoặc availTL bằng đúng (TRUE), pSelDsY[idx] được thu nhận như sau:~~

pSelDsY[idx] = (F3[1][0] *
pY[− SubWidthC][SubHeightC * y − 1] +
F3[0][1] *
pY[−1 − SubWidthC][SubHeightC * y] +

F3[1][1] * pY[−SubWidthC][SubHeightC * y] +

(378)

F3[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F3[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +
4) >> 3

~~– Nếu không phải (y bằng 0), pSelDsY[idx] được thu nhận như sau:~~

~~pSelDsY[idx] = (F2[0] * pY[−1 − SubWidthC][0] +~~
~~F2[1] * pY[−SubWidthC][0] +~~ (379)
~~F2[2] * pY[1 − SubWidthC][0] + 2) >> 2~~

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1 − SubWidthC][Sub HeightC * y] +

F4[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

F4[1][1] * pY[−SubWidthC][SubHeightC * y] +(380)

F4[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +

F4[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F4[2][2] * pY[1 − SubWidthC][SubHeightC * y + 1] + 4) >> 3

5. Khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx − cntL]][-1] với idx = cntL..cntL + cntT − 1, và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY[idx] với idx = 0..cntL + cntT − 1 được chỉ rõ như sau:
   - Biến x được thiết lập bằng pickPosT[idx − cntL].
   - Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

     pSelDsY[idx] = pY[x][− 1] (381)

   - Nếu không phải, phần sau đây được áp dụng:
     - Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

       ~~Nếu x lớn hơn 0, phần sau đây được áp dụng:~~

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F3[1][0] * pY[SubWidthC * x][− 1 − SubHeightC] +

F3[0][1] * pY[SubWidthC * x − 1][−SubHeightC] +

F3[1][1] * pY[SubWidthC * x][−SubHeightC] + (382)

F3[2][1] * pY[SubWidthC * x + 1][−SubHeightC] +

F3[1][2] * pY[SubWidthC * x][1 − SubHeightC] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] + (383)

F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2

~~Nếu không phải (x bằng 0), phần sau đây được áp dụng:~~

~~Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = (F3[1][0] * pY[−1][− 1 − SubHeightC] +~~

~~F3[0][1] * pY[−1][−SubHeightC] +~~

~~F3[1][1] * pY[0][−SubHeightC] + (384)~~

~~F3[2][1] * pY[1][−SubHeightC] +~~

~~F3[1][2] * pY[−1][1 − SubHeightC] + 4) >> 3~~

~~— Nếu không phải, nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = (F2[0] * pY[−1][−1] + F2[1] * pY[0][−1] +~~
~~F2[2] * pY[1][−1] + 2) >> 2 (385)~~

~~— Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = (F2[0] * pY[0][−1] + F2[1] * pY[0][−2] +~~
~~F2[2] * pY[0][−1] + 2) >> 2 (386)~~

~~— Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = pY[0][−1]~~
~~(387)~~

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

~~— Nếu x lớn hơn 0, phần sau đây được áp dụng:~~

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[SubWidthC x − 1][−1] +
F4[0][2] * pY[SubWidthC * x − 1][−2] +
F4[1][1] * pY[SubWidthC * x][−1] +(388)
F4[1][2] * pY[SubWidthC * x][−2] +
F4[2][1] * pY[SubWidthC * x + 1][−1] +

F4[2][2] * pY[SubWidthC * x + 1][−2] + 4)

>> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] + (389)

F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2

~~Nếu không phải (x bằng 0), phần sau đây được áp dụng:~~

~~Nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = (F4[0][1] * pY[−1][−1] + F4[0][2] * pY[−1][−2] +~~

~~F4[1][1] * pY[0][−1] + F4[1][2] * pY[0][−2] + (390)~~

~~F4[2][1] * pY[1][−1] + F4[2][2] * pY[1][−2] + 4) >> 3~~

~~Nếu không phải, nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = (F2[0] * pY[−1][−1] + F2[1] * pY[0][−1] +~~

~~F2[2] * pY[1][−1] + 2) >> 2 (391)~~

~~Nếu không phải, nếu availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = (F1[1] * pY[0][−2] + F1[0] * pY[0][−1] + 1) >> 1 (392)~~

~~Nếu không phải (availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:~~

~~pSelDsY[idx] = pY[0][−1] (393)~~

6. Khi cntT+ cntL không bằng 0, các biến minY, maxY, minC và maxC được thu nhận như sau:

   – Khi cntT + cntL bằng 2, pSelComp[3] được thiết lập bằng pSelComp[0], pSelComp[2] được thiết lập bằng pSelComp[1], pSelComp[0] được thiết lập bằng pSelComp[1], và pSelComp[1] được thiết lập bằng pSelComp[3], với Comp được thay thế bởi DsY và C.

   – Các mảng minGrpIdx và maxGrpIdx được thu nhận như sau:

   minGrpIdx[0] = 0 (394)

   minGrpIdx[1] = 2 (395)

   maxGrpIdx[0] = 1 (396)

   maxGrpIdx[1] = 3 (397)

   – Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[minGrpIdx[1]], minGrpIdx[0] và minGrpIdx[1] được hoán đổi như sau:

   (minGrpIdx[0], minGrpIdx[1]) = Swap(minGrpIdx[0], minGrpIdx[1]) (398)

   – Khi pSelDsY[maxGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], maxGrpIdx[0] và maxGrpIdx[1] được hoán đổi như sau:

   (maxGrpIdx[0], maxGrpIdx[1]) = Swap(maxGrpIdx[0], maxGrpIdx[1]) (399)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], các mảng minGrpIdx và maxGrpIdx được hoán đổi như sau:

(minGrpIdx, maxGrpIdx) = Swap(minGrpIdx, maxGrpIdx) (400)

– Khi pSelDsY[minGrpIdx[1]] lớn hơn pSelDsY[maxGrpIdx[0]], minGrpIdx[1] và maxGrpIdx[0] được hoán đổi như sau:

(minGrpIdx[1], maxGrpIdx[0]) = Swap(minGrpIdx[1], maxGrpIdx[0]) (401)

– Các biến maxY, maxC, minY và minC được thu nhận như sau:

maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1 (402)

maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1 (403)

minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1 (404)

minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1 (405)

7. Các biến a, b, và k được thu nhận như sau:

– Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

k = 0 (406)

a = 0 (407)

b = 1 << (BitDepth − 1) (408)

– Nếu không phải, phần sau đây được áp dụng:

diff = maxY − minY (409)

– Nếu diff không bằng 0, phần sau đây được áp dụng:

diffC = maxC − minC (410)

$$x = Floor(Log2(diff)) \quad (411)$$

$$normDiff = ((diff << 4) >> x) \& 15 \quad (412)$$

$$x += (normDiff != 0) ? 1 : 0 \quad (413)$$

$$y = Floor(Log2(Abs (diffC))) + 1 \quad (414)$$

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y \quad (415)$$

$$k = ((3 + x - y) < 1) ? 1 : 3 + x - y \quad (416)$$

$$a = ((3 + x - y) < 1) ? Sign(a) * 15 : a \quad (417)$$

$$b = minC - ((a * minY) >> k) \quad (418)$$

Trong đó divSigTable[ ] được chỉ rõ như sau:

$$divSigTable[ ] = \{ 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 \} \quad (419)$$

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

$$k = 0 \quad (420)$$

$$a = 0 \quad (421)$$

$$b = minC \quad (422)$$

8. Các mẫu dự đoán predSamples[x][y] với
   x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

$$predSamples[x][y] = Clip1(((pDsY[x][y] * a) >> k) + b) \quad (423)$$

Lưu ý – Xử lý này sử dụng sps_chroma_vertical_collocated_flag. Tuy nhiên, để đơn giản hóa cách thức thực hiện, xử lý này không sử dụng sps_chroma_horizontal_collocated_flag.

Không có các đánh dấu thay đổi, mô tả kỹ thuật được thể hiện nêu trên có thể được biểu diễn như sau:

1. Các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] với x = 0..nTbW * SubWidthC − 1, y= 0..nTbH * SubHeightC − 1 được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y).

2. Các mẫu độ chói lân cận pY[x][y] được thu nhận như sau:

   – Khi numSampL lớn hơn 0, các mẫu độ chói bên trái lân cận pY[x][y] với x = −1..−3, y = 0..SubHeightC * numSampL − 1, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x , yTbY +y).

   – Khi numSampT lớn hơn 0, các mẫu độ chói trên cùng lân cận pY[x][y] với x = 0..SubWidthC * numSampT − 1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

   – Khi availTL bằng đúng (TRUE), các mẫu độ chói trên cùng-bên trái lân cận pY[x][y] với x = −1, y = −1, −2, được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY+ x, yTbY + y).

   – Khi availT bằng sai (FALSE), các mẫu độ chói trên cùng lân cận pY[x][y] với x = -1..SubWidthC * numSampT − 1, y = −1, -2, được thiết lập bằng các mẫu độ chói pY[x][0].

   – Khi availL bằng sai (FALSE), các mẫu độ chói bên trái lân cận pY[x][y] với x = −1, -2, -3, y = -1..SubHeightC * numSampL − 1, được thiết lập bằng các mẫu độ chói pY[0][y].

3. Các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

   – Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 1..nTbW − 1, y = 1..nTbH − 1 được thu nhận như sau:

pDstY[x][y] = pY[x][y] (355)

– Nếu không phải, phần sau đây được áp dụng:

– Mảng các hệ số bộ lọc một chiều F1 và F2, và các mảng hệ số bộ lọc hai chiều F3 và F4 được chỉ rõ như sau.

F2[0] = 1, F2[1] = 2, F2[2] = 1 (357)

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2 (358)

– Nếu cả SubWidthC và SubHeightC bằng 2, phần sau đây được áp dụng:

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1 (360)

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1 (361)

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1 (362)

– Nếu không phải, phần sau đây được áp dụng:

F3[1][1] = 8 (363)

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2, (364)

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

– pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F3[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

F3[0][1] * pY[SubWidthC * x − 1][SubHeightC * y]
+

F3[1][1] * pY[SubWidthC * x][SubHeightC * y] +
(365)
F3[2][1] *
pY[SubWidthC * x + 1][SubHeightC * y] +

F3[1][2] * pY[SubWidthC * x][SubHeightC * y + 1]
+ 4) >> 3

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

pDsY[x][y] = (F4[0][1] * pY[SubWidthC * x − 1][Su
bHeightC * y] +

F4[0][2] * pY[SubWidthC * x − 1][SubHeightC * y +
1] +

F4[1][1] * pY[SubWidthC * x][SubHeightC * y] +
(374)

F4[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

F4[2][1] * pY[SubWidthC * x + 1][SubHeightC * y]
+

F4[2][2] * pY[SubWidthC * x + 1][SubHeightC * y + 1] + 4) >> 3

4. Khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[−1][pickPosL[idx]] với idx = 0..cntL − 1, và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY[idx] với idx = 0..cntL − 1 được thu nhận như sau:

   – Biến y được thiết lập bằng pickPosL[idx].

   – Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

   pSelDsY[idx] = pY[− 1][y]

   (377)

   – Nếu không phải, phần sau đây được áp dụng:

   – Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

   pSelDsY[idx] = (F3[1][0] *
   pY[− SubWidthC][SubHeightC * y − 1] +
   F3[0][1] *
   pY[−1 − SubWidthC][SubHeightC * y] +

   F3[1][1] * pY[−SubWidthC][SubHeightC * y] +

   (378)

   F3[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

   F3[1][2] * pY[−SubWidthC][SubHeightC * y + 1] + 4) >> 3

   – Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1 − SubWidthC][Sub HeightC * y] +

F4[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

F4[1][1] * pY[−SubWidthC][SubHeightC * y] + (380)

F4[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +

F4[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F4[2][2] * pY[1 − SubWidthC][SubHeightC * y + 1] + 4) >> 3

5. Khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx − cntL]][-1] với idx = cntL..cntL + cntT − 1, và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY[idx] với idx = 0..cntL + cntT − 1 được chỉ rõ như sau:
   - Biến x được thiết lập bằng pickPosT[idx − cntL].
   - Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

     pSelDsY[idx] = pY[x][− 1] (381)

   - Nếu không phải, phần sau đây được áp dụng:
     - Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:
       - Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F3[1][0] * pY[SubWidthC * x][− 1 − SubHeightC] +

F3[0][1] * pY[SubWidthC * x − 1][−SubHeightC] +

F3[1][1] * pY[SubWidthC * x][−SubHeightC] + (382)

F3[2][1] * pY[SubWidthC * x + 1][−SubHeightC] +

F3[1][2] * pY[SubWidthC * x][1 − SubHeightC] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] + (383)

F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[SubWidthC x − 1][−1] +

F4[0][2] * pY[SubWidthC * x − 1][−2] +
F4[1][1] * pY[SubWidthC * x][−1] +(388)
F4[1][2] * pY[SubWidthC * x][−2] +
F4[2][1] * pY[SubWidthC * x + 1][−1] +

F4[2][2] * pY[SubWidthC * x + 1][−2] + 4)

>> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] + F2[1] * pY[SubWidthC * x][−1] + (389)

F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2

6. Khi cntT+ cntL không bằng 0, các biến minY, maxY, minC và maxC được thu nhận như sau:

– Khi cntT + cntL bằng 2, pSelComp[3] được thiết lập bằng pSelComp[0], pSelComp[2] được thiết lập bằng pSelComp[1], pSelComp[0] được thiết lập bằng pSelComp[1], và pSelComp[1] được thiết lập bằng pSelComp[3], với Comp được thay thế bởi DsY và C.

– Các mảng minGrpIdx và maxGrpIdx được thu nhận như sau:

minGrpIdx[0] = 0 (394)

minGrpIdx[1] = 2 (395)

maxGrpIdx[0] = 1 (396)

maxGrpIdx[1] = 3 (397)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[minGrpIdx[1]], minGrpIdx[0] và minGrpIdx[1] được hoán đổi như sau:

(minGrpIdx[0], minGrpIdx[1]) = Swap(minGrpIdx[0], minGrpIdx[1]) (398)

– Khi pSelDsY[maxGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], maxGrpIdx[0] và maxGrpIdx[1] được hoán đổi như sau:

(maxGrpIdx[0], maxGrpIdx[1]) = Swap(maxGrpIdx[0], max GrpIdx[1]) (399)

– Khi pSelDsY[minGrpIdx[0]] lớn hơn pSelDsY[maxGrpIdx[1]], các mảng minGrpIdx và maxGrpIdx được hoán đổi như sau:

(minGrpIdx, maxGrpIdx) = Swap(minGrpIdx, maxGrpIdx) (400)

– Khi pSelDsY[minGrpIdx[1]] lớn hơn pSelDsY[maxGrpIdx[0]], minGrpIdx[1] và maxGrpIdx[0] được hoán đổi như sau:

(minGrpIdx[1], maxGrpIdx[0]) = Swap(minGrpIdx[1], maxG rpIdx[0]) (401)

– Các biến maxY, maxC, minY và minC được thu nhận như sau:

maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1 (402)

maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1 (403)

minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1 (404)

minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1 (405)

7. Các biến a, b, và k được thu nhận như sau:

– Nếu numSampL bằng 0, và numSampT bằng 0, phần sau đây được áp dụng:

k = 0 (406)

a = 0 (407)

b = 1 << (BitDepth − 1) (408)

– Nếu không phải, phần sau đây được áp dụng:

diff = maxY − minY (409)

– Nếu diff không bằng 0, phần sau đây được áp dụng:

$$diffC = maxC - minC \quad (410)$$

$$x = Floor(Log2(diff)) \quad (411)$$

$$normDiff = ((diff << 4) >> x) \& 15 \quad (412)$$

$$x += (normDiff != 0) ? 1 : 0 \quad (413)$$

$$y = Floor(Log2(Abs (diffC))) + 1 \quad (414)$$

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y \quad (415)$$

$$k = ((3 + x - y) < 1) ? 1 : 3 + x - y \quad (416)$$

$$a = ((3 + x - y) < 1) ? Sign(a) * 15 : a \quad (417)$$

$$b = minC - ((a * minY) >> k) \quad (418)$$

Trong đó divSigTable[ ] được chỉ rõ như sau:

$$divSigTable[ ] = \{ 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 \} \quad (419)$$

– Nếu không phải (diff bằng 0), phần sau đây được áp dụng:

$$k = 0 \quad (420)$$

$$a = 0 \quad (421)$$

$$b = minC \quad (422)$$

8. Các mẫu dự đoán predSamples[x][y] với
x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

$$predSamples[x][y] = Clip1(((pDsY[x][y] * a) >> k) + b) \quad (423)$$

Lưu ý – Xử lý này sử dụng sps_chroma_vertical_collocated_flag. Tuy nhiên, để đơn giản hóa cách thức thực hiện, xử lý này không sử dụng sps_chroma_horizontal_collocated_flag.

Các phương pháp và phương án sau đây được thực hiện bởi thiết bị giải mã được đề xuất. Thiết bị giải mã có thể là bộ giải mã viđeo 30 của Fig.1A, Fig.1B, hoặc bộ giải mã 30 của Fig.3. Theo giải pháp này, có đề xuất phương pháp dự đoán khối sắc độ theo phương án 1400 của sáng chế như được minh họa trên Fig.14. Phương pháp này có thể là một phần của phương pháp giải mã chuỗi viđeo được mã hóa.

Như được thể hiện trên Fig.14, trong bước 1401, thiết bị phân tích từ dòng bit các phần tử cú pháp. Các phần tử cú pháp bao gồm phần tử cú pháp thứ nhất mà chỉ báo sự lựa chọn của bộ lọc đối với mẫu độ chói. Ví dụ, phần tử cú pháp thứ nhất là sps_cclm_colocated_chroma_flag, mà là tham số SPS.

Các phần tử cú pháp có thể bao gồm phần tử cú pháp thứ hai, phần tử cú pháp thứ hai được sử dụng để chỉ báo rằng chế độ dự đoán nội bộ được sử dụng cho giải mã hiện tại là chế độ CCLM. Chế độ CCLM bao gồm chế độ dự đoán nội bộ INTRA_LT_CCLM, INTRA_L_CCLM hoặc INTRA_T_CCLM.

Trong bước 1402, thiết bị áp dụng bộ lọc tới các mẫu độ chói được khôi phục của thành phần độ chói của khối hiện tại ít nhất dựa trên phần tử cú pháp thứ nhất, để thu nhận các mẫu độ chói được khôi phục được lọc của khối hiện tại. Bộ lọc thuộc về tập hợp của các bộ lọc, và tập hợp của các bộ lọc chỉ bao gồm hai bộ lọc. Bộ lọc là bộ lọc thứ nhất khi giá trị của phần tử cú pháp là giá trị thứ nhất hoặc bộ lọc là bộ lọc thứ hai khi giá trị của phần tử cú pháp là giá trị thứ hai. Ví dụ, khi sps_chroma_vertical_collocated_flag bằng 1, bộ lọc F3 được áp dụng, hoặc khi sps_chroma_vertical_collocated_flag bằng 0, bộ lọc F4 được áp dụng.

Cụ thể, các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống pDsY[x][y] với x = 0..nTbW − 1, y = 0..nTbH − 1 được thu nhận như sau:

– Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

$$pDstY[x][y] = pY[x][y]$$

– Nếu không phải, phần sau đây được áp dụng:

– Mảng các hệ số bộ lọc một chiều F2 được chỉ rõ như sau.

F2[0] = 1, F2[1] = 2, F2[2] = 1

– Nếu không phải, các mảng hệ số bộ lọc hai chiều F3 và F4 được chỉ rõ như sau:

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

pDsY[x][y] = (F3[1][0] * pY[SubWidthC * x][SubHeightC * y − 1] +

F3[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F3[1][1] * pY[SubWidthC * x][SubHeightC * y] +
F3[2][1] *
pY[SubWidthC * x + 1][SubHeightC * y] +

F3[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] + 4) >> 3

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

pDsY[x][y] = (F4[0][1] * pY[SubWidthC * x − 1][SubHeightC * y] +

F4[0][2] * pY[SubWidthC * x − 1][SubHeightC * y + 1] +

F4[1][1] * pY[SubWidthC * x][SubHeightC * y] +

F4[1][2] * pY[SubWidthC * x][SubHeightC * y + 1] +

F4[2][1] * pY[SubWidthC * x + 1][SubHeightC * y] +

F4[2][2] * pY[SubWidthC * x + 1][SubHeightC * y + 1] + 4) >> 3

Trong bước 1403, thiết bị áp dụng bộ lọc tới các mẫu độ chói bên trái lân cận ít nhất dựa trên phần tử cú pháp thứ nhất, để thu nhận các mẫu độ chói bên trái lân cận được lấy mẫu xuống. Bộ lọc thuộc về tập hợp của các bộ lọc, và tập hợp của các bộ lọc chỉ bao gồm hai bộ lọc. Bộ lọc là bộ lọc thứ nhất khi giá trị của phần tử cú pháp là giá trị thứ nhất hoặc bộ lọc là bộ lọc thứ hai khi giá trị của phần tử cú pháp là giá trị thứ hai. Ví dụ, khi sps_chroma_vertical_collocated_flag bằng 1, bộ lọc F3 được áp dụng, hoặc khi sps_chroma_vertical_collocated_flag bằng 0, bộ lọc F4 được áp dụng.

Cụ thể, khi numSampL lớn hơn 0, các mẫu sắc độ bên trái lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[−1][pickPosL[idx]] với idx = 0..cntL − 1, và các mẫu độ chói bên trái lân cận được lấy mẫu xuống được lựa chọn pSelDsY[idx] với idx = 0..cntL − 1 được thu nhận như sau:

- Biến y được thiết lập bằng pickPosL[idx].
- Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

pSelDsY[idx] = pY[− 1][y]

– Nếu không phải, phần sau đây được áp dụng:

– Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:

pSelDsY[idx] = (F3[1][0] *
pY[− SubWidthC][SubHeightC * y − 1] +
F3[0][1] *
pY[−1 − SubWidthC][SubHeightC * y] +

F3[1][1] * pY[−SubWidthC][SubHeightC * y] +

F3[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F3[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +
4) >> 3

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1 − SubWidthC][Sub HeightC * y] +

F4[0][2] * pY[−1 − SubWidthC][SubHeightC * y + 1] +

F4[1][1] * pY[−SubWidthC][SubHeightC * y] +

F4[1][2] * pY[−SubWidthC][SubHeightC * y + 1] +

F4[2][1] * pY[1 − SubWidthC][SubHeightC * y] +

F4[2][2] * pY[1 − SubWidthC][SubHeightC * y + 1]

+ 4) >> 3

Trong bước 1404, thiết bị áp dụng bộ lọc tới các mẫu độ chói trên cùng lân cận ít nhất dựa trên phần tử cú pháp thứ nhất, để thu nhận các mẫu độ chói trên cùng lân cận được lấy mẫu xuống. Bộ lọc thuộc về tập hợp của các bộ lọc, và tập hợp của các bộ lọc chỉ bao gồm hai bộ lọc. Bộ lọc là bộ lọc thứ nhất khi giá trị của phần tử cú pháp là giá trị thứ nhất hoặc bộ lọc là bộ lọc thứ hai khi giá trị của phần tử cú pháp là giá trị thứ hai. Ví dụ, khi sps_chroma_vertical_collocated_flag bằng 1, bộ lọc F3 được áp dụng, hoặc khi sps_chroma_vertical_collocated_flag bằng 0, bộ lọc F4 được áp dụng.

