(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) **1-0049771**

(51)[2020.01] **H04N 19/117**; H04N 19/136

(13) **B**

(21) 1-2022-00283

(22) 15/07/2020

(86) PCT/CN2020/102003 15/07/2020

(87) WO2021/008539 21/01/2021

(30) PCT/CN2019/096059 15/07/2019 CN; PCT/CN2019/096398 17/07/2019 CN

(45) 25/08/2025 449

(43) 25/03/2022 408A

(73) 1. BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. (CN)
Room B-0035, 2/F, No. 3 Building No. 30, Shixing Road, Shijingshan District Beijing 100041, China
2. BYTEDANCE INC. (US)
12655 West Jefferson Boulevard Sixth Floor, Suite No. 137 Los Angeles, California 90066, USA

(72) ZHANG, Li (CN); ZHANG, Kai (CN); LIU, Hongbin (CN); WANG, Yue (CN).

(74) Công ty Luật TNHH Phạm và Liên danh (PHAM & ASSOCIATES)

(54) PHƯƠNG PHÁP VÀ THIẾT BỊ XỬ LÝ DỮ LIỆU VIĐEO, VẬT GHI MÁY TÍNH, VÀ PHƯƠNG PHÁP LƯU TRỮ DÒNG BIT CỦA VIĐEO

(21) 1-2022-00283

(57) Sáng chế đề cập đến phương pháp xử lý viđeo bao gồm bước xác định, đối với phép biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn dòng bit của viđeo, các građien của tập con của các mẫu trong vùng đối với hoạt động phân loại trong quá trình lọc. Vùng có chiều M×N và khối có chiều K×L, M, N, K, L là các số nguyên dương. Khối được đặt trong vùng. Phương pháp cũng bao gồm bước thực hiện biến đổi dựa trên việc xác định.

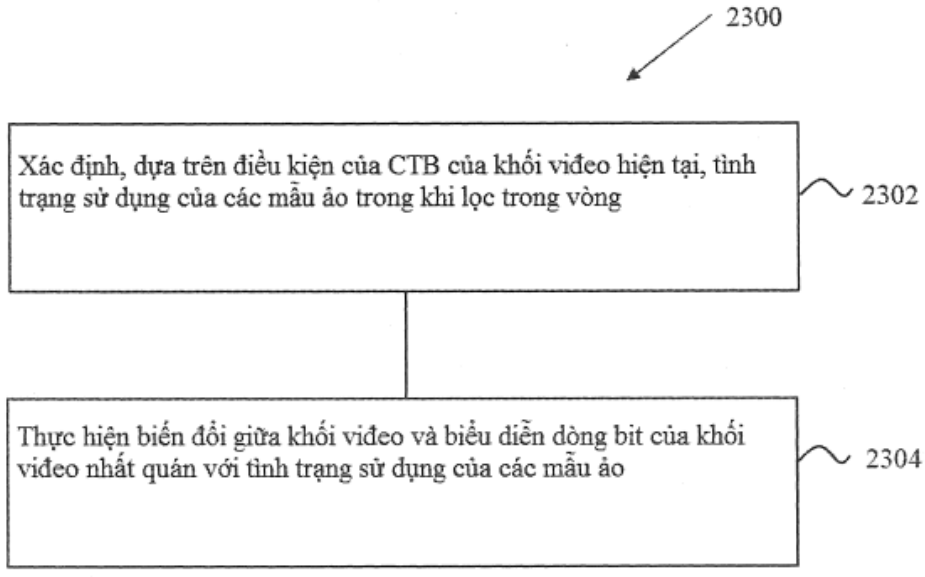


Fig.23

Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập chung đến các công nghệ mã hóa và giải mã viđeo.

Tình trạng kỹ thuật của sáng chế

Các chuẩn mã hóa viđeo đã phát triển chủ yếu qua sự phát triển của các chuẩn ITU-T và ISO/IEC đã biết. ITU-T đã phát triển chuẩn H.261 và H.263, ISO/IEC tạo ra MPEG-1 và MPEG-4 hình ảnh, và hai tổ chức đồng thời tạo ra các chuẩn H.262/MPEG-2 Viđeo và H.264/MPEG-4 mã hóa viđeo cao cấp (AVC) và H.265/mã hóa viđeo hiệu suất cao (HEVC). Kể từ H.262, các chuẩn mã hóa viđeo dựa trên cấu trúc mã hóa viđeo lai trong đó sử dụng dự báo thời gian và mã hóa biến đổi. Để khai thác các công nghệ mã hóa viđeo lai ngoài HEVC, nhóm khai thác viđeo chung (JVET) đã được thành lập bởi VCEG và MPEG đồng thời vào năm 2015. Kể từ đó, nhiều phương pháp mới đã được chấp thuận bởi JVET và được đưa vào phần mềm chuẩn được đặt tên là mô hình khai thác chung (JEM). Vào tháng 4, 2018, JVET giữa VCEG (Q6/16) và ISO/IEC JTC1 SC29/WG11 (MPEG) được tạo ra để hoạt động trên chuẩn mã hóa viđeo đa dụng (VVC) thế hệ tiếp theo nhằm giảm 50% tốc độ bit so với HEVC.

Bản chất kỹ thuật của sáng chế

Nhờ sử dụng các kỹ thuật giải mã hoặc chuyển mã, mã hóa viđeo được bộc lộ, các phương án thực hiện of các bộ mã hóa hoặc các bộ giải mã viđeo có thể xử lý các đường biên ảo của các khối cây mã hóa để cho hiệu quả nén tốt hơn và triển khai đơn giản hơn các công cụ mã hóa hoặc giải mã.

Một khía cạnh ví dụ đề cập đến phương pháp xử lý viđeo. Phương pháp bao gồm xác định, đối với phép biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn dòng bit của viđeo, các gradien của tập hợp con của các mẫu trong vùng đối

với hoạt động phân loại trong quá trình lọc. Vùng có kích thước M×N và khối có kích thước K×L, M, N, K, L là các số nguyên dương, và khối được đặt trong vùng. Phương pháp cũng bao gồm bước thực hiện biến đổi dựa trên việc xác định.

Khía cạnh ví dụ khác đề cập đến phương pháp xử lý viđeo. Phương pháp bao gồm xác định, đối với phép biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn dòng bit của viđeo, hoạt động đệm thống nhất trong quá trình lọc vòng thích ứng (ALF) có thể áp dụng cho các mẫu được đặt ở đường biên ảo 360 của nhiều vùng viđeo của ảnh viđeo bất kể vị trí của đường biên ảo 360 trong ảnh viđeo. Phương pháp cũng bao gồm bước thực hiện biến đổi dựa trên việc xác định.

Khía cạnh ví dụ khác đề cập đến phương pháp xử lý viđeo. Phương pháp bao gồm thực hiện biến đổi giữa các khối viđeo của ảnh viđeo và biểu diễn dòng bit của nó. Ở đây, các khối viđeo được xử lý nhờ sử dụng các nhóm lôgic của các khối cây mã và các khối cây mã được xử lý dựa trên liệu đường biên dưới của khối cây mã dưới nằm ngoài đường biên dưới của ảnh viđeo.

Khía cạnh ví dụ khác đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm các bước xác định, dựa trên điều kiện của khối cây mã của khối viđeo hiện tại, tình trạng sử dụng của các mẫu ảo trong khi lọc trong vòng và thực hiện biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo nhất quán với tình trạng sử dụng của các mẫu ảo.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm xác định, trong khi biến đổi giữa ảnh viđeo được tạo nhóm lôgic thành một hoặc nhiều lát viđeo hoặc khối viên viđeo, và biểu diễn dòng bit của ảnh viđeo, để vô hiệu hóa sử dụng các mẫu trong lát hoặc khối viên khác trong quá trình lọc vòng thích ứng và thực hiện biến đổi nhất quán với bước xác định.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm bước xác định, trong khi biến đổi giữa khối viđeo hiện tại của ảnh viđeo và biểu diễn dòng bit của khối viđeo hiện tại, rằng khối viđeo hiện tại bao gồm các mẫu được đặt ở đường biên của đơn vị viđeo của ảnh viđeo và thực hiện biến đổi dựa trên việc xác định, trong đó việc thực hiện biến đổi bao gồm tạo các mẫu ảo cho quá trình lọc trong vòng nhờ sử dụng phương pháp thống nhất giống

như đối với tất cả các loại đường biên trong ảnh viđeo.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm các bước xác định để áp dụng, trong khi biến đổi giữa khối viđeo hiện tại của ảnh viđeo và biểu diễn dòng bit của nó, một trong nhiều phương pháp chọn mẫu lọc trong vòng (ALF) khả dụng cho ảnh viđeo trong khi biến đổi và thực hiện biến đổi bằng cách áp dụng một trong nhiều phương pháp chọn mẫu ALF.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên vô hiệu hóa việc sử dụng các mẫu giao cắt đơn vị dữ liệu đường ống ảo (VPDU) của ảnh viđeo và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm các bước thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên xác định để sử dụng, đối với các vị trí của khối viđeo hiện tại đi qua đường biên đơn vị viđeo, các mẫu được tạo mà không sử dụng đệm và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm các bước thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên xác định lựa chọn, đối với hoạt động lọc trong vòng, bộ lọc có các kích thước sao cho các mẫu của khối viđeo hiện tại được sử dụng trong khi lọc trong vòng không giao với đường biên của đơn vị viđeo của ảnh viđeo và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm các bước thực hiện, dựa trên quy tắc đường biên, hoạt động lọc

trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên xác định lựa chọn, đối với hoạt động lọc trong vòng, các tham số xén hoặc các hệ số lọc dựa trên liệu các mẫu được đệm có cần cho lọc trong vòng hay không và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

Khía cạnh ví dụ khác nữa đề cập đến phương pháp xử lý viđeo khác. Phương pháp bao gồm bước thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên tùy thuộc vào định danh thành phần màu của khối viđeo hiện tại và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

Khía cạnh ví dụ khác nữa đề cập đến thiết bị mã hóa viđeo được tạo cấu hình để thực hiện phương pháp nêu trên.

Khía cạnh ví dụ khác nữa đề cập đến bộ giải mã viđeo được tạo cấu hình để thực hiện phương pháp nêu trên.

Khía cạnh ví dụ khác nữa đề cập đến phương tiện máy đọc được. Phương tiện lưu trữ mã mà, khi thực thi, khiến bộ xử lý thực hiện một hoặc nhiều phương pháp nêu trên.

Các khía cạnh và các dấu hiệu khác và nêu trên của sáng chế được mô tả chi tiết hơn trong các hình vẽ, phần mô tả và các điểm yêu cầu bảo hộ.

Mô tả vắn tắt các hình vẽ

Fig.1 là hình vẽ thể hiện ví dụ của ảnh có các đơn vị cây mã (CTU) độ sáng 18x12 được phân vùng thành 12 ô và 3 lát quét mành;

Fig.2 là hình vẽ thể hiện ví dụ của ảnh có các CTU độ sáng 18x12 được phân vùng thành 24 ô và 9 lát chữ nhật;

Fig.3 là hình vẽ thể hiện ví dụ của ảnh được phân vùng thành 4 ô, 11 khối viên, và 4 lát chữ nhật;

Fig.4A là hình vẽ thể hiện ví dụ của các khối cây mã (CTB) giao các đường biên ảnh khi $K = M$, $L < N$;

Fig.4B là hình vẽ thể hiện ví dụ của các CTB giao các đường biên ảnh khi K<M, L=N;

Fig.4C là hình vẽ thể hiện ví dụ của các CTB giao các đường biên ảnh khi K<M, L<N;

Fig.5 là hình vẽ thể hiện ví dụ của sơ đồ khối của bộ mã hóa;

Fig.6 là hình vẽ minh họa các mẫu ảnh và các đường biên khối chiều ngang và chiều dọc trên lưới 8×8, và các khối không chồng lấp của các mẫu 8×8, mà có thể được khử khối song song;

Fig.7 là hình vẽ thể hiện các ví dụ của các pixel được bao gồm trong quyết định bật/tắt bộ lọc và lựa chọn bộ lọc mạnh/yếu;

Fig.8 là hình vẽ thể hiện bốn mẫu hình định hướng 1-D;

Fig.9 là hình vẽ thể hiện ví dụ của các hình dạng bộ lọc ALF hình học (GALF) (bên trái: hình kim cương 5×5, giữa: hình kim cương 7×7, bên phải: hình kim cương 9×9);

Fig.10 là hình vẽ thể hiện các tọa độ tương đối để hỗ trợ lọc kim cương 5×5;

Fig.11 là hình vẽ thể hiện ví dụ của các tọa độ tương đối để hỗ trợ lọc kim cương 5×5;

Fig.12A là hình vẽ thể hiện bố trí lấy làm ví dụ cho các phép tính Laplacian được lấy mẫu phụ;

Fig.12B là hình vẽ thể hiện bố trí ví dụ khác cho các phép tính Laplacian được lấy mẫu phụ;

Fig.12C là hình vẽ thể hiện bố trí ví dụ khác cho các phép tính Laplacian được lấy mẫu phụ;

Fig.12D là hình vẽ thể hiện bố trí ví dụ khác nữa cho các phép tính Laplacian được lấy mẫu phụ;

Fig.13 là hình vẽ thể hiện ví dụ của yêu cầu bộ đệm dòng lọc vòng trong VTM-4.0 đối với thành phần độ sáng;

Fig.14 là hình vẽ thể hiện ví dụ của yêu cầu bộ đệm dòng lọc vòng trong VTM-4.0 đối với thành phần sắc độ;

Fig.15A là hình vẽ thể hiện ví dụ của phân loại khối ALF ở đường biên ảo khi N = 4;

Fig.15B là hình vẽ thể hiện ví dụ khác phân loại khối ALF ở đường biên ảo khi N = 4;

Fig.16A là hình vẽ minh họa ví dụ của lọc ALF độ sáng được chỉnh sửa ở đường biên ảo;

Fig.16B là hình vẽ minh họa ví dụ khác của lọc ALF độ sáng được chỉnh sửa ở đường biên ảo;

Fig.16C là hình vẽ minh họa ví dụ khác nữa của lọc ALF độ sáng được chỉnh sửa ở đường biên ảo;

Fig.17A là hình vẽ thể hiện ví dụ của lọc ALF sắc độ được chỉnh sửa ở đường biên ảo;

Fig.17B là hình vẽ thể hiện ví dụ khác lọc ALF sắc độ được chỉnh sửa ở đường biên ảo;

Fig.18A là hình vẽ thể hiện ví dụ của bù chuyển động bao quanh ngang;

Fig.18B là hình vẽ thể hiện ví dụ khác bù chuyển động bao quanh ngang;

Fig.19 là hình vẽ minh họa ví dụ của ALF được chỉnh sửa;

Fig.20 là hình vẽ thể hiện ví dụ xử lý các CTU trong ảnh viđeo;

Fig.21 là hình vẽ thể hiện ví dụ của ALF được chỉnh sửa đường biên;

Fig.22 là sơ đồ khối của ví dụ của thiết bị xử lý viđeo;

Fig.23 là lưu đồ đối với phương pháp xử lý viđeo lấy làm ví dụ;

Fig.24 là hình vẽ thể hiện ví dụ của ảnh của HEC trong bố trí 3x2;

Fig.25 là hình vẽ thể hiện ví dụ của số lượng đường được đệm đối với các mẫu của hai loại đường biên;

Fig.26 là hình vẽ thể hiện ví dụ xử lý các CTU trong ảnh;

Fig.27 là hình vẽ thể hiện ví dụ khác xử lý các CTU trong ảnh;

Fig.28 là hình vẽ thể hiện ví dụ khác mẫu hiện tại và các mẫu cần được truy nhập;

Fig.29 là hình vẽ thể hiện ví dụ khác đệm các mẫu lân cận "không khả dụng";

Fig.30 là hình vẽ thể hiện ví dụ của các mẫu cần được sử dụng trong quá trình phân loại ALF;

Fig.31 là sơ đồ khối của hệ thống xử lý viđeo lấy làm ví dụ trong đó sáng chế có thể được triển khai;

Fig.32 là hình vẽ thể hiện lưu đồ của phương pháp xử lý viđeo theo sáng chế; và

Fig.33 là hình vẽ thể hiện lưu đồ của phương pháp xử lý viđeo khác theo sáng chế.

Mô tả chi tiết sáng chế

Các đề mục được sử dụng trong sáng chế để dễ hiểu và không giới hạn các phương án thực hiện được bộc lộ trong mục ở chỉ mục đó. Ngoài ra, trong khi các phương án thực hiện cụ thể được mô tả dựa vào VVC hoặc các codec viđeo cụ thể khác, sáng chế cũng có thể áp dụng cho các công nghệ mã hóa viđeo khác. Ngoài ra, trong khi một số phương án thực hiện mô tả chi tiết các bước mã hóa viđeo, cần hiểu rằng các bước tương ứng giải mã mà hoàn tác mã hóa sẽ được triển khai bằng bộ giải mã. Ngoài ra, cụm từ xử lý viđeo bao gồm mã hóa hoặc nén viđeo, giải mã hoặc giải nén viđeo và chuyển mã viđeo trong đó các pixel viđeo được biểu diễn từ một định dạng được nén thành định dạng được nén khác hoặc ở tốc độ bit được nén khác.

1. Tóm tắt

Sáng chế đề cập đến các công nghệ mã hóa viđeo. Cụ thể là, đề cập đến mã hóa đường biên ảnh/lát/ô/khối viên và đường biên ảo đặc biệt đối với bộ lọc vòng thích ứng phi tuyến. Có thể được áp dụng cho chuẩn mã hóa viđeo hiện tại như HEVC, hoặc chuẩn (VVC) cần được hoàn thiện. Cũng có thể áp dụng cho các chuẩn mã hóa viđeo hoặc viđeo codec tương lai.

2. Mô tả ban đầu

Các chuẩn mã hóa viđeo đã phát triển chủ yếu qua sự phát triển của các chuẩn ITU-T và ISO/IEC đã biết. ITU-T đã phát triển chuẩn H.261 và H.263, ISO/IEC tạo ra MPEG-1 và MPEG-4 hình ảnh, và hai tổ chức đồng thời tạo ra các chuẩn H.262/MPEG-2 Video và H.264/MPEG-4 AVC và H.265/HEVC. Từ H.262, các chuẩn mã hóa viđeo dựa trên cấu trúc mã hóa viđeo lai trong đó sử dụng dự báo thời gian và mã hóa biến đổi. Để khai thác các công nghệ mã hóa viđeo lai ngoài HEVC, JVET đã được thành lập bởi VCEG và MPEG đồng thời vào năm 2015. Kể từ đó, nhiều phương pháp mới đã được chấp thuận bởi JVET và được đưa

vào phần mềm chuẩn được đặt tên là JEM. Vào tháng 4 2018, JVET giữa VCEG (Q6/16) và ISO/IEC JTC1 SC29/WG11 (MPEG) được tạo ra để hoạt động trên chuẩn VVC nhằm giảm 50% tốc độ bit so với HEVC.

2.1 Không gian màu và lấy mẫu phụ sắc độ

Không gian màu, cũng được biết đến như là mô hình màu (hoặc hệ thống màu), là mô hình toán học trừu tượng mà chỉ mô tả khoảng màu sắc dưới dạng các bộ ba số, thường là 3 hoặc 4 giá trị hoặc các thành phần màu sắc (chẳng hạn, RGB). Về cơ bản, không gian màu là mô tả chi tiết của hệ tọa độ và không gian phụ.

Để nén viđeo, các không gian màu thường được sử dụng nhất là YCbCr và RGB.

YCbCr, Y'CbCr, hoặc Y Pb/Cb Pr/Cr, cũng được viết là YCBCR hoặc Y'CBCR, là họ của không gian màus được sử dụng làm một phần của đường ống ảnh màu trong các hệ thống viđeo và chụp ảnh số. Y' là thành phần độ sáng và CB và CR là các thành phần sắc độ lệch lam và lệnh đỏ. Y' (có dấu phẩy) được phân biệt với Y, vốn là độ sáng, nghĩa là cường độ ánh sáng được mã hóa phi tuyến dựa trên các màu gốc RGB được hiệu chỉnh gamma.

Lấy mẫu phụ sắc độ là thực hiện mã hóa ảnh bằng cách triển khai độ phân giải nhỏ hơn cho thông tin sắc độ so với cho thông tin độ sáng, tận dụng độ tinh nhỏ hơn đối với chênh lệch màu sắc so với độ sáng của hệ thị lực của con người.

2.1.1 Định dạng màu sắc 4:4:4

Mỗi thành phần trong ba thành phần Y'CbCr có cùng tốc độ mẫu, do vậy không có lấy mẫu phụ sắc độ. Phương tiện này đôi lúc được sử dụng trong các máy quét film dòng cao cấp và hậu kỳ phim ảnh.

2.1.2 Định dạng màu sắc 4:2:2

Hai thành phần sắc độ được lấy mẫu ở nửa tốc độ mẫu độ sáng: độ phân giải sắc độ theo chiều ngang được chia đôi. Điều này giảm băng thông của tín hiệu viđeo không được nén đi 1/3 mà không gây ra khác biệt về hình ảnh

2.1.3 Định dạng màu sắc 4:2:0

Ở định dạng 4:2:0, lấy mẫu theo chiều ngang được tăng gấp đôi so với 4:1:1, nhưng do các kênh Cb và Cr chỉ được lấy mẫu trên mỗi đường xen kẽ trong sơ

đồ này, độ phân giải dọc được chia đôi. Do vậy, tốc độ dữ liệu là giống nhau. Mỗi trong số Cb và Cr được lấy mẫu phụ ở hệ số 2 cả theo chiều ngang và chiều dọc. Có ba biến thể của sơ đồ 4:2:0, có vạch tuyến theo chiều ngang và dọc khác nhau.

Ở chuẩn MPEG-2, Cb và Cr nằm cùng vị trí theo chiều ngang. Cb và Cr được đặt giữa các pixel theo chiều dọc (được đặt ngoài nút).

Trong chuẩn JPEG/JFIF, H.261, và MPEG-1, Cb và Cr được đặt ngoài nút, nửa đường giữa các mẫu độ sáng luân phiên.

Ở định dạng 4:2:0 DV, Cb và Cr nằm cùng vị trí theo chiều ngang. Theo chiều dọc, chúng nằm cùng vị trí trên các đường luân phiên.

2.2 Các đơn vị viđeo khác nhau

Ảnh được phân chia thành một hoặc nhiều hàng ô và một hoặc nhiều cột ô. Ô là chuỗi của các CTU bao phủ vùng chữ nhật của ảnh.

Ô được phân chia thành một hoặc nhiều khối viên, mỗi khối viên chứa nhiều hàng CTU trong ô.

Ô không được phân vùng thành nhiều khối viên cũng được gọi là khối viên. Tuy nhiên, khối viên là tập con thực của ô không được gọi là ô.

Lát chứa nhiều ô của ảnh hoặc nhiều khối viên của ô.

Hai chế độ của lát được hỗ trợ, tức là chế độ lát quét mành và chế độ lát chữ nhật. Trong chế độ lát quét mành, lát chứa chuỗi ô trong quét mành ô của ảnh. Trong chế độ lát chữ nhật, lát chứa nhiều khối viên của ảnh tạo chung thành vùng chữ nhật của ảnh. Các khối viên trong lát chữ nhật theo thứ tự quét khối viên của lát.

Fig.1 thể hiện ví dụ của phân vùng lát quét mành của ảnh, trong đó ảnh được phân chia thành 12 ô và 3 lát quét mành.

Fig.2 thể hiện ví dụ của phân vùng lát chữ nhật của ảnh, trong đó ảnh được phân chia thành 24 ô (6 cột ô và 4 hàng ô) và 9 lát chữ nhật.

Fig.3 thể hiện ví dụ của ảnh được phân vùng thành các ô, các khối viên, và lát chữ nhật, trong đó ảnh được phân chia thành 4 ô (2 cột ô và 2 hàng ô), 11 khối viên (ô trên bên trái chứa 1 khối viên, ô trên bên phải chứa 5 khối viên, ô dưới bên trái chứa 2 khối viên, và ô dưới bên phải chứa 3 khối viên), và 4 lát chữ

nhật.

2.2.1 Kích thước CTU/CTB

Trong VVC, kích thước CTU, được báo hiệu trong SPS bằng phần tử cú pháp **log2_ctu_size_minus2**, có thể nhỏ bằng 4x4.

7.3.2.3 Cú pháp RBSP tập tham số chuỗi

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_5bits	u(5)
profile_tier_level(sps_max_sub_layers_minus1)	
gra_enabled_flag	u(1)
sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
bit_depth_luma_minus8	ue(v)
bit_depth_chroma_minus8	ue(v)
log2_max_pic_order_cnt_lsb_minus4	ue(v)
sps_sub_layer_ordering_info_present_flag	u(1)
for(i = (sps_sub_layer_ordering_info_present_flag ? 0 : sps_max_sub_layers_minus1); i <= sps_max_sub_layers_minus1; i++) {	
sps_max_dec_pic_buffering_minus1[i]	ue(v)
sps_max_num_reorder_pics[i]	ue(v)
sps_max_latency_increase_plus1[i]	ue(v)
}	
long_term_ref_pics_flag	u(1)
sps_idr_rpl_present_flag	u(1)
rpl1_same_as_rpl0_flag	u(1)
for(i = 0; i < !rpl1_same_as_rpl0_flag ? 2 : 1; i++) {	
num_ref_pic_lists_in_sps[i]	ue(v)
for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)	
ref_pic_list_struct(i, j)	
}	
qtbtt_dual_tree_intra_flag	u(1)
log2_ctu_size_minus2	ue(v)
log2_min_luma_coding_block_size_minus2	ue(v)
partition_constraints_override_enabled_flag	u(1)
sps_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)

sps_log2_diff_min_qt_min_cb_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(sps_max_mtt_hierarchy_depth_inter_slices != 0) {	
sps_log2_diff_max_bt_min_qt_inter_slice	ue(v)
sps_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if (sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
...	
rbsp_trailing_bits()	
}	

log2_ctu_size_minus2 cộng 2 xác định kích thước CTB độ sáng của mỗi CTU.

log2_min_luma_coding_block_size_minus2 cộng 2 xác định kích thước khối mã độ sáng nhỏ nhất.

Các biến CtbLog2SizeY, CtbSizeY, MinCbLog2SizeY, MinCbSizeY, MinTbLog2SizeY, MaxTbLog2SizeY, MinTbSizeY, MaxTbSizeY, PicWidthInCtbsY, PicHeightInCtbsY, PicSizeInCtbsY, PicWidthInMinCbsY, PicHeightInMinCbsY, PicSizeInMinCbsY, PicSizeInSamplesY, PicWidthInSamplesC và PicHeightInSamplesC được dẫn xuất như sau:

CtbLog2SizeY = log2_ctu_size_minus2 + 2 (7-9)

CtbSizeY = 1 << CtbLog2SizeY (7-10)

MinCbLog2SizeY = log2_min_luma_coding_block_size_minus2 + 2 (7-11)

MinCbSizeY = 1 << MinCbLog2SizeY(7-12)

MinTbLog2SizeY = 2 (7-13)

MaxTbLog2SizeY = 6 (7-14)

MinTbSizeY = 1 << MinTbLog2SizeY(7-15)

MaxTbSizeY = 1 << MaxTbLog2SizeY (7-16)

PicWidthInCtbsY = Ceil(pic_width_in_luma_samples ÷ CtbSizeY) (7-17)

PicHeightInCtbsY = Ceil(pic_height_in_luma_samples ÷ CtbSizeY) (7-18)

PicSizeInCtbsY = PicWidthInCtbsY * PicHeightInCtbsY (7-19)

PicWidthInMinCbsY = pic_width_in_luma_samples/MinCbSizeY (7-20)

PicHeightInMinCbsY = pic_height_in_luma_samples/MinCbSizeY (7-21)

PicSizeInMinCbsY = PicWidthInMinCbsY * PicHeightInMinCbsY (7-22)

PicSizeInSamplesY =
pic_width_in_luma_samples * pic_height_in_luma_samples (7-23)

PicWidthInSamplesC = pic_width_in_luma_samples/SubWidthC (7-24)

PicHeightInSamplesC = pic_height_in_luma_samples/SubHeightC (7-25)

2.2.2 Các CTU trong ảnh

Giả sử kích thước CTB/LCU được chỉ báo bằng M x N (thường M bằng N, như được định nghĩa trong HEVC/VVC), và đối với CTB được đặt ở ảnh (hoặc ô hoặc lát hoặc các loại khác, biên ảnh được lấy làm ví dụ) border, K x L mẫu nằm trong biên ảnh trong đó K<M hoặc L<N. Đối với các CTB đó như được mô tả trên Fig.4A-4C, **kích thước CTB vẫn bằng MxN, tuy nhiên, đường biên dưới/đường biên phải của CTB nằm ngoài ảnh.**

Fig.4A thể hiện các CTB giao biên dưới của ảnh. Fig.4B thể hiện các CTB giao biên phải của ảnh. Fig.4C thể hiện các CTB giao biên dưới bên phải của ảnh

Các hình vẽ từ Fig.4A đến Fig.4C thể hiện các ví dụ của các CTB giao với các đường biên của ảnh, (a) K=M, L<N; (b) K<M, L=N; (c) K<M, L<N

2.3 Luồng mã hóa của codec viđeo đặc trưng

Fig.5 thể hiện ví dụ của sơ đồ khối của bộ mã hóa của VVC, mà chứa ba khối lọc trong vòng: bộ lọc tách khối (DF), bộ lọc độ lệch mẫu thích ứng (SAO) và ALF. Không giống DF, mà sử dụng các bộ lọc định trước, SAO và ALF sử dụng các mẫu ban đầu của ảnh hiện tại để giảm các sai số quân phương giữa các mẫu ban đầu và các mẫu được tái tạo bằng cách bổ sung độ lệch và bằng cách áp dụng bộ lọc đáp ứng xung hữu hạn (FIR), một cách lần lượt, có thông tin phụ được mã hóa báo hiệu các độ lệch và các hệ số lọc. ALF ở được đặt ở tầng xử lý cuối cùng của mỗi ảnh và có thể được xem là công cụ cố gắng bắt và sửa các giả tượng được tạo bởi các tầng trước đó.

2.4 Bộ lọc tách khối (DB)

Đầu vào của DB là các mẫu được tái tạo trước các bộ lọc trong vòng.

Các cạnh dọc trong ảnh được lọc trước. Sau đó, các cạnh ngang trong ảnh

được lọc với các mẫu được chỉnh sửa bởi quá trình lọc cạnh dọc làm đầu vào. Các cạnh ngang và dọc trong các CTB của mỗi CTU được xử lý riêng rẽ dựa trên đơn vị mã. Các cạnh dọc của các khối mã trong đơn vị mã được lọc bắt đầu với cạnh ở phía tay trái của các khối mã đi qua cạnhs về phía tay phải của các khối mã theo thứ tự hình học. Các cạnh ngang của các khối mã trong đơn vị mã được lọc bắt đầu với cạnh ở phía trên của các khối mã đi qua cạnhs về phía dưới của các khối mã theo thứ tự hình học.

Fig.6 minh họa các mẫu ảnh và các đường biên khối chiều ngang và chiều dọc trên lưới 8×8, và các khối không chồng lấp của các mẫu 8×8, mà có thể được khử khối song song.

2.4.1. Xác định đường biên

Việc lọc được áp dụng cho các đường biên khối 8x8. Ngoài ra, nó phải là đường biên khối biến đổi hoặc đường biên khối phụ mã hóa (chẳng hạn, do sử dụng dự báo chuyển động afin (ATMVP)). Đối với các đường biên không phải là các đường biên này, bộ lọc bị vô hiệu hóa.

2.4.1 Tính toán cường độ biên

Đối với đường biên khối biến đổi/đường biên khối phụ mã, nếu được đặt trong the 8x8 grid, có thể được lọc và thiết lập của $bS[xD_i][yD_j]$ (trong đó $[xD_i][yD_j]$ ký hiệu tọa độ) đối với cạnh này được định nghĩa trong Bảng 1 và Bảng 2, một cách lần lượt.

Bảng 1. Cường độ biên (khi SPS IBC bị vô hiệu hóa)

Độ ưu tiên	Điều kiện	Y	U	V
5	Ít nhất một trong các khối lân cận là nội vùng	2	2	2
4	Đường biên TU và ít nhất một trong các khối lân cận có các hệ số biến đổi khác 0	1	1	1
3	Các ảnh tham chiếu hoặc số MV (1 đối với dự báo đơn, 2 đối với dự báo kép) của các khối lân cận là khác nhau	1	N/A	N/A
2	Hiệu số tuyệt đối giữa các vectơ chuyển động của cùng ảnh tham chiếu thuộc các khối liền kề lớn hơn hoặc bằng một mẫu độ sáng số nguyên	1	N/A	N/A

1	Ngược lại	0	0	0

Bảng 2. Cường độ biên (khi SPS IBC được kích hoạt)

Độ ưu tiên	Điều kiện	Y	U	V
8	Ít nhất một trong các khối lân cận là nội vùng	2	2	2
7	Đường biên TU và ít nhất một trong các khối lân cận có các hệ số biến đổi khác 0	1	1	1
6	Chế độ dự báo của các khối liền kề là khác nhau (chẳng hạn, một là IBC, một là liên)	1		
5	Cả IBC và hiệu số tuyệt đối giữa các vectơ chuyển động thuộc các khối liền kề lớn hơn hoặc bằng một mẫu độ sáng số nguyên	1	N/A	N/A
4	Các ảnh tham chiếu hoặc số MV (1 đối với dự báo đơn, 2 đối với dự báo kép) của các khối lân cận là khác nhau	1	N/A	N/A
3	Hiệu số tuyệt đối giữa các vectơ chuyển động của cùng ảnh tham chiếu thuộc các khối liền kề lớn hơn hoặc bằng một mẫu độ sáng số nguyên	1	N/A	N/A
1	Ngược lại	0	0	0

2.4.3 Quyết định tách khối cho thành phần độ sáng

Quá trình quyết định tách khối được mô tả trong phần phụ này.

Fig.7 thể hiện các ví dụ của các pixel được bao gồm trong quyết định bật/tắt bộ lọc và lựa chọn bộ lọc mạnh/yếu.

Các bộ lọc độ sáng mạnh hơn - rộng hơn là các bộ lọc được sử dụng chỉ nếu tất cả điều kiện1, Điều kiện 2 và điều kiện 3 làTRUE.

Điều kiện 1 là "điều kiện khối lớn". Điều kiện này dò thấy liệu các mẫu ở phía P và phía Q thuộc các khối lớn, được biểu diễn bởi biến bSidePisLargeBlk và bSideQisLargeBlk một cách lần lượt. bSidePisLargeBlk và bSideQisLargeBlk được định nghĩa như sau.

bSidePisLargeBlk = ((loại cạnh là dọc và p_0 thuộc CU với width >= 32) | |

(loại cạnh là ngang và p_0 thuộc CU với height >= 32))? TRUE: FALSE

bSideQisLargeBlk = ((loại cạnh là dọc và q_0 thuộc CU với width >= 32) | |

(loại cạnh là ngang và q_0 thuộc CU với height >= 32))? TRUE: FALSE

Dựa trên bSidePisLargeBlk và bSideQisLargeBlk, điều kiện 1 được định nghĩa như sau.

Condition1 = (bSidePisLargeBlk || bSidePisLargeBlk) ? TRUE: FALSE

Tiếp theo, nếu **Condition 1 là** true, điều kiện 2 sẽ được kiểm tra thêm. Trước hết, các biến sau được dẫn xuất:

- dp0, dp3, dq0, dq3 được dẫn xuất trước như trong HEVC
- nếu (phía p lớn hơn hoặc bằng 32)

  $dp0 = (dp0 + Abs(p5_0 - 2 * p4_0 + p3_0) + 1) >> 1$

  $dp3 = (dp3 + Abs(p5_3 - 2 * p4_3 + p3_3) + 1) >> 1$
- nếu (phía q lớn hơn hoặc bằng 32)

  $dq0 = (dq0 + Abs(q5_0 - 2 * q4_0 + q3_0) + 1) >> 1$

  $dq3 = (dq3 + Abs(q5_3 - 2 * q4_3 + q3_3) + 1) >> 1$

Condition2 = (d < β) ? TRUE: FALSE

trong đó d= dp0 + dq0 + dp3 + dq3.

Nếu Condition1 và Condition2 là hợp lệ, liệu khối bất kỳ trong các khối sử dụng các khối phụ còn được kiểm tra:

```
If (bSidePisLargeBlk)
{
  If (mode block P == SUBBLOCKMODE)
    Sp =5
  else
    Sp =7
}
else
 Sp = 3
If (bSideQisLargeBlk)
 {
  If (mode block Q == SUBBLOCKMODE)
    Sq =5
```

```
        else
          Sq =7
        }
      else
       Sq = 3
```

Cuối cùng, nếu cả **Condition 1** và **Condition 2** là hợp lệ, phương pháp tách khối được đề xuất sẽ kiểm tra điều kiện 3 (điều kiện lọc khỏe khối lớn), which được định nghĩa như sau.

Trong Condition3 StrongFilterCondition, các biến sau được dẫn xuất:

dpq được dẫn xuất như trong HEVC.

$sp_3 = Abs(p_3 - p_0)$, được dẫn xuất như trong HEVC

```
if (phía p lớn hơn hoặc bằng 32)
   if(Sp==5)
     sp3 = (sp3 + Abs(p5 − p3) + 1) >> 1
   else
     sp3 = (sp3 + Abs(p7 − p3) + 1) >> 1
```

$sq_3 = Abs(q_0 - q_3)$, được dẫn xuất như trong HEVC

```
if (phía q lớn hơn hoặc bằng 32)
  If(Sq==5)
    sq3 = (sq3 + Abs(q5 − q3) + 1) >> 1
  else
    sq3 = (sq3 + Abs(q7 − q3) + 1) >> 1
```

Như trong chuẩn HEVC, StrongFilterCondition = (dpq nhỏ hơn ($\beta >> 2$), $sp_3 + sq_3$ nhỏ hơn ($3*\beta >> 5$), và $Abs(p_0 - q_0)$ nhỏ hơn $(5 * t_C + 1) >> 1$) ? TRUE : FALSE.

2.4.4 Bộ lọc tách khối mạnh hơn cho độ sáng (được thiết kế cho các khối lớn hơn)

Bộ lọc song tuyến tính được sử dụng khi các mẫu ở một phía của đường biên thuộc khối lớn. Mẫu thuộc khối lớn được định nghĩa như khi chiều rộng >= 32 đối với cạnh dọc, và khi chiều cao >= 32 đối với cạnh ngang.

Bộ lọc song tuyến tính được liệt kê dưới đây.

Các mẫu biên khối p_i với i=0 đến Sp-1 và q_i với j=0 đến Sq-1 (pi và qi là mẫu thứ i trong hàng để lọc cạnh dọc, hoặc mẫu thứ i trong cột để lọc cạnh ngang) trong quá trình tách khối HEVC nêu trên) sau đó được thay thế bằng nội suy tuyến tính như sau:

— $p_i' = (f_i * Middle_{s,t} + (64 - f_i) * P_s + 32) \gg 6)$, *được rút gọn thành* $p_i \pm tcPD_i$

— $q_j' = (g_j * Middle_{s,t} + (64 - g_j) * Q_s + 32) \gg 6)$, *được rút gọn thành* $q_j \pm tcPD_j$

trong đó số hạng $tcPD_i$ và $tcPD_j$ là hoạt động rút gọn tùy thuộc vị trí được mô tả trong phần 2.4.7 và g_j, f_i, $Middle_{s,t}$, P_s và Q_s được nêu dưới đây:

2.4.5 Điều khiển tách khối đối với sắc độ

Các bộ lọc mạnh sắc độ được sử dụng ở cả hai phía của đường biên khối. Ở đây, bộ lọc sắc độ được chọn khi cả hai phía của mép sắc độ lớn hơn hoặc bằng 8 (vị trí sắc độ), và quyết định dưới đây với ba điều kiện được thỏa mãn: điều kiện thứ nhất là để quyết định cường độ biên cũng như khối lớn. Bộ lọc được đề xuất có thể được áp dụng khi chiều rộng hoặc chiều cao của khối mà trực giao với cạnh khối bằng hoặc lớn hơn 8 trong miền mẫu sắc độ. Điều kiện thứ hai và thứ ba về cơ bản giống như đối với việc quyết định tách khối độ sáng HEVC, vốn là quyết định bật/tắt và quyết định lọc mạnh, một cách lần lượt.

Trong quyết định thứ nhất, cường độ biên (bS) được chỉnh sửa để lọc sắc độ và các điều kiện được kiểm tra tuần tự. Nếu điều kiện được thỏa mãn, thì các điều kiện còn lại có các độ ưu tiên thấp hơn được bỏ qua.

Tách khối sắc độ được thực hiện khi bS bằng 2, or bS bằng 1 khi dò thấy đường biên khối lớn.

Điều kiện thứ hai và thứ ba về cơ bản giống như quyết định lọc mạnh độ sáng HEVC như sau.

Ở điều kiện thứ hai:

d làn được dẫn xuất như trong tách khối độ sáng HEVC.

Điều kiện thứ hai sẽ là TRUE khi d nhỏ hơn β.

Ở điều kiện thứ ba StrongFilterCondition được dẫn xuất như sau:

dpq được dẫn xuất như trong HEVC.

$sp_3 = Abs(p_3 - p_0)$, được dẫn xuất như trong HEVC

$sq_3 = Abs(q_0 - q_3)$, được dẫn xuất như trong HEVC

Như trong thiết kế HEVC, StrongFilterCondition = (dpq nhỏ hơn (β >> 2), $sp_3 + sq_3$ nhỏ hơn (β >> 3), và Abs($p_0 - q_0$) nhỏ hơn (5 * t_C + 1) >> 1)

2.4.6 Bộ lọc tách khối mạnh đối với sắc độ

Bộ lọc tách khối mạnh sau đây đối với sắc độ được định nghĩa:

$p_2' = (3*p_3+2*p_2+p_1+p_0+q_0+4) >> 3$

$p_1' = (2*p_3+p_2+2*p_1+p_0+q_0+q_1+4) >> 3$

$p_0' = (p_3+p_2+p_1+2*p_0+q_0+q_1+q_2+4) >> 3$

Bộ lọc sắc độ được đề xuất thực hiện tách khối trên lưới mẫu sắc độ 4x4.

2.4.7 Hoạt động rút gọn tùy thuộc vị trí

Hoạt động rút gọn tùy thuộc vị trí tcPD được áp dụng cho các mẫu đầu ra của quá trình lọc độ sáng bao gồm các bộ lọc dài và mạnh đang chỉnh sửa 7, 5 và 3 mẫu ở đường biên. Giả sử phân phối sai số lượng tử, đề xuất tăng giá trị rút gọn đối với các mẫu sẽ có nhiễu lượng tử cao hơn, do vậy sẽ có độ lệch cao hơn của giá trị mẫu được tái tạo từ giá trị mẫu đúng.

Đối với mỗi đường biên P hoặc Q được lọc với bộ lọc bất đối xứng, tùy thuộc vào kết quả của quá trình ra quyết định trong phần 2.4.2, bảng của ngưỡng phụ thuộc vị trí được chọn từ hai bảng (chẳng hạn, Tc7 và Tc3 được tạo bảng dưới đây) mà được cấp cho bộ giải mã làm thông tin phụ:

Tc7 = {6, 5, 4, 3, 2, 1, 1}; Tc3 = {6, 4, 2};

tcPD = (Sp == 3) ? Tc3 : Tc7;

tcQD = (Sq == 3) ? Tc3 : Tc7;

Đối với các đường biên P hoặc Q được lọc với bộ lọc đối xứng ngắn, ngưỡng phụ thuộc vị trí có độ lớn nhỏ hơn được áp dụng:

Tc3 = {3, 2, 1};

Dưới đây định nghĩa ngưỡng, các giá trị mẫu p'_i và q'_i được lọc được rút gọn theo các giá trị rút gọn tcP và tcQ:

$p''_i = Clip3(p'_i + tcP_i, p'_i - tcP_i, p'_i);$

$q''_j = Clip3(q'_j + tcQ_j, q'_j - tcQ_j, q'_j);$

trong đó p'_i và q'_i là các giá trị mẫu được lọc, p''_i và q''_j là giá trị mẫu đầu ra sau khi rút gọn và tcP_i tcP_i là các ngưỡng xén được dẫn xuất từ tham số tc VVC và *tcPD* và *tcQD*. Hàm Clip3 là hàm rút gọn như được nêu trong VVC.

2.4.8 Điều chỉnh tách khối khối phụ

Để có thể tách khối thân thiện song song nhờ sử dụng cả bộ lọc dài và tách khối khối phụ các bộ lọc dài bị giới hạn để chỉnh sửa nhiều nhất 5 mẫu ở phía sử dụng tách khối khối phụ (AFFINE hoặc ATMVP hoặc DMVR) như được thể hiện trong điều khiển độ sáng đối với các bộ lọc dài. Ngoài ra, tách khối khối phụ được điều chỉnh sao cho việc các đường biên khối phụ trên lưới 8x8 gần với CU hoặc đường biên TU ngầm bị giới hạn để chỉnh sửa nhiều nhất hai mẫu ở mỗi phía.

Phần sau áp dụng cho các đường biên khối phụ không được căn chỉnh với đường biên CU.

```
If (mode block Q == SUBBLOCKMODE && edge !=0) {
 if (!(implicitTU && (edge == (64/4))))
  if (edge == 2 || edge == (orthogonalLength - 2) || edge == (56/4) || edge == (72/4))
    Sp = Sq = 2;
  else
    Sp = Sq = 3;
 else
  Sp = Sq = bSideQisLargeBlk ? 5:3
}
```

Trong đó edge bằng 0 tương ứng với đường biên CU, edge bằng 2 hoặc bằng orthogonalLength-2 tương ứng với 8 mẫu đường biên khối phụ từ đường biên CU v.v.. Trong đó TU ngầm là đúng nếu tách ngầm TU được sử dụng.

2.5 SAO

Đầu vào của SAO là các mẫu được tái tạo sau DB. Khái niệm của SAO là giảm méo dạng mẫu trung bình của vùng bằng cách trước hết phân loại các mẫu vùng thành nhiều phân loại có bộ phân loại được chọn, thu được độ lệch đối với mỗi phân loại, và sau đó bổ sung độ lệch vào mỗi mẫu của phân loại, trong đó chỉ số bộ phân loại và các độ lệch của vùng được mã hóa trong dòng bit. Trong chuẩn HEVC và VVC, vùng (đơn vị để báo hiệu tham số SAO) được định nghĩa là CTU.

Hai loại SAO có thể thỏa mãn các yêu cầu có độ phức tạp thấp được chấp thuận trong HEVC. Hai loại đó là độ lệch mép (EO) và độ lệch băng (BO), được nêu chi tiết hơn dưới đây. Chỉ số của loại SAO được mã hóa (nằm trong khoảng [0, 2]). Đối với EO, phân loại mẫu dựa trên so sánh giữa các mẫu hiện tại và các mẫu lân cận theo các mẫu hình định hướng 1-D: ngang, dọc, đường chéo 135°, và đường chéo 45°.

Fig.8 thể hiện bốn mẫu hình định hướng 1-D để phân loại mẫu EO: ngang (lớp EO = 0), dọc (lớp EO = 1), đường chéo 135° (lớp EO = 2), và đường chéo 45° (lớp EO = 3)

Đối với lớp EO cụ thể, mỗi mẫu bên trong CTB được phân loại thành một trong năm phân loại. Giá trị mẫu hiện tại, được dán nhãn là "c," được so sánh với hai lân cận dọc theo mẫu hình 1-D được chọn. Các quy tắc phân loại cho mỗi mẫu được tóm tắt trong Bảng I. Các phân loại 1 và 4 được liên kết với điểm lõm nội bộ và điểm đỉnh nội bộ dọc theo mẫu hình 1-D được chọn, một cách lần lượt. Các phân loại 2 và 3 được liên kết với các góc lồi và lõm dọc theo mẫu hình 1-D được chọn, một cách lần lượt. Nếu mẫu hiện tại không thuộc EO các phân loại 1–4, thì nó là phân loại 0 và SAO không được áp dụng.

Bảng 3: Các quy tắc phân loại mẫu đối với EO

Phân loại	Điều kiện
1	c<a và c<b
2	(c < a && c==b) ||(c == a && c < b)
3	(c > a && c==b) ||(c == a && c > b)
4	c > a && c > b
5	Không điều kiện nào trong các điều kiện trên

2.6 Bộ lọc vòng thích ứng dựa trên biến đổi hình học

Đầu vào của DB là các mẫu được tái tạo sau DB và SAO. **Quá trình lọc và phân loại mẫu dựa trên các mẫu được tái tạo sau DB và SAO.**

Theo một số phương án thực hiện, ALF dựa trên biến đổi hình học (GALF) có thích ứng lọc dựa trên khối được áp dụng. Đối với thành phần độ sáng, một bộ lọc trong số 25 bộ lọc được chọn cho mỗi khối 2×2, dựa trên hướng và độ hoạt động của gradien cục bộ.

2.6.1 Hình dạng bộ lọc

Theo một số phương án thực hiện, có tới ba hình dạng bộ lọc kim cương (như được thể hiện trên Fig.9) có thể được chọn đối với thành phần độ sáng. Chỉ số được báo hiệu ở mức ảnh để chỉ báo hình dạng bộ lọc được sử dụng đối với thành phần độ sáng. Mỗi hình vuông là mẫu, và Ci (i bằng từ 0 đến 6 (bên trái), từ 0 đến 12 (giữa), từ 0 đến 20 (bên phải)) ký hiệu hệ số cần được áp dụng cho mẫu. Đối với các thành phần sắc độ trong ảnh, hình kim cương 5×5 luôn được sử dụng.

2.6.1.1 Phân loại khối

Mỗi khối 2×2 được phân loại thành một trong 25 lớp. Chỉ số phân loại C được dẫn xuất dựa trên hướng D và giá trị hoạt động được lượng tử $\hat{A}$, như sau:

$$C = 5D + \hat{A}. \tag{1}$$

Để tính toán D và $\hat{A}$, các gradien của ngang, dọc và hai hướng đường chéo trước hết được tính toán nhờ sử dụng 1-D Laplacian:

$$g_v = \sum_{k=i-2}^{i+3} \sum_{l=j-2}^{j+3} V_{k,l}, \; V_{k,l} = |2R(k,l) - R(k,l-1) - R(k,l+1)|, \tag{2}$$

$$g_h = \sum_{k=i-2}^{i+3} \sum_{l=j-2}^{j+3} H_{k,l}, \; H_{k,l} = |2R(k,l) - R(k-1,l) - R(k+1,l)|, \tag{3}$$

$$g_{d1} = \sum_{k=i-2}^{i+3} \sum_{l=j-3}^{j+3} D1_{k,l}, D1_{k,l} = |2R(k,l) - R(k-1,l-1) - R(k+1,l+1)| \tag{4}$$

$$g_{d2} = \sum_{k=i-2}^{i+3} \sum_{j=j-2}^{j+3} D2_{k,l}, \; D2_{k,l} = |2R(k,l) - R(k-1,l+1) - R(k+1,l-1)| \tag{5}$$

Các chỉ số i và j là các tọa độ của mẫu trên bên trái trong khối 2×2 và $R(i,j)$ chỉ báo mẫu được tái tạo ở tọa độ (i,j).

Sau đó, D giá trị lớn nhất và nhỏ nhất của các gradien của các hướng ngang và dọc được thiết lập bằng:

$$g_{h,v}^{max} = max(g_h, g_v), \;\; g_{h,v}^{min} = min(g_h, g_v), \tag{6}$$

và các giá trị lớn nhất và nhỏ nhất của gradien của hai hướng đường chéo được thiết lập bằng:

$$g_{d0,d1}^{max} = max(g_{d0}, g_{d1}), \;\; g_{d0,d1}^{min} = min(g_{d0}, g_{d1}), \tag{7}$$

Để dẫn xuất giá trị của hướng D, các giá trị này được so sánh với nhau và với hai ngưỡng t_1 và t_2:

Bước 1. Nếu cả $g_{h,v}^{max} \leq t_1 \cdot g_{h,v}^{min}$ và $g_{d0,d1}^{max} \leq t_1 \cdot g_{d0,d1}^{min}$ là true, D được gán

bằng 0.

Bước 2. Nếu $g_{h,v}^{max}/g_{h,v}^{min} > g_{d0,d1}^{max}/g_{d0,d1}^{min}$, tiếp tục từ Bước 3; ngược lại tiếp tục từ Bước 4.

Bước 3. Nếu $g_{h,v}^{max} > t_2 \cdot g_{h,v}^{min}$, D được gán bằng 2; ngược lại D được gán bằng 1.

Bước 4. Nếu $g_{d0,d1}^{max} > t_2 \cdot g_{d0,d1}^{min}$, D được gán bằng 4; ngược lại D được gán bằng 3.

Giá trị hoạt động A được tính toán là:

$$A = \sum_{k=i-2}^{i+3} \sum_{l=j-2}^{j+3} \left(V_{k,l} + H_{k,l}\right). \tag{8}$$

A còn được lượng tử đến khoảng từ 0 đến 4, bao gồm hai giá trị đầu mút, và giá trị được lượng tử được ký hiệu là $\hat{A}$.