Cụ thể, khi numSampT lớn hơn 0, các mẫu sắc độ trên cùng lân cận được lựa chọn pSelC[idx] được thiết lập bằng p[pickPosT[idx − cntL]][-1] với idx = cntL..cntL + cntT − 1, và các mẫu độ chói trên cùng lân cận được lấy mẫu xuống pSelDsY[idx] với idx = 0..cntL + cntT − 1 được chỉ rõ như sau:

- Biến x được thiết lập bằng pickPosT[idx − cntL].
- Nếu cả SubWidthC và SubHeightC bằng 1, phần sau đây được áp dụng:

  pSelDsY[idx] = pY[x][− 1]

- Nếu không phải, phần sau đây được áp dụng:
  - Nếu sps_chroma_vertical_collocated_flag bằng 1, phần sau đây được áp dụng:
    - Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

      pSelDsY[idx] = (F3[1][0] * pY[SubWidthC * x][− 1 − SubHeightC] +

      F3[0][1] * pY[SubWidthC * x − 1][−SubHeightC] +

F3[1][1] * pY[SubWidthC * x][−SubHeightC] +

F3[2][1] * pY[SubWidthC * x + 1][−SubHeightC] +

F3[1][2] * pY[SubWidthC * x][1 − SubHeightC] + 4
) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] +

F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2

– Nếu không phải (sps_chroma_vertical_collocated_flag bằng 0), phần sau đây được áp dụng:

– Nếu bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[SubWidthC x − 1][−1] +
F4[0][2] * pY[SubWidthC * x − 1][−2] +
F4[1][1] * pY[SubWidthC * x][−1] +
F4[1][2] * pY[SubWidthC * x][−2] +
F4[2][1] * pY[SubWidthC * x + 1][−1] +

F4[2][2] * pY[SubWidthC * x + 1][−2] + 4) >> 3

– Nếu không phải (bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

$$pSelDsY[\ idx\ ] = (\ F2[\ 0\ ] * pY[\ SubWidthC * x - 1\ ][\ -1\ ] + F2[\ 1\ ] * pY[\ SubWidthC * x\ ][\ -1\ ] + F2[\ 2\ ] * pY[\ SubWidthC * x + 1][\ -1\ ] + 2\ ) >> 2.$$

Không có chuỗi xử lý giữa các bước 1402, 1403 và 1404.

Trong 1405, thiết bị thu nhận các hệ số mô hình tuyến tính dựa trên các mẫu độ chói được khôi phục được lọc. Các mẫu độ chói được khôi phục được lọc bao gồm các mẫu độ chói được khôi phục được lọc của khối hiện tại trong bước 1402, các mẫu độ chói bên trái lân cận được lấy mẫu xuống trong bước 1403, và/hoặc các mẫu độ chói trên cùng lân cận được lấy mẫu xuống trong bước 1404. Trong bước 1406, thiết bị thực hiện việc dự đoán thành phần chéo dựa trên các hệ số mô hình tuyến tính thu được và các mẫu độ chói được khôi phục được lọc của khối hiện tại để thu nhận giá trị dự đoán của thành phần sắc độ của khối hiện tại.

Các phương án được mô tả nêu trên có thể được thực hiện tring thiết bị mã 1500 như được họa trên Fig.15. Thiết bị mã 1500 bao gồm:

- bộ phân tích 1501, có cấu trúc để phân tích từ dòng bit phần tử cú pháp thứ nhất. Như được mô tả nêu trên, ví dụ, phần tử cú pháp thứ nhất là sps_cclm_colocated_chroma_flag, mà được chứa trong tham số SPS.

- bộ lọc 1502, có cấu trúc để áp dụng bộ lọc tới các mẫu độ chói được khôi phục của thành phần độ chói của khối hiện tại ít nhất dựa trên phần tử cú pháp thứ nhất, để thu nhận các mẫu độ chói được khôi phục được lọc của khối hiện tại, trong đó bộ lọc thuộc về tập hợp của các bộ lọc, và tập hợp của các bộ lọc chỉ bao gồm hai bộ lọc, và trong đó bộ lọc là bộ lọc thứ nhất khi giá trị của phần tử cú pháp là giá trị thứ nhất hoặc bộ lọc là bộ lọc thứ hai khi giá trị của phần tử cú pháp là giá trị thứ hai.

Ví dụ, khi sps_chroma_vertical_collocated_flag bằng 1, bộ lọc F3 được áp dụng, hoặc khi sps_chroma_vertical_collocated_flag bằng 0, bộ lọc F4 được áp dụng.

Bộ lọc 1502 có thể cũng áp dụng bộ lọc tới các mẫu độ chói bên trái lân cận ít nhất dựa trên phần tử cú pháp thứ nhất, để thu nhận các mẫu độ chói bên trái lân cận được lấy mẫu xuống, và áp dụng bộ lọc tới các mẫu độ chói trên cùng lân cận ít nhất dựa trên phần tử cú pháp thứ nhất, để thu nhận các mẫu độ chói trên cùng lân cận được lấy mẫu xuống, như được mô tả nêu trên.

- bộ thu nhận 1503, có cấu trúc để thu nhận các hệ số mô hình tuyến tính dựa trên các mẫu độ chói được khôi phục được lọc.

- bộ dự đoán 1504, có cấu trúc để thực hiện việc dự đoán thành phần chéo (như dự đoán sắc độ từ độ chói thành phần chéo hoặc dự đoán CCLM) dựa trên các hệ số mô hình tuyến tính thu được và các mẫu độ chói được khôi phục được lọc của khối hiện tại để thu nhận giá trị dự đoán của thành phần sắc độ của khối hiện tại.

Cụ thể, các phương án sau đây được đề xuất ở đây:

Phương án 1.A phương pháp dự đoán nội bộ nhờ sử dụng mô hình tuyến tính, bao gồm:

- xác định bộ lọc đối với thành phần độ chói của khối hiện tại, dựa trên khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về;

-áp dụng bộ lọc được xác định tới vùng của các mẫu độ chói được khôi phục của thành phần độ chói của khối hiện tại và các mẫu độ chói trong vị trí được lựa chọn mà lân cận (một hoặc một vài hàng/cột liền kề với phía bên trái hoặc trên cùng của khối hiện tại) tới khối hiện tại, để thu nhận các mẫu độ chói được khôi phục được lọc (ví dụ mẫu độ chói được khôi phục được lọc bên trong khối hiện tại(như thành phần độ chói của khối hiện tại));

thu nhận, dựa trên các mẫu độ chói được khôi phục được lọc như là đầu vào của thu nhận mô hình tuyến tính (ví dụ tập hợp của các mẫu độ chói bao gồm các mẫu độ chói được khôi phục được lọc bên trong khối hiện tại và các mẫu độ

chói lân cận được lọc phía ngoài khối hiện tại, ví dụ, bộ lọc được xác định có thể cũng được áp dụng tới các mẫu độ chói lân cận phía ngoài khối hiện tại), các hệ số mô hình tuyến tính; và

thực hiện dự đoán thành phần chéo dựa trên các hệ số mô hình tuyến tính thu được và các mẫu độ chói được khôi phục được lọc của khối hiện tại (ví dụ các mẫu độ chói được khôi phục được lọc bên trong khối hiện tại (như thành phần độ chói của khối hiện tại)) để thu nhận biến dự đoán của khối sắc độ hiện tại.

Phương án 2. Phương pháp theo phương ánt 1, trong đó bước xác định bộ lọc, bao gồm:

xác định bộ lọc dựa trên vị trí của mẫu độ chói trong khối hiện tại và khuôn dạng sắc độ; hoặc

xác định các bộ lọc tương ứng cho các mẫu độ chói mà thuộc về khối hiện tại, dựa trên các vị trí tương ứng của các mẫu độ chói trong khối hiện tại và khuôn dạng sắc độ.

Phương án 3. Phương pháp theo phương án 1, trong đó bước xác định bộ lọc, bao gồm: xác định bộ lọc dựa trên một hoặc nhiều trong số thông tin sau đây:

thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC, mà có thể được thu nhận từ bảng theo khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về),

khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về (ví dụ, trong đó khuôn dạng sắc độ được sử dụng để thu nhận thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC)),

vị trí của mẫu độ chói trong khối hiện tại,

số lượng mẫu độ chói mà thuộc về khối hiện tại,

độ rộng và độ cao của khối hiện tại, và/hoặc

vị trí của mẫu sắc độ được lấy mẫu con so với mẫu độ chói trong khối hiện tại.

Phương án 4. Phương pháp theo phương án 3, trong đó khi mẫu sắc độ được lấy mẫu con không được sắp xếp cùng vị trí với mẫu độ chói tương ứng, quan hệ

được thiết lập trước thứ nhất (như bảng 4) giữa các bộ lọc và thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC, hoặc như các giá trị của độ rộng và độ cao của khối hiện tại) được sử dụng để xác định bộ lọc; và/hoặc,

khi mẫu sắc độ được lấy mẫu con được sắp xếp cùng vị trí với mẫu độ chói tương ứng, quan hệ được thiết lập trước thứ hai hoặc thứ ba (như các bảng 2 hoặc bảng 3) giữa các bộ lọc và thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC, hoặc như các giá trị của độ rộng và độ cao của khối hiện tại) được sử dụng cho việc xác định bộ lọc.

Phương án 5. Phương pháp theo phương án 4, trong đó quan hệ thứ hai hoặc thứ ba (như các bảng 2 hoặc bảng 3) giữa các bộ lọc và thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC, hoặc như các giá trị của độ rộng và độ cao của khối hiện tại) được xác định trên cơ sở của số lượng mẫu độ chói (như mẫu độ chói khả dụng) mà thuộc về khối hiện tại.

Phương án 6. Phương pháp theo bất kỳ một trong số phương án nêu trên, trong đó bộ lọc được xác định có điều kiện như sau:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không được thỏa mãn, phần sau đây được áp dụng đối với tập hợp của các bộ lọc {F3, F5, F6};

F3[0] = 1, F3[1] = 2, F3[2] = 1

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F5[0][1] = 1, F5[1][1] = 4, F3[2][1] = 1, F5[1][0] = 1, F5[1][2] = 1

F6[0][1] = 1, F6[1][1] = 2, F6[2][1] = 1,

F6[0][2] = 1, F6[1][2] = 2, F6[2][2] = 1,

F2[0] = 1, F2[1] = 1

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F5[0][1] = 0, F5[1][1] = 8, F3[2][1] = 0, F5[1][0] = 0, F5[1][2] = 0

F6[0][1] = 2, F6[1][1] = 4, F6[2][1] = 2,

F6[0][2] = 0, F6[1][2] = 0, F6[2][2] = 0,

F2[0] = 2, F2[1] = 0.

Phương án 7. Phương pháp theo bất kỳ một trong số phương án nêu trên, trong đó bộ lọc được xác định có điều kiện như sau nhờ sử dụng ký hiệu của các bộ lọc tuần tự:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không được thỏa mãn, phần sau đây được áp dụng đối với tập hợp của các bộ lọc {F1, F2, F3}:

F1[0] = 2, F1[1] = 0

F2[0] = 1, F2[1] = 2, F2[2] = 1;

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F1[0] = 1, F1[1] = 1,

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1,

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1,

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1;

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F3[0][1] = 0, F3[1][1] = 8, F3[2][1] = 0, F3[1][0] = 0, F3[1][2] = 0

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2,

F4[0][2] = 0, F4[1][2] = 0, F4[2][2] = 0.

Phương án 7a. Phương pháp phương án 1-6, trong đó bộ lọc được xác định có điều kiện như sau nhờ sử dụng ký hiệu của các bộ lọc tuần tự:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không được thỏa mãn, phần sau đây được áp dụng đối với tập hợp của các bộ lọc {F1, F2, F3}:

F1[0] = 4, F1[1] = 0

F2[0] = 1, F2[1] = 2, F2[2] = 1;

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F1[0] = 3, F1[1] = 1,

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1,

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1,

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1;

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F3[0][1] = 0, F3[1][1] = 8, F3[2][1] = 0, F3[1][0] = 0, F3[1][2] = 0

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2,

F4[0][2] = 0, F4[1][2] = 0, F4[2][2] = 0

Phương án 7b. Phương pháp theo phương án 7, trong đó việc lọc bởi một trong số các bộ lọc Fa∈{F1 … F4} được thực hiện nhờ sử dụng hoạt động tạo đệm và bộ lọc Fb, mà có nhiều các hệ số không phải 0 hơn so với bộ lọc Fa.

Phương án 7c. Phương pháp theo phương án 7b, trong đó Fa là F1 và Fb là F2, và trong đó bộ lọc được xác định có điều kiện như sau nhờ sử dụng ký hiệu của các bộ lọc tuần tự:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định trong tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không được thỏa mãn:

F2[0] = 1, F2[1] = 2, F2[2] = 1;

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1,

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1,

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1;

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F3[0][1] = 0, F3[1][1] = 8, F3[2][1] = 0, F3[1][0] = 0, F3[1][2] = 0

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2,

F4[0][2] = 0, F4[1][2] = 0, F4[2][2] = 0

Phương án 8. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó bộ lọc bao gồm các hệ số không phải 0 tại các vị trí mà liền kề theo chiều dọc và/hoặc ngang với vị trí của mẫu độ chói được khôi phục được lọc, khi thành phần sắc độ của khối hiện tại không được lấy mẫu con.

(như $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$, trong đó vị trí trung tâm với hệ số "4" tương ứng với vị trí của mẫu độ chói được khôi phục được lọc)

Phương án 9. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khuôn dạng sắc độ bao gồm khuôn dạng sắc độ YCbCr 4:4:4, khuôn dạng sắc độ YCbCr 4:2:0, khuôn dạng sắc độ YCbCr 4:2:2, hoặc đơn sắc.

Phương án 10. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó tập hợp của các mẫu độ chói được sử dụng như là đầu vào của thu nhận mô hình tuyến tính, bao gồm:

các mẫu được khôi phục độ chói biên mà được lấy mẫu con từ các mẫu độ chói được khôi phục được lọc (như ${Rec'}_L[x, y]$).

Phương án 11. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó biến dự đoán đối với khối sắc độ hiện tại được thu nhận dựa trên:

$$pred_C(i, j) = \alpha \cdot {rec_L}'(i, j) + \beta$$

Trong đó $pred_C(i, j)$ biểu diễn mẫu sắc độ, và $rec_L(i, j)$ biểu diễn mẫu độ chói được khôi phục tương ứng (ví dụ, vị trí của mẫu độ chói được khôi phục tương ứng nằm bên trong khối hiện tại).

Phương án 12. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó mô hình tuyến tính là mô hình tuyến tính đa chiều (MDLM), và các hệ số mô hình tuyến tính được sử dụng để thu nhận MDLM.

Phương án 13. Phương pháp mã hóa được thực hiện bởi thiết bị mã hóa, bao gồm:

thực hiện việc dự đoán nội bộ nhờ sử dụng mô hình tuyến tính (như mô hình tuyến tính thành phần chéo (CCLM), hoặc mô hình tuyến tính đa chiều (MDLM)); và

tạo ra dòng bit bao gồm các phần tử cú pháp, trong đó các phần tử cú pháp bao gồm phần tử cú pháp mà chỉ báo việc lựa chọn bộ lọc dùng cho mẫu độ chói mà thuộc về khối (như lựa chọn bộ lọc độ chói của CCLM, cụ thể, cờ SPS, như sps_cclm_colocated_chroma_flag).

Phương án 14. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi giá trị của phần tử cú pháp là 1 hoặc đúng, bộ lọc không được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán;

khi giá trị của phần tử cú pháp là 0 hoặc sai, bộ lọc không được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán.

Phương án 15. Phương pháp giải mã được thực hiện bởi thiết bị giải mã, bao gồm:

phân tích từ dòng bit các phần tử cú pháp, trong đó các phần tử cú pháp bao gồm phần tử cú pháp mà chỉ báo việc lựa chọn bộ lọc dùng cho mẫu độ chói mà thuộc về khối (như lựa chọn bộ lọc độ chói của CCLM, cụ thể, cờ SPS, như sps_cclm_colocated_chroma_flag); và

thực hiện việc dự đoán nội bộ nhờ sử dụng mô hình tuyến tính được chỉ báo (như CCLM).

Phương án 16. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi giá trị của phần tử cú pháp là 0 hoặc sai, bộ lọc được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán;

khi giá trị của phần tử cú pháp là 1 hoặc đúng, bộ lọc không được áp dụng tới mẫu độ chói để xác định mô hình tuyến tính và dự đoán.

Phương án 17. Phương pháp thực hiện dự đoán nội bộ nhờ sử dụng mô hình tuyến tính, bao gồm:

xác định tập hợp của các bộ lọc lấy mẫu xuống dựa trên thông tin khuôn dạng sắc độ, trong đó thông tin khuôn dạng sắc độ chỉ báo khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về;

thu nhận các mẫu độ chói được lấy mẫu xuống của các mẫu độ chói được khôi phục trong khối độ chói của khối hiện tại và các mẫu tham chiếu độ chói được lấy mẫu xuống của các mẫu độ chói lân cận được lựa chọn nằm ngoài khối độ chói bằng cách áp dụng bộ lọc lấy mẫu xuống tương ứng trong số tập hợp của các bộ lọc lấy mẫu xuống;

xác định một hoặc nhiều các hệ số mô hình tuyến tính dựa trên các mẫu tham chiếu độ chói được lấy mẫu xuống của các mẫu độ chói lân cận được lựa chọn và các mẫu tham chiếu sắc độ mà tương ứng với các mẫu tham chiếu độ chói được lấy mẫu xuống; và

thu nhận các mẫu dự đoán của khối sắc độ mà tương ứng với khối độ chói dựa trên các hệ số mô hình tuyến tính và các mẫu độ chói được lấy mẫu xuống của các mẫu độ chói được khôi phục trong khối độ chói.

Phương án 18. Phương pháp theo phương án 17, trong đó bước xác định tập hợp của các bộ lọc lấy mẫu xuống dựa trên thông tin khuôn dạng sắc độ, bao gồm:

xác định thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC) dựa trên thông tin khuôn dạng sắc độ; và

xác định tập hợp của các bộ lọc lấy mẫu xuống dựa trên thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC).

Phương án 19. Phương pháp theo phương án 17, trong đó bước xác định tập hợp của các bộ lọc lấy mẫu xuống dựa trên khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về, bao gồm một hoặc nhiều trong số phần sau đây:

khi khuôn dạng sắc độ là khuôn dạng màu 4:2:0, xác định tập hợp thứ nhất của các bộ lọc lấy mẫu xuống mà được sử dụng cho khối độ chói của khối hiện tại;

khi khuôn dạng sắc độ là khuôn dạng màu 4:2:2, xác định tập hợp thứ hai của các bộ lọc lấy mẫu xuống mà được sử dụng cho khối độ chói của khối hiện tại; hoặc

khi khuôn dạng sắc độ là khuôn dạng màu 4:4:4, xác định tập hợp thứ ba của các bộ lọc lấy mẫu xuống mà được sử dụng cho khối độ chói của khối hiện tại.

Phương án 20. Phương pháp theo phương án 18, trong đó bước xác định tập hợp của các bộ lọc lấy mẫu xuống(tập hợp của các hệ số bộ lọc lấy mẫu xuống) dựa trên thông tin tỷ lệ lấy mẫu con (như SubWidthC và SubHeightC), bao gồm một hoặc nhiều trong số phần sau đây:

khi SubWidthC=2 và SubHeightC=2, xác định tập hợp thứ nhất của các bộ lọc lấy mẫu xuống (tập hợp của các hệ số bộ lọc lấy mẫu xuống) mà được sử dụng cho khối độ chói của khối hiện tại;

khi SubWidthC=2 và SubHeightC=1, xác định tập hợp thứ hai của các bộ lọc lấy mẫu xuống (tập hợp của các hệ số bộ lọc lấy mẫu xuống) mà được sử dụng cho khối độ chói của khối hiện tại; hoặc

khi SubWidthC=1 và SubHeightC=1, xác định tập hợp thứ ba của các bộ lọc lấy mẫu xuống (tập hợp của các hệ số bộ lọc lấy mẫu xuống) mà được sử dụng cho khối độ chói của khối hiện tại.

Phương án 21. Phương pháp theo bất kỳ một trong số phương án 17 đến 20, trong đó nếu mẫu sắc độ được lấy mẫu con được sắp xếp cùng vị trí với mẫu độ chói tương ứng trong khối hiện tại,

Bước thu nhận các mẫu độ chói được lấy mẫu xuống của các mẫu độ chói được khôi phục trong khối độ chói của khối hiện tại và các mẫu tham chiếu độ chói được lấy mẫu xuống của các mẫu độ chói lân cận được lựa chọn nằm ngoài khối độ chói bằng cách áp dụng bộ lọc lấy mẫu xuống tương ứng trong số tập hợp của các bộ lọc lấy mẫu xuống, bao gồm:

thu nhận mẫu độ chói được lấy mẫu xuống của mẫu độ chói được khôi phục trong khối độ chói bằng cách áp dụng bộ lọc lấy mẫu xuống thứ nhất tới vùng không gian của các mẫu độ chói được khôi phục, trong đó vùng không gian của các mẫu độ chói được khôi phục bao gồm: các mẫu độ chói được khôi phục tại các vị trí mà liền kề theo chiều ngang và/hoặc dọc với vị trí của mẫu độ chói được khôi phục (mẫu độ chói được lấy mẫu xuống); và

thu nhận mẫu tham chiếu độ chói được lấy mẫu xuống của mẫu độ chói lân cận được lựa chọn bằng cách áp dụng các bộ lọc lấy mẫu xuống thứ nhất tới vùng không gian của các mẫu độ chói được khôi phục, trong đó vùng không gian của các mẫu độ chói được khôi phục bao gồm: các mẫu độ chói được khôi phục tại các vị trí mà liền kề theo chiều ngang và/hoặc chiều dọc với vị trí của mẫu độ chói lân cận được lựa chọn (mẫu tham chiếu độ chói được lấy mẫu xuống);

trong đó bộ lọc lấy mẫu xuống thứ nhất nằm trong số tập hợp của các bộ lọc lấy mẫu xuống.