Đối với cả hai thành phần sắc độ trong ảnh, không phương pháp phân loại nào được áp dụng, chẳng hạn, một tập hợp hệ số ALF được áp dụng cho mỗi thành phần sắc độ.

2.6.1.2 Các phép biến đổi hình học của các hệ số lọc

Fig.10 thể hiện tọa độ tương đối để hỗ trợ lọc kim cương 5×5: Bên trái: Đường chéo; Trung tâm: Lật dọc, Bên phải: quay.

Trước khi lọc mỗi khối 2×2, các phép biến đổi hình học, chẳng hạn, chuyển động quay hoặc đường chéo và lật dọc được áp dụng cho các hệ số lọc $f(k, l)$, được liên kết với tọa độ (k, l), tùy thuộc vào các giá trị gradien được tính toán cho khối đó. Điều này tương đương với việc áp dụng các phép biến đổi này cho các mẫu trong vùng hỗ trợ lọc. Ý tưởng là khiến các khối khác nhau mà ALF được áp dụng giống nhau hơn bằng cách căn chỉnh hướng của nó.

Ba phép biến đổi hình học, bao gồm đường chéo, lật dọc và chuyển động quay được đưa vào:

$$\begin{aligned} &\text{Đường chéo: } f_D(k, l) = f(l, k), \\ &\text{Lật dọc: } f_V(k, l) = f(k, K - l - 1), \\ &\text{Quay: } f_R(k, l) = f(K - l - 1, k). \end{aligned} \tag{9}$$

trong đó K là kích thước của bộ lọc và $0 \leq k, l \leq K - 1$ là các tọa độ hệ số,

sao cho vị trí (0,0) ở góc trên bên trái và vị trí $(K-1, K-1)$ ở góc dưới bên phải. Các phép biến đổi được áp dụng cho các hệ số lọc $f(k, l)$ tùy thuộc vào các giá trị gradien được tính toán cho khối đó. Mối quan hệ giữa phép biến đổi và bốn gradien của bốn hướng được tóm tắt trong Bảng 4. Fig.9 thể hiện các hệ số được biến đổi cho mỗi vị trí dựa trên hình kim cương 5x5.

Bảng 4: Ánh xạ của gradien được tính toán cho một khối và các phép biến đổi

Các giá trị gradien	Biến đổi
$g_{d2} < g_{d1}$ và $g_h < g_v$	Không biến đổi
$g_{d2} < g_{d1}$ và $g_v < g_h$	Đường chéo
$g_{d1} < g_{d2}$ và $g_h < g_v$	Lật dọc
$g_{d1} < g_{d2}$ và $g_v < g_h$	Quay

2.6.1.3 Báo hiệu các tham số lọc

Theo một số phương án thực hiện, các tham số lọc GALF được báo hiệu cho CTU thứ nhất, chẳng hạn, sau tiêu đề lát và trước các tham số SAO của CTU thứ nhất. Có tới 25 tập hợp hệ số lọc độ sáng có thể được báo hiệu. Để giảm phụ tải bit, các hệ số lọc thuộc phân loại khác có thể được hợp nhất. Các hệ số GALF của các ảnh tham chiếu cũng được lưu trữ và được phép sử dụng lại dưới dạng các hệ số GALF của ảnh hiện tại. ảnh hiện tại có thể chọn sử dụng các hệ số GALF được lưu trữ đối với các ảnh tham chiếu và bỏ qua báo hiệu các hệ số GALF. Trong trường hợp này, riêng chỉ số đến một trong các ảnh tham chiếu được báo hiệu, và các hệ số GALF được lưu trữ của ảnh tham chiếu được chỉ báo được kế thừa đối với ảnh hiện tại.

Để hỗ trợ dự báo thời gian GALF, danh sách ứng viên của các tập hợp lọc GALF được duy trì. Lúc bắt đầu giải mã chuỗi mới, danh sách ứng viên trống. Sau khi giải mã một ảnh, tập hợp bộ lọc tương ứng có thể được bổ sung vào danh sách ứng viên. Khi kích thước của danh sách ứng viên đạt đến giá trị được phép lớn nhất (chẳng hạn, 6), tập hợp bộ lọc mới ghi đè lên tập hợp cũ nhất theo thứ tự giải mã, và tức là, quy tắc vào trước ra trước (FIFO) được áp dụng để cập nhật danh sách ứng viên. Để tránh trùng lặp, tập hợp chỉ có thể được bổ sung vào danh sách khi ảnh tương ứng không sử dụng dự báo thời gian GALF. Để hỗ trợ tính khả mở theo thời gian, có nhiều danh sách ứng viên của các tập hợp bộ lọc, và mỗi danh sách ứng viên được liên kết với lớp thời gian. Cụ thể hơn, mỗi mảng

được gán bởi chỉ số lớp thời gian (TempIdx) có thể bao gồm các tập hợp bộ lọc của các ảnh được giải mã trước đó với TempIdx bằng hoặc nhỏ hơn. Chẳng hạn, mảng thứ k được gán để liên kết với TempIdx bằng k, và nó chỉ chứa các tập hợp bộ lọc từ các ảnh có TempIdx nhỏ hơn hoặc bằng k. Sau khi mã hóa ảnh cụ thể, các tập hợp bộ lọc được liên kết với ảnh sẽ được sử dụng để cập nhật các mảng đó được liên kết với TempIdx bằng hoặc cao hơn.

Dự báo thời gian của các hệ số GALF được sử dụng cho các khung được mã hóa ngoài để giảm tối đa phụ tải báo hiệu. Đối với các khung nội vùng, dự báo thời gian không khả dụng, và tập hợp 16 bộ lọc cố định được gán cho mỗi lớp. Để chỉ báo việc sử dụng bộ lọc cố định, cờ đối với mỗi lớp được báo hiệu và nếu cần, chỉ số của bộ lọc cố định được chọn. Thậm chí khi bộ lọc cố định được chọn cho lớp cụ thể, các hệ số của bộ lọc thích ứng $f(k,l)$ vẫn có thể được gửi cho lớp này trong trường hợp đó các hệ số của bộ lọc mà sẽ được áp dụng cho ảnh được tái tạo là tổng của cả hai tập hợp hệ số.

Quá trình lọc của thành phần độ sáng có thể được điều khiển ở mức CU. Cờ được báo hiệu để chỉ báo liệu GALF được áp dụng cho thành phần độ sáng của CU. Đối với thành phần sắc độ, liệu GALF có được áp dụng hay không được chỉ báo ở riêng mức ảnh.

2.6.1.4 Quá trình lọc

Ở phía bộ giải mã, khi GALF được kích hoạt đối với khối, mỗi mẫu $R(i,j)$ trong khối được lọc, thu được giá trị mẫu $R'(i,j)$ như được thể hiện dưới đây, trong đó L ký hiệu chiều dài bộ lọc, $f_{m,n}$ là hệ số lọc, và $f(k,l)$ ký hiệu các hệ số lọc được giải mã.

$$R'(i,j) = \sum_{k=-L/2}^{L/2} \sum_{l=-L/2}^{L/2} f(k,l) \times R(i+k, j+l) \quad (10)$$

Fig.11 thể hiện ví dụ của các tọa độ tương đối được sử dụng để hỗ trợ lọc kim cương 5x5 giả sử tọa độ của mẫu hiện tại (i, j) là (0, 0). Các mẫu trong các tọa độ khác nhau được điền đầy màu giống nhau được nhân với các hệ số lọc tương tự.

2.7 ALF dựa trên biến đổi hình học (GALF)

2.7.1 GALF lấy làm ví dụ

Theo một số phương án thực hiện, quá trình lọc của ALF, được thực hiện như

sau:

$$O(x,y) = \sum_{(i,j)} w(i,j).I(x+i,y+j), \quad (11)$$

trong đó các mẫu $I(x+i,y+j)$ là các mẫu đầu vào, $O(x,y)$ là mẫu đầu ra được lọc (chẳng hạn, kết quả lọc), và $w(i,j)$ ký hiệu các hệ số lọc. Thực tế, trong VTM4.0 nó được triển khai nhờ sử dụng phép tính số học số nguyên để tính độ chính xác dấu chấm tĩnh:

$$O(x,y) = \left(\sum_{i=-\frac{L}{2}}^{\frac{L}{2}} \sum_{j=-\frac{L}{2}}^{\frac{L}{2}} w(i,j).I(x+i,y+j) + 64\right) \gg 7, \quad (12)$$

trong đó L ký hiệu chiều dài bộ lọc, và trong đó $w(i,j)$ là các hệ số lọc trong phép tính độ chính xác dấu chấm tĩnh.

Thiết kế hiện tại của GALF trong VVC có các thay đổi chính sau:

(1) Hình dạng bộ lọc thích ứng được loại bỏ. Chỉ hình dạng bộ lọc 7x7 được phép đối với thành phần độ sáng và hình dạng bộ lọc 5x5 được phép đối với thành phần sắc độ.

(2) Báo hiệu các tham số ALF được loại bỏ khỏi mức lát/ảnh đến mức CTU.

(3) Tính toán chỉ số lớp được thực hiện ở mức 4x4 thay vì 2x2. Ngoài ra, theo một số phương án thực hiện, tính toán Laplacian được lấy mẫu phụ phương pháp phân loại ALF được sử dụng. Cụ thể hơn, không cần tính toán các gradien ngang/dọc/ đường chéo 45°/135° đối với mỗi mẫu trong một khối. Thay vào đó, lấy mẫu phụ 1:2 được sử dụng.

Fig.12A đến Fig.12D thể hiện tính toán Laplacian được lấy mẫu phụ với CE2.6.2. Fig.12A minh họa các vị trí được lấy mẫu phụ cho gradien dọc, Fig.12 minh họa các vị trí được lấy mẫu phụ đối với gradien ngang, Fig.12C minh họa các vị trí được lấy mẫu phụ đối với gradien đường chéo, và Fig.12D minh họa các vị trí được lấy mẫu phụ đối với gradien đường chéo.

2.8 ALF phi tuyến tính

2.8.1 Tạo lại lọc

Phương trình (11) có thể được tạo dạng lại, mà không tác động đến hiệu suất mã hóa, trong biểu thức sau:

$$O(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).\big(I(x+i,y+j) - I(x,y)\big), \quad (13)$$

trong đó $w(i,j)$ là các hệ số lọc giống nhau như trong phương trình (11) [ngoại trừ $w(0,0)$ bằng 1 trong phương trình (13) trong khi bằng $1-\sum_{(i,j)\neq(0,0)} w(i,j)$ trong phương trình (11)].

Nhờ sử dụng công thức lọc nêu trên (13), VVC đưa vào phi tuyến tính để khiến ALF hiệu quả hơn bằng cách sử dụng hàm rút gọn đơn giản để giảm tác động của các giá trị mẫu lân cận ($I(x+i,y+j)$) khi chúng quá khác với giá trị mẫu hiện tại ($I(x,y)$) được lọc.

Cụ thể hơn, bộ lọc ALF được chỉnh sửa như sau:

$$O'(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).K(I(x+i,y+j) - I(x,y), k(i,j)), \quad (14)$$

trong đó $K(d,b) = \min(b, \max(-b,d))$ là hàm rút gọn, và $k(i,j)$ là các tham số rút gọn, mà phụ thuộc vào (i,j) hệ số lọc. Bộ mã hóa thực hiện tối ưu hóa để tìm $k(i,j)$ tốt nhất.

Theo một số phương án thực hiện, các tham số rút gọn $k(i,j)$ được xác định cho mỗi bộ lọc ALF, một giá trị rút gọn được báo hiệu theo hệ số lọc. Nghĩa là có tới 12 giá trị rút gọn có thể được báo hiệu trong dòng bit theo bộ lọc độ sáng và có tới 6 giá trị rút gọn cho bộ lọc sắc độ.

Để giới hạn chi phí báo hiệu và độ phức tạp của bộ mã hóa, chỉ 4 giá trị cố định giống nhau đối với các lát INTER và INTRA được sử dụng.

Do phương sai của các hiệu số cục bộ thường cao hơn đối với độ sáng so với đối với sắc độ, hai tập hợp khác nhau đối với các bộ lọc độ sáng và sắc độ được áp dụng. Giá trị mẫu lớn nhất (ở đây là 1024 đối với chiều sâu bit 10 bit) trong mỗi tập cũng được đưa vào, sao cho việc rút gọn có thể bị vô hiệu hóa nếu không cần thiết.

Theo một số phương án thực hiện, tập hợp giá trị rút gọn được sử dụng được nêu trong Bảng 5. 4 giá trị đã được chọn bằng cách phân tách gần như bằng nhau, trong miền lôgarit, toàn bộ khoảng giá trị mẫu (được mã hóa trên 10 bit) đối với độ sáng, và khoảng từ 4 đến 1024 đối với sắc độ.

Chính xác hơn, bảng độ sáng của giá trị rút gọns đã thu được bởi công thức sau:

$$\text{AlfClip}_L = \left\{ \text{round}\left(\left((M)^{\frac{1}{N}} \right)^{N-n+1} \right) \text{ for } n \in 1..N \right\}, \text{ với M=}2^{10} \text{ và N=4.} \quad (15)$$

Tương tự, các bảng sắc độ của các giá trị rút gọn thu được theo công thức sau:

$$\text{AlfClipc}= \left\{ \text{round}\left(A.\left(\left(\frac{M}{A}\right)^{\frac{1}{N-1}} \right)^{N-n} \right) \text{ for } n \in 1..N \right\}, \text{ với } M=2^{10},\ N=4 \text{ và } A=4. \quad (16)$$

Bảng 5: Các giá trị rút gọn được ủy quyền

	Nhóm ô INTRA/INTER
LUMA	{1024, 181, 32, 6}
CHROMA	{1024, 161, 25, 4}

Các giá trị rút gọn được chọn được mã hóa trong phần tử cú pháp "alf_data" nhờ sử dụng phương tiện mã hóa Golomb tương ứng với chỉ số của giá trị rút gọn trong Bảng 5 nêu trên. Phương tiện mã hóa này giống như phương tiện mã hóa cho chỉ số lọc.

2.9 Đường biên ảo

Trong phần cứng và phần mềm được nhúng, xử lý dựa trên ảnh trên thực tế không được chấp nhận do yêu cầu đệm ảnh cao. Nhờ sử dụng bộ đệm ảnh trên chip rất đắt đỏ và sử dụng bộ đệm ảnh ngoài chip tăng đáng kể truy nhập bộ nhớ ngoài, tiêu thụ công suất, và độ trễ truy nhập dữ liệu. Do vậy, DF, SAO, và ALF sẽ được thay đổi từ giải mã dựa trên ảnh đến dựa trên LCU trong các sản phẩm thực tế. Khi xử lý dựa trên LCU được sử dụng để DF, SAO, và ALF, toàn bộ quá trình giải mã có thể được thực hiện Theo từng LCU theo kiểu quét mành có đường ống LCU để xử lý song song có nhiều LCU. Trong trường hợp này, các bộ đệm dòng được yêu cầu đối với DF, SAO, và ALF do việc xử lý một hàng LCU yêu cầu các pixel từ hàng LCU nêu trên. Nếu các bộ đệm dòng ngoài chip (chẳng hạn, DRAM) được sử dụng, băng thông bộ nhớ ngoài và tiêu thụ công suất sẽ được tăng lên; nếu các bộ đệm dòng ngoài chip (chẳng hạn, SRAM) được sử dụng, diện tích chip sẽ được tăng lên. Do vậy, mặc dù các bộ đệm dòng đã nhỏ hơn nhiều so với bộ đệm ảnh, vẫn mong muốn giảm bộ đệm dòng.

Theo một số phương án thực hiện, như được thể hiện trên Fig.13, tổng số bộ đệm dòng được yêu cầu là 11,25 các dòng đối với thành phần độ sáng. Giải thích về yêu cầu bộ đệm dòng như sau: Việc tách khối cạnh ngang chồng lấp với cạnh CTU không thể được thực hiện dưới dạng các quyết định và việc lọc yêu cầu các

dòng K, L, M, M từ CTU thứ nhất và các dòng O, P từ CTU dưới. Do vậy, việc tách khối các cạnh ngang chồng lấp với đường biên CTU bị hủy bỏ cho đến khi đến CTU dưới. Do vậy đối với các dòng K, L, M, N các mẫu độ sáng được tái tạo phải được lưu trữ trong bộ đệm dòng (4 dòng). Sau đó, bộ lọc SAO có thể được thực hiện với các dòng từ A đến J. Dòng J có thể được lọc SAO do việc tách khối không thay đổi các mẫu trong dòng K. Để lọc SAO của dòng K, quyết định phân loại EO chỉ được lưu trữ trong bộ đệm dòng (vốn là 0,25 dòng độ sáng). Việc lọc ALF có thể chỉ được thực hiện đối với các dòng từ A đến F. Như được thể hiện trên Fig.13, phân loại ALF được thực hiện cho mỗi khối 4x4. Phương pháp phân loại mỗi khối 4x4 cần cửa sổ hoạt động có kích thước 8 x 8 mà sau đó cần cửa sổ 9x9 để tính toán Laplacian 1D để xác định gradien.

Do vậy, để phân loại khối của khối 4x4 chồng lấp với các dòng G, H, I, J cần các mẫu được lọc SAO dưới đường biên ảo. Ngoài ra, các mẫu được lọc SAO của các dòng D, E, F được yêu cầu để phân loại ALF. Ngoài ra, lọc ALF của dòng G cần ba dòng được lọc SAO D, E, F từ các dòng nêu trên. Do vậy, toàn bộ yêu cầu bộ đệm dòng như sau:

- Các dòng từ K đến N (các pixel DF ngang): 4 dòng
- Các dòng từ D đến J (các pixel được lọc SAO): 7 dòng
- Các giá trị bộ phân loại EO SAO giữa dòng J và dòng K: 0,25 dòng

Do vậy, tổng số dòng độ sáng được yêu cầu bằng 7 + 4 + 0,25 = 11,25.

Tương tự, yêu cầu bộ đệm dòng của thành phần sắc độ được minh họa trên Fig.14. Yêu cầu bộ đệm dòng đối với thành phần sắc độ được nâng lên thành 6,25 dòng.

Để loại bỏ yêu cầu các bộ đệm dòng của SAO và ALF, khái niệm của đường biên ảo (VB) được đưa vào VVC mới nhất. Như được thể hiện trên Fig.13, các VB được dịch lên phía trên N pixel theo các đường biên LCU ngang. Đối với mỗi LCU, SAO và ALF có thể xử lý các pixel phía trên VB trước khi đến LCU dưới nhưng không thể xử lý các pixel dưới VB cho đến khi đến LCU dưới, gây ra bởi DF. Khi xem xét chi phí lắp đặt phần cứng, không gian giữa VB được đề xuất và đường biên LCU ngang được gán bằng bốn pixel đối với độ sáng (chẳng hạn, N=4 trên Fig.13) và hai pixel đối với sắc độ (chẳng hạn, N=2 trên Fig.9).

2.9.1 Phân loại khối ALF được chỉnh sửa khi kích thước VB N bằng 4

Fig.15A và Fig.15B mô tả phân loại khối được chỉnh sửa đối với trường hợp khi đường biên ảo bằng 4 dòng phía trên đường biên CTU (N = 4). Như được mô tả trên Fig.15A, đối với khối 4x4 bắt đầu ở dòng G, phân loại khối chỉ sử dụng các dòng từ E đến J. Tuy nhiên, việc tính toán građien Laplacian đối với các mẫu thuộc dòng J đòi hỏi thêm một dòng nữa dưới đây (dòng K). Do vậy, dòng K được đệm với dòng J.

Tương tự, như được mô tả trên Fig.15B, đối với khối 4x4 bắt đầu ở dòng K, phân loại khối chỉ sử dụng các dòng từ K đến P. Tuy nhiên, việc tính toán građien Laplacian đối với các mẫu thuộc dòng K đòi hỏi thêm một dòng phía trên (dòng J). Do vậy, dòng J được đệm với dòng K.

2.9.2 Đệm hai phía cho các mẫu giao với các đường biên ảo

Như được mô tả trên các hình vẽ từ Fig.16A đến Fig.16C, phiên bản rút gọn của các bộ lọc được sử dụng để lọc các mẫu độ sáng thuộc các dòng gần với các đường biên ảo. Lấy Fig.16A làm ví dụ, khi lọc dòng M như được ký hiệu trên Fig.13, chẳng hạn, mẫu giữa của hỗ trợ kim cương 7x7 ở dòng M. yêu cầu truy nhập một dòng phía trên VB (ký hiệu bởi đường nét đậm). Trong trường hợp này, các mẫu phía trên VB được sao chép từ mẫu dưới bên phải dưới VB, chẳng hạn mẫu P0 trong đường nét liền được sao chép với vị trí nét đứt phía trên. Một cách đối xứng, mẫu P3 trong đường nét liền cũng được sao chép đến vị trí nét đứt dưới bên phải thậm chí mẫu đối với vị trí đó là khả dụng. Các mẫu được sao chép chỉ được sử dụng trong quá trình lọc độ sáng.

Phương pháp đệm được sử dụng cho các đường biên ảo ALF có thể được ký hiệu là 'Đệm hai phía' trong đó nếu một mẫu được đặt ở (i, j) (chẳng hạn, P0A có đường nét đứt trên Fig.16B) được đệm, thì mẫu tương ứng được đặt ở (m, n) (chẳng hạn, P3B có đường nét đứt trên Fig.16B) mà chia sẻ hệ số lọc tương tự cũng được đệm thậm chí khi mẫu khả dụng, như được mô tả trên các hình vẽ Fig.16A đến Fig.16C và Fig.17A và Fig.17B. Trên các hình vẽ Fig.16A đến Fig.16C, hỗ trợ lọc kim cương 7x7, tâm là mẫu hiện tại cần được lọc. Fig.16A thể hiện một đường được yêu cầu phía trên/phía dưới VB cần được đệm. Fig.16B thể hiện 2 dòng được yêu cầu trên/dưới VB cần được đệm. Fig.16C thể hiện 3

dòng được yêu cầu trên/dưới VB cần được đệm.

Tương tự, như được mô tả trên các hình vẽ Fig.17A và Fig.17B, phương pháp đệm hai phía cũng được sử dụng để lọc ALF sắc độ. Fig.17A và Fig.17B thể hiện bước lọc ALF sắc độ được chỉnh sửa ở đường biên ảo (5x5 hỗ trợ lọc kim cương, tâm là mẫu hiện tại cần được lọc). Fig.17A thể hiện 1 dòng được yêu cầu trên/dưới VB cần được đệm. Fig.17B thể hiện 2 dòng được yêu cầu trên/dưới VB cần được đệm.

2.9.3 Cách khác để thực hiện Đệm hai phía khi ALF phi tuyến tính bị vô hiệu hóa

Khi ALF phi tuyến tính bị vô hiệu hóa đối với CTB, chẳng hạn, các tham số rút gọn $k(i,j)$ trong phương trình (14) bằng (1<< Bitdepth), quá trình đệm có thể được thay thế bằng cách chỉnh sửa các hệ số lọc (cũng được biết đến là ALF dựa trên hệ số được chỉnh sửa, MALF). Chẳng hạn, khi lọc các mẫu ở dòng L/I, hệ số lọc c5 được cải biến thành c5', trong trường hợp này, không cần sao chép các mẫu độ sáng từ P0A nét liền sang P0A nét đứt và P3B nét liền sang P3B nét đứt Fig.18A. Trong trường hợp này, việc đệm hai phía và MALF sẽ tạo các kết quả giống nhau, giả sử mẫu hiện tại cần được lọc ở được đặt ở (x, y).

$$c5.K\big(I(x-1,y-1)-I(x,y),k(-1,-1)\big)+c1.K\big(I(x-1,y-2)-I(x,y),k(-1,-2)\big)$$
$$=(c5+c1).K(I(x-1,y-1)-I(x,y),k(-1,-1)) \quad (17)$$

do K(d, b) = d và $I(x-1,y-1)= I(x-1,y-2)$ do đệm.

Tuy nhiên, khi ALF phi tuyến tính được kích hoạt, MALF và đệm hai phía có thể tạo các kết quả được lọc khác nhau, do các tham số phi tuyến được liên kết với mỗi hệ số, chẳng hạn đối với các hệ số lọc c5 và c1, các tham số rút gọn là khác nhau. Do vậy,

$$c5.K\big(I(x-1,y-1)-I(x,y),k(-1,-1)\big)+c1.K\big(I(x-1,y-2)-I(x,y),k(-1,-2)\big)$$
$$!=(c5+c1).K(I(x-1,y-1)-I(x,y),k(-1,-1)) \quad (18)$$

do K(d, b) != d, thậm chí $I(x-1,y-1)= I(x-1,y-2)$ do đệm.

2.10 Mô tả về lọc ALF

Các phần mới được thêm vào được chỉ báo trong đoạn in đậm nghiêng gạch chân. Các phần được xóa được chỉ báo nhờ sử dụng [[]].

7.3.2.4 Cú pháp RBSP của tập hợp tham số ảnh

pic_parameter_set_rbsp() {	Mô tả
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
output_flag_present_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
uniform_tile_spacing_flag	u(1)
if(uniform_tile_spacing_flag) {	
tile_cols_width_minus1	ue(v)
tile_rows_height_minus1	ue(v)
} else {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
brick_splitting_present_flag	u(1)
for(i = 0; brick_splitting_present_flag && i < NumTilesInPic; i++) {	
brick_split_flag[i]	u(1)
if(brick_split_flag[i]) {	
uniform_brick_spacing_flag[i]	u(1)
if(uniform_brick_spacing_flag[i])	
brick_height_minus1[i]	ue(v)
else {	
num_brick_rows_minus1[i]	ue(v)
for(j = 0; j < num_brick_rows_minus1[i]; j++)	
brick_row_height_minus1[i][j]	ue(v)
}	
}	
}	
single_brick_per_slice_flag	u(1)
if(!single_brick_per_slice_flag)	
rect_slice_flag	u(1)
if(rect_slice_flag && !single_brick_per_slice_flag) {	
num_slices_in_pic_minus1	ue(v)
for(i = 0; i <= num_slices_in_pic_minus1; i++) {	
if(i > 0)	
top_left_brick_idx[i]	u(v)
bottom_right_brick_idx_delta[i]	u(v)
}	
}	
loop_filter_across_bricks_enabled_flag	***u(1)***
if(loop_filter_across_bricks_enabled_flag)	
loop_filter_across_slices_enabled_flag	***u(1)***
}	
if(rect_slice_flag) {	
signalled_slice_id_flag	u(1)
if(signalled_slice_id_flag) {	
signalled_slice_id_length_minus1	ue(v)
for(i = 0; i <= num_slices_in_pic_minus1; i++)	
slice_id[i]	u(v)

}	
}	
entropy_coding_sync_enabled_flag	u(1)
cabac_init_present_flag	u(1)
for(i = 0; i < 2; i++)	
num_ref_idx_default_active_minus1[i]	ue(v)
rpl1_idx_present_flag	u(1)
init_qp_minus26	se(v)
transform_skip_enabled_flag	u(1)
if(transform_skip_enabled_flag)	
log2_transform_skip_max_size_minus2	ue(v)
cu_qp_delta_enabled_flag	u(1)
if(cu_qp_delta_enabled_flag)	
cu_qp_delta_subdiv	ue(v)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbcr_qp_offset	se(v)
pps_slice_chroma_qp_offsets_present_flag	u(1)
weighted_pred_flag	u(1)
weighted_bipred_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
deblocking_filter_override_enabled_flag	u(1)
pps_deblocking_filter_disabled_flag	u(1)
if(!pps_deblocking_filter_disabled_flag) {	
pps_beta_offset_div2	se(v)
pps_tc_offset_div2	se(v)
}	
}	
pps_loop_filter_across_virtual_boundaries_disabled_flag	u(1)
if(pps_loop_filter_across_virtual_boundaries_disabled_flag) {	
pps_num_ver_virtual_boundaries	u(2)
for(i = 0; i < pps_num_ver_virtual_boundaries; i++)	
pps_virtual_boundaries_pos_x[i]	u(v)
pps_num_hor_virtual_boundaries	u(2)
for(i = 0; i < pps_num_hor_virtual_boundaries; i++)	
pps_virtual_boundaries_pos_y[i]	u(v)
}	
pps_extension_flag	u(1)
if(pps_extension_flag)	
trong khi(more_rbsp_data())	
pps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

loop_filter_across_bricks_enabled_flag bằng 1 xác định rằng các hoạt động lọc trong vòng có thể được thực hiện giữa các đường biên của khối viên trong các ảnh chỉ đến PPS. loop_filter_across_bricks_enabled_flag bằng 0 xác định rằng các hoạt động lọc trong vòng không được thực hiện giữa các đường

biên của khối viên trong các ảnh chỉ đến PPS. Các hoạt động lọc trong vòng bao gồm các hoạt động DBF, lọc SAO, và ALF. Khi không xuất hiện, giá trị của loop_filter_across_bricks_enabled_flag được suy ra bằng 1.

loop_filter_across_slices_enabled_flag bằng 1 xác định rằng các hoạt động lọc trong vòng có thể được thực hiện giữa các đường biên lát trong các ảnh chỉ đến PPS. loop_filter_across_slice_enabled_flag bằng 0 xác định rằng các hoạt động lọc trong vòng không được thực hiện giữa các đường biên lát trong các ảnh chỉ đến PPS. Các hoạt động lọc trong vòng bao gồm các hoạt động DBF, lọc SAO, và ALF. Khi không xuất hiện, giá trị của loop_filter_across_slices_enabled_flag được suy ra bằng 0.

pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 xác định rằng các hoạt động lọc trong vòng bị vô hiệu hóa giữa các đường biên ảo trong các ảnh chỉ đến PPS. pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 0 xác định rằng không vô hiệu hóa các hoạt động lọc trong vòng được áp dụng trong các ảnh chỉ đến PPS. Các hoạt động lọc trong vòng bao gồm các hoạt động DBF, lọc SAO, và ALF. Khi không xuất hiện, giá trị của pps_loop_filter_across_virtual_boundaries_disabled_flag được suy ra bằng 0.

pps_num_ver_virtual_boundaries xác định số lượng phần tử cú pháp pps_virtual_boundaries_pos_x[i] xuất hiện trong PPS. Khi pps_num_ver_virtual_boundaries không xuất hiện, nó được suy ra bằng 0.

8.8.5.2 Quá trình lọc CTB đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

- mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF,
- mảng mẫu ảnh độ sáng được tái tạo được lọc $alfPicture_L$,
- vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa $alfPicture_L$.

Mệnh đề của quá trình dẫn xuất chỉ số lọc 8.8.5.3 được gọi với vị trí (xCtb, yCtb) và mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ làm các đầu vào, và filtIdx[x][y] và transposeIdx[x][y] với x, y = 0..CtbSizeY − 1 làm đầu ra.

Để dẫn xuất các mẫu độ sáng được tái tạo được lọc $alfPicture_L$[x][y], mỗi mẫu độ sáng được tái tạo bên trong CTB độ sáng hiện tại $recPicture_L$[x][y] được lọc như sau với x, y = 0..CtbSizeY − 1:

– Mảng của các hệ số lọc độ sáng f[j] và mảng của các giá trị rút gọn độ sáng c[j] tương ứng với bộ lọc được xác định bởi filtIdx[x][y] được dẫn xuất như sau với j = 0..11:

– Nếu AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] nhỏ hơn 16, phương trình sau áp dụng:

i = AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] (8-1172)

f[j] = AlfFixFiltCoeff[AlfClassToFiltMap[i][filtidx]][j] (8-1173)

$c[j] = 2^{BitdepthY}$ (8-1174)

– Ngược lại (AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] lớn hơn hoặc bằng 16, phương trình sau áp dụng:

i = slice_alf_aps_id_luma[AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] − 16] (8-1175)

f[j] = $AlfCoeff_L$[i][filtIdx[x][y]][j] (8-1176)

c[j] = $AlfClip_L$[i][filtIdx[x][y]][j] (8-1177)

– Các hệ số lọc độ sáng và chỉ số của các giá trị rút gọn idx được dẫn xuất tùy thuộc vào transposeIdx[x][y] như sau:

– Nếu transposeIndex[x][y] bằng 1, phương trình sau áp dụng:

idx[] = {9, 4, 10, 8, 1, 5, 11, 7, 3, 0, 2, 6} (8-1178)

– Ngược lại, nếu transposeIndex[x][y] bằng 2, phương trình sau áp dụng:

idx[] = {0, 3, 2, 1, 8, 7, 6, 5, 4, 9, 10, 11} (8-1179)

– Ngược lại, nếu transposeIndex[x][y] bằng 3, phương trình sau áp dụng:

idx[] = {9, 8, 10, 4, 3, 7, 11, 5, 1, 0, 2, 6} (8-1180)

– Ngược lại, phương trình sau áp dụng:

idx[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11} (8-1181)

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −3..3 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i)

(8-1182)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n]− xCtb − x lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i)

(8-1183)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1184)

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j)

(8-1185)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j)

(8-1186)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1187)

– ***Biến applyVirtualBoundary được dẫn xuất như sau:***

– ***Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:***

– ***Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.***

– ***Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.***

– ***Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.***

– ***Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.***

– ***Ngược lại, applyVirtualBoundary được gán bằng 1.***

– Các độ lệch mẫu được tái tạo r1, r2 và r3 được xác định trong Bảng 8-22 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary.

– Biến curr được dẫn xuất như sau:

curr = $recPicture_L[h_x, v_y]$ (8-1188)

– Biến sum được dẫn xuất như sau:

sum = f[idx[0]] * (Clip3(−c[idx[0]], c[idx[0]], $recPicture_L[h_x, v_{y+r3}]$ − curr) + Clip3(−c[idx[0]], c[idx[0]], $recPicture_L[h_x, v_{y-r3}]$ − curr)) +
f[idx[1]] * (Clip3(−c[idx[1]], c[idx[1]], $recPicture_L[h_{x+1}, v_{y+r2}]$ − curr) + Clip3(−c[idx[1]], c[idx[1]], $recPicture_L[h_{x-1}, v_{y-r2}]$ − curr)) +
f[idx[2]] * (Clip3(−c[idx[2]], c[idx[2]], $recPicture_L[h_x, v_{y+r2}]$ − curr) + Clip3(−c[idx[2]], c[idx[2]], $recPicture_L[h_x, v_{y-r2}]$ − curr)) +
f[idx[3]] * (Clip3(−c[idx[3]], c[idx[3]], $recPicture_L[h_{x-1}, v_{y+r2}]$ − curr) + Clip3(−c[idx[3]], c[idx[3]], $recPicture_L[h_{x+1}, v_{y-r2}]$ − curr)) +
f[idx[4]] * (Clip3(−c[idx[4]], c[idx[4]], $recPicture_L[h_{x+2}, v_{y+r1}]$ − curr) + Clip3(−c[idx[4]], c[idx[4]], $recPicture_L[h_{x-2}, v_{y-r1}]$ − curr)) +
f[idx[5]] * (Clip3(−c[idx[5]], c[idx[5]], $recPicture_L[h_{x+1}, v_{y+r1}]$ − curr) + Clip3(−c[idx[5]], c[idx[5]], $recPicture_L[h_{x-1}, v_{y-r1}]$ − curr)) +
f[idx[6]] * (Clip3(−c[idx[6]], c[idx[6]], $recPicture_L[h_x, v_{y+r1}]$ − curr) + Clip3(−c[idx[6]], c[idx[6]], $recPicture_L[h_x, v_{y-r1}]$ − curr)) + (8-1189)

f[idx[7]] * (Clip3(−c[idx[7]], c[idx[7]], recPicture$_L$[h_{x-1}, v_{y+r1}] − curr) +
Clip3(−c[idx[7]], c[idx[7]], recPicture$_L$[h_{x+1}, v_{y-r1}] − curr)) +
f[idx[8]] * (Clip3(−c[idx[8]], c[idx[8]], recPicture$_L$[h_{x-2}, v_{y+r1}] − curr) +
Clip3(−c[idx[8]], c[idx[8]], recPicture$_L$[h_{x+2}, v_{y-r1}] − curr)) +
f[idx[9]] * (Clip3(−c[idx[9]], c[idx[9]], recPicture$_L$[h_{x+3}, v_y] − curr) +
Clip3(−c[idx[9]], c[idx[9]], recPicture$_L$[h_{x-3}, v_y] − curr)) +
f[idx[10]] * (Clip3(−c[idx[10]], c[idx[10]], recPicture$_L$[h_{x+2}, v_y] − curr) +
Clip3(−c[idx[10]], c[idx[10]], recPicture$_L$[h_{x-2}, v_y] − curr)) +
f[idx[11]] * (Clip3(−c[idx[11]], c[idx[11]], recPicture$_L$[h_{x+1}, v_y] − curr) +
Clip3(−c[idx[11]], c[idx[11]], recPicture$_L$[h_{x-1}, v_y] − curr))
sum = curr + ((sum + 64) >> 7) (8-1190)

– Mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa alfPicture$_L$[xCtb + x][yCtb + y] được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[xCtb+ x][yCtb + y] đều bằng 1, phương trình sau áp dụng:

alfPicture$_L$[xCtb + x][yCtb + y] = recPicture$_L$[h_x, v_y] (8-1191)

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 hoặc pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

alfPicture$_L$[xCtb + x][yCtb + y] = Clip3(0, (1 << BitDepth$_Y$) − 1, sum) (8-1192)

Bảng 8-22 – Mô tả r1, r2, và r3 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary

Điều kiện	**r1**	**r2**	**r3**
(y = = CtbSizeY − 5 || y = = CtbSizeY − 4) && (applyVirtualBoundary = = 1)	0	0	0
(y = = CtbSizeY − 6 || y = = CtbSizeY − 3) && (applyVirtualBoundary = = 1)	1	1	1
(y = = CtbSizeY − 7 || y = = CtbSizeY − 2) && (applyVirtualBoundary = = 1)	1	2	2
ngược lại	1	2	3

8.8.5.4 Quá trình lọc CTB đối với các mẫu sắc độ

Các đầu vào của quá trình này là:

– mảng mẫu ảnh sắc độ được tái tạo recPicture trước quá trình ALF,

– mảng mẫu ảnh sắc độ được tái tạo được lọc alfPicture,

– vị trí sắc độ (xCtbC, yCtbC) xác định mẫu trên bên trái của CTB sắc độ hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture.

Chiều rộng và chiều cao của CTB sắc độ hiện tại ctbWidthC and ctbHeightC được dẫn xuất như sau:

ctbWidthC = CtbSizeY/SubWidthC (8-1230)

ctbHeightC = CtbSizeY/SubHeightC (8-1231)

Để dẫn xuất các mẫu sắc độ được tái tạo được lọc alfPicture[x][y], mỗi mẫu sắc độ được tái tạo bên trong CTB sắc độ hiện tại recPicture[x][y] được lọc như sau với x = 0..ctbWidthC − 1, y = 0..ctbHeightC − 1:

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu sắc độ tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu sắc độ với i, j = −2..2 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtbC + x − PpsVirtualBoundariesPosX[n]/SubWidthC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n]/SubWidthC, (8-1232)
pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n]/SubWidthC − xCtbC − x lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n]/SubWidthC − 1, xCtbC + x + i)
(8-1233)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i)
(8-1234)

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtbC + y − PpsVirtualBoundariesPosY[n]/SubHeightC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n]/SubHeightC , (8-1235)
pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n]/SubHeightC − yCtbC − y lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n]/SubHeightC − 1, yCtbC + y + j)
(8-1236)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j)
(8-1237)

– Biến applyVirtualBoundary được dẫn xuất như sau:

– ***Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:***

– ***Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.***

– ***Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.***

– ***Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.***

– ***Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.***

– ***Ngược lại, applyVirtualBoundary được gán bằng 1.***

– Các độ lệch mẫu được tái tạo r1 và r2 được xác định trong Bảng 8-22 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary.

– Biến curr được dẫn xuất như sau:

curr = recPicture[h_x, v_y] (8-1238)

– Mảng của các hệ số lọc sắc độ f[j] và mảng của các giá trị rút gọn sắc độ c[j] được dẫn xuất như sau với j = 0..5:

f[j] = AlfCoeff$_C$[slice_alf_aps_id_chroma][j] (8-1239)

c[j] = AlfClip$_C$[slice_alf_aps_id_chroma][j] (8-1240)

– Biến sum được dẫn xuất như sau:

$$\begin{aligned}sum = f[0] * (& Clip3(-c[0], c[0], recPicture[h_x, v_{y+r2}] - curr) + \\ & Clip3(-c[0], c[0], recPicture[h_x, v_{y-r2}] - curr)) + \\ f[1] * & (Clip3(-c[1], c[1], recPicture[h_{x+1}, v_{y+r1}] - curr) + \\ & Clip3(-c[1], c[1], recPicture[h_{x-1}, v_{y-r1}] - curr)) + \\ f[2] * & (Clip3(-c[2], c[2], recPicture[h_x, v_{y+r1}] - curr) + \\ & Clip3(-c[2], c[2], recPicture[h_x, v_{y-r1}] - curr)) + \\ f[3] * & (Clip3(-c[3], c[3], recPicture[h_{x-1}, v_{y+r1}] - curr) + \\ & Clip3(-c[3], c[3], recPicture[h_{x+1}, v_{y-r1}] - curr)) + \\ f[4] * & (Clip3(-c[4], c[4], recPicture[h_{x+2}, v_y] - curr) + \\ & Clip3(-c[4], c[4], recPicture[h_{x-2}, v_y] - curr)) + \\ f[5] * & (Clip3(-c[5], c[5], recPicture[h_{x+1}, v_y] - curr) + \\ & Clip3(-c[5], c[5], recPicture[h_{x-1}, v_y] - curr))\end{aligned} \quad (8\text{-}1241)$$

$$sum = curr + (sum + 64) >> 7) \quad (8\text{-}1242)$$

– Mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture[xCtbC + x][yCtbC + y] được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[(xCtbC + x) * SubWidthC][(yCtbC + y) * SubHeightC] đều bằng 1, phương trình sau áp dụng:

$$alfPicture[xCtbC + x][yCtbC + y] = recPicture_L[h_x, v_y] \quad (8\text{-}1243)$$

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 hoặc pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

$$alfPicture[xCtbC + x][yCtbC + y] = Clip3(0, (1 << BitDepth_C) - 1, sum) \quad (8\text{-}1244)$$

2.11 Ví dụ về xử lý CTU

Theo thiết kế VVC hiện tại, nếu đường biên dưới của one CTB là đường biên dưới của lát/khối viên, phương pháp xử lý đường biên ảo ALF bị vô hiệu hóa. Chẳng hạn, một ảnh được tách thành nhiều CTU và 2 lát như được mô tả Fig.19.

Giả sử kích thước CTU là MxM (chẳng hạn, M = 64), theo định nghĩa đường biên ảo, 4 dòng cuối cùng trong CTB được xử lý dưới đường biên ảo. Khi lắp đặt phần cứng, điều kiện sau áp dụng:

- Nếu đường biên dưới của CTB là đường biên dưới của ảnh (chẳng hạn, CTU-D), nó xử lý khối (M+4) × M bao gồm 4 dòng từ hàng CTU nêu trên và tất cả các dòng trong CTU hiện tại.

- Ngược lại, nếu đường biên dưới của CTB là đường biên dưới của lát (hoặc khối viên) (chẳng hạn, CTU-C) và loop_filter_across_slice_enabled_flag (hoặc loop_filter_across_bricks_enabled_flag) bằng 0, nó xử lý khối (M+4) × M bao gồm 4 dòng từ hàng CTU nêu trên và tất cả các dòng trong CTU hiện tại.

- Ngược lại, nếu CTU/CTB trong hàng CTU thứ nhất trong lát/khối viên/ô (chẳng hạn, CTU-A), nó xử lý khối M × (M-4) không bao gồm 4 dòng cuối cùng.

- Ngược lại, nếu CTU/CTB không thuộc hàng CTU thứ nhất của lát/khối viên/ô (chẳng hạn, CTU-B) và không nằm trong hàng CTU cuối cùng của lát/khối viên/ô, nó xử lý khối M × M bao gồm 4 dòng từ hàng CTU nêu trên và không bao gồm 4 dòng cuối cùng trong CTU hiện tại.

Fig.19 thể hiện ví dụ xử lý các CTU trong ảnh.

2.12 Mã hóa viđeo 360°

Quá trình bù chuyển động bao quanh ngang trong VTM5 là công cụ mã hóa cụ thể 360° được thiết kế để cải thiện chất lượng hình ảnh của viđeo 360° được tái tạo trong định dạng chiếu đẳng chữ nhật (ERP). Trong quá trình bù chuyển động đã biết, khi vectơ chuyển động đề cập đến các mẫu ngoài các đường biên ảnh của ảnh tham chiếu, bước đệm lặp lại được áp dụng để dẫn xuất các giá trị của các mẫu ngoài biên bằng cách sao chép từ các lân cận gần nhất đó trên đường biên ảnh tương ứng. Đối với viđeo 360°, phương pháp đệm lặp lại không thích hợp, và có thể gây ra các giả tượng hình ảnh được gọi là "các giả tượng nối" trong viđeo cổng nhìn được tái tạo. Do viđeo 360° được ghi nhận trên hình cầu và vốn đã không có "đường biên," các mẫu tham chiếu nằm ngoài các đường biên của ảnh tham chiếu trong miền được chiếu có thể luôn thu được từ các mẫu lân cận trong miền cầu. Đối với định dạng chiếu tổng quát, có thể khó dẫn xuất các mẫu lân cận tương ứng trong miền cầu, do nó bao gồm phép biến đổi tọa độ 2D sang 3D và 3D sang 2D, cũng như nội suy mẫu đối với các vị trí mẫu phân số. Vấn đề này đơn giản hơn nhiều đối với các đường biên trái và phải của định dạng chiếu ERP, do các lân cận hình cầu ngoài biên ảnh bên trái có thể thu được từ các mẫu bên trong đường biên ảnh bên phải, và ngược lại.

Fig.20 thể hiện ví dụ của bù chuyển động bao quanh ngang trong VVC.

Quá trình bù chuyển động bao quanh ngang như được mô tả trên Fig.20.

Khi một phần của khối tham chiếu nằm ngoài đường biên trái (hoặc phải) của ảnh tham chiếu trong miền được chiếu, thay cho bước đệm lặp lại, phần "ngoài biên" được lấy từ các lân cận hình cầu tương ứng của ảnh tham chiếu về phía đường biên phải (hoặc trái) trong miền được chiếu. **Bước đệm lặp lại chỉ được sử dụng để các đường biên ảnh trên và dưới**. Như được mô tả trên Fig.20, quá trình bù chuyển động bao quanh ngang có thể được kết hợp với phương pháp đệm không trực chuẩn thường được sử dụng trong mã hóa viđeo 360°. Trong chuẩn VVC, điều này đạt được bằng cách báo hiệu phần tử cú pháp mức cao để chỉ báo độ lệch bao quanh, nên được gán bằng chiều rộng ảnh ERP trước khi đệm; cú pháp này được sử dụng để điều chỉnh tương ứng vị trí bao quanh theo chiều ngang. Cú pháp này không bị ảnh hưởng bởi lượng đệm cụ thể trên các đường biên ảnh bên trái và bên phải, và do vậy về bản chất hỗ trợ đệm không đối xứng ảnh ERP, chẳng hạn, khi đệm trái và phải là khác nhau. Quá trình bù chuyển động bao quanh ngang cung cấp thông tin ý nghĩa hơn để bù chuyển động khi các mẫu tham chiếu nằm ngoài đường biên bên trái và bên phải của ảnh tham chiếu.

Đối với các định dạng chiếu bao gồm các mặt, bất kể loại bố trí đóng gói khung nhỏ gọn nào được sử dụng, các điểm gián đoạn xuất hiện giữa hai hoặc nhiều mặt liền kề trong ảnh được đóng khung. Chẳng hạn, xem xét cấu hình đóng gói khung 3×2 được mô tả trên Fig.24, ba mặt ở nửa trên liên tục trong hình học 3D, ba mặt trong nửa dưới liên tục trong hình học 3D, nhưng các nửa trên và dưới của ảnh đóng khung là gián đoạn trong hình học 3D. Nếu các hoạt động lọc trong vòng được thực hiện giữa điểm gián đoạn này, các giả tượng nối mặt có thể thấy rõ trong viđeo được tái tạo.

Để giảm bớt các giả tượng nối mặt, các hoạt động lọc trong vòng có thể bị vô hiệu hóa giữa các điểm gián đoạn trong ảnh được đóng khung. Cú pháp được đề xuất để báo hiệu các đường biên ảo dọc và/hoặc ngang mà qua đó các hoạt động lọc trong vòng bị vô hiệu hóa. Được so sánh với việc sử dụng hai ô, one đối với mỗi tập hợp mặt liên tiếp, và để vô hiệu hóa các hoạt động lọc trong vòng giữa các ô ảnh, phương pháp báo hiệu được đề xuất linh hoạt hơn do nó không đòi hỏi kích thước mặt phải là bội số của kích thước CTU.

2.13 Các vùng độc lập bị giới hạn chuyển động dựa trên ảnh con lấy làm ví dụ

Theo một số phương án thực hiện, các dấu hiệu sau được bao gồm:

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

2) Chỉ báo tồn tại của các ảnh con được chỉ báo trong SPS, cùng với thông tin mức chuỗi khác của các ảnh con.

3) Liệu ảnh con được xử lý như là ảnh trong quá trình giải mã (không bao gồm các hoạt động lọc trong vòng) có thể được điều khiển bằng dòng bit.

4) Liệu lọc trong vòng giữa các đường biên ảnh con bị vô hiệu hóa có thể được điều khiển bằng dòng bit đối với mỗi ảnh con. Các quá trình DBF, SAO, và ALF được cập nhật để điều khiển các hoạt động lọc trong vòng giữa các đường biên ảnh con.

5) Để cho đơn giản, như là điểm bắt đầu, chiều rộng ảnh con, chiều cao, độ lệch ngang, và độ lệch dọc được báo hiệu theo các đơn vị của các mẫu độ sáng trong SPS. các đường biên ảnh con bị giới hạn là các đường biên lát.

6) Xử lý ảnh con như là ảnh trong quá trình giải mã (không bao gồm các hoạt động lọc trong vòng) được xác định bằng cách cập nhật một chút cú pháp coding_tree_unit(), và các cập nhật đôi với các quá trình giải mã sau:

- o Quá trình dẫn xuất đối với dự báo MV độ sáng thời gian (cải tiến)
- o Quá trình nội suy song tuyến tính mẫu độ sáng
- o Quá trình lọc nội suy 8 bậc mẫu độ sáng
- o Quá trình lọc nội suy mẫu sắc độ

7) Các ID ảnh con được báo hiệu tường minh trong SPS và được bao gồm trong các tiêu đề nhóm ô để có thể trích xuất các chuỗi ảnh con mà không cần thay đổi các đơn vị VCL NAL.

Các tập hợp ảnh con đầu ra (OSPS) được đề xuất để xác định các điểm trích xuất trực chuẩn và tương hợp đối với các ảnh con và các tập hợp của nó.

3. Các vấn đề kỹ thuật được giải quyết bởi các giải pháp theo sáng chế

Thiết kế VVC hiện tại có các vấn đề sau:

1. Thiết lập hiện tại kích hoạt đường biên ảo ALF phụ thuộc vào liệu đường biên dưới của CTB là đường biên dưới của ảnh. Nếu nó đúng, then đường biên

ảo ALF bị vô hiệu hóa, chẳng hạn CTU-D trên Fig.19. Tuy nhiên, có thể là đường biên dưới của CTB nằm ngoài đường biên dưới của ảnh, chẳng hạn ảnh 256x240 được tách thành 4 CTU 128x128, trong trường hợp này, đường biên ảo ALF sẽ bị thiết lập sai thành true đối với 2 CTU cuối cùng mà có các mẫu ngoài đường biên ảnh dưới.

2. Cách thức xử lý đường biên ảo ALF bị vô hiệu hóa đối với đường biên ảnh dưới và đường biên lát/ô/khối viên. Vô hiệu hóa VB dọc theo đường biên lát/khối viên có thể tạo bong bóng đường ống hoặc yêu cầu xử lý 68 dòng trên đơn vị dữ liệu đường ống ảo (VPDU, 64x64 trong chuẩn VVC) giả sử kích thước LCU là 64x64. Chẳng hạn:

a. Đối với các bộ giải mã không biết các đường biên lát/khối viên/ô ngay trước (tức là, các ứng dụng độ trễ thấp), các bộ đệm dòng ALF cần được phục hồi. Liệu nội dung trong các bộ đệm dòng quen thuộc hay không đối với lọc ALF phụ thuộc liệu CTU hiện tại cũng là CTU đường biên lát/khối viên/ô, thông tin này, tuy nhiên, chưa được biết cho đến khi lát/khối viên/ô tiếp theo được giải mã.

b. Đối với các bộ giải mã biết các đường biên lát/khối viên/ô ngay trước, các bộ giải mã cần sống với các bong bóng đường ống (rất không khả thi) hoặc vận hành ALF ở tốc độ 68 dòng trên 64x64 VDPU luôn luôn (overprovision), để tránh sử dụng các bộ đệm dòng ALF.