Phương án 22. Phương pháp theo phương án 21, trong đó nếu khuôn dạng màu 4:2:2 được sử dụng (SubWidthC=2 và SubHeightC=1), bộ lọc lấy mẫu xuống thứ nhất là bộ lọc không tách biệt 1D; hoặc

nếu khuôn dạng màu 4:2:0 được sử dụng (SubWidthC=2 và SubHeightC=2), bộ lọc lấy mẫu xuống thứ nhất là bộ lọc không tách biệt 2D.

Phương án 23. Phương pháp theo phương án 21, trong đó nếu khuôn dạng màu 4:2:2 được sử dụng (SubWidthC=2 và SubHeightC=1), bộ lọc lấy mẫu xuống thứ nhất (bộ lọc không tách biệt 1D F2 được trích dẫn theo tiêu chuẩn) được biểu diễn bởi [1, 2, 1] hoặc $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$, trong đó các hệ số không phải 0 tại các vị trí mà liền kề theo chiều ngang với vị trí của mẫu độ chói được khôi phục được lọc, trong đó vị trí trung tâm với hệ số "2" tương ứng với vị trí của mẫu độ chói được khôi phục được lọc).

Phương án 24. Phương pháp theo phương án 21, trong đó nếu khuôn dạng màu 4:2:0 được sử dụng (SubWidthC=2 và SubHeightC=2), bộ lọc lấy mẫu xuống thứ nhất (bộ lọc không tách biệt 2D, F3 được trích dẫn theo tiêu chuẩn VVC) được biểu diễn bởi $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$, trong đó các hệ số không phải 0 tại các vị trí mà liền kề theo chiều ngang và/hoặc dọc với vị trí của mẫu độ chói được khôi phục được lọc, trong đó vị trí trung tâm với hệ số "4" tương ứng với vị trí của mẫu độ chói được khôi phục được lọc).

Phương án 25. Phương pháp theo bất kỳ một trong số phương án 17 đến 20, trong đó nếu mẫu sắc độ được lấy mẫu con không được sắp xếp cùng vị trí với mẫu độ chói tương ứng trong khối hiện tại,

bước thu nhận các mẫu độ chói được lấy mẫu xuống của các mẫu độ chói được khôi phục trong khối độ chói của khối hiện tại và các mẫu tham chiếu độ chói được lấy mẫu xuống của các mẫu độ chói lân cận được lựa chọn nằm ngoài khối độ chói bằng cách áp dụng bộ lọc lấy mẫu xuống tương ứng trong số tập hợp của các bộ lọc lấy mẫu xuống, bao gồm;

thu nhận mẫu độ chói được lấy mẫu xuống của mẫu độ chói được khôi phục trong khối độ chói bằng cách áp dụng bộ lọc lấy mẫu xuống thứ hai tới vùng không gian của các mẫu độ chói được khôi phục, trong đó vùng không gian của các mẫu độ chói được khôi phục bao gồm: các mẫu độ chói được khôi phục tại các vị trí mà liền kề theo chiều ngang và/hoặc dọc với vị trí của mẫu độ chói được khôi phục (mẫu độ chói được lấy mẫu xuống); và

thu nhận mẫu tham chiếu độ chói được lấy mẫu xuống của mẫu độ chói lân cận được lựa chọn bằng cách áp dụng các bộ lọc lấy mẫu xuống thứ hai tới vùng không gian của các mẫu độ chói được khôi phục, trong đó vùng không gian của các mẫu độ chói được khôi phục bao gồm: các mẫu độ chói được khôi phục tại các vị trí mà liền kề theo chiều ngang và/hoặc chiều dọc với vị trí của mẫu độ chói lân cận được lựa chọn (mẫu tham chiếu độ chói được lấy mẫu xuống);

trong đó bộ lọc lấy mẫu xuống thứ hai nằm trong số tập hợp của các bộ lọc lấy mẫu xuống.

Phương án 26. Phương pháp theo phương án 25, trong đó nếu khuôn dạng màu 4:2:2 được sử dụng (SubWidthC=2 và SubHeightC=1), bộ lọc lấy mẫu xuống thứ hai là bộ lọc không tách biệt 1D; hoặc

nếu khuôn dạng màu 4:2:0 được sử dụng (SubWidthC=2 và SubHeightC=2), bộ lọc lấy mẫu xuống thứ hai là bộ lọc không tách biệt 2D.

Phương án 27. Phương pháp theo phương án 25, trong đó nếu khuôn dạng màu 4:2:2 được sử dụng (SubWidthC=2 và SubHeightC=1), bộ lọc lấy mẫu xuống thứ hai (bộ lọc không tách biệt 1D F1, F2 được trích dẫn theo tiêu chuẩn) được

biểu diễn bởi [2,0] hoặc [1, 2, 1] hoặc $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$, trong đó các hệ số không phải 0 tại các vị trí mà liền kề theo chiều ngang với vị trí của mẫu độ chói được khôi phục được lọc, trong đó vị trí trung tâm với hệ số "2" tương ứng với vị trí của mẫu độ chói được khôi phục được lọc).

Phương án 28. Phương pháp theo phương án 25, trong đó nếu khuôn dạng màu 4:2:0 được sử dụng (SubWidthC=2 và SubHeightC=2), bộ lọc lấy mẫu xuống thứ hai (bộ lọc không tách biệt 2D, F4 được trích dẫn theo tiêu chuẩn VVC) được biểu diễn bởi $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$, trong đó các hệ số không phải 0 tại các vị trí mà liền kề theo chiều ngang và/hoặc dọc với vị trí của mẫu độ chói được khôi phục được lọc, trong đó vị trí trung tâm với hệ số "2" tương ứng với vị trí của mẫu độ chói được khôi phục được lọc).

Phương án 29. Phương pháp theo bất kỳ một trong số phương án 21 đến 24, trong đó mẫu sắc độ được lấy mẫu con được sắp xếp cùng vị trí với mẫu độ chói tương ứng trong khối hiện tại, được biểu diễn bởi bất kỳ một trong số phần sau đây:

Loại mẫu sắc độ 2, hoặc

Loại mẫu sắc độ 4.

Phương án 30. Phương pháp theo bất kỳ một trong số phương án 25 đến 28, trong đó mẫu sắc độ được lấy mẫu con không được sắp xếp cùng vị trí với mẫu độ chói tương ứng trong khối hiện tại, được biểu diễn bởi bất kỳ một trong số phần sau đây:

Loại mẫu sắc độ 0,

Loại mẫu sắc độ 1,

Loại mẫu sắc độ 3, hoặc

Loại mẫu sắc độ 5.

Phương án 31. Phương pháp theo bất kỳ một trong số phương án nêu trên, trong đó bộ lọc được xác định có điều kiện như sau:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không được thỏa mãn, phần sau đây được áp dụng đối với tập hợp của các bộ lọc {F3, F5, F6};

F3[0] = 1, F3[1] = 2, F3[2] = 1

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F5[0][1] = 1, F5[1][1] = 4, F3[2][1] = 1, F5[1][0] = 1, F5[1][2] = 1

F6[0][1] = 1, F6[1][1] = 2, F6[2][1] = 1,

F6[0][2] = 1, F6[1][2] = 2, F6[2][2] = 1,

F2[0] = 1, F2[1] = 1

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F5[0][1] = 0, F5[1][1] = 8, F3[2][1] = 0, F5[1][0] = 0, F5[1][2] = 0

F6[0][1] = 2, F6[1][1] = 4, F6[2][1] = 2,

F6[0][2] = 0, F6[1][2] = 0, F6[2][2] = 0,

F2[0] = 2, F2[1] = 0.

Phương án 32. Phương pháp theo bất kỳ một trong số phương án nêu trên, trong đó bộ lọc được xác định có điều kiện như sau nhờ sử dụng ký hiệu của các bộ lọc tuần tự:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không

được thỏa mãn, phần sau đây được áp dụng đối với tập hợp của các bộ lọc {F1, F2, F3}:

F1[0] = 2 hoặc 4, F1[1] = 0

F2[0] = 1, F2[1] = 2, F2[2] = 1;

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F1[0] = 1 hoặc 3, F1[1] = 1,

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1,

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1,

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1;

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F3[0][1] = 0, F3[1][1] = 8, F3[2][1] = 0, F3[1][0] = 0, F3[1][2] = 0

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2,

F4[0][2] = 0, F4[1][2] = 0, F4[2][2] = 0

Phương án 33. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2, hoặc, khi các thành phần độ chói và sắc độ được sắp xếp cùng vị trí (được thể hiện là "loại mẫu sắc độ 2" và "loại mẫu sắc độ 4" trên Fig.10B), việc lấy mẫu từ thành phần độ chói được thực hiện với độ dịch so với phía trên cùng của khối độ chói.

Phương án 34. Phương pháp theo bất kỳ một trong số các phương án nêu trên, khi mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:2 hoặc biên khối là biên dòng CTU, bộ lọc [1 2 1]/4 được áp dụng tới các mẫu tham chiếu.

Phương án 35. Phương pháp theo phương án 34, trong đó nếu availTL bằng đúng (TRUE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng đối với F được thiết lập thành F3:

pSelDsY [idx] =

= (F[0] * pY [− 1] [− 1] +

+ F[1] * pY[0] [− 1] +

+ F[2] * pY[1] [− 1] +

+ 2) >> 2.

Phương án 36. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi biên khối không phải là biên dòng CTU, mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:0 (hoặc bất kỳ khuôn dạng sắc độ khác mà sử dụng lấy mẫu con sắc độ dọc), bộ lọc [1 2 1 ; 1 2 1] / 8 được áp dụng tới các mẫu tham chiếu.

Phương án 37. Phương pháp theo phương án 36, trong đó khi nếu availTL bằng đúng (TRUE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F6:

pSelDsY [idx] =

= (F[0][1] * pY[− 1][-1] +

+ F[0][2] * pY[− 1][-2] +

+ F[1][1] * pY[0][-1] +

+ F[1][2] * pY[0][-2] +

+ F[2][1] * pY[1][-1] +

+ F[2][2] * pY[1][−2] + 4) >> 3.

Phương án 38. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi mẫu trên cùng-bên trái không khả dụng và khuôn dạng sắc độ được

chỉ rõ là YUV 4:2:2 hoặc biên khối là biên dòng CTU, bộ lọc đường vòng [1]/1 được áp dụng tới mẫu tham chiếu.

Phương án 39. Phương pháp theo phương án 38, trong đó khi availTL bằng sai (FALSE) và bCTUboundary bằng đúng (TRUE), phần sau đây được áp dụng:

pSelDsY [idx] = pY[0][−1].

Phương án 40. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi biên khối không phải là biên dòng CTU, mẫu trên cùng-bên trái là khả dụng và khuôn dạng sắc độ được chỉ rõ là YUV 4:2:0 (hoặc bất kỳ khuôn dạng sắc độ khác mà sử dụng lấy mẫu con sắc độ dọc), bộ lọc [1 1] / 2 được áp dụng tới các mẫu tham chiếu.

Phương án 41. Phương pháp theo phương án 40, trong đó khi availTL bằng sai (FALSE) và bCTUboundary bằng sai (FALSE), phần sau đây được áp dụng đối với F được thiết lập thành F2:

pSelDsY [idx] = (F[1] * pY[0][−2] + F[0] * pY[0][−1] + 1) >> 1

Phương án 42. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi ít nhất điều kiện thứ nhất không được thỏa mãn (ví dụ, bCTUboundary bằng sai (FALSE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[SubWidthC x − 1][−1] +
F4[0][2] * pY[SubWidthC * x − 1][−2] +
F4[1][1] * pY[SubWidthC * x][−1] +
F4[1][2] * pY[SubWidthC * x][−2] +
F4[2][1] * pY[SubWidthC * x + 1][−1] +
F4[2][2] * pY[SubWidthC * x + 1][−2] + 4) >> 3.

Phương án 43. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi ít nhất điều kiện thứ nhất được thỏa mãn (ví dụ, bCTUboundary bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[SubWidthC * x − 1][−1] +
F2[1] * pY[SubWidthC * x][−1] +
F2[2] * pY[SubWidthC * x + 1][−1] + 2) >> 2.

Phương án 44. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi ít nhất điều kiện thứ nhất không được thỏa mãn và điều kiện thứ hai được thỏa mãn (ví dụ, bCTUboundary bằng sai (FALSE) và availTL bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F4[0][1] * pY[−1][−1] + F4[0][2] * pY[−1][−2] +
F4[1][1] * pY[0][−1] + F4[1][2] * pY[0][−2] +
F4[2][1] * pY[1][−1] + F4[2][2] * pY[1][−2] + 4) >> 3.

Phương án 45. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi ít nhất điều kiện thứ nhất không được thỏa mãn và điều kiện thứ hai được không thỏa mãn (ví dụ, bCTUboundary bằng đúng (TRUE) và availTL bằng đúng (TRUE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F2[0] * pY[−1][−1] + F2[1] * pY[0][−1] +
F2[2] * pY[1][− 1] + 2) >> 2.

Phương án 46. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi ít nhất điều kiện thứ nhất không được thỏa mãn và điều kiện thứ hai không được thỏa mãn (bCTUboundary bằng sai (FALSE) và availTL bằng sai (FALSE)), phần sau đây được áp dụng:

pSelDsY[idx] = (F1[1] * pY[0][−2] + F1[0] * pY[0][−1] + 1) >> 1;

hoặc

pSelDsY[idx] = (F1[1] * pY[0][−2] + F1[0] * pY[0][−1] + 2) >> 2.

Phương án 47. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó khi ít nhất điều kiện thứ nhất được thỏa mãn và điều kiện thứ hai không được thỏa mãn (bCTUboundary bằng đúng (TRUE) và availTL bằng sai (FALSE)), phần sau đây được áp dụng:

pSelDsY[idx] = pY[0][−1].

Phương án 48. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó nếu điều kiện thứ hai được thỏa mãn (ví dụ, availTL bằng đúng (TRUE)), phần sau đây được áp dụng:

pDsY[0][y] = (F1[0] * pY[0][SubHeightC * y] +
F1[1] * pY[0][SubHeightC * y + 1] + 2) >> 2.

Phương án 49. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó thao tác tạo đệm được sử dụng để thực hiện lựa chọn bộ lọc trên biên của khối sắc độ được khôi phục.

Phương án 50. Phương pháp theo bất kỳ một trong số các phương án nêu trên, phương pháp này còn bao gồm:

áp dụng xử lý lọc khi vị trí của vùng được lọc nằm trên biên khối.

Phương án 51. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó các hệ số bộ lọc là phụ thuộc vị trí.

Phương án 52. Phương pháp theo phương án 51, trong đó khi phía trái không khả dụng, các mẫu của phía trái được thu nhận bằng cách sao chép theo chiều ngang các mẫu được khôi phục ngoài cùng bên trái của khối được khôi phục.

Phương án 53. Phương pháp theo phương án 51, trong đó khi phía trên cùng không khả dụng, các mẫu của phía trên cùng được thu nhận bằng cách sao chép theo chiều dọc các mẫu được khôi phục ngoài cùng trên cùng của khối được khôi phục.

Phương án 54. Phương pháp theo bất kỳ một trong số các phương án nêu trên, trong đó việc lọc bởi một trong số các bộ lọc Fa∈{F1 … F4} được thực hiện bằng cách sử dụng hoạt động tạo đệm và bộ lọc Fb, trong đó bộ lọc Fb có nhiều hệ số không phải 0 hơn so với bộ lọc Fa.

Phương án 55. Phương pháp theo phương án 54, trong đó khi Fa là F1 và Fb là F2, bộ lọc F1 được thực hiện như là kết hợp của việc tạo đệm và lọc nhờ sử dụng bộ lọc F2.

Phương án 56. Phương pháp theo bất kỳ một trong số phương án 49 đến 55, trong đó bộ lọc được xác định có điều kiện như sau:

nếu điều kiện thứ nhất (như thông tin tỷ lệ lấy mẫu con thu được từ bảng được xác định trong tiêu chuẩn kỹ thuật, như SubWidthC==1 và SubHeightC==1) không được thỏa mãn:

F2[0] = 1, F2[1] = 2, F2[2] = 1;

F3[i][j] = F4[i][j] = 0, với i = 0..2, j = 0..2

Nếu điều kiện thứ hai (như thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) được thỏa mãn,

F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][0] = 1, F3[1][2] = 1,

F4[0][1] = 1, F4[1][1] = 2, F4[2][1] = 1,

F4[0][2] = 1, F4[1][2] = 2, F4[2][2] = 1;

Nếu không phải, (ví dụ nếu điều kiện thứ hai (thông tin tỷ lệ lấy mẫu con thu được từ bảng, như SubWidthC==2 và SubHeightC==2) không được thỏa mãn),

F3[0][1] = 0, F3[1][1] = 8, F3[2][1] = 0, F3[1][0] = 0, F3[1][2] = 0

F4[0][1] = 2, F4[1][1] = 4, F4[2][1] = 2,

F4[0][2] = 0, F4[1][2] = 0, F4[2][2] = 0.

Phương án 57. Phương pháp theo bất kỳ một trong số phương án 49 đến 56, trong đó hoạt động tạo đệm bao gồm:

khi availT bằng sai (FALSE), các mẫu độ chói trên cùng lân cận pY[x][y] với x = -1..SubWidthC * numSampT − 1, y = −1, -2, được thiết lập bằng các mẫu độ chói pY[x][0]; hoặc

khi availL bằng sai (FALSE), các mẫu độ chói bên trái lân cận pY[x][y] với x = −1, -2, -3, y = -1..SubHeightC * numSampL − 1, được thiết lập bằng các mẫu độ chói pY[0][y].

Theo các khía cạnh trước đó hoặc bất kỳ phương án có thể được thực hiện của các khía cạnh này, để áp dụng bộ lọc tới các mẫu độ chói được khôi phục trong khối độ chói của khối hiện tại, và/hoặc, tới các mẫu độ chói trong vị trí được lựa chọn mà lân cận với khối hiện tại, bộ lọc thuộc về tập hợp của các bộ lọc lấy mẫu xuống, và tập hợp của các bộ lọc chỉ bao gồm số lượng bộ lọc được làm giảm. Điều này có ý nghĩa số lượng bộ lọc trong tập hợp của các bộ lọc lấy mẫu xuống được làm giảm. Ví dụ, phụ thuộc vào giá trị của phần tử cú pháp (như sps_chroma_vertical_collocated_flag), tập hợp của các bộ lọc chỉ bao gồm các bộ lọc F3 và F4, mà có nghĩa rằng chỉ F3 và F4 có thể được lựa chọn. Một cách tương ứng, có thể làm giảm số lượng nhánh đối với các bộ lọc được áp dụng tới các mẫu độ chói được khôi phục, và/hoặc, tới các mẫu độ chói trong vị trí được lựa chọn mà lân cận với khối hiện tại.

FIG.16 là sơ đồ khối thể hiện hệ thống cung cấp nội dung 3100 để thực hiện dịch vụ phân phối nội dung. Hệ thống cung cấp nội dung 3100 này bao gồm thiết bị thu 3102, thiết bị đầu cuối 3106, và một cách tùy chọn bao gồm màn hình 3126. Thiết bị thu 3102 truyền thông với thiết bị đầu cuối 3106 qua liên kết truyền thông 3104. Liên kết truyền thông có thể bao gồm kênh truyền thông 13 được mô tả nêu trên. Liên kết truyền thông 3104 bao gồm nhưng không giới hạn ở WIFI, Ethernet, Cáp, không dây (3G / 4G / 5G), USB hoặc bất kỳ loại kết hợp của chúng hoặc tương tự.

Thiết bị thu 3102 tạo ra dữ liệu và có thể mã hóa dữ liệu bằng phương pháp mã hóa như được thể hiện trong các phương án nêu trên. Ngoài ra, thiết bị thu 3102 có thể phân phối dữ liệu tới máy chủ tạo dòng (không được thể hiện trên hình vẽ), và máy chủ mã hóa dữ liệu và truyền dữ liệu được mã hóa tới thiết bị đầu cuối 3106. Thiết bị thu 3102 bao gồm nhưng không giới hạn ở camera, điện thoại thông minh hoặc máy tính bảng, máy tính hoặc máy tính xách tay, hệ thống hội nghị truyền hình, PDA, thiết bị gắn trên xe, hoặc sự kết hợp của bất kỳ thiết bị trong số chúng hoặc loại tương tự. Ví dụ, thiết bị thu 3102 có thể bao gồm thiết bị nguồn 12 như được mô tả nêu trên. Khi dữ liệu bao gồm viđeo, bộ mã hóa viđeo 20 được chứa trong thiết bị thu 3102 có thể một cách thực tế thực

hiện xử lý mã hóa viđeo. Khi dữ liệu bao gồm auđiô (tức là, thoại), bộ mã hóa auđiô được chứa trong thiết bị thu 3102 có thể một cách thực tế thực hiện xử lý mã hóa auđiô. Đối với một số tình huống thực tế, thiết bị thu 3102 phân phối dữ liệu auđiô thanh và video được mã hóa bằng cách ghép kênh chúng với nhau. Đối với các tình huống thực tế khác, chẳng hạn như trong hệ thống hội nghị truyền hình, dữ liệu auđiô được mã hóa và dữ liệu video được mã hóa không được ghép kênh. Thiết bị thu 3102 phân phối dữ liệu auđiô được mã hóa và dữ liệu video được mã hóa đến thiết bị đầu cuối 3106 một cách riêng biệt.