3. Các cách khác nhau để xử lý đường biên ảo và đường biên đơn vị viđeo, chẳng hạn, các phương pháp đệm khác nhau hiện có. Đồng thời, nhiều hơn một phương pháp đệm có thể được thực hiện đối với dòng khi nó ở nhiều đường biên.

a. Trong một ví dụ, nếu đường biên dưới của khối là đường biên ảo 360° và đường biên ảo ALF cũng được áp dụng cho khối này, trong trường hợp này, phương pháp đệm đường biên ảo 360° có thể trước hết được áp dụng để tạo các mẫu ảo dưới đường biên ảo 360°. Sau đó, các mẫu ảo này được đặt dưới đường biên ảo 360° được xử lý như là khả dụng. Và phương pháp đệm 2 phía ALF có thể còn được áp dụng theo các hình vẽ từ Fig.16A đến Fig.16C. Ví dụ được mô tả trên Fig.25.

4. Cách thức xử lý đường biên ảo có thể là tối ưu phụ, do các mẫu được đệm

được sử dụng có thể ít hiệu quả hơn.

5. Khi ALF phi tuyến tính bị vô hiệu hóa, MALF và phương pháp đệm hai phía sẽ có thể tạo các kết quả tương tự để lọc mẫu mà yêu cầu truy nhập các mẫu giao đường biên ảo. Tuy nhiên, khi the ALF phi tuyến tính được kích hoạt, hai phương pháp sẽ mang lại các kết quả khác nhau. Sẽ có lợi khi căn chỉnh hai trường hợp.

6. Lát có thể là hình chữ nhật, hoặc không phải hình chữ nhật, such như được mô tả trên Fig.28. Trong trường hợp này, đối với CTU, có thể không trùng với các đường biên bất kỳ (chẳng hạn, ảnh/lát/ô/khối viên). Tuy nhiên, có thể cần truy nhập các mẫu ngoài lát hiện tại. Nếu lọc giao đường biên lát (chẳng hạn, loop_filter_across_slices_enabled_flag là false) bị vô hiệu hóa, cách thức thực hiện phân loại ALF và quá trình lọc không được biết.

7. Ảnh con là vùng chữ nhật của một hoặc nhiều lát trong ảnh. Ảnh con chứa một hoặc nhiều lát bao phủ chung vùng chữ nhật của ảnh. Bảng cú pháp được cải biến như sau để bao gồm khái niệm của ảnh con (in đậm, in nghiêng, và gạch chân).

7.3.2.3 Cú pháp RBSP tập tham số chuỗi

seq_parameter_set_rbsp() {	Mô tả
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_5bits	u(5)
profile_tier_level(sps_max_sub_layers_minus1)	
gra_enabled_flag	u(1)
sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
sub_pics_present_flag	***u(1)***
if(sub_pics_present_flag) {	
max_grid_idxs_minus1	***u(8)***
sub_pic_grid_col_width_minus1	***u(v)***
sub_pic_grid_row_height_minus1	***u(v)***
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
sub_pic_grid_idx[i][j]	***u(v)***
for(i = 0; i <= NumSubPics; i++) {	
sub_pic_treated_as_pic_flag[i]	***u(1)***
loop_filter_across_sub_pic_enabled_flag[i]	***u(1)***

}	
}	
bit_depth_luma_minus8	ue(v)
bit_depth_chroma_minus8	ue(v)
log2_max_pic_order_cnt_lsb_minus4	ue(v)
if(sps_max_sub_layers_minus1 > 0)	

Cần lưu trữ việc kích hoạt lọc giữa các ảnh con được điều khiển đối với mỗi ảnh con. Tuy nhiên, việc điều khiển kích hoạt lọc giữa lát/ô/khối viên được điều khiển ở mức ảnh được báo hiệu một lần để điều khiển tất cả các lát /ô/khối viên trong một ảnh.

8. Phân loại ALF được thực hiện trong khối 4x4, tức là, tất cả các mẫu trong một khối 4x4 chia sẻ cùng kết quả phân loại. Tuy nhiên, chính xác hơn, các mẫu trong cửa sổ 8x8 chứa khối 4x4 hiện tại cần tính toán các gradien. Trong trường hợp này, các mẫu 10x10 cần được truy nhập, như được mô tả trên Fig.30. Nếu một vài mẫu được đặt trong đơn vị viđeo khác (chẳng hạn, lát/ô/khối viên/ảnh con/"đường biên ảo 360" trên hoặc bên trái hoặc phải hoặc dưới khác nhau/trên hoặc dưới "đường biên ảo ALF"), cách thức tính toán phân loại cần được định nghĩa.

4. Các ví dụ của các kỹ thuật và các phương án thực hiện

Việc liệt kê dưới đây nên được xem là ví dụ để giải thích các khái niệm chung. Các kỹ thuật được liệt kê không nên được hiểu theo nghĩa hẹp. Ngoài ra, các kỹ thuật này có thể được kết hợp theo cách bất kỳ.

Phương pháp đệm được sử dụng cho các đường biên ảo ALF có thể được ký hiệu là 'Đệm hai phía' trong đó nếu một mẫu được đặt ở (i, j) được đệm, thì mẫu tương ứng được đặt ở (m, n) mà chia sẻ hệ số lọc tương tự cũng được đệm thậm chí khi mẫu khả dụng, như được mô tả trên các hình vẽ từ Fig.12 đến Fig.13.

Phương pháp đệm được sử dụng để các đường biên ảnh/các đường biên ảo viđeo 360°, các đường biên thông thường (chẳng hạn,các đường biên trên và dưới) có thể được ký hiệu là 'Đệm một phía' trong đó nếu một mẫu cần được sử dụng nằm ngoài các đường biên, nó được sao chép từ một đường khả dụng bên trong ảnh.

Phương pháp đệm được sử dụng để các đường biên bên trái và bên phải của

viđeo 360° có thể được ký hiệu là 'wrapping-base Padding' trong đó nếu một mẫu cần được sử dụng nằm ngoài các đường biên, nó được sao chép nhờ sử dụng các kết quả được bù chuyển động.

Trong phần thảo luận sau, mẫu là "ở đường biên của đơn vị viđeo" có thể nghĩa là khoảng cách giữa mẫu này và đường biên của đơn vị viđeo nhỏ hoặc không lớn hơn ngưỡng. "Dòng" có thể đề cập đến các mẫu ở một vị trí ngang tương tự hoặc các mẫu ở một vị trí dọc tương tự. (chẳng hạn, các mẫu trong cùng hàng và/hoặc các mẫu trong cùng cột). Hàm Abs(x) được định nghĩa như sau:

$$Abs(x) = \begin{cases} x & ; \quad x >= 0 \\ -x & ; \quad x < 0 \end{cases}.$$

Trong phần thảo luận sau, "mẫu ảo" đề cập đến mẫu được tạo mà có thể khác với mẫu được tái tạo (có thể được xử lý bằng cách tách khối và/hoặc SAO). Mẫu ảo có thể được sử dụng để tiến hành ALF đối với mẫu khác. Mẫu ảo có thể được tạo bằng cách đệm.

'Phương pháp xử lý ALF đường biên ảo được kích hoạt cho một khối' có thể chỉ báo rằng **applyVirtualBoundary** theo sáng chế được gán bằng true. 'Kích hoạt đường biên ảo' có thể chỉ báo rằng khối hiện tại được tách thành ít nhất hai phần bằng đường biên ảo và các mẫu được đặt ở một phần không được phép sử dụng các mẫu ở phần khác trong quá trình lọc (chẳng hạn, ALF). Đường biên ảo có thể là K hàng phía trên đường biên dưới của một khối.

Trong các phần mô tả sau, các mẫu lân cận có thể là các mẫu được yêu cầu cho quá trình phân loại bộ lọc và/hoặc lọc.

Theo sáng chế, mẫu lân cận là "không khả dụng" nếu nó nằm ngoài ảnh hiện tại, hoặc ảnh con hiện tại, hoặc ô ảnh hiện tại, hoặc lát hiện tại, hoặc khối viên hiện tại, hoặc CTU hiện tại, hoặc khối xử lý hiện tại (chẳng hạn khối xử lý ALF hoặc khối xử lý ALF hẹp), hoặc đơn vị vi đeo hiện tại khác bất kỳ.

1. Việc xác định 'đường biên dưới của CTB hiện tại là đường biên dưới của ảnh' được thay thế bằng 'đường biên dưới của CTB hiện tại là đường biên dưới của ảnh hoặc ngoài ảnh'.

a. Theo cách khác, ngoài ra, trong trường hợp này, phương pháp xử lý đường biên ảo ALF có thể bị vô hiệu hóa.

2. Liệu kích hoạt sử dụng mẫu ảos (chẳng hạn, liệu kích hoạt đường biên ảo (chẳng hạn, gán **applyVirtualBoundary** bằng true hoặc false)) trong quá trình lọc trong vòng có thể phụ thuộc vào kích thước CTB.

a. Trong một ví dụ, **applyVirtualBoundary** luôn được gán bằng false đối với kích thước CTU/CTB cụ thể, chẳng hạn, đối với kích thước CTU/CTB bằng KxL (chẳng hạn, K=L=4).

b. Trong một ví dụ, **applyVirtualBoundary** luôn được gán bằng false đối với các kích thước CTU/CTB cụ thể không lớn hơn hoặc nhỏ hơn KxL (chẳng hạn, K=L=8).

c. Theo cách khác, ALF bị vô hiệu hóa đối với các kích thước CTU/CTB cụ thể, chẳng hạn 4x4, 8x8.

3. Liệu kích hoạt sử dụng các mẫu ảo (chẳng hạn, được đệm từ các mẫu được tái tạo) trong các quá trình lọc trong vòng (chẳng hạn, ALF) có thể phụ thuộc vào liệu đường biên dưới của khối là đường biên dưới của đơn vị viđeo vốn có độ hạt mịn hơn được so sánh với ảnh (chẳng hạn, lát/ô/khối viên) hoặc đường biên ảo.

a. Trong một ví dụ, phương pháp xử lý đường biên ảo ALF có thể được kích hoạt (chẳng hạn, **applyVirtualBoundary** được gán bằng true) đối với khối cây mã (CTB) nếu đường biên dưới của CTB là đường biên của đơn vị viđeo hoặc đường biên ảo.

i. Theo cách khác, ngoài ra, nếu đường biên dưới không phải là đường biên ảnh dưới hoặc nếu đường biên dưới nằm ngoài ảnh, phương pháp nêu trên được kích hoạt.

b. Khi đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, phương pháp xử lý đường biên ảo ALF vẫn có thể được kích hoạt (chẳng hạn, **applyVirtualBoundary** được gán bằng true).

c. Trong một ví dụ, liệu kích hoạt phương pháp xử lý đường biên ảo ALF (chẳng hạn, giá trị của **applyVirtualBoundary**) đối với CTB có thể chỉ phụ thuộc vào mối quan hệ giữa đường biên dưới của CTB và đường biên dưới củảnh.

i. Trong một ví dụ, **applyVirtualBoundary** được gán bằng false chỉ nếu đường biên dưới của CTB là đường biên dưới của ảnh chứa CTB hoặc nếu đường biên dưới nằm ngoài ảnh.

ii. Trong một ví dụ, **applyVirtualBoundary** được gán bằng true khi đường biên dưới của CTB KHÔNG phải là đường biên dưới của ảnh chứa CTB.

d. Trong một ví dụ, đối với khi giải mã CTU-C trên các hình vẽ từ Fig.18A đến Fig.18C, mẫu M × M có thể được lọc với K dòng từ CTU nêu trên và không bao gồm K dòng dưới đường biên ảo.

4. Đề xuất vô hiệu hóa sử dụng các mẫu giữa các đường biên khối viên/lát trong quá trình lọc (chẳng hạn, ALF) thậm chí khi các cờ sử dụng điều khiển được báo hiệu cho các bộ lọc vòng giao các đường biên khối viên/lát (chẳng hạn, **loop_filter_across_bricks_enabled_flag/loop_filter_across_slices_enabled_flag**) **là** true.

a. Theo cách khác, ngoài ra, **loop_filter_across_bricks_enabled_flag/loop_filter_across_slices_enabled_flag** được báo hiệu có thể chỉ điều khiển quá trình lọc của DBF và SAO không bao gồm ALF.

b. Trong một ví dụ, mẫu ảo có thể được sử dụng thay cho mẫu được tái tạo ở vị trí tương ứng để tiến hành ALF đối với mẫu khác.

5. Khi một khối (chẳng hạn, CTB) chứa mẫu được đặt ở đường biên của đơn vị viđeo (chẳng hạn, lát/khối viên/ô/đường biên ảo viđeo 360° hoặc các đường biên thông thường/đường biên ảnh), cách thức tạo mẫu ảo bên trong hoặc ngoài đơn vị viđeo (chẳng hạn, các phương pháp đệm) để lọc trong vòng chẳng hạn ALF có thể được thống nhất đối với các loại đường biên khác.

a. Theo cách khác, ngoài ra, phương pháp của các đường biên ảo (chẳng hạn, phương pháp đệm hai phía) có thể được áp dụng cho khối để xử lý mẫu ở đường biên để lọc trong vòng.

b. Theo cách khác, ngoài ra, các phương pháp nêu trên có thể được áp dụng khi khối chứa mẫu được đặt ở đường biên dưới của đơn vị viđeo.

c. Trong một ví dụ, khi giải mã K dòng của một khối, nếu K dòng dưới đường biên ảo của khối (chẳng hạn, K dòng cuối cùng trong CTU-B trên Fig.17A

và Fig.17B) và đường biên dưới của khối là đường biên dưới của đơn vị viđeo, các mẫu ảo có thể được tạo trong quá trình phân loại ALF/lọc để tránh sử dụng các mẫu khác ngoài K dòng này, chẳng hạn, phương pháp đệm hai phía có thể được áp dụng.

i. Theo cách khác, ALF có thể bị vô hiệu hóa đối với K dòng cuối cùng đó.

d. Trong một ví dụ, khi một khối ở nhiều đường biên, các pixel/các mẫu được sử dụng để phân loại ALF có thể bị giới hạn không giao với đường biên bất kỳ trong nhiều đường biên này.

i. Trong một ví dụ, đối với mẫu, nếu mẫu lân cận cụ thể của nó "không khả dụng" (chẳng hạn, giao với đường biên bất kỳ trong nhiều đường biên), riêng hoặc tất cả các loại građien/hướng có thể không được tính toán cho mẫu đó.

1. Trong một ví dụ, các građien của mẫu có thể được xử lý bằng 0.

2. Trong một ví dụ, các građien của mẫu có thể được xử lý là "không khả dụng" và có thể không được bổ sung vào hoạt động (chẳng hạn, được định nghĩa trong phương trình (8) của phần 2.6.1.1) được dẫn xuất trong quá trình phân loại ALF.

ii. Trong một ví dụ, hoạt động/hướng được dẫn xuất trong quá trình phân loại ALF có thể được chia tỷ lệ bằng hệ số khi chỉ một phần mẫu được sử dụng trong quá trong quá trình phân loại ALF là "khả dụng" (chẳng hạn, không giao với đường biên bất kỳ trong các đường biên này).

iii. Trong một ví dụ, đối với khối đường biên, giả sử các građien/hướng được yêu cầu được tính toán cho N mẫu trong quá trình phân loại ALF, và các građien có thể chỉ được tính toán cho M mẫu (chẳng hạn, nếu mẫu lân cận cụ thể của mẫu không "khả dụng", thì građien không thể được tính toán cho nó), thì độ hoạt động có thể được nhân với N/M.

1. Theo cách khác, có thể được nhân với hệ số phụ thuộc vào N/M. Chẳng hạn, số này có thể là M^N (N là số nguyên), chẳng hạn, M = 2.

e. Trong một ví dụ, các građien của các mẫu bán phần trong cửa sổ MxN (chẳng hạn, M=N=8 theo thiết kế hiện tại, M cột và N hàng) có thể được sử dụng để phân loại.

i. Chẳng hạn, đối với khối N1*N2 hiện tại (N1=N2=4 theo thiết kế hiện tại), M*N có tâm ở khối N1*N2.

ii. Trong một ví dụ, các gradien của các mẫu không truy nhập các mẫu giữa các đường biên bất kỳ có thể được sử dụng.

1. Theo cách khác, ngoài ra, khi tính toán gradien của mẫu đặt ở một hoặc nhiều đường biên, việc đệm (chẳng hạn, đệm 1 phía) có thể được thực hiện nếu một số mẫu lân cận của mẫu hiện tại "không khả dụng".

2. Theo cách khác, ngoài ra, K (chẳng hạn, K = 1, 2) dòng không khả dụng nêu trên có thể được đệm nếu mẫu hiện tại ở được đặt ở đường biên trên của đơn vị viđeo (chẳng hạn lát/khối viên/ô/ các đường biên ảo viđeo 360° hoặc các đường biên ảo ALF).

3. Theo cách khác, ngoài ra, K (chẳng hạn, K = 1, 2) cột bên trái không khả dụng có thể được đệm nếu mẫu hiện tại ở được đặt ở đường biên bên trái của đơn vị viđeo.

4. Theo cách khác, ngoài ra, K (chẳng hạn, K = 1, 2) cột bên phải không khả dụng có thể được đệm nếu mẫu hiện tại ở được đặt ở đường biên phải của đơn vị viđeo.

5. Theo cách khác, ngoài ra, K (chẳng hạn, K = 1, 2) dòng dưới không khả dụng có thể được đệm nếu mẫu hiện tại ở được đặt ở đường biên dưới của đơn vị viđeo.

6. Theo cách khác, ngoài ra, nếu mẫu hiện tại ở được đặt ở đường biên trên và đường biên trái của đơn vị viđeo, K1 (chẳng hạn, K1 = 1, 2) dòng trên không khả dụng có thể được đệm trước để tạo cửa sổ M*(N + K1), sau đó, K2 (chẳng hạn, K2 = 1, 2) cột trái không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

a. Theo cách khác, K2 (chẳng hạn, K2 = 1, 2) cột trái không khả dụng có thể được đệm trước để tạo cửa sổ (M+K2)*N, sau đó, K1 (chẳng hạn, K1 = 1, 2) dòng trên không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

7. Theo cách khác, ngoài ra, nếu mẫu hiện tại ở được đặt ở đường biên trên và đường biên phải của đơn vị viđeo, K1 (chẳng hạn, K1 = 1, 2) dòng trên không khả dụng có thể được đệm trước để tạo cửa sổ M*(N + K1), sau đó, K2 (chẳng

hạn, K2 = 1, 2) cột phải không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

a. Theo cách khác, K2 (chẳng hạn, K2 = 1, 2) cột phải không khả dụng có thể được đệm trước để tạo cửa sổ (M+K2)*N, sau đó, K1 (chẳng hạn, K1 = 1, 2) dòng trên không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

8. Theo cách khác, ngoài ra, nếu mẫu hiện tại ở được đặt ở đường biên dưới và đường biên phải của đơn vị viđeo, K1 (chẳng hạn, K1 = 1, 2) dòng dưới không khả dụng có thể được đệm trước để tạo cửa sổ M*(N + K1), sau đó, K2 (chẳng hạn, K2 = 1, 2) cột phải không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

a. Theo cách khác, K2 (chẳng hạn, K2 = 1, 2) cột phải không khả dụng có thể được đệm trước để tạo cửa sổ (M+K2)*N, sau đó, K1 (chẳng hạn, K1 = 1, 2) dòng dưới không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

9. Theo cách khác, ngoài ra, nếu mẫu hiện tại ở được đặt ở đường biên dưới và đường biên trái của đơn vị viđeo, K1 (chẳng hạn, K1 = 1, 2) dòng dưới không khả dụng có thể được đệm trước để tạo cửa sổ M*(N + K1), sau đó, K2 (chẳng hạn, K2 = 1, 2) cột trái không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

a. Theo cách khác, K2 (chẳng hạn, K2 = 1, 2) cột trái không khả dụng có thể được đệm trước để tạo cửa sổ (M+K2)*N, sau đó, K1 (chẳng hạn, K1 = 1, 2) dòng dưới không khả dụng có thể được đệm để tạo cửa sổ (M + K2)*(N + K1).

10. Theo cách khác, ngoài ra, các mẫu được đệm có thể được sử dụng để tính toán các gradien.

iii. Trong một ví dụ, đối với khối ở đường biên trên/dưới của đơn vị viđeo (chẳng hạn lát/khối viên/ô/ các đường biên ảo viđeo 360° hoặc các đường biên ảo ALF), các gradien của các mẫu trong cửa sổ M*(N – C1) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các gradien của C1 dòng trên/dưới của cửa sổ M*N không được sử dụng trong phân loại.

iv. Trong một ví dụ, đối với khối ở đường biên trái/phải của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1)*N có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 cột trái/phải của cửa sổ M*N không được sử dụng trong phân loại.

v. Trong một ví dụ, đối với khối ở đường biên trên và đường biên dưới của đơn vị viđeo, các građien của các mẫu trong cửa sổ M*(N – C1 – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 dòng trên và C2 dòng dưới của cửa sổ M*N không được sử dụng trong phân loại.

vi. Trong một ví dụ, đối với khối ở đường biên trên và đường biên trái của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1)*(N – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 dòng trên và C2 cột trái của cửa sổ M*N không được sử dụng trong phân loại.

vii. Trong một ví dụ, đối với khối ở đường biên trên và đường biên phải của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1)*(N – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 dòng trên và C2 cột phải của cửa sổ M*N không được sử dụng trong phân loại.

viii. Trong một ví dụ, đối với khối ở đường biên dưới và đường biên trái của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1)*(N – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 dòng dưới và C2 cột trái của cửa sổ M*N không được sử dụng trong phân loại.

ix. Trong một ví dụ, đối với khối ở đường biên dưới và đường biên phải của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1)*(N – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 dòng dưới và C2 cột phải của cửa sổ M*N không được sử dụng trong phân loại.

x. Trong một ví dụ, đối với khối ở đường biên trái vả đường biên phải của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1 – C2)*N có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 cột trái và C2 cột phải của cửa sổ M*N không được sử dụng trong phân loại.

xi. Trong một ví dụ, đối với khối ở đường biên trên, đường biên dưới và đường biên trái của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C3)*(N – C1 – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của top C1 các dòng, và C2 dòng dưới và C3 cột trái của cửa sổ M*N không được sử dụng trong phân loại.

xii. Trong một ví dụ, đối với khối ở đường biên trên, đường biên dưới và đường biên phải của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C3)*(N – C1 – C2) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 dòng trên, và C2 dòng dưới và C3 cột phải của cửa sổ M*N không được sử dụng trong phân loại.

xiii. Trong một ví dụ, đối với khối ở đường biên trái, đường biên phải và đường biên trên của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1 – C2)*(N – C3) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 cột trái, và C2 cột phải và C3 dòng trên của cửa sổ M*N không được sử dụng trong phân loại.

xiv. Trong một ví dụ, đối với khối ở đường biên trái, đường biên phải và đường biên dưới của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1 – C2)*(N – C3) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 cột trái, và C2 cột phải và C3 dòng dưới của cửa sổ M*N không được sử dụng trong phân loại.

xv. Trong một ví dụ, đối với khối ở đường biên trái, đường biên phải, đường biên trên và đường biên dưới của đơn vị viđeo, các građien của các mẫu trong cửa sổ (M – C1 – C2)*(N – C3 – C4) có thể được sử dụng để phân loại của khối.

1. Theo cách khác, ngoài ra, các građien của C1 cột trái, và C2 cột phải, C3 dòng trên và C4 dòng dưới của cửa sổ M*N không được sử dụng trong phân loại.

xvi. Trong một ví dụ, C1, C2, C3 và C4 bằng 2.

xvii. Trong một ví dụ, các gradien của các mẫu không có các mẫu lân cận "không khả dụng" bất kỳ được yêu cầu trong tính toán gradien có thể được sử dụng.

f. Trong một ví dụ, khi một dòng ở nhiều đường biên (chẳng hạn, khoảng cách giữa dòng đến đường biên nhỏ hơn ngưỡng), quá trình đệm được thực hiện chỉ một lần bất kể bao nhiêu đường biên có thể có.

i. Theo cách khác, ngoài ra, bao nhiêu dòng lân cận phải được đệm có thể phụ thuộc vào vị trí của dòng hiện tại so với tất cả các đường biên.

ii. Chẳng hạn, bao nhiêu dòng lân cận phải được đệm có thể được quyết định bởi các khoảng cách giữa dòng hiện tại và hai đường biên, chẳng hạn khi dòng hiện tại nằm trong hai đường biên, với hai đường biên nằm trên và dưới.

iii. Chẳng hạn, bao nhiêu dòng lân cận phải được đệm có thể được quyết định bởi khoảng cách giữa dòng hiện tại và đường biên gần nhất, chẳng hạn khi dòng hiện tại nằm trong hai đường biên, với hai đường biên nằm trên và dưới.

iv. Chẳng hạn, bao nhiêu dòng lân cận phải được đệm có thể được tính toán đối với mỗi đường biên độc lập, và dòng lớn nhất được chọn làm số dòng được đệm cuối cùng.

v. Trong một ví dụ, bao nhiêu dòng lân cận phải được đệm có thể được quyết định đối với mỗi phía (chẳng hạn, phía trên và phía dưới) của dòng.

vi. Trong một ví dụ, đối với phương pháp đệm hai phía, bao nhiêu dòng lân cận phải được đệm có thể được quyết định đồng thời với hai phía.

vii. Theo cách khác, ngoài ra, phương pháp đệm 2 phía được sử dụng bởi ALF được áp dụng.

g. Trong một ví dụ, khi một dòng ở nhiều đường biên và có ít nhất một đường biên ở mỗi phía (chẳng hạn, phía trên và phía dưới) của dòng, ALF có thể bị vô hiệu hóa cho nó.

h. Trong một ví dụ, khi số lượng dòng được đệm được yêu cầu bởi dòng hiện tại lớn hơn ngưỡng, ALF có thể bị vô hiệu hóa cho dòng hiện tại.

i. Trong một ví dụ, khi số lượng dòng được đệm ở phía bất kỳ lớn hơn ngưỡng, ALF có thể bị vô hiệu hóa cho dòng hiện tại.

ii. Trong một ví dụ, khi tổng số các dòng được đệm ở cả hai phía lớn hơn ngưỡng, ALF có thể bị vô hiệu hóa cho dòng hiện tại.

i. Theo cách khác, ngoài ra, các phương pháp nêu trên có thể được áp dụng khi khối chứa mẫu được đặt ở đường biên dưới của đơn vị viđeo và lọc trong vòng chẳng hạn ALF được kích hoạt cho khối.

j. Theo cách khác, ngoài ra, các phương pháp nêu trên có thể được áp dụng theo các điều kiện cụ thể, chẳng hạn khi khối chứa mẫu được đặt ở đường biên dưới của đơn vị viđeo và lọc giao với các đường biên không được phép (chẳng hạn, **pps_loop_filter_across_virtual_boundaries_disabled_flag/ loop_filter_across_slices_enabled_flag/loop_filter_across_slices_enabled_flag** là true).

k. Phương pháp được đề xuất cũng có thể áp dụng cho các mẫu/các khối được đặt ở các đường biên ảo.

6. Khi mẫu có ít nhất hai đường biên của một khối (chẳng hạn, ít nhất một đường biên nằm trên đường hiện tại là đường biên ảo ALF, và dưới là đường biên khác), bao nhiêu dòng được đệm không thuần túy được xác định bởi khoảng cách giữa đường hiện tại so với đường biên ảo ALF. Thay vào đó, nó được xác định bởi khoảng cáchs giữa đường hiện tại so với hai đường biên.

a. Trong một ví dụ, số lượng dòng để đệm mỗi bên được gán bằng (M – min (D0, D1)).

b. Trong một ví dụ, số lượng dòng để đệm mỗi bên được gán bằng (M – max (D0, D1)).

c. Trong ví dụ nêu trên, D0, D1 ký hiệu khoảng cách giữa dòng hiện tại và các đường biên trên/dưới.

d. Trong ví dụ nêu trên, M ký hiệu số lượng dòng thở đường biên ảo ALF từ dưới của một CTU.

7. Ít nhất hai cách chọn mẫu trong phân loại ALF và/hoặc quá trình lọc ALF tuyến tính hoặc phi tuyến có thể được định nghĩa, với một trong số chúng chọn các mẫu trước phương pháp lọc trong vòng bất kỳ được áp dụng; và quá trình còn lại chọn các mẫu sau khi một hoặc nhiều phương pháp lọc trong vòng được áp dụng nhưng trước khi ALF được áp dụng.

a. Trong một ví dụ, việc lựa chọn các cách khác nhau có thể phụ thuộc vào vị trí của các mẫu cần được lọc.

b. Trong một ví dụ, mẫu ở đường biên dưới của đơn vị viđeo (chẳng hạn CTB) có thể được chọn với phương pháp thứ nhất khi nó được sử dụng trong ALF đối với mẫu khác. Ngược lại (nó không nằm ở biên), phương pháp thứ hai được chọn.

8. Đề xuất vô hiệu hóa sử dụng các mẫu giao với đường biên VPDU (chẳng hạn, vùng 64x64) trong quá trình lọc.

a. Trong một ví dụ, khi mẫu được yêu cầu bởi quá trình phân loại ALF nằm ngoài đường biên VPDU, hoặc dưới đường biên ảo, có thể được thay thế bằng mẫu ảo hoặc các kết quả phân loại cho mẫu có thể được sao chép từ mẫu được liên kết với các mẫu khác, chẳng hạn được đệm từ các mẫu khả dụng.

b. Trong một ví dụ, khi mẫu được yêu cầu bởi quá trình lọc nằm ngoài đường biên VPDU, hoặc dưới đường biên ảo, có thể được thay thế bằng mẫu ảo, chẳng hạn được đệm từ các mẫu khả dụng.

c. Trong một ví dụ, phương pháp xử lý đường biên ảo ALF có thể được kích hoạt (chẳng hạn, **applyVirtualBoundary** được gán bằng true) đối với khối nếu nó chứa các mẫu được đặt ở đường biên của VPDU.

d. Theo cách khác, việc sử dụng các mẫu giao với đường biên VPDU ngangcó thể bị vô hiệu hóa trong quá trình lọc.

i. Trong một ví dụ, khi mẫu được yêu cầu bởi quá trình lọc nằm dưới đường biên VPDU ngang, hoặc dưới đường biên ảo, có thể được thay thế bằng mẫu ảo, chẳng hạn được đệm từ các mẫu khả dụng.

e. Theo cách khác, việc sử dụng các mẫu giao đường biên VPDU dọc có thể bị vô hiệu hóa trong quá trình lọc.

i. Trong một ví dụ, khi mẫu được yêu cầu bởi quá trình lọc nằm ngoài đường biên VPDU dọc, hoặc nằm dưới đường biên ảo, có thể được thay thế bằng mẫu ảo, chẳng hạn được đệm từ các mẫu khả dụng.

9. Thay cho sử dụng các mẫu được đệm (chẳng hạn, khả dụng, trên/dưới các đường biên ảo, trên/dưới các đường của đơn vị viđeo) trong quá trình phân loại ALF/lọc, đề xuất sử dụng các mẫu được tái tạo trước tất cả các bộ lọc trong vòng.

a. Theo cách khác, ngoài ra, khái niệm của đệm hai phía được áp dụng qua các mẫu đệm từ các mẫu được tái tạo trước tất cả các bộ lọc trong vòng.

i. Trong một ví dụ, nếu mẫu trong phần hỗ trợ bộ lọc là từ các mẫu được tái tạo trước tất cả các bộ lọc trong vòng, mẫu đối xứng (chẳng hạn, đối xứng quanh gốc, chẳng hạn, mẫu hiện tại) trong phần đỡ bộ lọc cũng sẽ sử dụng mẫu được tái tạo trước tất cả các bộ lọc trong vòng.

1. Giả sử tọa độ của mẫu hiện tại cần được lọc là (0, 0) và mẫu được đặt ở (i, j) là mẫu được tái tạo trước tất cả các bộ lọc trong vòng, sau đó mẫu được đặt ở (-i, -j) là mẫu được tái tạo trước tất cả các bộ lọc trong vòng.

2. Giả sử tọa độ của mẫu hiện tại cần được lọc là (x, y) và mẫu được đặt ở (x + i, y + j) là mẫu được tái tạo trước tất cả các bộ lọc trong vòng, sau đó mẫu được đặt ở (x – i, y – j) là mẫu được tái tạo trước tất cả các bộ lọc trong vòng.

b. Theo cách khác, ngoài ra, khi bước tạo dạng lại trong vòng (cũng được biết đến là LMCS) được kích hoạt, các mẫu được tái tạo trước tất cả các bộ lọc trong vòng là các mẫu trong miền ban đầu được biến đổi từ miền được tạo dạng lại.

10. Thay cho sử dụng các mẫu được đệm (chẳng hạn, khả dụng, trên/dưới các đường biên ảo, trên/dưới các đường của đơn vị viđeo) trong quá trình lọc ALF, đề xuất sử dụng các phần hỗ trợ bộ lọc ALF khác nhau.

a. Trong một ví dụ, giả sử mẫu cần được đệm trong phương pháp nêu trên, thay cho thực hiện đệm, hệ số lọc được liên kết với mẫu được gán bằng 0.

i. Trong trường hợp này, phần hỗ trợ bộ lọc được cải biến bằng cách không bao gồm các mẫu được yêu cầu đệm.

ii. Theo cách khác, ngoài ra, các hệ số lọc được áp dụng cho các mẫu khác ngoại trừ mẫu hiện tại luôn không đổi, tuy nhiên, hệ số lọc được áp dụng cho mẫu hiện tại có thể được chỉnh sửa, chẳng hạn ((1<< C_BD) – tổng của tất cả các hệ số lọc được áp dụng cho các mẫu không cần được đệm) trong đó C_BD chỉ báo độ sâu bit của hệ số lọc.

1. Lấy các hình vẽ Fig.18A và Fig.18B chẳng hạn, khi lọc các dòng L và I, hệ số lọc c12 được áp dụng cho mẫu hiện tại được cải biến thành ((1<< C_BD) – 2*(c4+c5+c6+c7+c8+c9+c10+c11)).

b. Trong một ví dụ, giả sử mẫu (x1, y1) được đệm từ (x2, y2) trong phương pháp nêu trên, thay cho thực hiện đệm, hệ số lọc được liên kết với (x1, y1) được bổ sung vào hệ số của vị trí (x2, y2) bất kể bộ lọc phi tuyến được kích hoạt hoặc bị vô hiệu hóa.

i. Theo cách khác, ngoài ra, tham số rút gọn cho (x2, y2) có thể được dẫn xuất ngay khi chạy.

1. Trong một ví dụ, có thể được gán bằng tham số rút gọn được giải mã cho (x2, y2).

2. Theo cách khác, có thể được gán bằng giá trị được trả về của hàm có các tham số rút gọn được giải mã đối với (x1, y1) và (x2, y2) làm các đầu vào, chẳng hạn, giá trị lớn hoặc giá trị nhỏ hơn.

11. Việc lựa chọn các tham số rút gọn/các hệ số lọc/các phần hỗ trợ bộ lọc có thể phụ thuộc vào liệu mẫu lọc cần truy nhập các mẫu được đệm (chẳng hạn, khả dụng, trên/dưới các đường biên ảo, trên/dưới các đường của đơn vị viđeo).

a. Trong một ví dụ, các tham số rút gọn/các hệ số lọc/các phần hỗ trợ bộ lọc khác nhau có thể được sử dụng cho các mẫu với chỉ số lớp tương tự nhưng đối với một số mẫu yêu cầu truy nhập mẫu được đệm và mẫu còn lại không.

b. Trong một ví dụ, các tham số rút gọn/các hệ số lọc/các phần hỗ trợ bộ lọc để lọc các mẫu yêu cầu truy nhập các mẫu được đệm có thể được báo hiệu trong mức CTU/vùng/lát/ô.

c. Trong một ví dụ, các tham số rút gọn/các hệ số lọc/các phần hỗ trợ bộ lọc để lọc các mẫu yêu cầu truy nhập các mẫu được đệm có thể được dẫn xuất từ đó được sử dụng để lọc các mẫu không yêu cầu truy nhập các mẫu được đệm.

i. Trong một ví dụ, các đề mục 9a hoặc 9b có thể được áp dụng.

12. Cách thức xử lý mẫu ở biên để lọc trong vòng (chẳng hạn ALF) có thể phụ thuộc vào thành phần màu sắc và/hoặc định dạng màu sắc.

a. Chẳng hạn, định nghĩa về "nằm ở đường biên" có thể là khác nhau đối với các thành phần màu sắc khac snhau. Trong một ví dụ, mẫu độ sáng nằm ở đường biên dưới nếu khoảng cách giữa nó và đường biên dưới nhỏ hơn T1; mẫu sắc độ nằm ở đường biên dưới nếu khoảng cách giữa nó và đường biên dưới nhỏ hơn T2. T1 và T2 có thể khác nhau.

i. Trong một ví dụ, T1 và T2 có thể khác nhau khi định dạng màu sắc không phải là 4:4:4.

13. Khi đường biên dưới/trên/trái/phải của một CTU/VPDU cũng là đường biên của lát/ô/khối viên/vùng con có mã hóa độc lập, thứ tự cố định của nhiều quá trình đệm được áp dụng.

a. Trong một ví dụ, ở bước thứ nhất, phương pháp đệm (chẳng hạn, đệm 1 phía) của lát/ô/khối viên trước hết được áp dụng. Sau đó, phương pháp đệm xử lý ALF các đường biên ảo (chẳng hạn, phương pháp đệm 2 phía) còn được áp dụng trong suốt bước thứ hai. Trong trường hợp này, các mẫu được đệm sau bước thứ nhất được đánh dấu là không khả dụng và có thể được sử dụng để quyết định bao nhiêu dòng được đệm trong quá trình ALF đường biên ảo. Quy tắc tương tự (chẳng hạn, các hình vẽ Fig.16A đến Fig.16C) để xử lý các CTU không được đặt ở các đường biên đó được sử dụng.

14. Các phương pháp được đề xuất có thể được áp dụng cho một hoặc nhiều đường biên giữa hai ảnh con.

a. Đường biên áp dụng các phương pháp được đề xuất có thể là đường biên ngang.

b. Đường biên áp dụng các phương pháp được đề xuất có thể là đường biên dọc.

15. Các phương pháp được đề xuất nêu trên có thể được áp dụng cho các mẫu/các khối ở các đường biên dọc.

16. Liệu và/hoặc cách thức phương pháp đề xuất được áp dụng ở "đường biên ảo 360" có thể phụ thuộc vào vị trí của "đường biên ảo 360".

a. Trong một ví dụ, khi "đường biên ảo 360" trùng với đường biên CTU, phương pháp được đề xuất có thể được áp dụng. Chẳng hạn, chỉ đệm 2 phía có thể được áp dụng ở ALF đối với các mẫu ở "đường biên ảo 360".

b. Trong một ví dụ, khi "đường biên ảo 360" không trùng với đường biên CTU, phương pháp được đề xuất không thể được áp dụng. Chẳng hạn, chỉ đệm 1 phía có thể được áp dụng ở ALF đối với các mẫu ở "đường biên ảo 360".

c. Trong một ví dụ, phương pháp đệm tương tự có thể được áp dụng ở ALF đối với các mẫu ở "đường biên ảo 360" bất kể vị trí của "đường biên ảo 360".

i. Chẳng hạn, đệm 1 phía có thể được áp dụng ở ALF đối với các mẫu ở "đường biên ảo 360".

ii. Chẳng hạn, đệm 2 phía có thể được áp dụng ở ALF đối với các mẫu ở "đường biên ảo 360".

d. Trong một ví dụ, đối với các mẫu ở nhiều đường biên trong đó ít nhất một đường biên là "đường biên ảo 360" và ít nhất một trong "đường biên ảo 360" không trùng với đường biên CTU, phương pháp được đề xuất không thể được áp dụng.

i. Chẳng hạn, các mẫu giữa đường biên bất kỳ trong nhiều đường biên có thể được đệm bằng cách đệm 1 phía.

1. Theo cách khác, ngoài ra, nếu có "đường biên ảo", đệm 2 phía có thể được áp dụng trong ALF sau khi đệm 1 phía.

e. Trong một ví dụ, đối với các mẫu được đặt giữa hai loại đường biên, nếu một trong số chúng là "đường biên ảo 360", và còn lại không phải, bước đệm được gọi chỉ một lần trong quá trình ALF.

i. Trong một ví dụ, phương pháp đệm xử lý các đường biên ảo ALF (chẳng hạn, phương pháp đệm 2 phía) có thể được gọi.

1. Theo cách khác, phương pháp đệm xử lý các đường biên ảnh (hoặc lát/ô/khối viên/ảnh con) (chẳng hạn, đệm 1 phía) có thể được gọi.

ii. Theo cách khác, hai hoặc nhiều quá trình đệm có thể được áp dụng theo thứ tự.

1. Trong một ví dụ, phương pháp đệm xử lý các đường biên ảnh (hoặc lát/ô/khối viên/ảnh con) (chẳng hạn, đệm 1 phía) có thể được áp dụng trước, sau đó, phương pháp đệm xử lý các đường biên ảo ALF (chẳng hạn, phương pháp đệm 2 phía) có thể còn được gọi.

a. Theo cách khác, ngoài ra, các mẫu được đệm sau bước đệm thứ nhất được xử lý như là khả dụng trong quá trình đệm thứ hai.

iii. Trong một ví dụ, đối với các mẫu được đặt giữa hai hoặc nhiều loại đường biên, (chẳng hạn, đường biên lát/đường biên ô/đường biên khối viên/đường biên ảo 360"/"đường biên ảo ALF"/đường biên ảnh con), nếu chỉ một trong các đường biên là "đường biên ảo 360" (như được thể hiện trên Fig.24, chẳng hạn, đường

biên thứ nhất là "đường biên ảo 360", và đường biên thứ hai là "đường biên ảo ALF" hoặc lát/khối viên/đường biên ô/đường biên ảnh con; hoặc ngược lại), phương pháp được đề xuất có thể được áp dụng. Chẳng hạn, chỉ đệm 2 phía có thể được áp dụng trong ALF đối với các mẫu này.

1. Theo cách khác, nếu nhiều loại đường biên này là "đường biên ảo 360" hoặc đường biên ảnh, phương pháp được đề xuất không thể được áp dụng. Chẳng hạn, chỉ đệm 1 phía có thể được áp dụng trong ALF đối với các mẫu này.

f. Trong một ví dụ, đối với các mẫu được đặt giữa hai hoặc nhiều loại đường biên, và nếu ít nhất một trong các đường biên là "đường biên ảo 360" và không trùng với đường biên CTU, phương pháp được đề xuất không thể được áp dụng.

i. Trong trường hợp này, có thể được xem là giải pháp kỹ thuật đã biết để xử lý các mẫu chỉ ở "đường biên ảo 360" mà không ở các loại đường biên khác.

ii. Trong một ví dụ, chỉ đệm 1 phía có thể được áp dụng trong ALF đối với các mẫu này.

g. Trong một ví dụ, đối với các mẫu được đặt giữa hai hoặc nhiều loại đường biên, và nếu ít nhất một trong các đường biên là "đường biên ảo 360", phương pháp được đề xuất không thể được áp dụng.

i. Trong trường hợp này, có thể được xem là giải pháp kỹ thuật đã biết để xử lý các mẫu chỉ ở "đường biên ảo 360" mà không ở các loại đường biên khác.

ii. Trong một ví dụ, chỉ đệm 1 phía có thể được áp dụng trong ALF đối với các mẫu này.

17. Khi mẫu tham chiếu được yêu cầu trong quá trình lọc ALF (chẳng hạn, P0i với i là A/B/C/D trên Fig.16C khi lọc mẫu hiện tại P0) và/hoặc quá trình phân loại ALF là "không khả dụng", chẳng hạn, do việc mẫu được đặt ở đơn vị viđeo khác (chẳng hạn, lát/khối viên/ô/ảnh con) với mẫu hiện tại và lọc nhờ sử dụng các mẫu giao với đơn vị viđeo (chẳng hạn, các đường biên lát/khối viên/ô/ảnh con) không được phép, mẫu "không khả dụng" có thể được đệm với các mẫu "khả dụng" (chẳng hạn, các mẫu trong cùng lát/khối viên/ô/ảnh con với mẫu hiện tại).

a. Trong một ví dụ, mẫu tham chiếu "không khả dụng" có thể trước hết được rút gọn thành vị trí ngang "khả dụng" gần nhất, sau đó, mẫu tham chiếu "không khả dụng" được rút gọn thành vị trí dọc "khả dụng" gần nhất nếu cần.

b. Trong một ví dụ, mẫu tham chiếu "không khả dụng" có thể trước hết được rút gọn thành vị trí dọc "khả dụng" gần nhất, sau đó, mẫu "không khả dụng" được rút gọn thành vị trí ngang "khả dụng" gần nhất nếu cần.

c. Trong một ví dụ, tọa độ của mẫu tham chiếu "không khả dụng" được rút gọn thành tọa độ của mẫu "khả dụng" gần nhất của nó (chẳng hạn, khoảng cách nhỏ nhất) theo chiều ngang.

i. Trong một ví dụ, đối với hai mẫu có các tọa độ (x1, y1) và (x2, y2), khoảng cách ngang giữa chúng có thể được tính toán là Abs(x1 – x2).

d. Trong một ví dụ, tọa độ của mẫu tham chiếu "không khả dụng" được rút gọn thành tọa độ của mẫu "khả dụng" gần nhất của nó (chẳng hạn, khoảng cách nhỏ nhất) theo chiều dọc.

i. Trong một ví dụ, đối với hai mẫu có các tọa độ (x1, y1) và (x2, y2), khoảng cách dọc giữa chúng có thể được tính toán là Abs(y1 – y2).

e. Trong một ví dụ, mẫu "không khả dụng" được rút gọn thành mẫu "khả dụng" gần nhất của nó (chẳng hạn, khoảng cách nhỏ nhất).

i. Trong một ví dụ, đối với hai mẫu có các tọa độ (x1, y1) và (x2, y2), khoảng cách giữa chúng có thể được tính toán là (x1 – x2)* (x1 – x2) + (y1 – y2)*(y1 – y2).

ii. Theo cách khác, khoảng cách giữa hai pixel có thể được tính toán là Abs(x1 – x2) + Abs(y1 – y2).

f. Theo cách khác, quá trình lọc bị vô hiệu hóa đối với mẫu hiện tại.

g. Theo cách khác, quá trình phân loại trong ALF (chẳng hạn, tính toán građien đối với mẫu hiện tại) có thể không được phép sử dụng các mẫu tham chiếu không khả dụng.

18. Cách thức dẫn xuất mẫu được đệm của các mẫu tham chiếu không khả dụng có thể phụ thuộc vào liệu CTU trùng với các đường biên bất kỳ.

a. Trong một ví dụ, khi CTU hiện tại không trùng với các loại đường biên bất kỳ, nhưng quá trình lọc (chẳng hạn, phân loại ALF/ALF quá trình lọc) đối

với mẫu hiện tại cần truy nhập mẫu tham chiếu trong đơn vị viđeo khác (chẳng hạn, lát), các phương pháp được mô tả trong mục 16 có thể được áp dụng.

i. Theo cách khác, ngoài ra, khi CTU hiện tại không trùng với các loại đường biên bất kỳ, nhưng quá trình lọc (chẳng hạn, phân loại ALF/ALF quá trình lọc) đối với mẫu hiện tại cần truy nhập mẫu tham chiếu trong đơn vị viđeo khác (chẳng hạn, lát) và lọc giao với đường biên lát không được phép, các phương pháp được mô tả trong mục 16 có thể được áp dụng.

ii. Theo cách khác, ngoài ra, khi CTU hiện tại không trùng với các loại đường biên bất kỳ, nhưng quá trình lọc (chẳng hạn, phân loại ALF/ALF quá trình lọc) đối với mẫu hiện tại cần truy nhập mẫu tham chiếu trong đơn vị viđeo khác (chẳng hạn, lát) và mẫu tham chiếu trong cùng đơn vị viđeo và lọc giao với đường biên lát không được phép, các phương pháp được mô tả trong mục 16 có thể được áp dụng.

b. Trong một ví dụ, khi CTU hiện tại trùng với ít nhất một loại đường biên, các phương pháp đệm thống nhất có thể được áp dụng (chẳng hạn, đệm 2 phía hoặc 1 phía).

i. Theo cách khác, khi CTU hiện tại trùng với nhiều loại đường biên và lọc giao với các đường biên đó không được phép, các phương pháp đệm thống nhất có thể được áp dụng (chẳng hạn, đệm 2 phía hoặc 1 phía).

c. Trong một ví dụ, chỉ các mẫu "không khả dụng" không thể được đệm bằng cách đệm 2 phía và/hoặc đệm 1 phía có thể được đệm nhờ sử dụng các phương pháp được mô tả trong mục 16.

19. Liệu quá trình lọc (chẳng hạn, lọc tách khối, SAO, ALF, lọc hai biên, lọc biến đổi Hadamard v.v..) có thể truy nhập các mẫu giữa các đường của đơn vị viđeo (chẳng hạn, lát/khối viên/ô/đường biên ảnh con) có thể được điều khiển ở các mức khác nhau, chẳng hạn được tự điều khiển, thay cho được điều khiển cho tất cả các đơn vị viđeo trong chuỗi/ảnh.

a. Theo cách khác, một phần tử cú pháp có thể được báo hiệu cho lát trong PPS/tiêu đề lát để chỉ báo liệu quá trình lọc có thể giao với đường biên lát cho lát.

b. Theo cách khác, một phần tử cú pháp có thể được báo hiệu cho khối viên/ô trong PPS để chỉ báo liệu quá trình lọc có thể giao với khối viên/đường biên ô cho khối viên/ô.

c. Trong một ví dụ, các phần tử cú pháp có thể được báo hiệu trong SPS/PPS để chỉ báo liệu quá trình lọc có thể giao với đường biên khối viên và/hoặc đường biên ô và/hoặc đường biên lát và/hoặc "đường biên ảo 360°" đối với viđeo/ảnh.

i. Trong một ví dụ, các phần tử cú pháp riêng rẽ có thể được báo hiệu cho các loại đường biên khác nhau.

ii. Trong một ví dụ, một phần tử cú pháp có thể được báo hiệu cho tất cả các loại đường biên.

iii. Trong một ví dụ, một phần tử cú pháp có thể được báo hiệu cho vài loại đường biên.

1. Chẳng hạn, 1 phần tử cú pháp có thể được báo hiệu cho cả đường biên khối viên lẫn đường biên ô.

d. Trong một ví dụ, phần tử cú pháp có thể được báo hiệu trong SPS để chỉ báo liệu có các chỉ báo mức PPS/lát trên quá trình lọc.

i. Trong một ví dụ, các phần tử cú pháp riêng rẽ có thể được báo hiệu cho các loại đường biên khác nhau.

ii. Trong một ví dụ, một phần tử cú pháp có thể được báo hiệu cho tất cả các loại đường biên.

iii. Trong một ví dụ, một phần tử cú pháp có thể được báo hiệu cho vài loại đường biên.

1. Chẳng hạn, 1 phần tử cú pháp có thể được báo hiệu cho cả đường biên khối viên và đường biên ô.

iv. Các chỉ báo về liệu quá trình lọc có thể giao với lát/khối viên/ô/đường biên ảnh con có thể được báo hiệu trong PPS/tiêu đề lát chỉ khi phần tử cú pháp tương ứng trong SPS bằng giá trị cụ thể.

1. Theo cách khác, các chỉ báo về liệu quá trình lọc có thể giao với lát/khối viên/ô/đường biên ảnh con có thể không được báo hiệu trong PPS/tiêu đề lát khi phần tử cú pháp tương ứng trong SPS bằng các giá trị cụ thể.

a. Trong trường hợp này, quá trình lọc có thể không được phép giao với lát/khối viên/ô/đường biên ảnh con nếu chỉ báo trong SPS bằng giá trị cụ thể.

b. Trong trường hợp này, quá trình lọc có thể giao với lát/khối viên/ô/đường biên ảnh con nếu chỉ báo trong SPS bằng giá trị cụ thể.

5. Các phương án thực hiện

Trong các phần sau, một số ví dụ về cách thức phiên bản hiện tại của chuẩn VVC được cải biến để thực hiện một số phương án thực hiện sáng chế được mô tả. Các phần mới được thêm vào được chỉ báo trong đoạn in đậm nghiêng gạch chân. Các phần được xóa được chỉ báo nhờ sử dụng [[]].

5.1 Phương án thực hiện #1

loop_filter_across_bricks_enabled_flag bằng 1 xác định rằng các hoạt động lọc trong vòng có thể được thực hiện giữa các đường biên của khối viên trong các ảnh chỉ đến PPS. loop_filter_across_bricks_enabled_flag bằng 0 xác định rằng các hoạt động lọc trong vòng không được thực hiện giữa các đường biên của khối viên trong các ảnh chỉ đến PPS. Các hoạt động lọc trong vòng bao gồm các hoạt động DBF, lọc SAO [[, và ALF]]. Khi không xuất hiện, giá trị của loop_filter_across_bricks_enabled_flag được suy ra bằng 1.

loop_filter_across_slices_enabled_flag bằng 1 xác định rằng các hoạt động lọc trong vòng có thể được thực hiện giữa các đường biên lát trong các ảnh chỉ đến PPS. loop_filter_across_slice_enabled_flag bằng 0 xác định rằng các hoạt động lọc trong vòng không được thực hiện giữa các đường biên lát trong các ảnh chỉ đến PPS. Các hoạt động lọc trong vòng bao gồm các hoạt động DBF, lọc SAO [[, và ALF]]. Khi không xuất hiện, giá trị của loop_filter_across_slices_enabled_flag được suy ra bằng 0.