Trong hệ thống cung cấp nội dung 3100, thiết bị đầu cuối 310 thu và tái tạo dữ liệu được mã hóa. Thiết bị đầu cuối 3106 có thể là thiết bị có khả năng thu và khôi phục dữ liệu, như điện thoại thông minh hoặc máy tính bảng 3108, máy tính hoặc laptop 3110, bộ ghi viđeo mạng (NVR)/ bộ ghi viđeo số (DVR) 3112, TV 3114, hộp giải mã (STB) 3116, hệ thống hội nghị truyền hình 3118, hệ thống giám sát viđeo 3120, thiết bị hỗ trợ cá nhân số (PDA) 3122, thiết bị lắp trên xe 3124, hoặc kết hợp bất kỳ của chúng, hoặc loại tương tự có khả năng giải mã dữ liệu được mã hóa nêu trên. Ví dụ, thiết bị đầu cuối 3106 có thể bao gồm thiết bị đích 14 như được mô tả nêu trên. Khi dữ liệu được mã hóa bao gồm viđeo, bộ giải mã viđeo 30 được chứa trong thiết bị đầu cuối được ưu tiên để thực hiện việc giải mã viđeo. Khi dữ liệu được mã hóa bao gồm auđio, bộ giải mã auđio được chứa trong thiết bị đầu cuối được ưu tiên để thực hiện xử lý giải mã auđio.

Đối với thiết bị đầu cuối có màn hình, ví dụ, điện thoại thông minh hoặc máy tính bảng 3108, máy tính hoặc máy tính xách tay 3110, bộ ghi viđeo mạng (NVR)/bộ ghi viđeo số (DVR) 3112, TV 3114, thiết bị hỗ trợ cá nhân số (PDA) 3122 hoặc thiết bị được gắn trên xe 3124, thiết bị đầu cuối có thể cung cấp dữ liệu được giải mã tới màn hình của thiết bị này. Đối với thiết bị đầu cuối không được trang bị màn hình, chẳng hạn như STB 3116, hệ thống hội nghị truyền hình 3118 hoặc hệ thống giám sát viđeo 3120, màn hình ngoài 3126 được kết nối trong đó để thu và hiển thị dữ liệu được giải mã.

Khi mỗi thiết bị trong hệ thống thực hiện việc mã hóa hoặc giải mã, thiết bị mã hóa ảnh hoặc thiết bị giải mã ảnh như được thể hiện trong các phương án nêu trên có thể được sử dụng.

FIG.17 là sơ đồ thể hiện cấu trúc của ví dụ về thiết bị đầu cuối 3106. Sau khi thiết bị đầu cuối 3106 thu dòng từ thiết bị thu 3102, bộ tiến hành giao thức 3202 phân tích giao thức truyền của dòng. Giao thức bao gồm nhưng không giới hạn ở Giao thức tạo dòng thời gian thực (RTSP), Giao thức truyền siêu văn bản (HTTP), Giao thức tạo dòng trực tiếp HTTP (HLS), MPEG-DASH, Giao thức truyền tải thời gian thực (RTP), Giao thức nhắn tin thời gian thực (RTMP)), hoặc bất kỳ loại kết hợp của chúng, hoặc loại tương tự.

Sau khi bộ tiến hành giao thức 3202 xử lý dòng, tệp dòng được tạo. Tệp được xuất ra bộ giải ghép kênh 3204. Bộ giải ghép kênh 3204 có thể tách biệt dữ liệu được ghép kênh thành dữ liệu audio được mã hóa và dữ liệu viđeo được mã hóa. Như được mô tả nêu trên, đối với các tình huống thực tế khác, chẳng hạn như trong hệ thống hội nghị truyền hình, dữ liệu audiô được mã hóa và dữ liệu video được mã hóa không được ghép kênh. Trong trường hợp này, dữ liệu được mã hóa được truyền tới bộ giải mã viđeo 3206 và bộ giải mã audio 3208 mà không được truyền thông qua bộ giải ghép kênh 3204.

Thông qua xử lý giải ghép kênh, dòng sơ cấp (ES-elementary stream) viđeo, audio ES và phụ đề tùy chọn được tạo ra. Bộ giải mã viđeo 3206 mà bao gồm bộ giải mã viđeo 30 như được giải thích trong các phương án nêu trên, giải mã dòng sơ cấp (ES) viđeo bởi phương pháp giải mã như được thể hiện trong các phương án nêu trên để tạo ra khung viđeo, và cấp dữ liệu này tới tới bộ đồng bộ 3212. Bộ giải mã audio 3208 giải mã dòng sơ cấp audio để tạo ra khung audio, và cấp dữ liệu này tới bộ đồng bộ 3212. Ngoài ra, khung viđeo có thể được lưu trữ trong bộ đệm (không được thể hiện trên FIG.17) trước khi cấp khung video tới bộ đồng bộ 3212. Tương tự, khung audio có thể được lưu trữ trong bộ đệm (không được thể hiện trên FIG.17) trước khi cấp khung audio tới bộ đồng bộ 3212.

Bộ đồng bộ 3212 đồng bộ khung viđeo và khung auđio, và cấp viđeo/auđio tới màn hình viđeo/auđio 3214. Ví dụ, bộ đồng bộ 3212 đồng bộ việc trình diễn thông tin viđeo và thông tin auđio. Thông tin có thể mã hóa trong cú pháp nhờ sử dụng các dấu thời gian liên quan đến việc trình diễn của auđio được mã hóa và dữ liệu trực quan và các dấu thời gian liên quan đến việc phân phát dòng dữ liệu.

Nếu phụ đề được bao gồm trong dòng, bộ giải mã phụ đề 3210 sẽ giải mã phụ đề và đồng bộ hóa nó với khung video và khung auđio, cung cấp video/auđio/phụ đề tới màn hình viđeo/auđio/phụ đề 3216.

Sáng chế không bị giới hạn ở hệ thống nêu trên, và thiết bị mã hóa ảnh hoặc thiết bị giải mã ảnh trong các phương án nêu trên có thể được kết hợp vào hệ thống khác, ví dụ, hệ thống xe hơi.

Các toán tử toán học

Các toán tử toán học được sử dụng trong sáng chế là tương tự như các toán tử được sử dụng trong ngôn ngữ lập trình C. Tuy nhiên, các kết quả của phép chia nguyên và các phép dịch số học được định nghĩa chính xác hơn, và các phép toán khác được xác định, như sự mũ hóa và phép chia có giá trị thực. Các quy ước đánh số và đếm thường bắt đầu từ 0, ví dụ, "thứ nhất" tương đương với thứ 0, "thứ hai" tương đương với thứ 1, v.v.

Các toán tử số học

Các toán tử số học sau đây được định nghĩa như sau:

+	Phép cộng
–	Phép trừ (dưới dạng toán tử hai đối số) hoặc phủ định (dưới dạng toán tử tiền tố một ngôi)
*	Phép nhân, bao gồm phép nhân ma trận

x^y	Lũy thừa. Chỉ rõ x nâng lên lũy thừa của y. Trong các ngữ cảnh khác, ký hiệu này được sử dụng để ghi chỉ số trên không nhằm mục đích để hiểu như là lũy thừa.
/	Phép chia số nguyên với việc cắt kết quả về phía 0. Ví dụ, 7 / 4 và −7 / −4 được cắt thành 1 và −7 / 4 và 7 / −4 được cắt về −1.
÷	Được sử dụng để ký hiệu phép chia trong các phương trình toán học mà không có ý định cắt hoặc làm tròn.
$\frac{x}{y}$	Được sử dụng để ký hiệu phép chia trong các phương trình toán học mà không có ý định cắt hoặc làm tròn.
$\sum_{i=x}^{y} f(i)$	Tổng của f(i) với i nhận tất cả các giá trị nguyên từ x đến và bao gồm cả y.
x% y	Môđun. Phần dư của x chia cho y, chỉ được xác định cho các số nguyên x và y với x> = 0 và y> 0.

Các toán tử logic

Các toán tử logic sau được định nghĩa như sau:

x && y	lôgic Boolean "và" của x và y
x | | y	lôgic Boolean "hoặc" của x và y
!	Lôgic Boolean "không phải (not)"
x ? y :z	Nếu x là đúng (TRUE) hoặc không bằng 0, đánh giá là giá trị của y; nếu không phải, đánh giá là giá trị của z.

Các toán tử quan hệ

Các toán tử quan hệ sau đây được định nghĩa như sau:

>	Lớn hơn
>=	Lớn hơn hoặc bằng
<	Nhỏ hơn
<=	Nhỏ hơn hoặc bằng
==	bằng
!=	Không bằng

Khi toán tử quan hệ được áp dụng tới phần tử cú pháp hoặc biến đã được gán giá trị "na" (không áp dụng), giá trị "na" được coi là giá trị khác biệt đối với phần tử cú pháp hoặc biến. Giá trị "na" được xem là không bằng bất kỳ giá trị khác.

Toán tử thao tác bit (bit-wise)

Các toán tử thao tác bit sau đây được định nghĩa như sau:

&	Thao tác bit "và (and)". Khi thao tác trên các đối số nguyên, thao tác trên biểu diễn phần bù của hai của giá trị nguyên. Khi thao tác trên một đối số nhị phân mà chứa ít bit hơn đối số khác, đối số ngắn hơn được mở rộng bằng cách thêm nhiều bit quan trọng hơn bằng 0.
|	Thao tác bit "hoặc (or)". Khi thao tác trên các đối số nguyên, thao tác trên biểu diễn phần bù của hai của giá trị nguyên. Khi thao tác trên một đối số nhị phân mà chứa ít bit hơn đối số khác, đối số ngắn hơn được mở rộng bằng cách thêm nhiều bit quan trọng hơn bằng 0.
^	Thao tác bit "hoặc loại trừ (exclusive or)". Khi thao tác trên các đối số nguyên, thao tác trên biểu diễn phần bù của hai của giá trị nguyên. Khi thao tác trên một đối số nhị phân mà chứa ít bit hơn đối số khác, đối số ngắn hơn được mở rộng bằng cách thêm nhiều bit quan trọng hơn bằng 0.

x >> y Dịch phải số học của biểu diễn nguyên phần bù của hai của x bởi y ký tự nhị phân. Hàm này chỉ được xác định đối với các giá trị nguyên không âm của y. Các bit được dịch chuyển thành các bit quan trọng nhất (MSB) do kết quả của việc dịch phải có giá trị bằng MSB của x trước phép dịch.

x << y Dịch trái số học của biểu diễn nguyên phần bù của hai của x bởi y ký tự nhị phân. Hàm này chỉ được xác định đối với các giá trị nguyên không âm của y. Các bit được dịch chuyển thành các bit ít quan trọng nhất (LSB) do kết quả của việc dịch trái có giá trị bằng 0.

Các toán tử gán

Các toán tử số học sau đây được định nghĩa như sau:

= Toán tử gán

+ + tăng, tức là, *x*+ + tương đương với *x* = *x* + 1; khi được sử dụng trong chỉ số mảng, đánh giá là giá trị của biến trước phép toán tăng.

− − Giảm, tức là, *x*− − tương đương với *x* = *x* − 1; khi được sử dụng trong chỉ số mảng, đánh giá là giá trị của biến trước pháp toán giảm.

+ = Tăng theo số lượng được chỉ định, tức là x + = 3 tương đương với x = x + 3, và
x + = (−3) tương đương với x = x + (−3).

- = Giảm theo số lượng đã chỉ định, tức là x - = 3 tương đương với x = x - 3, và
x - = (−3) tương đương với x = x - (−3).

Ký hiệu phạm vi

Ký hiệu sau được sử dụng để chỉ rõ phạm vi của các giá trị:

x = y..z x nhận các giá trị nguyên bắt đầu từ đoạn y đến z, với x, y và z là các số nguyên và z lớn hơn y.

Các hàm toán học

Các hàm toán học sau được định nghĩa:

$$Abs(x) = \begin{cases} x & ; \quad x >= 0 \\ -x & ; \quad x < 0 \end{cases}$$

Asin (x) hàm sin nghịch đảo lượng giác, hoạt động trên đối số x nằm trong đoạn từ −1,0 đến 1,0, với giá trị đầu ra trong đoạn từ $-\pi \div 2$ đến $\pi \div 2$, tính bằng đơn vị radian

Atan (x) hàm tang nghịch đảo lượng giác, hoạt động trên đối số x, với giá trị đầu ra trong đoạn từ $-\pi \div 2$ đến $\pi \div 2$, tính bằng đơn vị radian

$$Atan2(y, x) = \begin{cases} Atan\left(\frac{y}{x}\right) & ; \quad x > 0 \\ Atan\left(\frac{y}{x}\right) + \pi & ; \quad x < 0 \;\&\&\; y >= 0 \\ Atan\left(\frac{y}{x}\right) - \pi & ; \quad x < 0 \;\&\&\; y < 0 \\ +\frac{\pi}{2} & ; \quad x = = 0 \;\&\&\; y >= 0 \\ -\frac{\pi}{2} & ; \quad \text{trường hợp khác} \end{cases}$$

Ceil(x) số nguyên nhỏ nhất lớn hơn hoặc bằng x.

$Clip1_Y(x) = Clip3(0, (1 << BitDepth_Y) - 1, x)$

$Clip1_C(x) = Clip3(0, (1 << BitDepth_C) - 1, x)$

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{trường hợp khác} \end{cases}$$

Cos (x) hàm cosin lượng giác hoạt động trên đối số x tính bằng đơn vị radian.

Floor(x) số nguyên lớn nhất nhỏ hơn hoặc bằng x.

$$\text{GetCurrMsb}(a, b, c, d) = \begin{cases} c + d & ; \quad b - a >= d / 2 \\ c - d & ; \quad a - b > d / 2 \\ c & ; \quad \text{trường hợp khác} \end{cases}$$

Ln (x) logarit tự nhiên của x (logarit cơ số e, trong đó e là hằng số logarit tự nhiên 2,718 281 828 ...).

Log 2 (x) logarit cơ số 2 của x.

Log10(x) logarit cơ số 10 của x.

$$\text{Min}(x, y) = \begin{cases} x & ; \quad x <= y \\ y & ; \quad x > y \end{cases}$$

$$\text{Max}(x, y) = \begin{cases} x & ; \quad x >= y \\ y & ; \quad x < y \end{cases}$$

$$\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0.5)$$

$$\text{Sign}(x) = \begin{cases} 1 & ; \quad x > 0 \\ 0 & ; \quad x == 0 \\ -1 & ; \quad x < 0 \end{cases}$$

Sin (x) hàm sin lượng giác hoạt động trên đối số x tính bằng đơn vị radian

$$\text{Sqrt}(x) = \sqrt{x}$$

$$\text{Swap}(x, y) = (y, x)$$

Tan (x) hàm tang lượng giác hoạt động trên đối số x tính bằng đơn vị radian

Thứ tự ưu tiên phép toán

Khi thứ tự ưu tiên trong một biểu thức không được chỉ báo rõ ràng bằng cách sử dụng dấu ngoặc đơn, các quy tắc sau sẽ được áp dụng:

- Các phép toán có mức độ ưu tiên cao hơn được đánh giá trước bất kỳ phép toán có mức độ ưu tiên thấp hơn.
- Các phép toán có cùng thứ tự ưu tiên được đánh giá tuần tự từ trái sang phải.

Bảng dưới đây chỉ rõ mức độ ưu tiên của các phép toán từ cao nhất đến thấp nhất; vị trí cao hơn trong bảng chỉ báo mức độ ưu tiên cao hơn.

Đối với những toán tử mà cũng được sử dụng trong ngôn ngữ lập trình C, thứ tự ưu tiên được sử dụng trong bản mô tả này giống như được sử dụng trong ngôn ngữ lập trình C.

Bảng: Mức độ ưu tiên phép toán từ cao nhất (ở đầu bảng) đến thấp nhất (ở cuối bảng)

các phép toán (với các toán hạng x, y và z)
"x++", "x– –"
"!x", "–x" (dưới dạng toán tử tiền tố một ngôi)
x^y
"x * y", "x / y", "x ÷ y", "$\frac{x}{y}$", "x % y"
"x + y", "x – y" (dưới dạng toán tử hai đối số), "$\sum_{i=x}^{y} f(i)$"
"x << y", "x >> y"
"x < y", "x <= y", "x > y", "x >= y"
"x = = y", "x != y"
"x & y"
"x | y"
"x && y"
"x || y"
"x ? y : z"
"x..y"
"x = y", "x += y", "x –= y"

Phần mô tả văn bản của các phép toán logic

Trong văn bản này, câu lệnh của các phép toán logic sẽ được mô tả về mặt toán học ở dạng sau đây:

```
if( điều kiện 0 )
   câu lệnh 0
else if( điều kiện 1 )
   câu lệnh 1
...
else /* ghi chú thông tin điều kiện còn lại */
   câu lệnh n
```

có thể được mô tả theo cách thức sau:

... như sau / ... những điều sau áp dụng:

- If điều kiện 0, câu lệnh 0
- Nếu không phải, if điều kiện 1, câu lệnh 1
- ...
- Nếu không phải (ghi chú thông tin về điều kiện còn lại), câu lệnh n

Mỗi "If ... Nếu không phải, if ... Nếu không phải, ..." câu lệnh trong văn bản được đưa ra với "... như sau" hoặc "... Những điều sau đây được áp dụng" ngay sau "If ... ". Điều kiện cuối cùng của "If ... Nếu không phải, if ... Nếu không phải, ..." luôn là "Nếu không phải, ...". Ghép xen "If ... Nếu không phải, if ... Nếu không phải, ..." các câu lệnh có thể được nhận dạng bằng cách ghép "... Như sau" hoặc "... Những điều sau đây được áp dụng" với kết thúc "Nếu không phải, ...".

Trong văn bản này, câu lệnh của các phép toán logic sẽ được mô tả về mặt toán học ở dạng sau đây:

```
if( điều kiện 0a    &&    điều kiện 0b )
   câu lệnh 0
else if( điều kiện 1a    ||    điều kiện 1b )
   câu lệnh 1
...
else
   câu lệnh n
```

có thể được mô tả theo cách thức sau:

... như sau / ... những điều sau áp dụng:

- Nếu tất cả điều kiện sau đây là đúng, câu lệnh 0:
  - điều kiện 0a

- điều kiện 0b

- Nếu không phải, nếu một hoặc nhiều điều kiện sau đây là đúng, câu lệnh 1:
  - điều kiện 1a
  - điều kiện 1b
- ...
- Nếu không phải, câu lệnh n

Trong văn bản này, câu lệnh của các phép toán logic sẽ được mô tả về mặt toán học ở dạng sau đây:

if(điều kiện 0)
  câu lệnh 0
if(điều kiện 1)
  câu lệnh 1

có thể được mô tả theo cách thức sau:

khi điều kiện 0, câu lệnh 0

khi điều kiện 1, câu lệnh 1.

Mặc dù các phương án của sáng chế đã được mô tả về cơ bản dựa trên mã hóa viđeo, lưu ý rằng các phương án về hệ thống mã 10, bộ mã hóa 20 và bộ giải mã 30 (và một cách tương ứng là hệ thống 10) và các phương án khác được mô tả ở đây có thể cũng được cấu hình đối với xử lý hoặc mã hóa ảnh tĩnh, tức là xử lý hoặc mã hóa ảnh riêng biệt độc lập với bất kỳ ảnh đứng trước hoặc liên tiếp như trong mã viđeo. Nói chung chỉ các bộ dự đoán liên đới 244 (bộ mã hóa) và 344 (bộ giải mã) có thể không khả dụng trong trường hợp việc mã hóa xử lý ảnh bị giới hạn ở ảnh đơn 17. Tất cả các chức năng khác (cũng được gọi là các công cụ hoặc các kỹ thuật) của bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 có thể cũng được sử dụng để xử lý ảnh tĩnh, ví dụ tính toán phần dư 204/304, biến đổi 206, lượng tử hóa 208, lượng tử hóa ngược 210/310, biến đổi (ngược) 212/312, phân

chia 262/362, dự đoán nội bộ 254/354, và/hoặc lọc vòng 220, 320, và mã hóa entropy 270 và giải mã entropy 304.

Các phương án, ví dụ của bộ mã hóa 20 và bộ giải mã 30, và các chức năng được mô tả ở đây, ví dụ có viện dẫn tới bộ mã hóa 20 và bộ giải mã 30, có thể được thực hiện trong phần cứng, phần mềm, vi chương trình, hoặc bất kỳ kết hợp của chúng. Nếu được thực hiện trong phần mềm, các chức năng có thể được lưu trữ trong phương tiện đọc được bởi máy tính hoặc được truyền qua phương tiện truyền thông như là một hoặc nhiều lệnh hoặc mã và được thực thi bởi bộ xử lý dựa trên phần cứng. Phương tiện đọc được bởi máy tính có thể bao gồm phương tiện lưu trữ đọc được bởi máy tính, mà tương ứng với phương tiện hữu hình như phương tiện lưu trữ dữ liệu, hoặc phương tiện truyền thông bao gồm bất kỳ phương tiện mà hỗ trợ truyền chương trình máy tính từ nơi này tới nơi khác, ví dụ, theo giao thức truyền thông. Theo cách này, phương tiện đọc được bởi máy tính, nói chung có thể tương ứng với (1) phương tiện lưu trữ hữu hình đọc được bởi máy tính mà là bất biến hoặc (2) phương tiện truyền thông như tín hiệu hoặc sóng mang. Phương tiện lưu trữ dữ liệu có thể là bất kỳ phương tiện khả dụng mà có thể được truy nhập bởi một hoặc nhiều máy tính hoặc một hoặc nhiều bộ xử lý để gọi ra các lệnh, mã, và/hoặc các cấu trúc dữ liệu để thực hiện các kỹ thuật được mô tả trong sáng chế. Sản phẩm chương trình máy tính có thể bao gồm phương tiện đọc được bởi máy tính.

Bằng ví dụ không bị giới hạn, phương tiện lưu trữ đọc được bởi máy tính này có thể bao gồm RAM, ROM, EEPROM, CD-ROM hoặc bộ lưu trữ đĩa quang khác, bộ lưu trữ đĩa từ hoặc thiết bị lưu trữ từ khác, bộ nhớ chớp, hoặc bất kỳ phương tiện khác mà có thể được sử dụng để lưu trữ mã chương trình mong muốn dưới dạng của lệnh hoặc cấu trúc dữ liệu và có thể được truy nhập bởi máy tính. Ngoài ra, bất kỳ kết nối cũng được gọi một cách thích hợp là phương tiện đọc được bởi máy tính. Ví dụ, nếu các lệnh được truyền từ trang mạng, máy chủ hoặc nguồn từ xa khác bằng cách sử dụng cáp đồng trục, cáp sợi quang, cặp dây xoắn, đường dây thuê bao số (Digital Subscriber Line-DSL) hoặc các kỹ thuật vô tuyến như tia hồng ngoại, vô tuyến và viba, thì cáp đồng trục, cáp sợi quang,

cặp dây xoắn, DSL hoặc kỹ thuật không dây như tia hồng ngoại, vô tuyến và viba nằm trong định nghĩa của phương tiện. Tuy nhiên, sẽ được hiểu rằng phương tiện lưu trữ đọc được bởi máy tính và phương tiện lưu trữ dữ liệu không bao gồm các kết nối, sóng mang, tín hiệu, hoặc phương tiện chuyển tiếp khác, mà thay vì đó liên quan đến phương tiện lưu trữ hữu hình bất biến. Đĩa như được sử dụng ở đây, bao gồm đĩa nén (CD), đĩa laze, đĩa quang, đĩa đa năng số (DVD), đĩa mềm và đĩa Blu-ray, trong đó đĩa thường tái tạo dữ liệu bằng từ tính, trong khi đĩa tái tạo dữ liệu quang học bằng tia laze . Các kết hợp của bất kỳ phương tiện nêu trên cũng sẽ được nằm trong phạm vi của phương tiện đọc được bởi máy tính.

Các lệnh có thể được thực thi bởi một hoặc nhiều bộ xử lý, chẳng hạn như một hoặc nhiều bộ xử lý tín số (DSP), bộ vi xử lý đa năng, mạch tích hợp ứng dụng riêng (ASIC), mảng logic có thể lập trình dạng trường (FPGA) hoặc mạch logic tích hợp hoặc rời rạc tương đương khác . Do đó, thuật ngữ "bộ xử lý", như được sử dụng ở đây có thể đề cập đến bất kỳ cấu trúc nêu trên hoặc bất kỳ cấu trúc khác phù hợp để thực hiện các kỹ thuật được mô tả ở đây. Ngoài ra, trong một vài khía cạnh, chức năng được mô tả ở đây có thể được đề xuất trong phần cứng dành riêng và/hoặc các môđun phần mềm được cấu hình để mã hóa và giải mã, hoặc được tích hợp trong bộ mã hóa-giải mã được kết hợp. Ngoài ra, các kỹ thuật có thể đều được thực hiện trong một hoặc nhiều mạch hoặc các phần tử logic.

Các kỹ thuật của sáng chế có thể được thực hiện trong nhiều loại thiết bị hoặc cơ cấu, bao gồm bộ cầm tay không dây, mạch tích hợp (integrated circuit, IC) hoặc tập hợp IC (ví dụ, bộ chip). Các thành phần, môđun, hoặc bộ phận khác nhau được mô tả trong sáng chế để nhấn mạnh các khía cạnh chức năng của các thiết bị có cấu trúc để thực hiện các kỹ thuật được bộc lộ, nhưng không cần thiết được thực hiện bởi các bộ phần cứng khác nhau. Ngoài ra, như được mô tả nêu trên, các bộ phận khác nhau có thể được kết hợp trong bộ phần cứng mã hóa-giải mã hoặc được cung cấp bởi một tập hợp các bộ phần cứng tương tác, bao gồm một

hoặc nhiều bộ xử lý như được mô tả nêu trên, kết hợp với phần mềm và/hoặc vi chương trình phù hợp.

YÊU CẦU BẢO HỘ

1. Phương pháp dự đoán khối sắc độ, trong đó phương pháp này bao gồm:

phân tích từ dòng bit phần tử cú pháp thứ nhất,

xử lý, dựa trên việc khuôn dạng sắc độ của ảnh mà khối hiện tại thuộc về có phải là YUV 4:2:0 hay không và ít nhất phần tử cú pháp thứ nhất, các mẫu độ chói của khối hiện tại, để thu nhận các mẫu độ chói được lọc của khối hiện tại;

trong đó bước xử lý, dựa trên ít nhất phần tử cú pháp thứ nhất, các mẫu độ chói của khối hiện tại, bao gồm:

xử lý các mẫu độ chói của khối hiện tại bằng cách sử dụng tập hợp thứ nhất của các hệ số lọc khi giá trị của phần tử cú pháp thứ nhất là giá trị thứ nhất; hoặc

xử lý các mẫu độ chói của khối hiện tại bằng cách sử dụng tập hợp thứ hai của các hệ số lọc khi giá trị của phần tử cú pháp thứ nhất là giá trị thứ hai; và

thu nhận, dựa trên các mẫu độ chói được lọc, giá trị dự đoán của các mẫu sắc độ của khối hiện tại.

2. Phương pháp theo điểm 1, trong đó phần tử cú pháp thứ nhất là sps_chroma_vertical_collocated_flag.

3. Phương pháp theo điểm 2,, trong đó bước xử lý các mẫu độ chói của khối hiện tại bằng cách sử dụng tập hợp thứ nhất của các hệ số lọc khi giá trị của phần tử cú pháp thứ nhất là giá trị thứ nhất, bao gồm

khi ít nhất sps_chroma_vertical_collocated_flag bằng 1, các mẫu độ chói được lọc được thu nhận như sau:

pDsY[x][y] = (F3[1][0]*pY[SubWidthC*x][SubHeightC*y–1] +
F3[0][1]*pY[SubWidthC*x–1][SubHeightC*y] +
F3[1][1]*pY[SubWidthC*x][SubHeightC*y] +
F3[2][1]*pY[SubWidthC*x + 1][SubHeightC*y] +
F3[1][2]*pY[SubWidthC*x][SubHeightC*y + 1] + 4) >> 3

trong đó tập hợp thứ nhất của các hệ số lọc bao gồm F3[1][0] = 1, F3[0][1] = 1, F3[1][1] = 4, F3[2][1] = 1, F3[1][2] = 1, trong đó pY[x][y] biểu diễn các mẫu độ chói, pDsY[x][y] biểu diễn các mẫu độ chói được lọc, SubWidthC là độ rộng của khối hiện tại; và SubHeightC là độ cao của khối hiện tại.

4. Phương pháp theo điểm 2,, trong đó bước xử lý các mẫu độ chói của khối hiện tại bằng cách sử dụng tập hợp thứ hai của các hệ số lọc khi giá trị của phần tử cú pháp thứ nhất là giá trị thứ hai, bao gồm

khi ít nhất sps_chroma_vertical_collocated_flag bằng 0, các mẫu độ chói được lọc được thu nhận như sau:

pDsY[x][y]=(F4[0][1]*pY[SubWidthC*x–1][SubHeightC*y] +
F4[0][2]*pY[SubWidthC*x–1][SubHeightC*y+1] +
F4[1][1]*pY[SubWidthC*x][SubHeightC*y] +
F4[1][2]*pY[SubWidthC*x][SubHeightC*y + 1] +
F4[2][1]*pY[SubWidthC*x + 1][SubHeightC*y] +
F4[2][2]*pY[SubWidthC*x + 1][SubHeightC*y + 1] + 4) >> 3

trong đó, tập hợp thứ hai của các hệ số lọc bao gồm F4[0][1] = 1, F4[0][2] = 1, F4[1][1] = 2, F4[1][2] = 2, F4[2][1] = 1, F4[2][2] = 1; trong đó pY[x][y] biểu diễn các mẫu độ chói, pDsY[x][y] biểu diễn các mẫu độ chói được lọc, SubWidthC là độ rộng của khối hiện tại; và SubHeightC là độ cao của khối hiện tại.

5. Phương pháp theo điểm bất kỳ trong số các điểm 1 đến 4, trong đó bước thu nhận, dựa trên các mẫu độ chói được lọc, giá trị dự đoán của các mẫu sắc độ của khối hiện tại, bao gồm:

giá trị dự đoán của các mẫu sắc độ được thu nhận như sau:

predSamples[x][y] = Clip1(((pDsY[x][y]*a) >> k) + b)

trong đó predSamples[x][y] biểu diễn giá trị dự đoán của các mẫu sắc độ của khối hiện tại, pDsY[x][y] biểu diễn các mẫu độ chói được lọc, và x = 0..nTbW–1; y = 0..nTbH–1, trong đó nTbW chỉ báo độ rộng của khối biến đổi, và nTbH chỉ báo độ cao của khối biến đổi.

6. Phương pháp theo điểm bất kỳ trong số các điểm 1 đến 5, trong đó phương pháp này còn bao gồm:

phân tích dòng bit để thu nhận phần tử cú pháp thứ hai, trong đó phần tử cú pháp thứ hai được sử dụng để chỉ báo rằng chế độ dự đoán nội bộ được sử dụng

để giải mã hiện tại là một trong số chế độ dự đoán nội bộ INTRA_LT_CCLM, INTRA_L_CCLM hoặc INTRA_T_CCLM.

7. Phương pháp theo điểm bất kỳ trong số các điểm 1 đến 6, trong đó:

phương pháp này được sử dụng để giải mã khối ảnh hiện tại trong chuỗi viđeo, trong đó khối ảnh hiện tại bao gồm khối độ chói thứ nhất và khối sắc độ hiện tại, và ảnh trong chuỗi viđeo nằm trong khuôn dạng 4:2:0.

8. Phương pháp theo điểm bất kỳ trong số các điểm 1 đến 7, trong đó bước phân tích từ dòng bit phần tử cú pháp thứ nhất, bao gồm:

phân tích tham số tập hợp tham số chuỗi (SPS-sequence parameter set) trong dòng bit, để thu nhận phần tử cú pháp thứ nhất.

9. Phương pháp theo điểm bất kỳ trong số các điểm 1 đến 8, trong đó phương pháp này còn bao gồm:

áp dụng tập hợp của các hệ số lọc được xác định từ tập hợp thứ nhất của các hệ số lọc và tập hợp thứ hai của các hệ số lọc dựa trên phần tử cú pháp thứ nhất, tới các mẫu độ chói trong vị trí được lựa chọn lân cận với khối hiện tại,

trong đó các mẫu độ chói trong vị trí được lựa chọn lân cận với khối hiện tại là các mẫu của một hoặc một vài cột liền kề với phía trái của khối hiện tại và/hoặc các mẫu của một hoặc một vài hàng liền kề với phía trên cùng của khối hiện tại.

10. Phương pháp theo điểm bất kỳ trong số các điểm 1 đến 9, trong đó phương pháp này còn bao gồm:

nếu cả SubWidthC và SubHeightC bằng 1, các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống của khối hiện tại được thu nhận như sau:

$$pDstY[x][y] = pY[x][y]$$

trong đó pY[x][y] biểu diễn các mẫu độ chói được sắp xếp cùng vị trí, pDsY[x][y] biểu diễn các mẫu độ chói được sắp xếp cùng vị trí được lấy mẫu xuống, SubWidthC là độ rộng của khối hiện tại; và SubHeightC là độ cao của khối hiện tại.

11. Phương pháp theo điểm 10, trong đó các mẫu độ chói được sắp xếp cùng vị trí pY[x][y] được thiết lập bằng các mẫu độ chói được khôi phục trước xử lý lọc giải khối tại các vị trí (xTbY + x, yTbY + y);

trong đó x = 0..nTbW*SubWidthC–1, y = 0..nTbH*SubHeightC–1, (xTbY, yTbY) chỉ báo vị trí độ chói hiện tại, trong đó nTbW chỉ báo độ rộng của khối biến đổi, và nTbH chỉ báo độ cao của khối biến đổi.

12. Bộ giải mã (30) bao gồm mạch xử lý để thực hiện phương pháp theo điểm bất kỳ trong số các điểm 1 đến 11.

13. Phương tiện bất biến đọc được bởi máy tính mà mang mã chương trình mà, khi được thực thi bởi thiết bị máy tính, làm cho thiết bị máy tính thực hiện phương pháp theo điểm bất kỳ trong số các điểm 1 đến 11.

14. Thiết bị mã, trong đó thiết bị mã bao gồm bộ giải mã hoặc bộ mã hóa, bao gồm:

một hoặc nhiều bộ xử lý; và

phương tiện lưu trữ bất biến đọc được bởi máy tính được ghép nối tới các bộ xử lý và lưu trữ chương trình để thực thi bởi các bộ xử lý, trong đó chương trình, khi được thực thi bởi các bộ xử lý, cấu hình bộ giải mã hoặc bộ mã hóa để thực hiện phương pháp theo điểm bất kỳ trong số các điểm 1 đến 11.

15. Bộ giải mã, bao gồm:

một hoặc nhiều bộ xử lý; và

phương tiện lưu trữ bất biến đọc được bởi máy tính được ghép nối tới các bộ xử lý và lưu trữ chương trình để thực thi bởi các bộ xử lý, trong đó chương trình, khi được thực thi bởi các bộ xử lý, cấu hình bộ giải mã này để thực hiện phương pháp theo điểm bất kỳ trong số các điểm 1 đến 11.

16. Thiết bị giải mã dữ liệu viđeo, bao gồm:

bộ lưu trữ nhớ bất biến, được tạo cấu hình để lưu trữ dữ liệu viđeo dưới dạng của dòng bit; và

bộ giải mã viđeo, được tạo cấu hình để thực hiện phương pháp theo điểm bất kỳ trong số các điểm từ 1 đến 11.

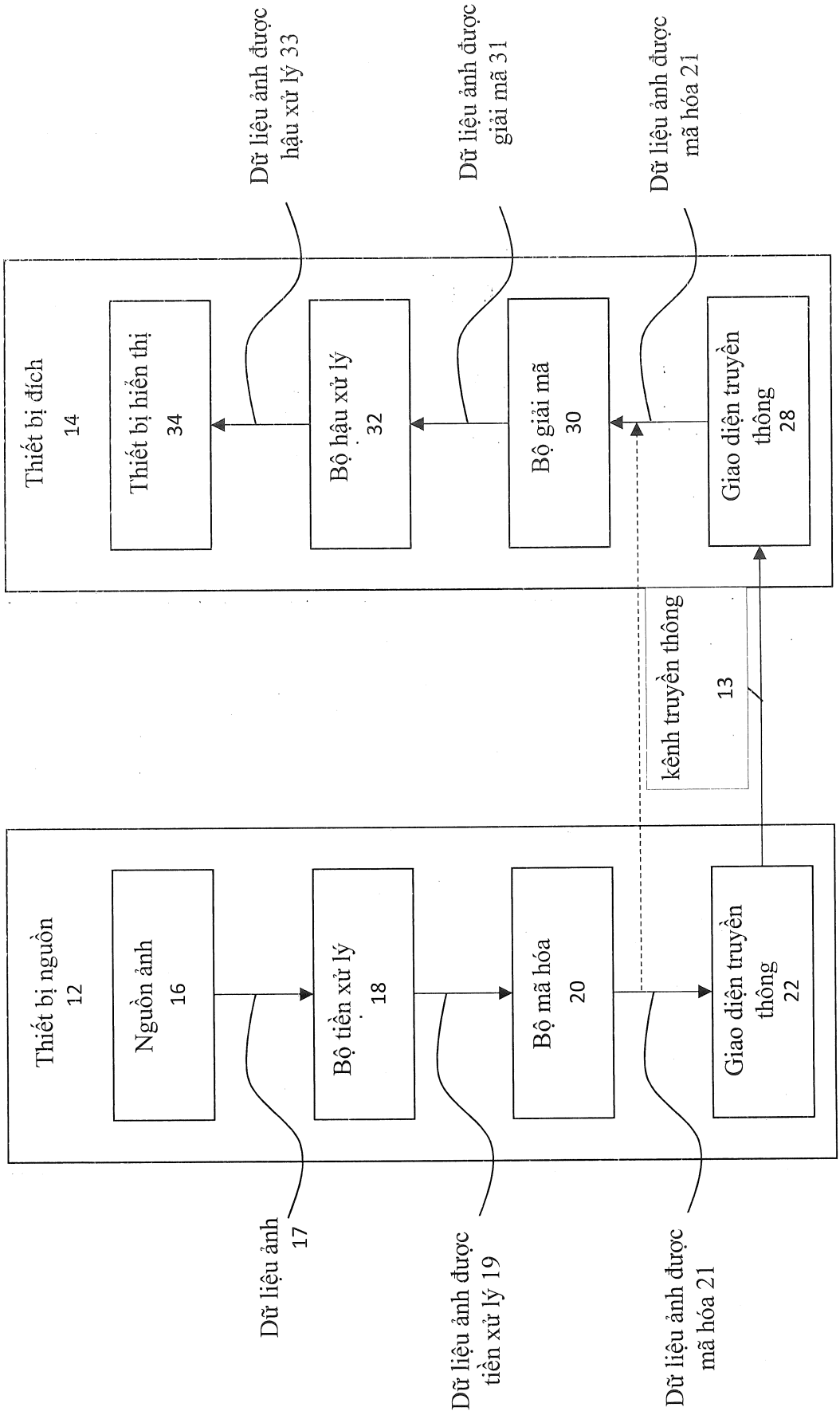

Thiết bị đích
14
Thiết bị hiển thị
34
Bộ hậu xử lý
32
Bộ giải mã
30
Giao diện truyền thông
28
Dữ liệu ảnh được hậu xử lý 33
Dữ liệu ảnh được giải mã 31
Dữ liệu ảnh được mã hóa 21
kênh truyền thông
13
Thiết bị nguồn
12
Nguồn ảnh
16
Bộ tiền xử lý
18
Bộ mã hóa
20
Giao diện truyền thông
22
Dữ liệu ảnh
17
Dữ liệu ảnh được tiền xử lý 19
Dữ liệu ảnh được mã hóa 21
10

FIG. 1A

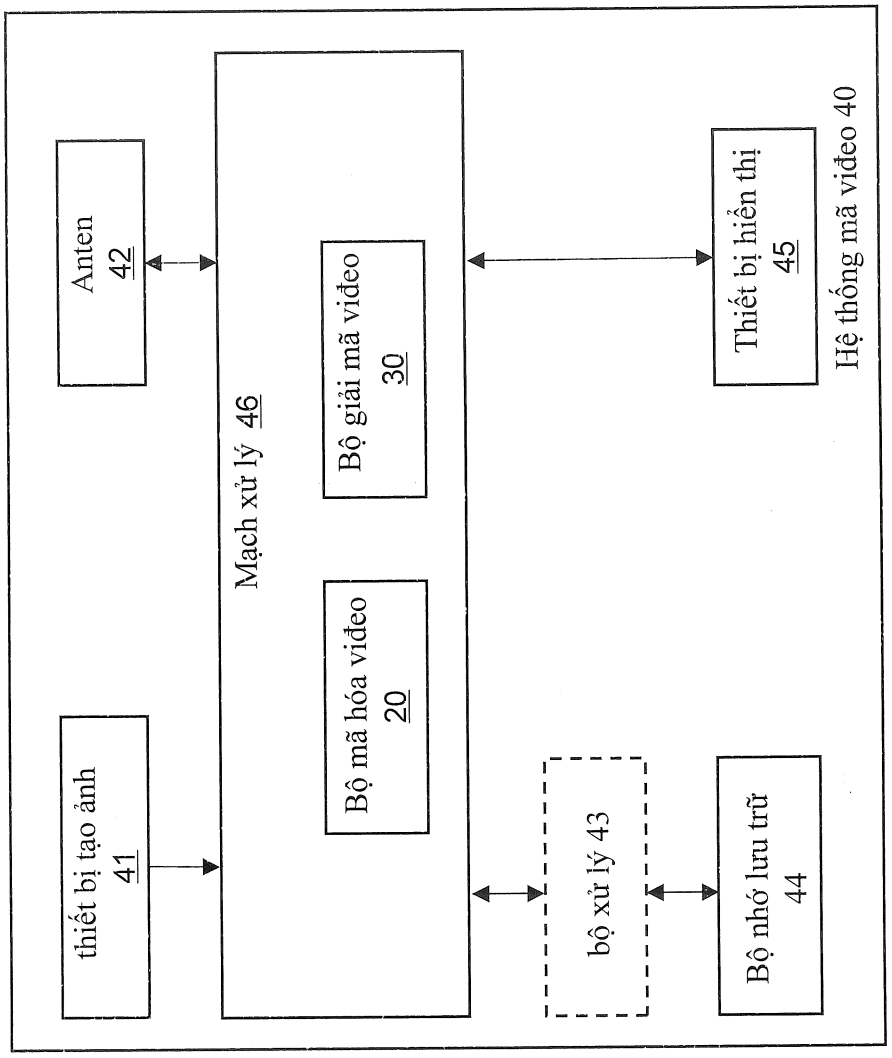

Hệ thống mã video 40
Anten 42
Mạch xử lý 46
Bộ giải mã video 30
Bộ mã hóa video 20
Thiết bị hiển thị 45
thiết bị tạo ảnh 41
bộ xử lý 43
Bộ nhớ lưu trữ 44