5.2 Phương án thực hiện #2

Fig.21 thể hiện xử lý các CTU trong ảnh. Các khác biệt được so sánh với Fig.19 được in đậm bằng các đường nét đứt.

5.3 Phương án thực hiện #3

8.8.5.2 Quá trình lọc CTB đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF,

– mảng mẫu ảnh độ sáng được tái tạo được lọc alfPicture$_L$,

– vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa alfPicture$_L$.

Mệnh đề của quá trình dẫn xuất chỉ số lọc 8.8.5.3 được gọi với vị trí (xCtb, yCtb) và mảng mẫu ảnh độ sáng được tái tạo recPicture$_L$ làm các đầu vào, và filtIdx[x][y] và transposeIdx[x][y] với x, y = 0..CtbSizeY − 1 làm đầu ra.

Để dẫn xuất các mẫu độ sáng được tái tạo được lọc alfPicture$_L$[x][y], mỗi mẫu độ sáng được tái tạo bên trong CTB độ sáng hiện tại recPicture$_L$[x][y] được lọc như sau với x, y = 0..CtbSizeY − 1:

– Mảng của các hệ số lọc độ sáng f[j] và mảng của các giá trị rút gọn độ sáng c[j] tương ứng với bộ lọc được xác định bởi filtIdx[x][y] được dẫn xuất như sau với j = 0..11:

– …

– Các hệ số lọc độ sáng và chỉ số của các giá trị rút gọn idx được dẫn xuất tùy thuộc vào transposeIdx[x][y] như sau:

– …

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −3..3 được dẫn xuất như sau:

– …

– Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu [[một hoặc nhiều]] điều kiện sau ***là*** true, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

[[– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.]]

– Ngược lại, applyVirtualBoundary được gán bằng 1.

– Các độ lệch mẫu được tái tạo r1, r2 và r3 được xác định trong Bảng 8-22 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary.

– ...

8.8.5.4 Quá trình lọc CTB đối với các mẫu sắc độ

Các đầu vào của quá trình này là:

– mảng mẫu ảnh sắc độ được tái tạo recPicture trước quá trình ALF,

– mảng mẫu ảnh sắc độ được tái tạo được lọc alfPicture,

– vị trí sắc độ (xCtbC, yCtbC) xác định mẫu trên bên trái của CTB sắc độ hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture.

Chiều rộng và chiều cao của CTB sắc độ hiện tại ctbWidthC và ctbHeightC được dẫn xuất như sau:

$$ctbWidthC = CtbSizeY/SubWidthC \quad (8\text{-}1230)$$

$$ctbHeightC = CtbSizeY/SubHeightC \quad (8\text{-}1231)$$

Để dẫn xuất các mẫu sắc độ được tái tạo được lọc alfPicture[x][y], mỗi mẫu sắc độ được tái tạo bên trong CTB sắc độ hiện tại recPicture[x][y] được lọc như sau với x = 0..ctbWidthC − 1, y = 0..ctbHeightC − 1:

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu sắc độ tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu sắc độ với i, j = −2..2 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtbC + x − PpsVirtualBoundariesPosX[n]/SubWidthC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

$$h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n]/SubWidthC, \quad (8\text{-}1232)$$
$$pic_width_in_luma_samples/SubWidthC - 1, xCtbC + x + i)$$

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n]/SubWidthC − xCtbC − x lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n]/SubWidthC − 1, xCtbC + x + i) (8-1233)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i) (8-1234)

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtbC + y − PpsVirtualBoundariesPosY[n]/SubHeightC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n]/SubHeightC , pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j) (8-1235)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n]/SubHeightC − yCtbC − y lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n]/SubHeightC − 1, yCtbC + y + j) (8-1236)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j) (8-1237)

– Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu [[một hoặc nhiều]] điều kiện sau ***là*** true, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– [[Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh and pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.]]

– Ngược lại, applyVirtualBoundary được gán bằng 1.

– Các độ lệch mẫu được tái tạo r1 và r2 được xác định trong Bảng 8-22 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary.

...

Theo cách khác, điều kiện "đường biên dưới của khối cây mã hiện tại là đường biên dưới của ảnh" có thể được thay thế bằng "đường biên dưới của CTB hiện tại là đường biên dưới của ảnh hoặc ngoài ảnh."

5.4 Phương án thực hiện #4

Phương án thực hiện thể hiện ví dụ không cho phép sử dụng các mẫu dưới vùng VPDU trong quá trình phân loại ALF (tương ứng với đề mục 7 trong phần 4).

8.8.5.3 Quá trình dẫn xuất đối với chỉ số lọc và chuyển vị ALF đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại,

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF.

Các đầu ra của quá trình này là

– mảng chỉ số lọc phân loại filtIdx[x][y] với x, y = 0..CtbSizeY − 1,

– mảng chỉ số chuyển vị transposeIdx[x][y] với x, y = 0..CtbSizeY − 1.

Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −2..5 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i)

(8-1193)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n] − xCtb − x lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i) (8-1194)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1195)

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j)

(8-1196)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1197)

– Ngược lại, phương trình sau áp dụng:

– Nếu yCtb + CtbSizeY lớn hơn hoặc bằng pic_height_in_luma_samples, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1198)

– Ngược lại, nếu y nhỏ hơn CtbSizeY − 4, phương trình sau áp dụng:

v_{y+j} = Clip3(0, yCtb + CtbSizeY − 5, yCtb + y + j) (8-1199)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(yCtb + CtbSizeY − 4, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1200)

Mảng chỉ số lọc phân loại filtIdx và mảng chỉ số chuyển vị transposeIdx được dẫn xuất bằng các bước có thứ tự sau:

1. Các biến filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] với x, y = − 2..CtbSizeY + 1 được dẫn xuất như sau:

– Nếu cả x và y là các số chẵn hoặc cả x và y là các số không chẵn, phương trình sau áp dụng:

filtH[x][y] = Abs((recPicture[h_x, v_y] << 1) − recPicture[h_{x-1}, v_y] − recPicture[h_{x+1}, v_y]) (8-1201)

filtV[x][y] = Abs((recPicture[h_x, v_y] << 1) − recPicture[h_x, v_{y-1}] − recPicture[h_x, v_{y+1}]) (8-1202)

filtD0[x][y] = Abs((recPicture[h_x, v_y] << 1) − recPicture[h_{x-1}, v_{y-1}] − recPicture[h_{x+1}, v_{y+1}]) (8-1203)

filtD1[x][y] = Abs((recPicture[h_x, v_y] << 1) − recPicture[h_{x+1}, v_{y-1}] − recPicture[h_{x-1}, v_{y+1}]) (8-1204)

– Ngược lại, filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] được gán bằng 0.

2. Các biến minY, maxY và ac được dẫn xuất như sau:

– Nếu (y << 2) bằng ***(min(CtbSizeY, 64)-8)*** [[(CtbSizeY − 8)]] và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples – 1, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.

– Ngược lại, nếu (y << 2) bằng ***(min(CtbSizeY, 64)-4)*** [[(CtbSizeY − 4)]] và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples – 1, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.

– Ngược lại, minY được gán bằng −2 và maxY được gán bằng 5 và ac được gán bằng 64.

3. Các biến varTempH1[x][y], varTempV1[x][y], varTempD01[x][y], varTempD11[x][y] và varTemp[x][y] với x, y = 0..(CtbSizeY − 1) >> 2 được dẫn xuất như sau:

sumH[x][y] = $\Sigma_i\Sigma_j$ filtH[$h_{(x << 2)+i}$ − xCtb][$v_{(y << 2)+j}$ − yCtb] với i = −2..5, j = minY..maxY (8-1205)

sumV[x][y] = $\Sigma_i\Sigma_j$ filtV[$h_{(x << 2)+i}$ − xCtb][$v_{(y << 2)+j}$ − yCtb] với i = −2..5, j = minY..maxY (8-1206)

sumD0[x][y] = $\Sigma_i\Sigma_j$ filtD0[$h_{(x << 2)+i}$ − xCtb][$v_{(y << 2)+j}$ − yCtb] với i = −2..5, j = minY..maxY (8-1207)

sumD1[x][y] = $\Sigma_i\Sigma_j$ filtD1[$h_{(x << 2)+i} - xCtb$][$v_{(y << 2)+j} - yCtb$] với i = −2..5, j = minY..maxY (8-1208)

sumOfHV[x][y] = sumH[x][y] + sumV[x][y] (8-1209)

4. Các biến dir1[x][y], dir2[x][y] và dirS[x][y] với x, y = 0..CtbSizeY − 1 được dẫn xuất như sau:

– Các biến hv1, hv0 và dirHV được dẫn xuất như sau:

…

– Các biến d1, d0 và dirD được dẫn xuất như sau:

…

5. Biến avgVar[x][y] với x, y = 0..CtbSizeY − 1 được dẫn xuất như sau:

varTab[] = {0, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4} (8-1227)

avgVar[x][y] = varTab[Clip3(0, 15, (sumOfHV[x >> 2][y >> 2] * ac) >> (3 + $BitDepth_Y$))] (8-1228)

6. Mảng chỉ số lọc phân loại filtIdx[x][y] và mảng chỉ số chuyển vị transposeIdx[x][y] với x = y = 0..CtbSizeY − 1 được dẫn xuất như sau:

transposeTable[] = {0, 1, 0, 2, 2, 3, 1, 3}

transposeIdx[x][y] = transposeTable[dir1[x][y] * 2 + (dir2[x][y] >> 1)]

filtIdx[x][y] = avgVar[x][y]

Khi dirS[x][y] không bằng 0, filtIdx[x][y] được cải biến như sau:

filtIdx[x][y] += (((dir1[x][y] & 0x1) << 1) + dirS[x][y]) * 5 (8-1229)

5.5 Phương án thực hiện #5

Đối với các mẫu đặt ở nhiều loại đường biên (chẳng hạn, đường biên lát/khối viên, đường biên ảo 360°), quá trình đệm chỉ được gọi một lần. Và bao nhiêu dòng được đệm theo phía phụ thuộc vào vị trí of mẫu hiện tại so với các đường biên.

Trong một ví dụ, phương pháp đệm 2 phía ALF được áp dụng. Theo cách khác, ngoài ra, trong phương pháp đệm 2 phía đối xứng, khi mẫu nằm ở hai đường biên, chẳng hạn, một đường biên ở phía trên và một đường biên ở phía dưới, bao nhiêu mẫu được đệm được quyết định bởi đường biên gần hơn như được thể hiện trên Fig.27. Đồng thời, khi dẫn xuất thông tin phân loại, chỉ 4 dòng giữa hai đường biên trên Fig.27 được sử dụng.

Fig.26 thể hiện ví dụ của các phương pháp đệm nếu 4 dòng mẫu của hai đường biên. Trong một ví dụ, đường biên thứ nhất trên Fig.26 có thể là đường biên ảo ALF; đường biên thứ hai trên Fig.25 có thể là đường biên lát/ô/khối viên hoặc đường biên ảo 360°.

5.6 Phương án thực hiện #6

8.8.5.2 Quá trình lọc CTB đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF,

– mảng mẫu ảnh độ sáng được tái tạo được lọc $alfPicture_L$,

–vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa $alfPicture_L$.

Mệnh đề của quá trình dẫn xuất chỉ số lọc 8.8.5.3 được gọi với vị trí (xCtb, yCtb) và mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ làm các đầu vào, và filtIdx[x][y] và transposeIdx[x][y] với x, y = 0..CtbSizeY − 1 làm đầu ra.

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −3..3 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i) (8-1197)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n]− xCtb − x lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i) (8-1198)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1199)

– [[Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j) (8-1200)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1201)]]

– [[Ngược lại, t]]***Phương trình sau áp dụng***:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1202)

– [[Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.

– Ngược lại, applyVirtualBoundary được gán bằng 1.]]

– ***Biến boundaryPos1 và boundaryPos2 được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên dọc đối với các mẫu độ sáng như được nêu trong 8.8.5.5 với yCtb bằng yCtb và y bằng y.***

Bảng 8-24 – Mô tả r1, r2, và r3 theo vị trí mẫu độ sáng ngang y và [[applyVirtualBoundary]] *boundaryPos1 và boundaryPos2.*

Điều kiện	**r1**	**r2**	**r3**
(y = = CtbSizeY − 5 || y = = CtbSizeY − 4) && (applyVirtualBoundary = = 1)	0	0	0
(y = = CtbSizeY − 6 || y = = CtbSizeY − 3) && (applyVirtualBoundary = = 1)	1	1	1
(y = = CtbSizeY − 7 || y = = CtbSizeY − 2) && (applyVirtualBoundary = = 1)	1	2	2
ngược lại	1	2	3

Điều kiện	*r1*	*r2*	*r3*
(y == boundaryPos1 − 1 || y == boundaryPos1) && (boundaryPos1 > −1 && (boundaryPos2 = = −1 || boundaryPos2 >= boundaryPos1 + 8))	*0*	*0*	*0*
(y == boundaryPos1 − 2 || y == boundaryPos1 + 1) && (boundaryPos1 > −1 && (boundaryPos2 = = −1 || boundaryPos2 >= boundaryPos1 + 8))	*1*	*1*	*1*
(y == boundaryPos1 − 3 || y == boundaryPos1 + 2) && (boundaryPos1 > −1 && (boundaryPos2 = = −1 || boundaryPos2 >= boundaryPos1 + 8))	*1*	*2*	*2*
(y = = boundaryPos1 − 1 || y = = boundaryPos1 || y = = boundaryPos2 − 1 || y == boundaryPos2) && (boundaryPos1 > −1 && boundaryPos2 = = boundaryPos1 + 4))	*0*	*0*	*0*
(y == boundaryPos1 − 2 || y == boundaryPos1 + 1 || y == boundaryPos2 − 2 || y == boundaryPos2 + 1) && (boundaryPos1 > −1 && boundaryPos2 = = boundaryPos1 + 4))	*1*	*1*	*1*
(y == boundaryPos1 − 3 || y == boundaryPos2 + 2) && (boundaryPos1 > −1 && boundaryPos2 = = boundaryPos1 + 4))	*1*	*2*	*2*
ngược lại	*1*	*2*	*3*

8.8.5.3 Quá trình dẫn xuất đối với chỉ số lọc và chuyển vị ALF đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại,

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF.

Các đầu ra của quá trình này là

– mảng chỉ số lọc phân loại filtIdx[x][y] với x, y = 0..CtbSizeY − 1,

– mảng chỉ số chuyển vị transposeIdx[x][y] với x, y = 0..CtbSizeY − 1.

Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −2..5 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i) (8-1208)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n] − xCtb − x lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i) (8-1209)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1210)

– [[Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j) (8-1211)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1212)

– Ngược lại, phương trình sau áp dụng:

– Nếu yCtb + CtbSizeY lớn hơn hoặc bằng pic_height_in_luma_samples, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1213)

– Ngược lại, nếu y nhỏ hơn CtbSizeY – 4, phương trình sau áp dụng:

v_{y+j} = Clip3(0, yCtb + CtbSizeY – 5, yCtb + y + j) (8-1214)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(yCtb + CtbSizeY – 4, pic_height_in_luma_samples – 1, yCtb + y + j) (8-1215)]]

– ***Biến boundaryPos1 and boundaryPos2 được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên dọc đối với các mẫu độ sáng như được nêu trong 8.8.5.5 với yCtb bằng yCtb và y bằng y.***

– ***Nếu boundaryPos1 bằng –1 và boundaryPos2 bằng –1, phương trình sau áp dụng:***

h_{x+i} = Clip3(0, pic_width_in_luma_samples – 1, xCtb + x + i) (8-1211)

– ***Ngược lại, nếu boundaryPos1 không bằng –1 và boundaryPos2 bằng –1, phương trình sau áp dụng:***

– ***Nếu y nhỏ hơn boundaryPos1, phương trình sau áp dụng:***

v_{y+j} = Clip3(0, yCtb + boundaryPos1 – 1, yCtb + y + j) (8-1215)

– ***Ngược lại, nếu y bằng hoặc lớn hơn boundaryPos1, phương trình sau áp dụng:***

v_{y+j} = Clip3(yCtb + boundaryPos1, pic_height_in_luma_samples – 1, yCtb + y + j) (8-1216)

– ***Ngược lại (boundaryPos1 không bằng –1 và boundaryPos2 không bằng –1), phương trình sau áp dụng:***

– ***Nếu y nhỏ hơn boundaryPos1, phương trình sau áp dụng:***

v_{y+j} = Clip3(0, yCtb + boundaryPos1 – 1, yCtb + y + j) (8-1215)

– ***Ngược lại, nếu y bằng hoặc lớn hơn boundaryPos2, phương trình sau áp dụng:***

v_{y+j} = Clip3(yCtb + boundaryPos2, pic_height_in_luma_samples – 1, yCtb + y + j) (8-1216)

– ***Ngược lại (y lớn hơn boundaryPos1 và bằng hoặc nhỏ hơn boundaryPos2), phương trình sau áp dụng:***

v_{y+j} = Clip3(yCtb + boundaryPos1, yCtb + boundaryPos2 – 1, yCtb + y + j) (8-1216)

Mảng chỉ số lọc phân loại filtIdx và mảng chỉ số chuyển vị transposeIdx được dẫn xuất bằng các bước có thứ tự sau:

1. Các biến filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] với x, y = − 2..CtbSizeY + 1 được dẫn xuất như sau:

– Nếu cả x và y là các số chẵn hoặc cả x và y là các số không chẵn, phương trình sau áp dụng:

$$filtH[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x-1}, v_y] - recPicture[h_{x+1}, v_y]) \quad (8\text{-}1216)$$

$$filtV[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_x, v_{y-1}] - recPicture[h_x, v_{y+1}]) \quad (8\text{-}1217)$$

$$filtD0[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x-1}, v_{y-1}] - recPicture[h_{x+1}, v_{y+1}]) \quad (8\text{-}1218)$$

$$filtD1[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x+1}, v_{y-1}] - recPicture[h_{x-1}, v_{y+1}]) \quad (8\text{-}1219)$$

– Ngược lại, filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] được gán bằng 0.

2. Các biến minY, maxY và ac được dẫn xuất như sau:

– Nếu (y << 2) bằng (CtbSizeY − 8) và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples − 1, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.

– Ngược lại, nếu (y << 2) bằng (CtbSizeY − 4) và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples − 1, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.

– ***Biến boundaryPos1 và boundaryPos2 được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên dọc đối với các mẫu độ sáng như được nêu trong 8.8.5.5 với yCtb bằng yCtb và y bằng y.***

– ***Nếu boundaryPos1 không bằng −1 và boundaryPos2 bằng −1, phương trình sau áp dụng:***

– ***Nếu y bằng boundaryPos1 – 4, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.***

– ***Ngược lại, nếu y bằng boundaryPos1, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.***

– ***Ngược lại, minY được gán bằng −2 và maxY được gán bằng 5 và ac được gán bằng 64.***

– ***Ngược lại, nếu boundaryPos1 không bằng −1 và boundaryPos2 không bằng −1, phương trình sau áp dụng:***

– ***Nếu y bằng boundaryPos1 – 4, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.***

– ***Ngược lại, nếu y bằng boundaryPos2, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.***

– ***Ngược lại, nếu y bằng boundaryPos1 và boundaryPos2 bằng hoặc lớn hơn boundaryPos2 + 8, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.***

– ***Ngược lại, nếu y bằng boundaryPos1 và boundaryPos2 bằng boundaryPos2 + 4, minY được gán bằng 0, maxY được gán bằng 3 và ac được gán bằng 128.***

– ***Ngược lại, nếu y bằng boundaryPos2 – 4 và boundaryPos2 bằng hoặc lớn hơn boundaryPos2 + 8, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.***

– ***Ngược lại, minY được gán bằng −2 và maxY được gán bằng 5 và ac được gán bằng 64.***

– ***Ngược lại (boundaryPos1 bằng −1 và boundaryPos2 bằng −1), minY được gán bằng −2 và maxY được gán bằng 5 và ac được gán bằng 64.***

– [[Ngược lại, minY được gán bằng −2 và maxY được gán bằng 5 và ac được gán bằng 64.]]

3. Các biến sumH[x][y], sumV[x][y], sumD0[x][y], sumD1[x][y] và sumOfHV[x][y] với x, y = 0..(CtbSizeY − 1) >> 2 được dẫn xuất như sau:

$sumH[x][y] = \Sigma_i\Sigma_j\ filtH[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = -2..5, j = minY..maxY$ (8-1220)

$sumV[x][y] = \Sigma_i\Sigma_j\ filtV[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = -2..5, j = minY..maxY$ (8-1221)

$sumD0[x][y] = \Sigma_i\Sigma_j\ filtD0[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = -2..5, j = minY..maxY$ (8-1222)

sumD1[x][y] = $\Sigma_i\Sigma_j$ filtD1[$h_{(x << 2)+i}$ − xCtb][$v_{(y << 2)+j}$ − yCtb] với i = −2..5, j = minY..maxY (8-1223)

sumOfHV[x][y] = sumH[x][y] + sumV[x][y] (8-1224)

...

8.8.5.4 Quá trình lọc CTB đối với các mẫu sắc độ

Các đầu vào của quá trình này là:

– mảng mẫu ảnh sắc độ được tái tạo recPicture trước quá trình ALF,

– mảng mẫu ảnh sắc độ được tái tạo được lọc alfPicture,

– vị trí sắc độ (xCtbC, yCtbC) xác định mẫu trên bên trái của CTB sắc độ hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture.

Chiều rộng và chiều cao của CTB sắc độ hiện tại ctbWidthC and ctbHeightC được dẫn xuất như sau:

ctbWidthC = CtbSizeY/SubWidthC (8-1245)

ctbHeightC = CtbSizeY/SubHeightC (8-1246)

Để dẫn xuất các mẫu sắc độ được tái tạo được lọc alfPicture[x][y], mỗi mẫu sắc độ được tái tạo bên trong CTB sắc độ hiện tại recPicture[x][y] được lọc như sau với x = 0..ctbWidthC − 1, y = 0..ctbHeightC − 1:

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu sắc độ tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu sắc độ với i, j = −2..2 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và xCtbC + x − PpsVirtualBoundariesPosX[n]/SubWidthC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n]/SubWidthC, (8-1247)
pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosX[n]/SubWidthC − xCtbC − x lớn hơn 0 và

nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n]/SubWidthC − 1, xCtbC + x + i) (8-1248)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i) (8-1249)

– [[If pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtbC + y − PpsVirtualBoundariesPosY[n]/SubHeightC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n]/SubHeightC , (8-1250)
pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n]/SubHeightC − yCtbC − y lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n]/SubHeightC − 1, yCtbC + y + j)(8-1251)

– Ngược lại,]] ***Phương trình sau áp dụng***:

v_{y+j} = Clip3(0, pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j) (8-1252)

– [[Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.

– Ngược lại, applyVirtualBoundary được gán bằng 1.]]

– Biến boundaryPos1 và boundaryPos2 được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên dọc đối với các mẫu độ sáng như được nêu trong 8.8.5.5 với yCtb bằng yCtb và y bằng y.

– Biến boundaryPos1 được gán bằng boundaryPos1/SubWidthC.

– Biến boundaryPos2 được gán bằng boundaryPos2/SubWidthC.

– Các độ lệch mẫu được tái tạo r1 và r2 được xác định trong Bảng 8-24 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary.