FIG. 1B

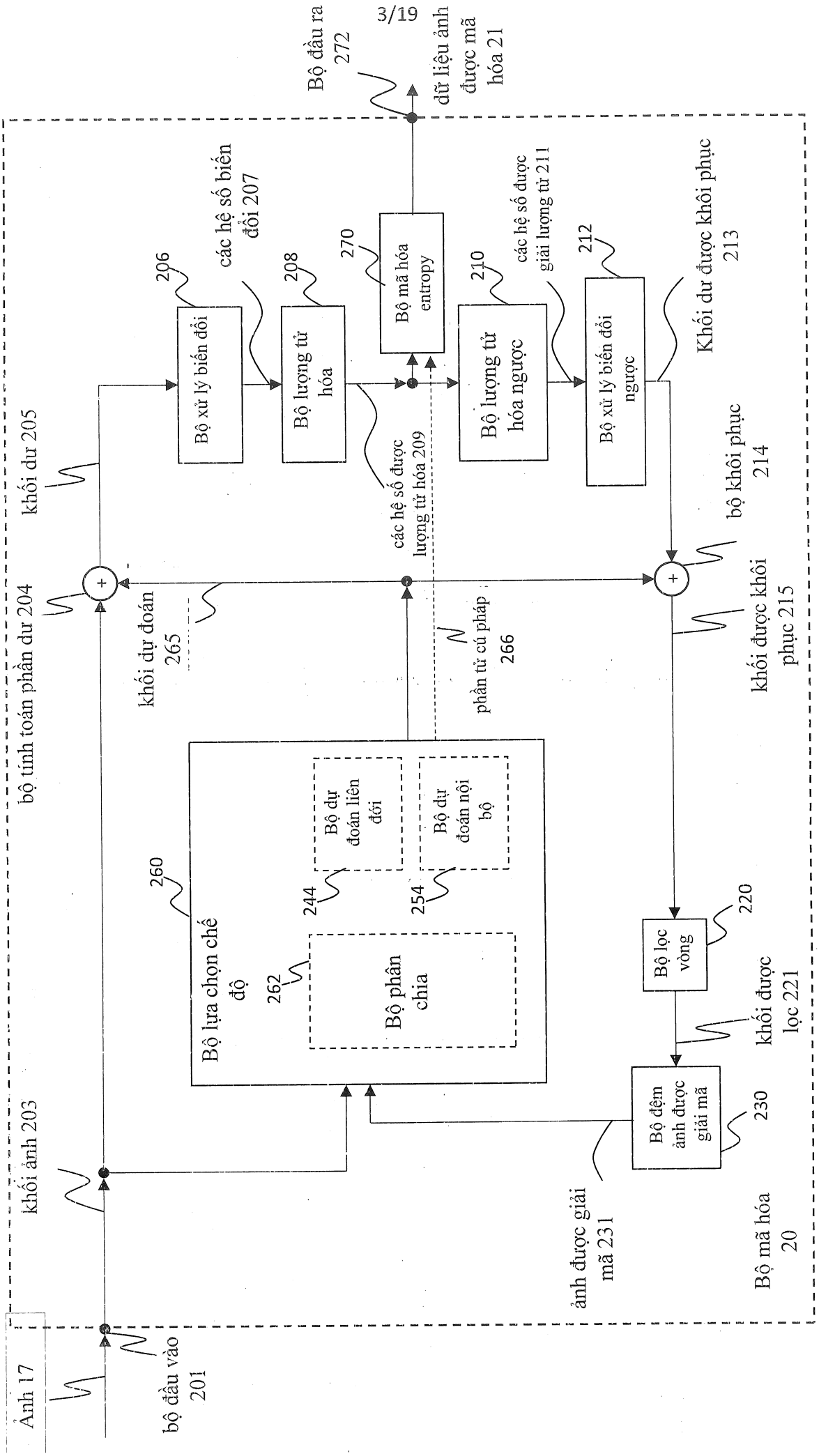

Ảnh 17
bộ đầu vào 201
bộ tính toán phần dư 204
khối ảnh 203
khối dư 205
khối dự đoán 265
phần tử cú pháp 266
Bộ lựa chọn chế độ 260
Bộ phân chia 262
Bộ dự đoán liên đới 244
Bộ dự đoán nội bộ 254
Bộ xử lý biến đổi 206
các hệ số biến đổi 207
Bộ lượng tử hóa 208
các hệ số được lượng tử hóa 209
Bộ mã hóa entropy 270
Bộ đầu ra 272
dữ liệu ảnh được mã hóa 21
Bộ lượng tử hóa ngược 210
các hệ số được giải lượng tử 211
Bộ xử lý biến đổi ngược 212
Khối dư được khôi phục 213
bộ khôi phục 214
khối được khôi phục 215
Bộ lọc vòng 220
khối được lọc 221
Bộ đệm ảnh được giải mã 230
ảnh được giải mã 231
Bộ mã hóa 20

FIG. 2

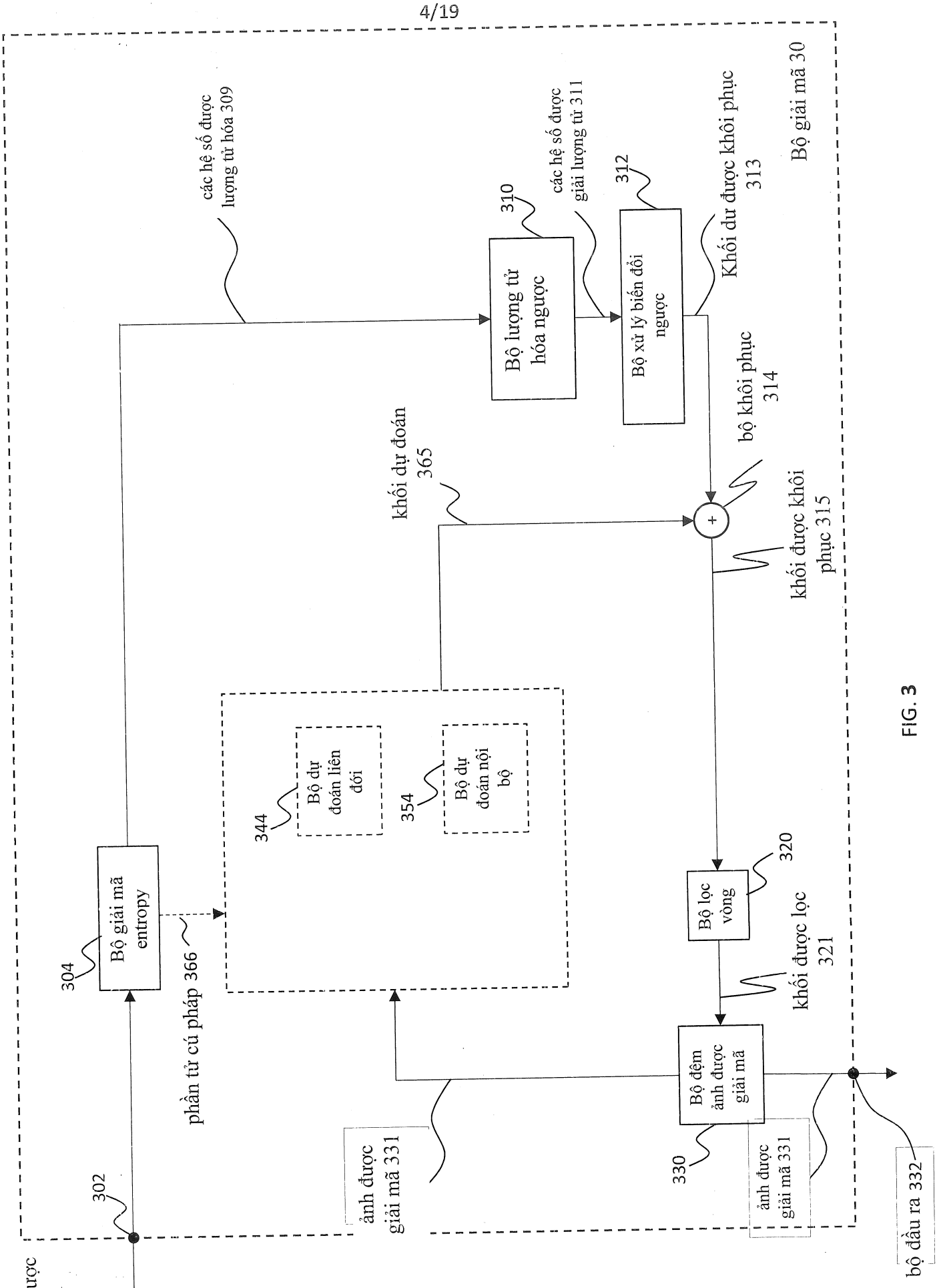

dữ liệu ảnh được mã hóa 21
302
Bộ giải mã entropy
304
phần tử cú pháp 366
các hệ số được lượng tử hóa 309
Bộ lượng tử hóa ngược
310
các hệ số được giải lượng tử 311
Bộ xử lý biến đổi ngược
312
Khối dư được khôi phục 313
Bộ giải mã 30
bộ khôi phục 314
khối dự đoán 365
khối được khôi phục 315
Bộ dự đoán liên đới
344
Bộ dự đoán nội bộ
354
Bộ lọc vòng
320
khối được lọc 321
Bộ đệm ảnh được giải mã
330
ảnh được giải mã 331
ảnh được giải mã 331
bộ đầu ra 332

FIG. 3

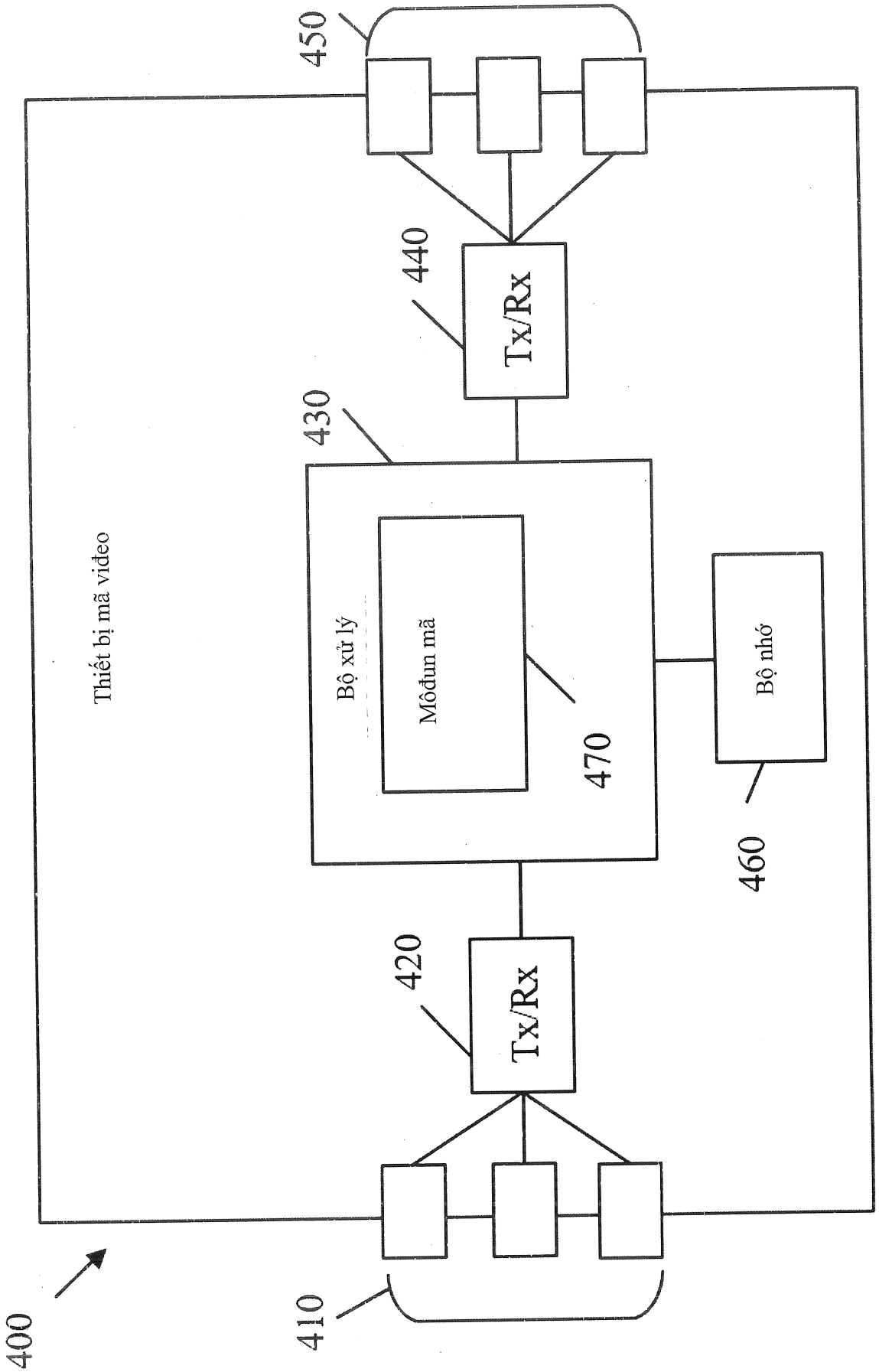

400
Thiết bị mã video
410
Cổng dòng xuống
420
Tx/Rx
430
Bộ xử lý
Môđun mã
470
460
Bộ nhớ
440
Tx/Rx
450
Cổng dòng lên

FIG. 4

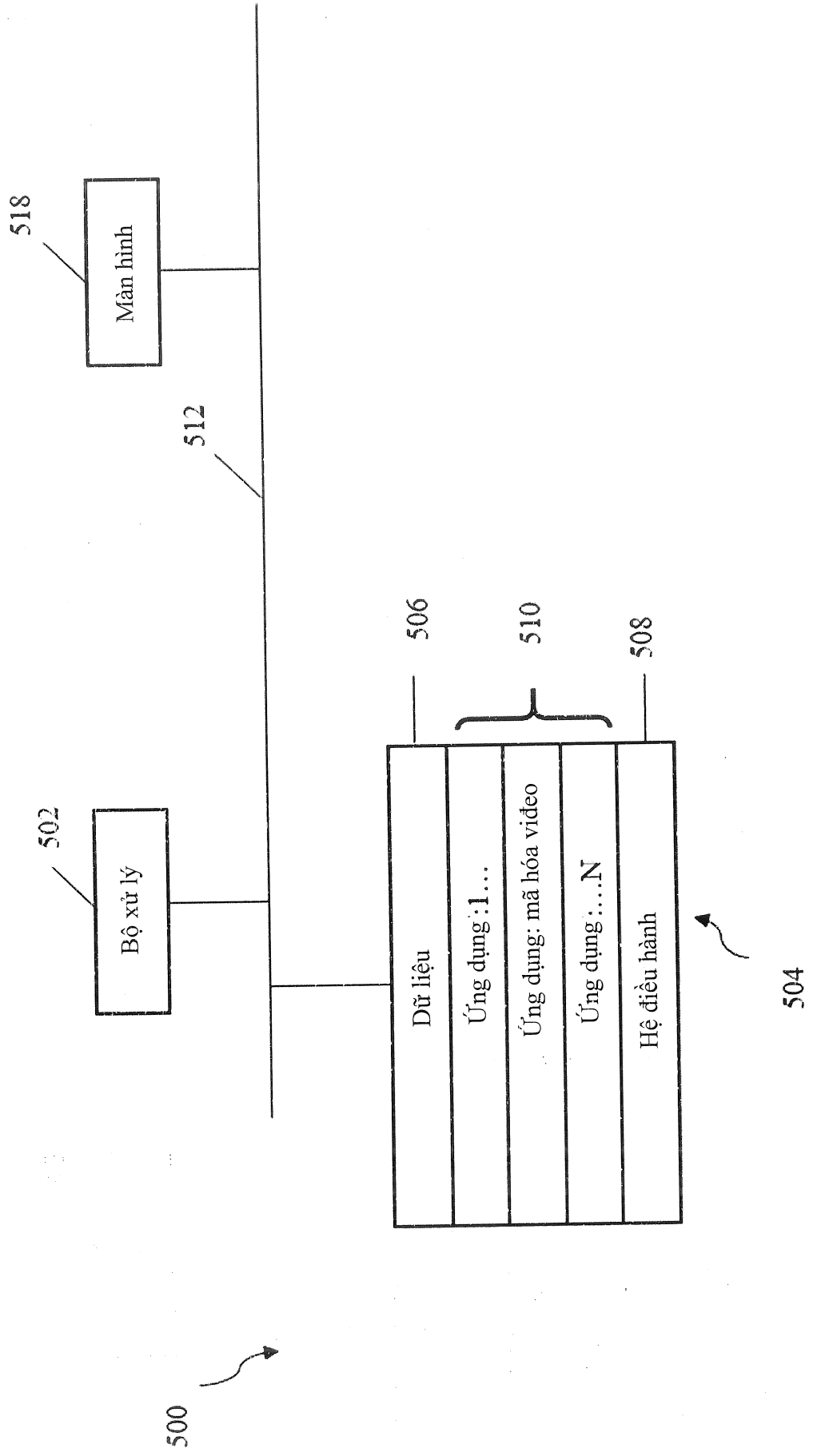

500
502
Bộ xử lý
504
506
508
510
512
518
Màn hình
Dữ liệu
Ứng dụng:1...
Ứng dụng: mã hóa video
Ứng dụng:...N
Hệ điều hành

FIG. 5

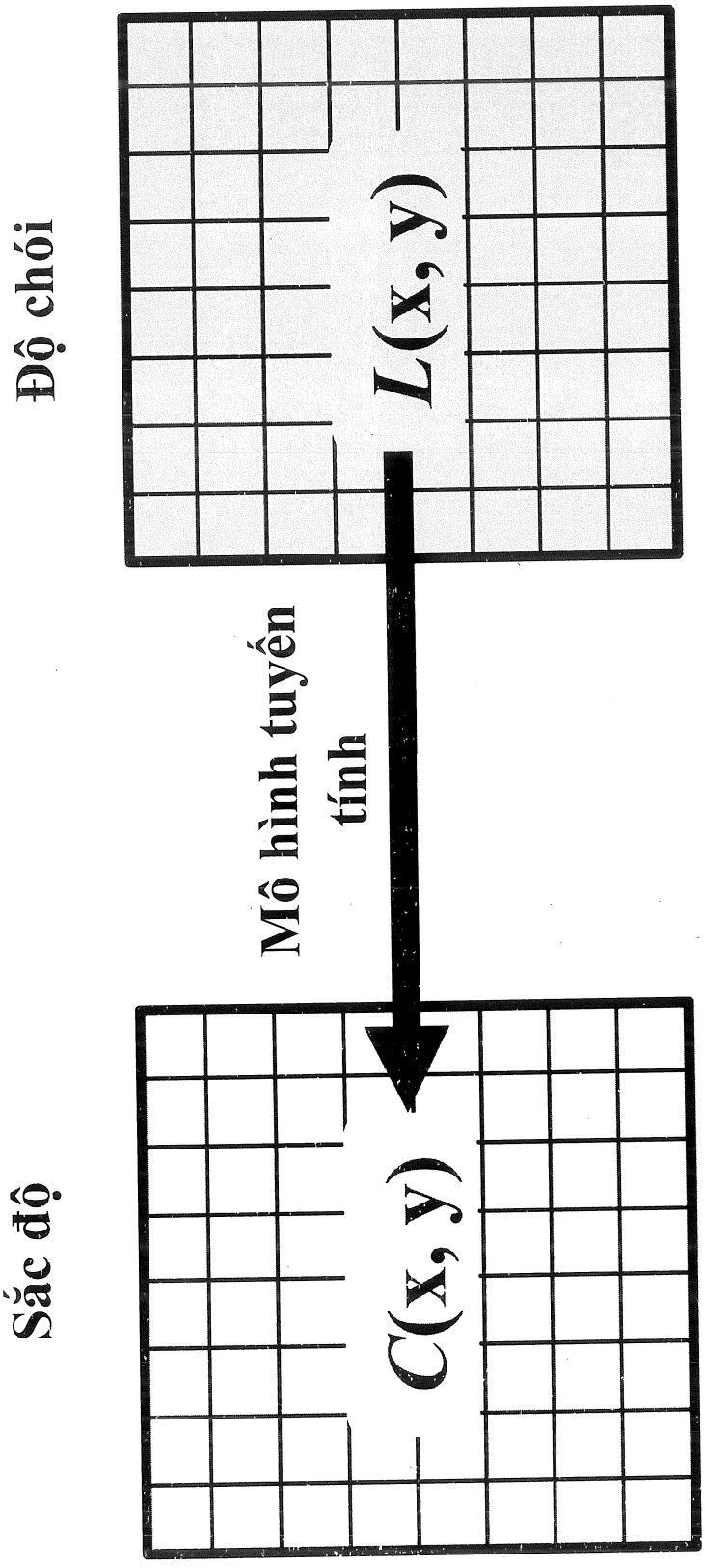


FIG. 6

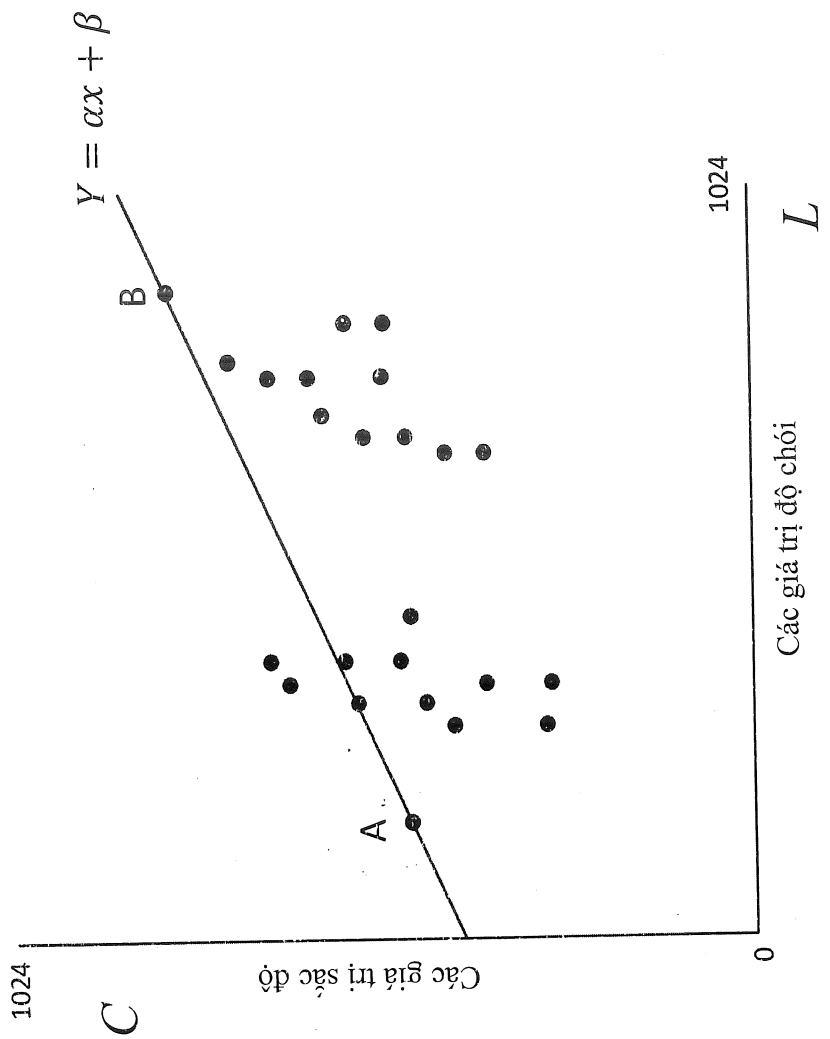

$Y = \alpha x + \beta$
B
A
1024
0
1024
L
Các giá trị độ chói
Các giá trị sắc độ
C

FIG. 7

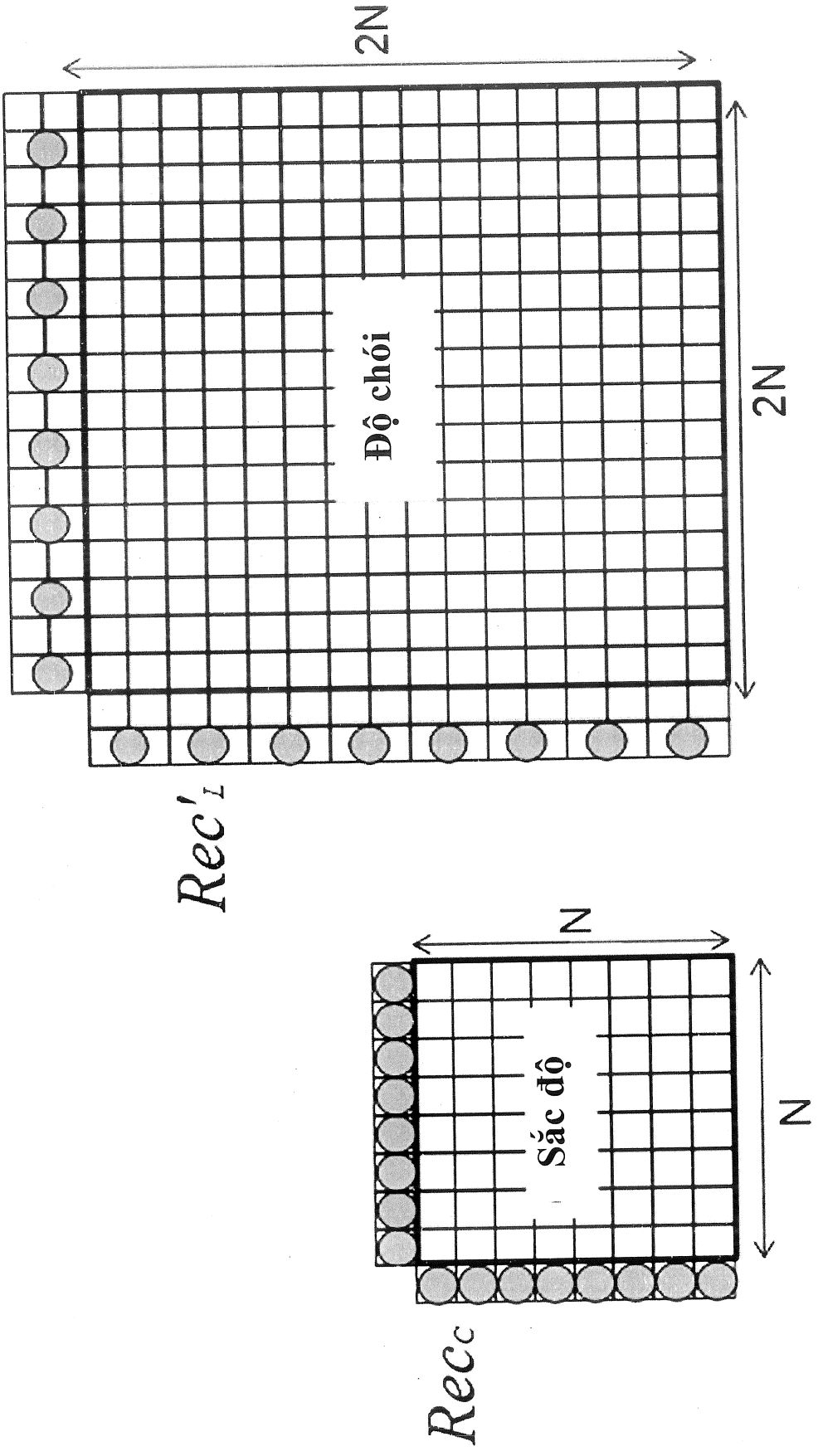


FIG. 8

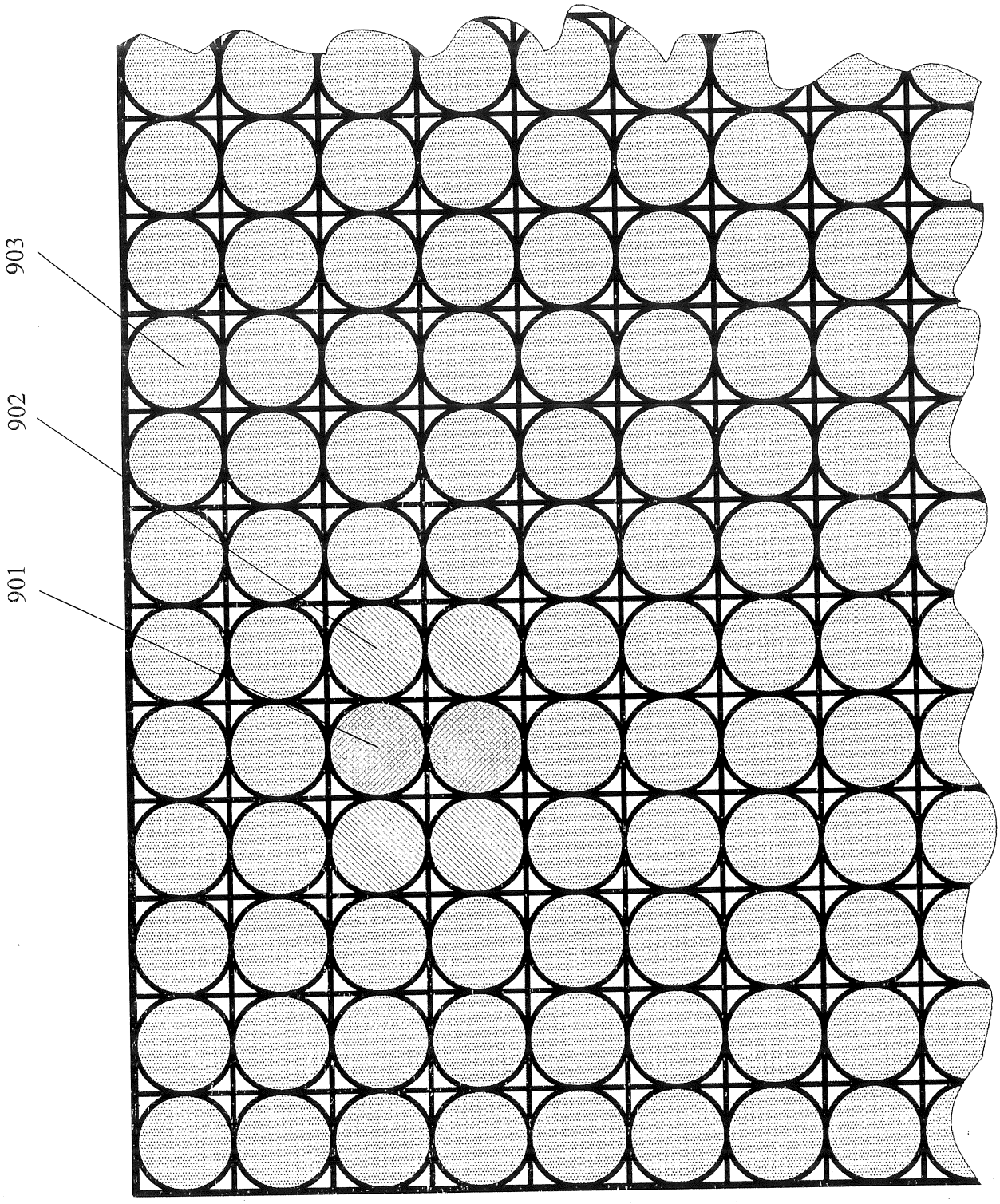
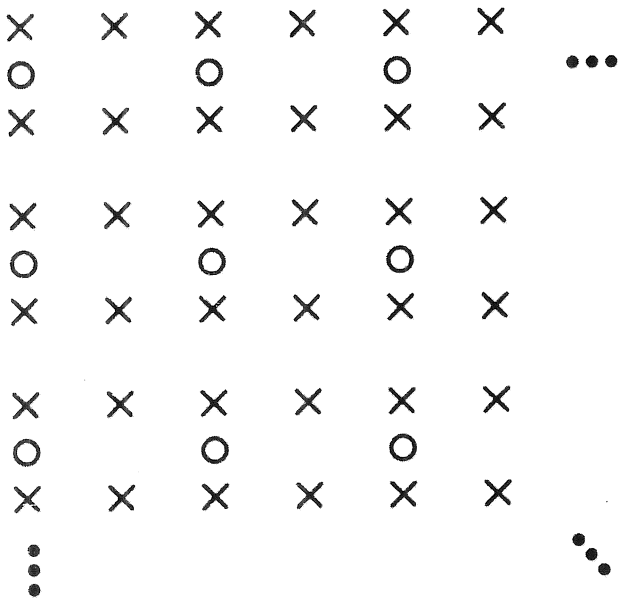


FIG. 9

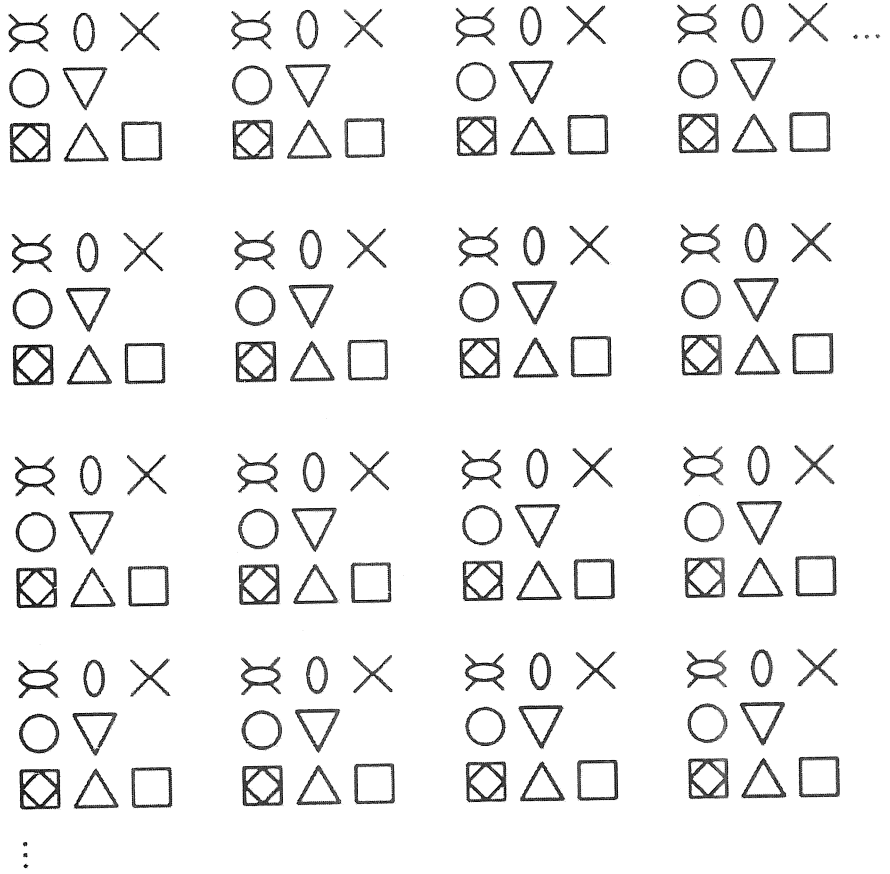


Chỉ dẫn:

× – Vị trí của mẫu độ chói

○ – Vị trí của mẫu sắc độ

FIG. 10A



Ý nghĩa của các ký tự:

các chỉ báo vị trí mẫu độ chói:

× = trường trên cùng mẫu độ chói

□ = trường dưới cùng mẫu độ chói

Các chỉ báo vị trí mẫu sắc độ, trong đó việc điền màu xám chỉ báo loại mẫu trường dưới cùng và việc không điền chỉ báo loại mẫu trường trên cùng

⬭ = loại mẫu sắc độ 2

0 = loại mẫu sắc độ 3

○ = loại mẫu sắc độ 0

▽ = loại mẫu sắc độ 1

◇ = loại mẫu sắc độ 4

△ = loại mẫu sắc độ 5

FIG. 10B

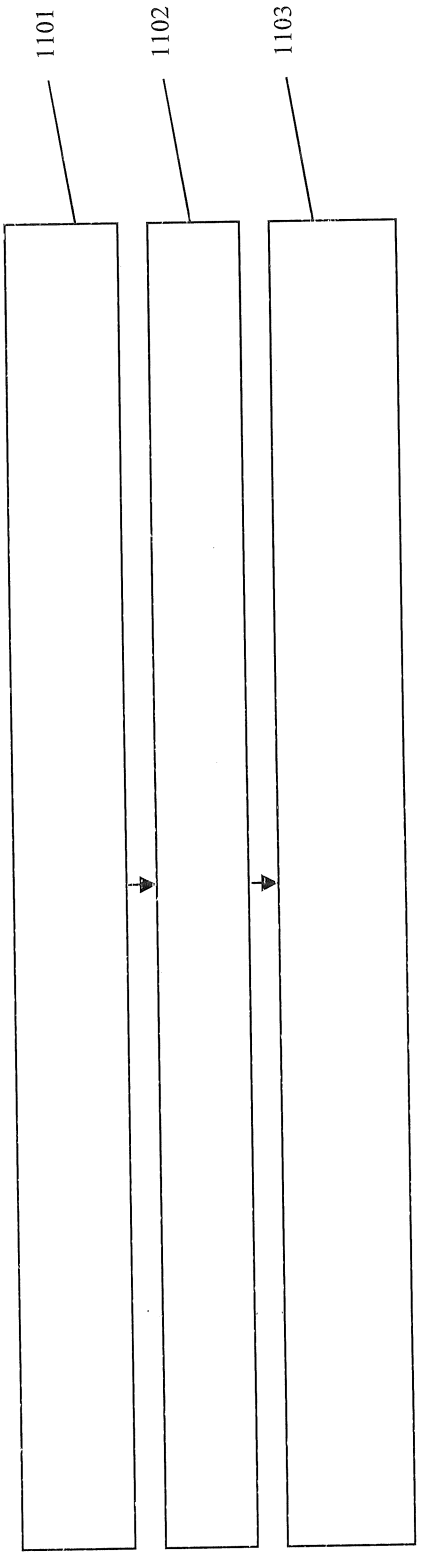


FIG. 11

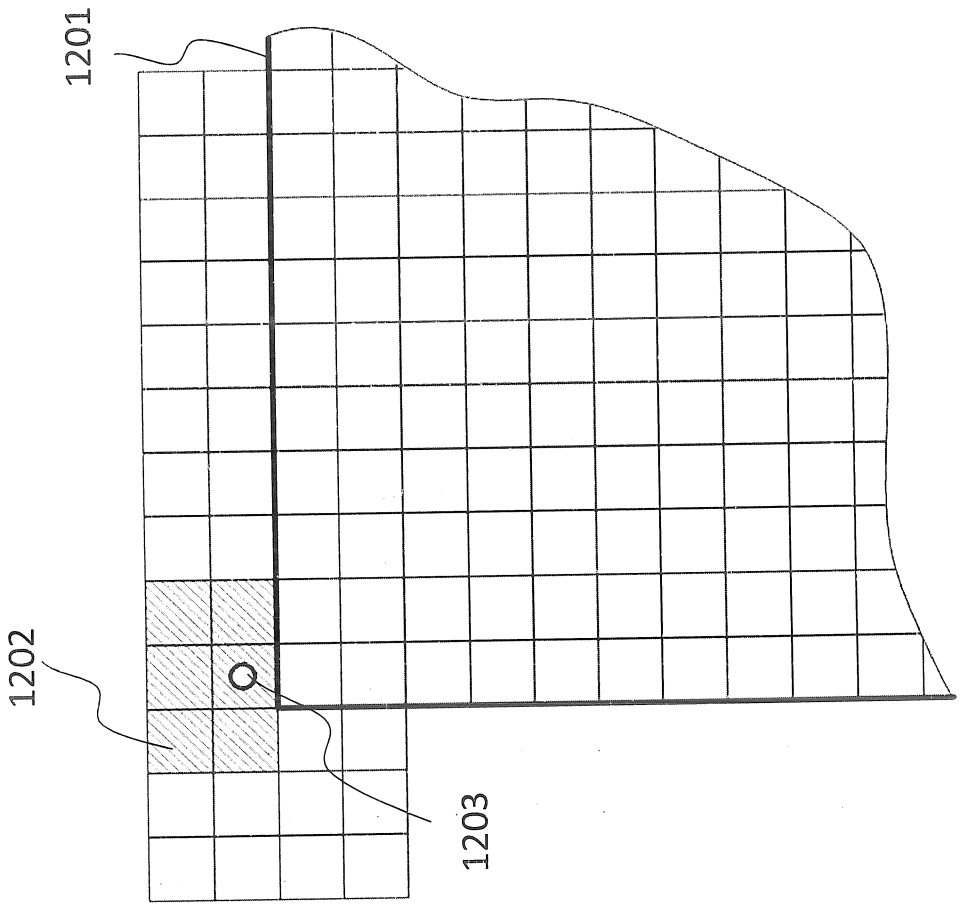


FIG. 12B

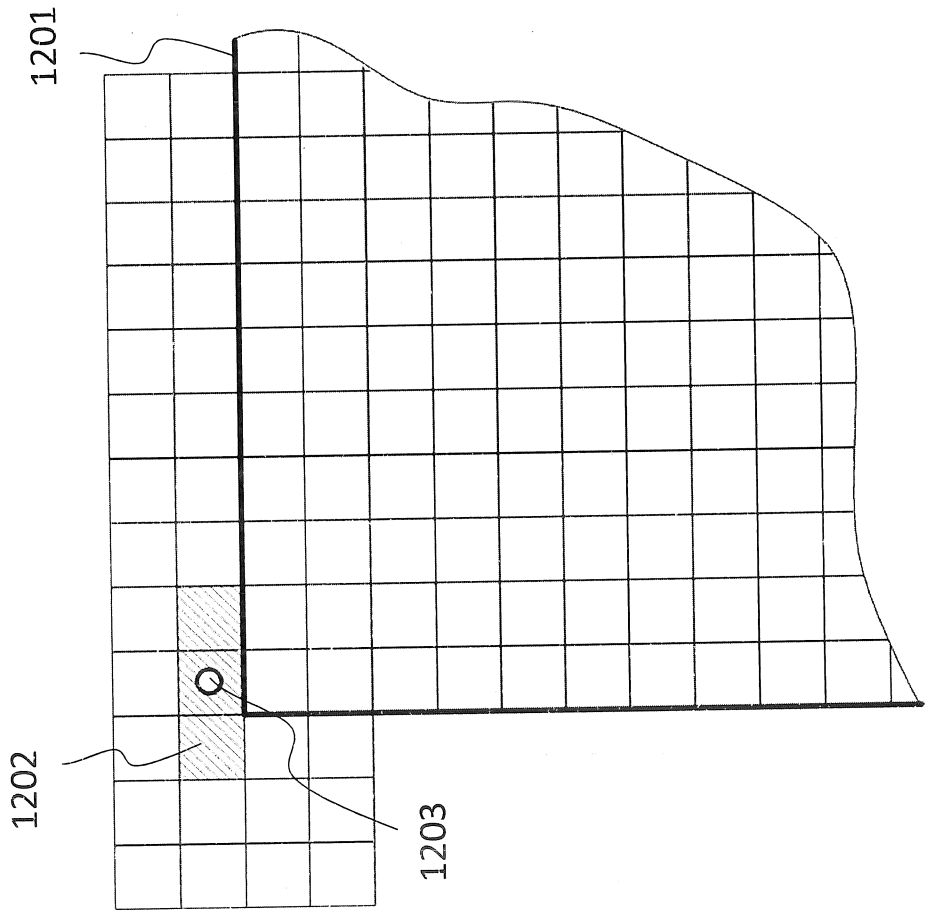


FIG. 12A

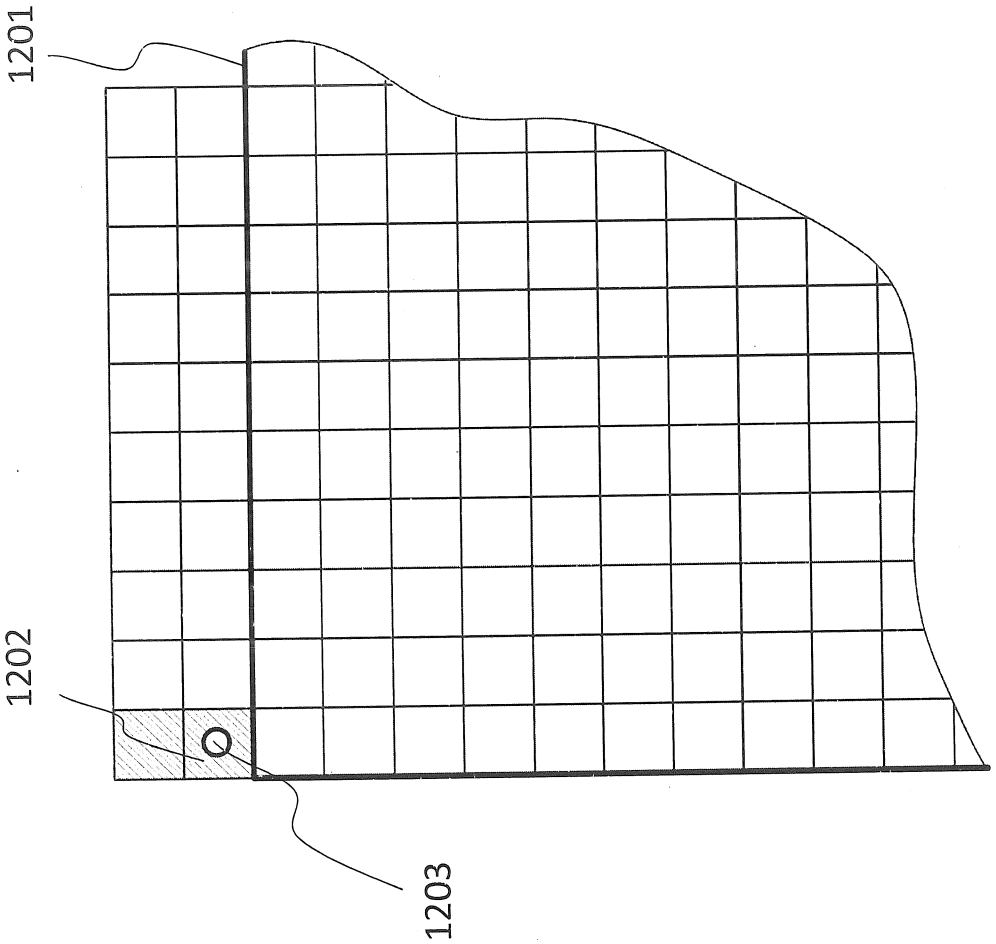


FIG. 12D

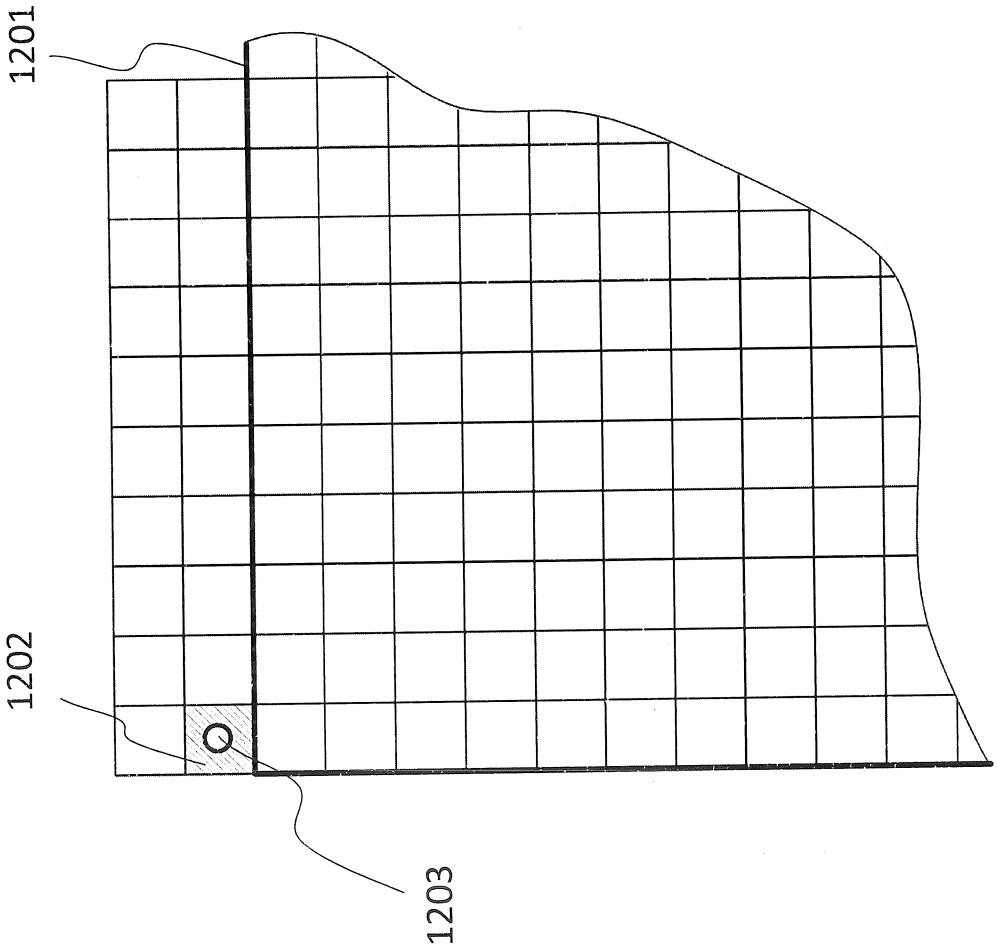


FIG. 12C

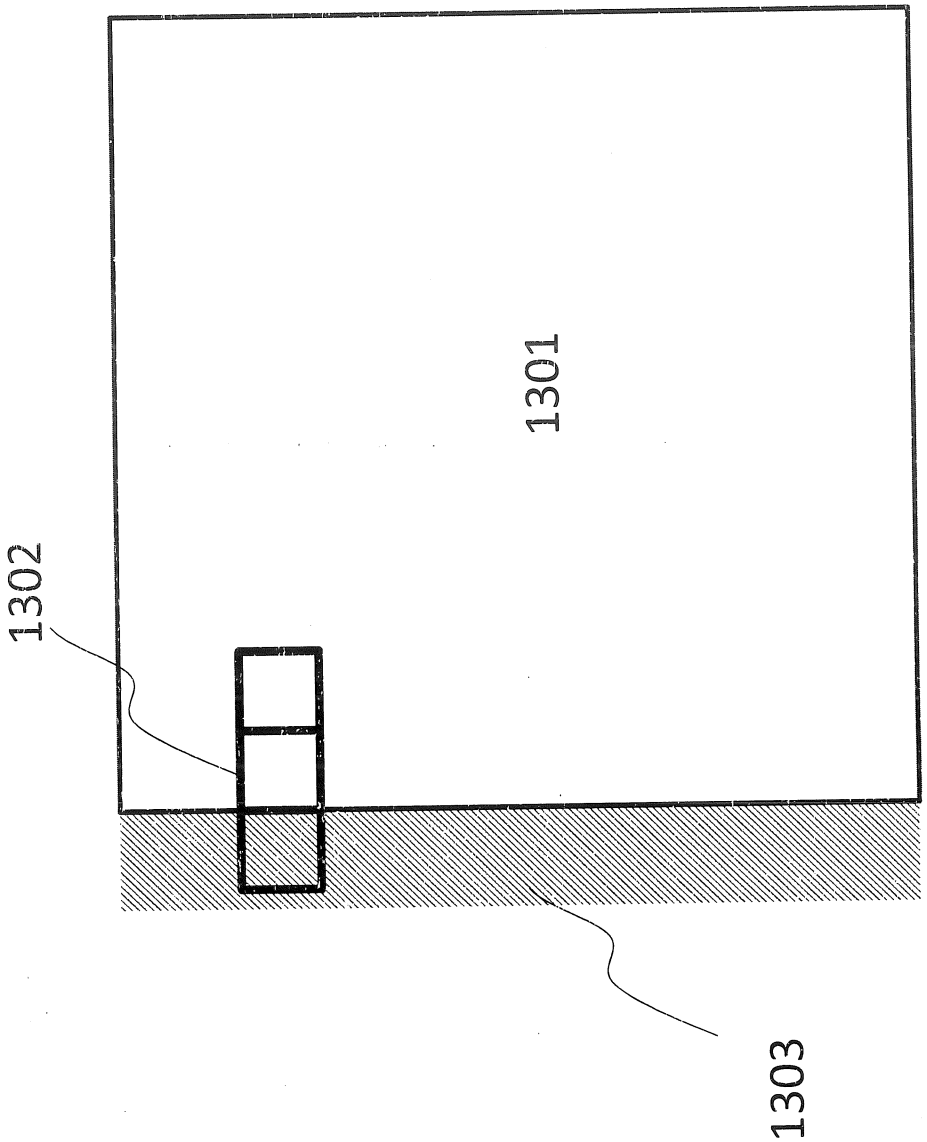


FIG. 13

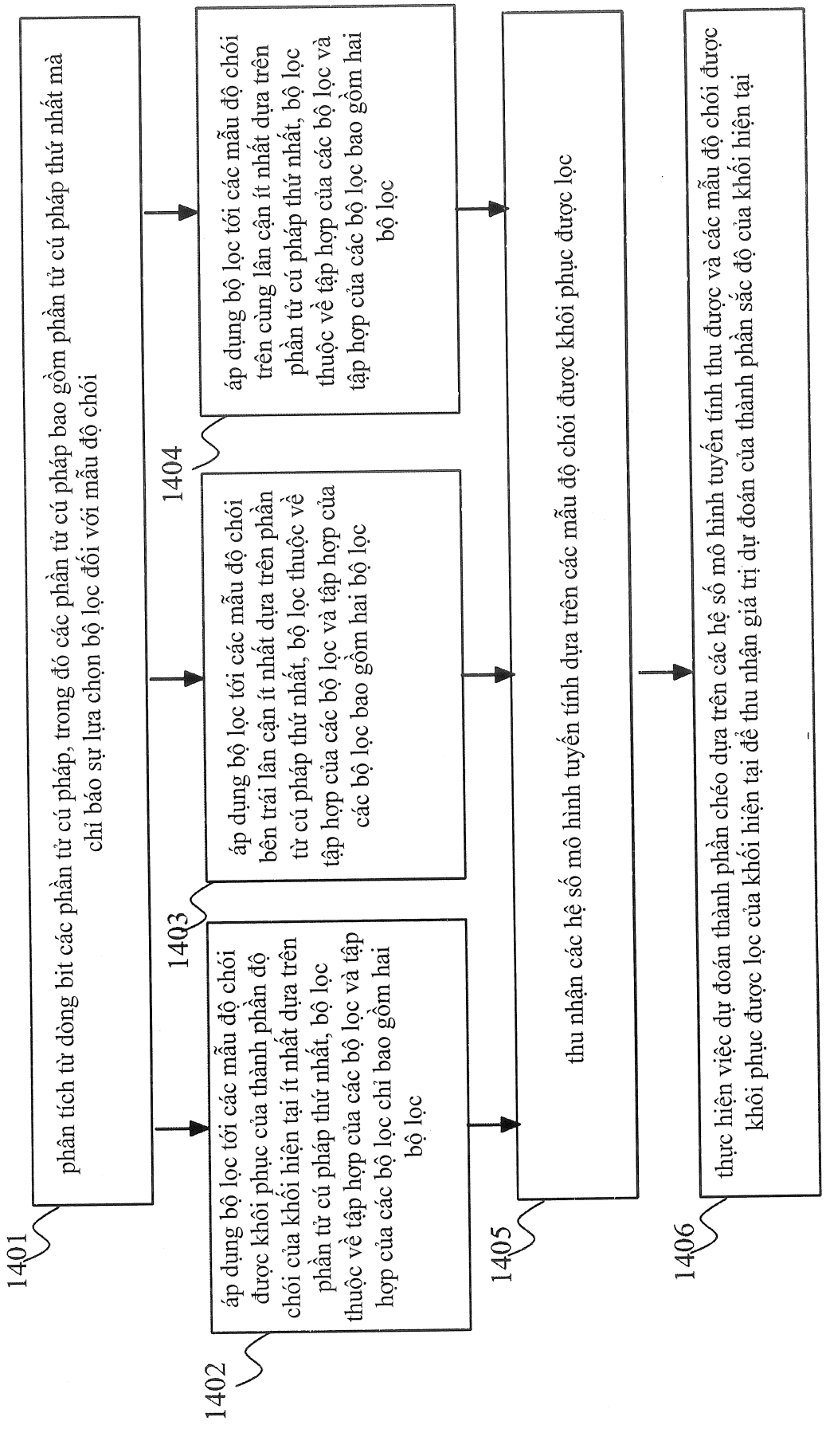

1400
phân tích từ dòng bit các phần tử cú pháp, trong đó các phần tử cú pháp bao gồm phần tử cú pháp thứ nhất mà chỉ báo sự lựa chọn bộ lọc đối với mẫu độ chói
1401
áp dụng bộ lọc tới các mẫu độ chói được khôi phục của thành phần độ chói của khối hiện tại ít nhất dựa trên phần tử cú pháp thứ nhất, bộ lọc thuộc về tập hợp của các bộ lọc và tập hợp của các bộ lọc chỉ bao gồm hai bộ lọc
1402
1403
áp dụng bộ lọc tới các mẫu độ chói bên trái lân cận ít nhất dựa trên phần từ cú pháp thứ nhất, bộ lọc thuộc về tập hợp của các bộ lọc và tập hợp của các bộ lọc bao gồm hai bộ lọc
1404
áp dụng bộ lọc tới các mẫu độ chói trên cùng lân cận ít nhất dựa trên phần tử cú pháp thứ nhất, bộ lọc thuộc về tập hợp của các bộ lọc và tập hợp của các bộ lọc bao gồm hai bộ lọc
thu nhận các hệ số mô hình tuyến tính dựa trên các mẫu độ chói được khôi phục được lọc
1405
thực hiện việc dự đoán thành phần chéo dựa trên các hệ số mô hình tuyến tính thu được và các mẫu độ chói được khôi phục được lọc của khối hiện tại để thu nhận giá trị dự đoán của thành phần sắc độ của khối hiện tại
1406

FIG. 14

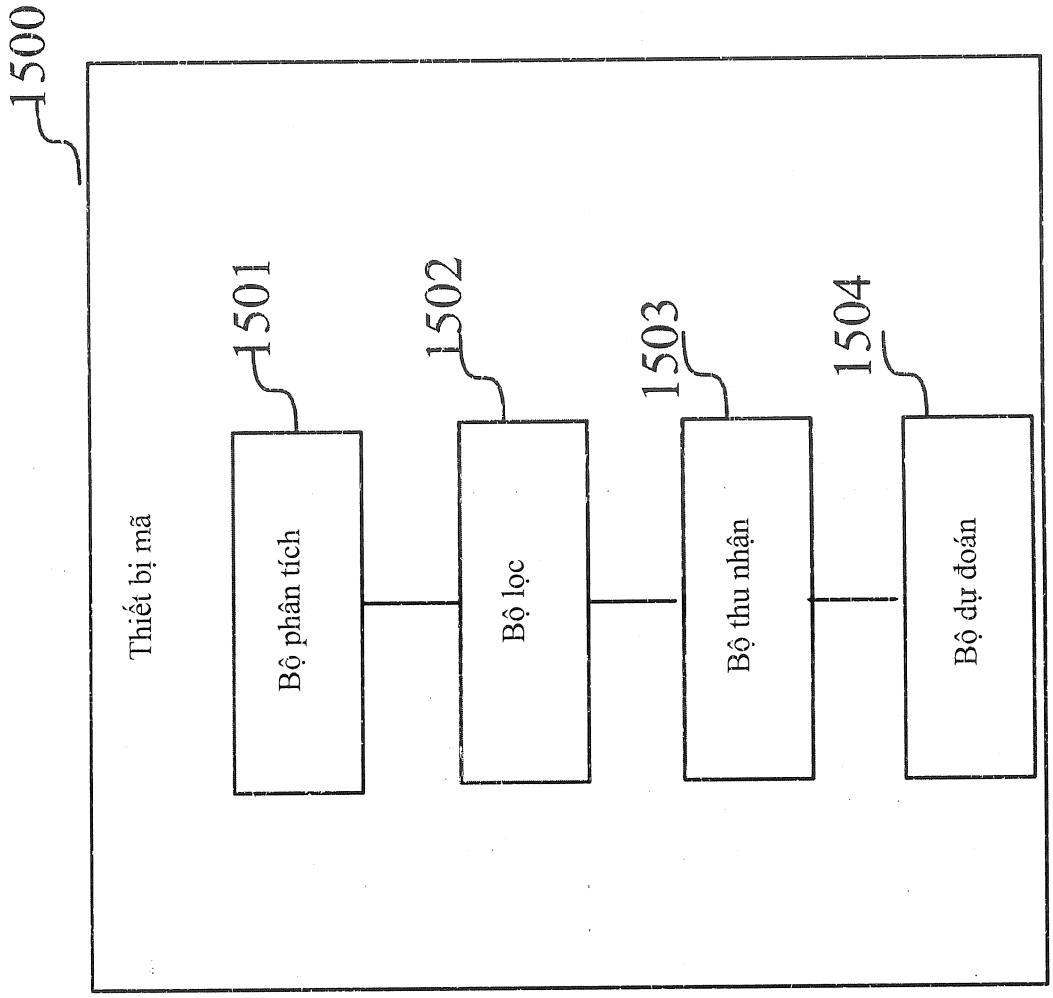

1500
Thiết bị mã
1501
Bộ phân tích
1502
Bộ lọc
1503
Bộ thu nhận
1504
Bộ dự đoán

FIG. 15

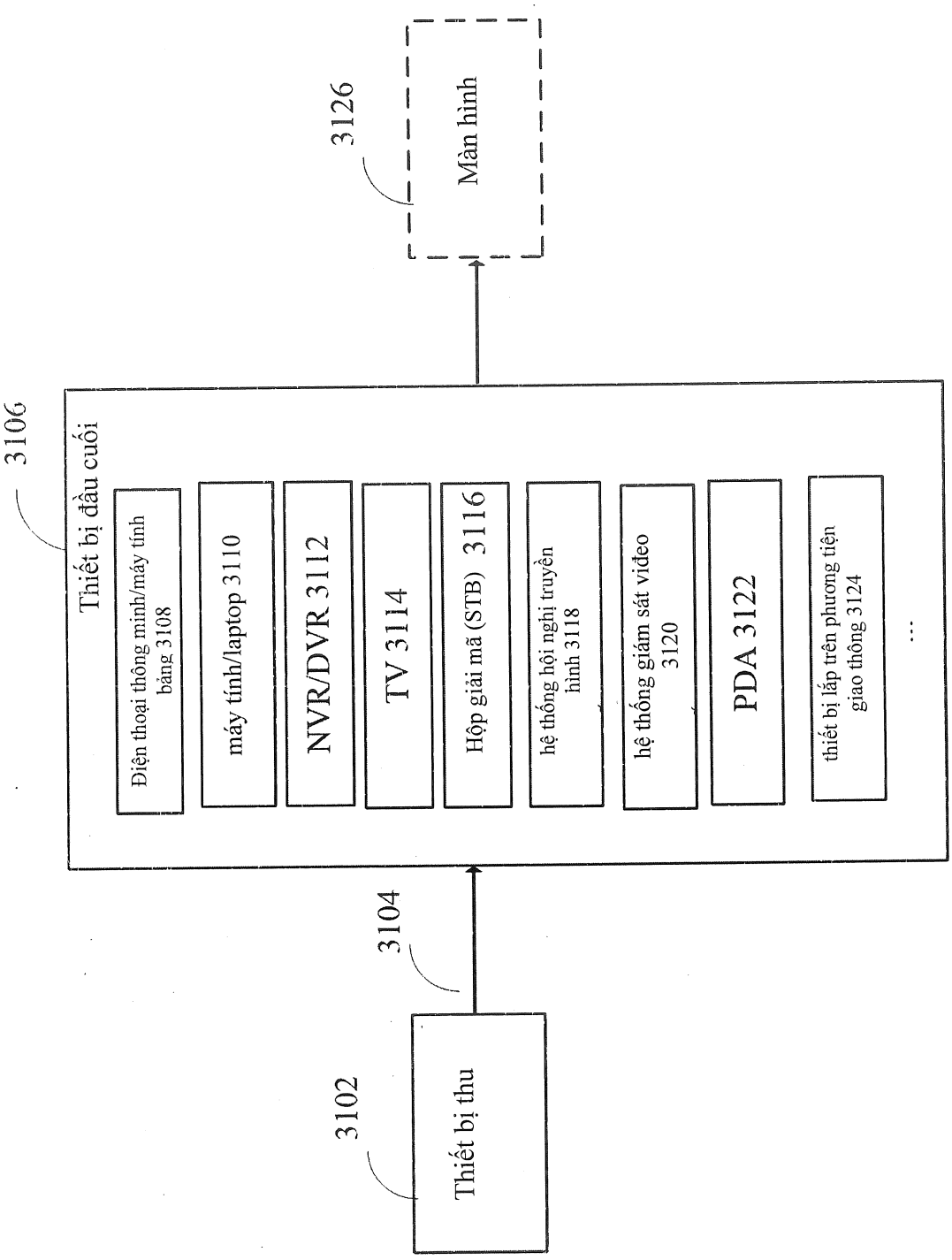

3126
Màn hình
3106
Thiết bị đầu cuối
Điện thoại thông minh/máy tính bảng 3108
máy tính/laptop 3110
NVR/DVR 3112
TV 3114
Hộp giải mã (STB) 3116
hệ thống hội nghị truyền hình 3118
hệ thống giám sát video 3120
PDA 3122
thiết bị lắp trên phương tiện giao thông 3124
...
3104
3102
Thiết bị thu

FIG. 16

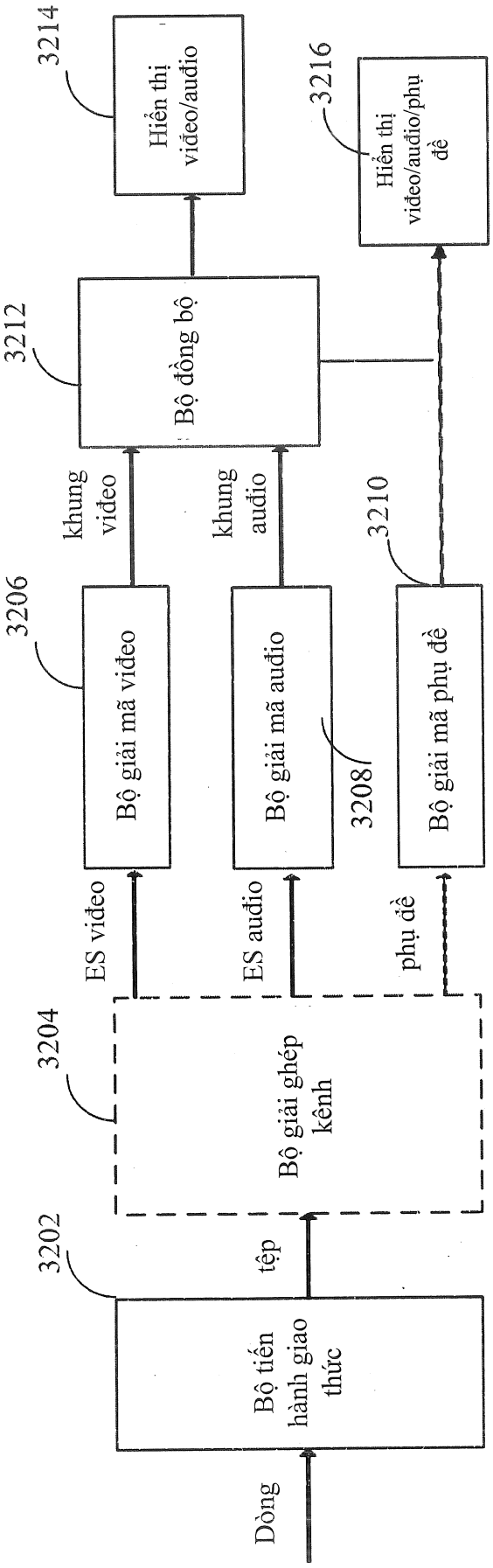

Dòng
Bộ tiến hành giao thức
3202
tệp
Bộ giải ghép kênh
3204
ES video
ES audio
phụ đề
Bộ giải mã video
Bộ giải mã audio
Bộ giải mã phụ đề
3206
3208
3210
khung video
khung audio
Bộ đồng bộ
3212
Hiển thị video/audio
3214
Hiển thị video/audio/phụ đề
3216

FIG. 17