– Biến curr được dẫn xuất như sau:

$curr = recPicture[h_x, v_y]$ (8-1253)

– Mảng của các hệ số lọc sắc độ f[j] và mảng của các giá trị rút gọn sắc độ c[j] được dẫn xuất như sau với j = 0..5:

$f[j] = AlfCoeff_C[slice_alf_aps_id_chroma][j]$ (8-1254)

$c[j] = AlfClip_C[slice_alf_aps_id_chroma][j]$ (8-1255)

– Biến sum được dẫn xuất như sau:

$$\begin{aligned} sum = {} & f[0] * (Clip3(-c[0], c[0], recPicture[h_x, v_{y+r2}] - curr) + \\ & Clip3(-c[0], c[0], recPicture[h_x, v_{y-r2}] - curr)) + \\ & f[1] * (Clip3(-c[1], c[1], recPicture[h_{x+1}, v_{y+r1}] - curr) + \\ & Clip3(-c[1], c[1], recPicture[h_{x-1}, v_{y-r1}] - curr)) + \\ & f[2] * (Clip3(-c[2], c[2], recPicture[h_x, v_{y+r1}] - curr) + \\ & Clip3(-c[2], c[2], recPicture[h_x, v_{y-r1}] - curr)) + \\ & f[3] * (Clip3(-c[3], c[3], recPicture[h_{x-1}, v_{y+r1}] - curr) + \\ & Clip3(-c[3], c[3], recPicture[h_{x+1}, v_{y-r1}] - curr)) + \\ & f[4] * (Clip3(-c[4], c[4], recPicture[h_{x+2}, v_y] - curr) + \\ & Clip3(-c[4], c[4], recPicture[h_{x-2}, v_y] - curr)) + \\ & f[5] * (Clip3(-c[5], c[5], recPicture[h_{x+1}, v_y] - curr) + \\ & Clip3(-c[5], c[5], recPicture[h_{x-1}, v_y] - curr)) \end{aligned} \quad (8\text{-}1256)$$

$sum = curr + (sum + 64) >> 7)$ (8-1257)

– Mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture[xCtbC + x][yCtbC + y] được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[(xCtbC + x) * SubWidthC][(yCtbC + y) * SubHeightC] đều bằng 1, phương trình sau áp dụng:

$alfPicture[xCtbC + x][yCtbC + y] = recPicture_L[h_x, v_y]$ (8-1258)

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 hoặc pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

alfPicture[xCtbC + x][yCtbC + y] = Clip3(0, (1 << BitDepthc) − 1, sum) (8-1259)

[[Bảng 8-25 – Mô tả r1 và r2 theo vị trí mẫu độ sáng ngang y và applyVirtualBoundary]]

[[điều kiện	**r1**	**r2**
(y = = ctbHeightC − 2 || y = = ctbHeightC − 3) && (applyVirtualBoundary = = 1)	0	0
(y = = ctbHeightC − 1 || y = = ctbHeightC − 4) && (applyVirtualBoundary = = 1)	1	1
ngược lại	1	2]]

Mảng 8-25 -- Mô tả r1 và r2 theo vị trí mẫu độ sáng ngang y và boundaryPos1 và boundaryPos2

Điều kiện	**r1**	**r2**
(y == boundaryPos1 − 1 || y == boundaryPos1) && (boundaryPos1 > −1 && (boundaryPos2 = = −1 || boundaryPos2 >= boundaryPos 1 + 4))	0	0
(y == boundaryPos1 − 2 || y == boundaryPos1 + 1) && (boundaryPos1 > −1 && (boundaryPos2 = = −1 || boundaryPos2 >= boundaryPos 1 + 4))	1	1
(y = = boundaryPos1 − 1 || y = = boundaryPos1 || y = = boundaryPos2 − 1 || y = = boundaryPos2) && (boundaryPos1 > −1 && boundaryPos2 = = boundaryPos1 + 2))	0	0
(y == boundaryPos1 − 2 || y == boundaryPos2 + 1) && (boundaryPos1 > −1 && boundaryPos2 = = boundaryPos1 + 2))	1	1
ngược lại	1	2

8.8.5.5Quá trình dẫn xuất vị trí đường biên dọc đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– ***vị trí độ sáng yCtb xác định vị trí dọc của mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.***

– ***vị trí độ sáng y xác định vị trí dọc của pixel hiện tại so với mẫu trên bên trái của CTB độ sáng hiện tại.***

Đầu ra của quá trình này là:

– ***vị trí biên dọc trên boundaryPos1.***

– ***vị trí biên dọc dưới boundaryPos2.***

– ***Biến boundaryPos1 được dẫn xuất như sau:***

– *Biến boundaryPos1 được gán bằng −1.*

– *Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:*

boundaryPos1 = PpsVirtualBoundariesPosY[n] (8-1201)

– *Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1 và PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:*

boundaryPos1 = PpsVirtualBoundariesPosY[n] (8-1201)

– *Biến boundaryPos2 được dẫn xuất như sau:*

– *Biến boundaryPos2 được gán bằng −1.*

– *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0, phương trình sau áp dụng:*

– *Nếu CtbSizeY − y nhỏ hơn 4, biến boundaryPos2 được gán bằng CtbSizeY.*

– *Ngược lại, nếu đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0, phương trình sau áp dụng:*

– *Nếu CtbSizeY − y nhỏ hơn 4, biến boundaryPos2 được gán bằng CtbSizeY.*

– *Ngược lại, nếu đường biên trên của CTB hiện tại là đường biên trên của khối viên và loop_filter_across_bricks_enabled_flag bằng 0, phương trình sau áp dụng:*

– *Nếu y nhỏ hơn 3, biến boundaryPos2 được gán bằng 0.*

– *Ngược lại, nếu đường biên trên của CTB hiện tại là đường biên trên của lát và loop_filter_across_slices_enabled_flag bằng 0, phương trình sau áp dụng:*

– *Nếu y nhỏ hơn 3, biến boundaryPos2 được gán bằng 0.*

– *Nếu biến boundaryPos1 bằng −1, phương trình sau áp dụng:*

– *Biến boundaryPos1 bằng boundaryPos2.*

– *Biến boundaryPos2 bằng −1.*

– *Biến applyVirtualBoundary được dẫn xuất như sau:*

– *Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:*

– *Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.*

– *Ngược lại, applyVirtualBoundary được gán bằng 1.*

– *Nếu applyVirtualBoundary bằng 1, phương trình sau áp dụng:*

– *Nếu boundaryPos1 bằng −1, boundaryPos1 được gán bằng CtbSizeY − 4.*

– *Ngược lại, nếu boundaryPos2 bằng −1, boundaryPos2 được gán bằng CtbSizeY − 4.*

– *Nếu boundaryPos1 không bằng −1 và boundaryPos1 không bằng −1, phương trình sau áp dụng:*

– *Nếu boundaryPos1 bằng boundaryPos2, boundaryPos2 được gán bằng −1.*

– *Ngược lại, nếu boundaryPos1 lớn hơn boundaryPos2, phương trình sau áp dụng:*

– *Biến tmpPos được gán bằng boundaryPos2.*

– *Biến boundaryPos2 được gán bằng boundaryPos1.*

Biến boundaryPos1 được gán bằng tmpPos.

5.7 Phương án thực hiện #7

Đối với CTU, có thể không trùng với các đường biên bất kỳ (chẳng hạn, ảnh/lát/ô/khối viên/đường biên ảnh con). Tuy nhiên, có thể cần truy nhập các mẫu ngoài khối hiện tại (chẳng hạn, ảnh/lát/ô/khối viên/ảnh con). Nếu lọc giao đường biên lát (chẳng hạn, loop_filter_across_slices_enabled_flag is false) bị vô hiệu hóa, cần đệm mẫu ngoài khối hiện tại.

Chẳng hạn, đối với mẫu 2801 (lấy mẫu độ sáng làm ví dụ) trên Fig.28, các mẫu được sử dụng trong ALF quá trình lọc có thể được đệm như trên Fig.29.

5.8 Phương án thực hiện #8

Theo phương án thực hiện, các ý tưởng chỉnh sau được áp dụng:

Khi kích hoạt các đường biên ảo ALF:

- Đối với các CTU không được đặt trong hàng CTU cuối cùng trong ảnh (chẳng hạn, đường biên dưới của các CTU không phải là đường biên dưới của ảnh hoặc vượt quá đường biên dưới của ảnh), đường biên ảo ALF được kích hoạt, tức là, một CTU có thể được tách thành hai hoặc nhiều phần, và các mẫu trong một phần không được phép sử dụng các mẫu trong phần khác.

- Đối với các CTU được đặt trong hàng CTU cuối cùng trong ảnh (chẳng hạn, đường biên dưới của các CTU là đường biên dưới của ảnh hoặc vượt quá đường biên dưới củảnh), đường biên ảo ALF được kích hoạt, tức là, một CTU có thể được tách thành hai hoặc nhiều phần, và các mẫu ở một phần không được phép sử dụng các mẫu trong phần khác.

Khi đệm các đường biên (bao gồm các đường biên ảo ALF, các đường biên lát/ô/khối viên/ảnh con, "các đường biên ảo 360") trong quá trình phân loại:

Đối với mẫu ở một (hoặc nhiều loại) đường biên, khi các mẫu lân cận giao với đường biên không được phép được sử dụng, đệm 1 phía được thực hiện để đệm các mẫu lân cận này.

Khi đệm các đường biên (bao gồm các đường biên ảo ALF, các đường biên lát/ô/khối viên/ảnh con, "360 các đường biên ảo") trong quá trình lọc ALF:

- Đối với mẫu ở một (hoặc nhiều loại) đường biên vốn là lát/ô/khối viên/đường biên ảnh con hoặc "đường biên ảo 360" trùng với đường biên CTU, khi các mẫu lân cận giao với đường biên không được phép được sử dụng, đệm 2 phía được thực hiện để đệm các mẫu lân cận này.

- Đối với mẫu ở một (hoặc nhiều loại) đường biên vốn là đường biên ảnh hoặc "đường biên ảo 360" không trùng với đường biên CTU, khi các mẫu lân cận giao với đường biên không được phép được sử dụng, đệm 1 phía được thực hiện để đệm các mẫu lân cận này.

8.8.5.2 Quá trình lọc CTB đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF,

– mảng mẫu ảnh độ sáng được tái tạo được lọc $alfPicture_L$,

–vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa $alfPicture_L$.

Mệnh đề của quá trình dẫn xuất chỉ số lọc 8.8.5.3 được gọi với vị trí (xCtb, yCtb) và mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ làm các đầu vào, và filtIdx[x][y] và transposeIdx[x][y] với x, y = 0..CtbSizeY − 1 làm đầu ra.

Để dẫn xuất các mẫu độ sáng được tái tạo được lọc $alfPicture_L$[x][y], mỗi mẫu độ sáng được tái tạo bên trong CTB độ sáng hiện tại $recPicture_L$[x][y] được lọc như sau với x, y = 0..CtbSizeY − 1:

– Mảng của các hệ số lọc độ sáng f[j] và mảng của các giá trị rút gọn độ sáng c[j] tương ứng với bộ lọc được xác định bởi filtIdx[x][y] được dẫn xuất như sau với j = 0..11:

– Nếu AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] nhỏ hơn 16, phương trình sau áp dụng:

i = AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] (8-1187)

f[j] = AlfFixFiltCoeff[AlfClassToFiltMap[i][filtIdx[x][y]]][j] (8-1188)

$c[j] = 2^{BitdepthY}$ (8-1189)

– Ngược lại, (AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] lớn hơn hoặc bằng 16, phương trình sau áp dụng:

i = slice_alf_aps_id_luma[AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] − 16] (8-1190)

f[j] = $AlfCoeff_L$[i][filtIdx[x][y]][j] (8-1191)

c[j] = $AlfClip_L$[i][filtIdx[x][y]][j] (8-1192)

– Các hệ số lọc độ sáng và chỉ số của các giá trị rút gọn idx được dẫn xuất tùy thuộc vào transposeIdx[x][y] như sau:

– Nếu transposeIndex[x][y] bằng 1, phương trình sau áp dụng:

idx[] = {9, 4, 10, 8, 1, 5, 11, 7, 3, 0, 2, 6} (8-1193)

– Ngược lại, nếu transposeIndex[x][y] bằng 2, phương trình sau áp dụng:

idx[] = {0, 3, 2, 1, 8, 7, 6, 5, 4, 9, 10, 11} (8-1194)

– Ngược lại, nếu transposeIndex[x][y] bằng 3, phương trình sau áp dụng:

idx[] = {9, 8, 10, 4, 3, 7, 11, 5, 1, 0, 2, 6} (8-1195)

– Ngược lại, phương trình sau áp dụng:

idx[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11} (8-1196)

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −3..3 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i) (8-1197)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosX[n]− xCtb − x lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i) (8-1198)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1199)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

h_{x+i} = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, h_{x+i}) (8-1184)]]

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j) (8-1200)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1201)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1202)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

v_{y+j} = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, v_{y+j}) (8-1184)

– Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh con và loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.

– Ngược lại, applyVirtualBoundary được gán bằng 1.]]

– ***Biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí biên ngang đối với các mẫu độ sáng như được nêu trong cluase 8.8.5.5。 với (xCtb, yCtb) và (x, y) làm các đầu vào.***

– Các độ lệch mẫu được tái tạo r1, r2 and r3 được xác định trong Bảng 8-24 theo vị trí mẫu độ sáng ngang y và ***clipTopPos và clipBottomPos*** [[applyVirtualBoundary]].

– ***Các độ lệch mẫu được tái tạo c1, c2 và c3 được xác định trong Bảng 8-25 theo vị trí mẫu độ sáng dọc x và clipLeftPos và clipRightPos.***

– Biến curr được dẫn xuất như sau:

$$curr = recPicture_L[h_x, v_y] \quad (8\text{-}1203)$$

- Biến sum được dẫn xuất như sau:

$$\begin{aligned}sum = {} & f[idx[0]] * (Clip3(-c[idx[0]], c[idx[0]], recPicture_L[h_x, v_{y+r3}] - curr) + \\ & Clip3(-c[idx[0]], c[idx[0]], recPicture_L[h_x, v_{y-r3}] - curr)) + \\ & f[idx[1]] * (Clip3(-c[idx[1]], c[idx[1]], recPicture_L[h_{x+c1}, v_{y+r2}] - curr) + \\ & Clip3(-c[idx[1]], c[idx[1]], recPicture_L[h_{x-c1}, v_{y-r2}] - curr)) + \\ & f[idx[2]] * (Clip3(-c[idx[2]], c[idx[2]], recPicture_L[h_x, v_{y+r2}] - curr) + \\ & Clip3(-c[idx[2]], c[idx[2]], recPicture_L[h_x, v_{y-r2}] - curr)) + \\ & f[idx[3]] * (Clip3(-c[idx[3]], c[idx[3]], recPicture_L[h_{x-c1}, v_{y+r2}] - curr) + \\ & Clip3(-c[idx[3]], c[idx[3]], recPicture_L[h_{x+c1}, v_{y-r2}] - curr)) + \\ & f[idx[4]] * (Clip3(-c[idx[4]], c[idx[4]], recPicture_L[h_{x+c2}, v_{y+r1}] - curr) + \\ & Clip3(-c[idx[4]], c[idx[4]], recPicture_L[h_{x-c2}, v_{y-r1}] - curr)) + \\ & f[idx[5]] * (Clip3(-c[idx[5]], c[idx[5]], recPicture_L[h_{x+c1}, v_{y+r1}] - curr) + \\ & Clip3(-c[idx[5]], c[idx[5]], recPicture_L[h_{x-c1}, v_{y-r1}] - curr)) + \\ & f[idx[6]] * (Clip3(-c[idx[6]], c[idx[6]], recPicture_L[h_x, v_{y+r1}] - curr) + \\ & Clip3(-c[idx[6]], c[idx[6]], recPicture_L[h_x, v_{y-r1}] - curr)) + \\ & f[idx[7]] * (Clip3(-c[idx[7]], c[idx[7]], recPicture_L[h_{x-c1}, v_{y+r1}] - curr) + \\ & Clip3(-c[idx[7]], c[idx[7]], recPicture_L[h_{x+c1}, v_{y-r1}] - curr)) + \\ & f[idx[8]] * (Clip3(-c[idx[8]], c[idx[8]], recPicture_L[h_{x-c2}, v_{y+r1}] - curr) + \\ & Clip3(-c[idx[8]], c[idx[8]], recPicture_L[h_{x+c2}, v_{y-r1}] - curr)) + \\ & f[idx[9]] * (Clip3(-c[idx[9]], c[idx[9]], recPicture_L[h_{x+c3}, v_y] - curr) + \\ & Clip3(-c[idx[9]], c[idx[9]], recPicture_L[h_{x-c3}, v_y] - curr)) + \\ & f[idx[10]] * (Clip3(-c[idx[10]], c[idx[10]], recPicture_L[h_{x+c2}, v_y] - curr) + \\ & Clip3(-c[idx[10]], c[idx[10]], recPicture_L[h_{x-c2}, v_y] - curr)) + \\ & f[idx[11]] * (Clip3(-c[idx[11]], c[idx[11]], recPicture_L[h_{x+c1}, v_y] - curr) + \\ & Clip3(-c[idx[11]], c[idx[11]], recPicture_L[h_{x-c1}, v_y] - curr)) \end{aligned} \quad (8\text{-}1204)$$

$$sum = curr + ((sum + 64) >> 7) \quad (8\text{-}1205)$$

– Mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa $alfPicture_L[xCtb + x][yCtb + y]$ được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[xCtb+ x][yCtb + y] đều bằng 1, phương trình sau áp dụng:

$alfPicture_L[xCtb + x][yCtb + y] = recPicture_L[h_x, v_y]$ (8-1206)

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 hoặc pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

$alfPicture_L[xCtb + x][yCtb + y] = Clip3(0, (1 << BitDepth_Y) - 1, sum)$ (8-1207)

Bảng 8-24 – Mô tả r1, r2, và r3 theo vị trí mẫu độ sáng ngang y, *clipTopPos và clipBottomPos* [[và applyVirtualBoundary]]

[[Điều kiện	r1	r2	r3
(y == CtbSizeY − 5 || y == CtbSizeY − 4) && (applyVirtualBoundary == 1)	0	0	0
(y == CtbSizeY − 6 || y == CtbSizeY − 3) && (applyVirtualBoundary == 1)	1	1	1
(y == CtbSizeY − 7 || y == CtbSizeY − 2) && (applyVirtualBoundary == 1)	1	2	2
ngược lại	1	2	3]]

Điều kiện	*r1*	*r2*	*r3*
(y == clipBottomPos − 1 || y == clipTopPos)	*0*	*0*	*0*
(y == clipBottomPos − 2 || y == clipTopPos + 1)	*1*	*1*	*1*
(y == clipBottomPos − 3 || y == clipTopPos + 2) && (clipBottomPos != clipTopPos + 4)	*1*	*2*	*2*
(y == clipTopPos + 2) && (clipBottomPos == clipTopPos + 4)	*1*	*1*	*1*
ngược lại	*1*	*2*	*3*

Bảng 8-24 – Mô tả c1, c2, và c3 theo vị trí mẫu độ sáng dọc x, clipLeftPos và clipRightPos

Điều kiện	*c1*	*c2*	*c3*
(xCtb + x == clipLeftPos || xCtb + x == clipRightPos -1)	*0*	*0*	*0*
(xCtb + x == clipLeftPos +1 || xCtb + x == clipRightPos -2)	*1*	*1*	*1*
(xCtb + x == clipLeftPos +2 || xCtb + x == clipRightPos -3)	*1*	*2*	*2*
ngược lại	*1*	*2*	*3*

8.8.5.3 Quá trình dẫn xuất đối với chỉ số lọc và chuyển vị ALF đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại,

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF.

Các đầu ra của quá trình này là

– mảng chỉ số lọc phân loại filtIdx[x][y] với x, y = 0..CtbSizeY − 1,

– mảng chỉ số chuyển vị transposeIdx[x][y] với x, y = 0..CtbSizeY − 1.

Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −2..5 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i) (8-1208)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosX[n] − xCtb − x lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i)

(8-1209)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i)

(8-1210)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

h_{x+i} = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, h_{x+i})

(8-1184)]]

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và***

yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j) (8-1211)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1212)

– Ngược lại, phương trình sau áp dụng:

– Nếu yCtb + CtbSizeY lớn hơn hoặc bằng pic_height_in_luma_samples, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1213)

– [[Ngược lại, nếu y nhỏ hơn CtbSizeY − 4, phương trình sau áp dụng:

v_{y+j} = Clip3(0, yCtb + CtbSizeY − 5, yCtb + y + j) (8-1214)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(yCtb + CtbSizeY − 4, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1215)

– Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

v_{y+j} = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, v_{y+j}) (8-1184)]]

– ***Biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên ngang đối với các mẫu độ sáng như được nêu trong mệnh đề 8.8.5.5。 với (xCtb, yCtb) và (x, y).***

– ***Nếu clipTopPos không bằng −128, phương trình sau áp dụng:***

v_{y+j} = Clip3(clipTopPos, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1213)

– ***Nếu clipBottomPos không bằng −128, phương trình sau áp dụng:***

v_{y+j} = Clip3(0, clipBottomPos − 1, yCtb + y + j) (8-1213)

– ***Nếu clipLeftPos không bằng −128, phương trình sau áp dụng:***

h_{x+i} = *Clip3(clipLeftPos,*

pic_width_in_luma_samples − 1, xCtb + x + i) *(8-1213)*

– ***Nếu clipRightPos không bằng −128, phương trình sau áp dụng:***

h_{x+i} = *Clip3(0, clipRightPos − 1, xCtb + x + i)* *(8-1213)*

Mảng chỉ số lọc phân loại filtIdx và mảng chỉ số chuyển vị transposeIdx được dẫn xuất bằng các bước có thứ tự sau:

1. Các biến filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] với x, y = − 2..CtbSizeY + 1 được dẫn xuất như sau:

– Nếu cả x và y là các số chẵn hoặc cả x và y là các số không chẵn, phương trình sau áp dụng:

$$filtH[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x-1}, v_y] - recPicture[h_{x+1}, v_y]) \quad (8\text{-}1216)$$

$$filtV[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_x, v_{y-1}] - recPicture[h_x, v_{y+1}]) \quad (8\text{-}1217)$$

$$filtD0[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x-1}, v_{y-1}] - recPicture[h_{x+1}, v_{y+1}]) \quad (8\text{-}1218)$$

$$filtD1[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x+1}, v_{y-1}] - recPicture[h_{x-1}, v_{y+1}]) \quad (8\text{-}1219)$$

– Ngược lại, filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] được gán bằng 0.

2. [[Các biến minY, maxY và ac được dẫn xuất như sau:

– Nếu (y << 2) bằng (CtbSizeY − 8) và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples − 1, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.

– Ngược lại, nếu (y << 2) bằng (CtbSizeY − 4) và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples − 1, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.]]

3. Các biến sumH[x][y], sumV[x][y], sumD0[x][y], sumD1[x][y] và sumOfHV[x][y] với x, y = 0..(CtbSizeY − 1) >> 2 được dẫn xuất như sau:

Các biến minY, maxY và ac được dẫn xuất như sau:

- *Nếu clipTopPos không bằng −128 and clipBottomPos bằng −128, phương trình sau áp dụng:*
  - *Nếu (y << 2) bằng clipTopPos, minY được gán bằng 0, maxY được gán bằng 5.*
- *Ngược lại, nếu clipTopPos bằng −128 và clipBottomPos không bằng −128, phương trình sau áp dụng:*
  - *Nếu (y << 2) bằng (clipBottomPos – 4), minY được gán bằng −2, maxY được gán bằng 3.*
- *Ngược lại, nếu clipTopPos không bằng −128 và clipBottomPos không bằng −128, phương trình sau áp dụng:*
  - *minY được gán bằng 0, maxY được gán bằng 3.*
- *Ngược lại, minY được gán bằng −2 và maxY được gán bằng 5 ~~and ac được gán bằng 64.~~*

Các biến minX, maxX và ac được dẫn xuất như sau:

- *Nếu clipLeftPos không bằng −128, phương trình sau áp dụng:*
  - *Nếu (x << 2) bằng clipLeftPos, minX được gán bằng 0, maxX được gán bằng 5.*
- *Ngược lại, nếu clipRightPos không bằng −128, phương trình sau áp dụng:*
  - *Nếu (x << 2) bằng (clipRightPos – 4), minX được gán bằng −2, maxX được gán bằng 3.*
- *Ngược lại, minX được gán bằng −2 và maxX được gán bằng 5.*

Biến ac được dẫn xuất theo Table 8-24.

Bảng 8-24 – Mô tả ac theo minX, minY, maxX, maxY

*(maxY – minY + 1) * (maxX – minX + 1)*	*8*8*	*8*6*	*8*4*	*6*6*	*6*4*
Ac	*64*	*96*	*128*	*112*	*192*

$sumH[x][y] = \Sigma_i\Sigma_j \; filtH[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = minX..maxX$ [[−2..5]], $j = minY..maxY$ (8-1220)

$sumV[x][y] = \Sigma_i\Sigma_j \; filtV[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = minX..maxX$ [[−2..5]], $j = minY..maxY$ (8-1221)

$sumD0[x][y] = \Sigma_i\Sigma_j\ filtD0[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = \underline{\boldsymbol{minX..maxX}}$ [[−2..5]], j = minY..maxY (8-1222)

$sumD1[x][y] = \Sigma_i\Sigma_j\ filtD1[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = \underline{\boldsymbol{minX..maxX}}$ [[−2..5]], j = minY..maxY (8-1223)

sumOfHV[x][y] = sumH[x][y] + sumV[x][y] (8-1224)

4. Các biến dir1[x][y], dir2[x][y] và dirS[x][y] với x, y = 0..CtbSizeY − 1 được dẫn xuất như sau:
   - Các biến hv1, hv0 and dirHV được dẫn xuất như sau:
     - Nếu sumV[x >> 2][y >> 2] lớn hơn sumH[x >> 2][y >> 2], phương trình sau áp dụng:

       hv1 = sumV[x >> 2][y >> 2] (8-1225)

       hv0 = sumH[x >> 2][y >> 2] 8-1226)

       dirHV = 1 (8-1227)
     - Ngược lại, phương trình sau áp dụng:

       hv1 = sumH[x >> 2][y >> 2] (8-1228)

       hv0 = sumV[x >> 2][y >> 2] (8-1229)

       dirHV = 3 (8-1230)
   - Các biến d1, d0 và dirD được dẫn xuất như sau:
     - Nếu sumD0[x >> 2][y >> 2] lớn hơn sumD1[x >> 2][y >> 2], phương trình sau áp dụng:

       d1 = sumD0[x >> 2][y >> 2] (8-1231)

       d0 = sumD1[x >> 2][y >> 2] (8-1232)

       dirD = 0 (8-1233)
     - Ngược lại, phương trình sau áp dụng:

d1 = sumD1[x >> 2][y >> 2]

(8-1234)

d0 = sumD0[x >> 2][y >> 2]

(8-1235)

dirD = 2 (8-1236)

– Các biến hvd1, hvd0, được dẫn xuất như sau:

hvd1 = (d1 * hv0 > hv1 * d0) ? d1 : hv1 (8-1237)

hvd0 = (d1 * hv0 > hv1 * d0) ? d0 : hv0 (8-1238)

– Các biến dirS[x][y], dir1[x][y] và dir2[x][y] được dẫn xuất như sau:

dir1[x][y] = (d1 * hv0 > hv1 * d0) ? dirD : dirHV (8-1239)

dir2[x][y] = (d1 * hv0 > hv1 * d0) ? dirHV : dirD (8-1240)

dirS[x][y] = (hvd1 > 2 * hvd0) ?1: ((hvd1 * 2 > 9 * hvd0) ? 2: 0) (8-1241)

5. Biến avgVar[x][y] với x, y = 0..CtbSizeY − 1 được dẫn xuất như sau:

varTab[] = {0, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4} (8-1242)

avgVar[x][y] = varTab[Clip3(0, 15, (sumOfHV[x >> 2][y >> 2] * ac) >> (3 + BitDepth$_Y$))] (8-1243)

6. Mảng chỉ số lọc phân loại filtIdx[x][y] và mảng chỉ số chuyển vị transposeIdx[x][y] với x = y = 0..CtbSizeY − 1 được dẫn xuất như sau:

transposeTable[] = {0, 1, 0, 2, 2, 3, 1, 3}

transposeIdx[x][y] = transposeTable[dir1[x][y] * 2 + (dir2[x][y] >> 1)]

filtIdx[x][y] = avgVar[x][y]

Khi dirS[x][y] không bằng 0, filtIdx[x][y] được cải biến như sau:

filtIdx[x][y] += (((dir1[x][y] & 0x1) << 1) + dirS[x][y]) * 5 (8-1244)

8.8.5.4 Quá trình lọc CTB đối với các mẫu sắc độ

Các đầu vào của quá trình này là:

– mảng mẫu ảnh sắc độ được tái tạo recPicture trước quá trình ALF,

– mảng mẫu ảnh sắc độ được tái tạo được lọc alfPicture,

– vị trí sắc độ (xCtbC, yCtbC) xác định mẫu trên bên trái của CTB sắc độ hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture.

Chiều rộng và chiều cao của CTB sắc độ hiện tại ctbWidthC và ctbHeightC được dẫn xuất như sau:

ctbWidthC = CtbSizeY/SubWidthC (8-1245)

ctbHeightC = CtbSizeY/SubHeightC (8-1246)

Để dẫn xuất các mẫu sắc độ được tái tạo được lọc alfPicture[x][y], mỗi mẫu sắc độ được tái tạo bên trong CTB sắc độ hiện tại recPicture[x][y] được lọc như sau với x = 0..ctbWidthC − 1, y = 0..ctbHeightC − 1:

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu sắc độ tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu sắc độ với i, j = −2..2 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** xCtbC + x − PpsVirtualBoundariesPosX[n]/SubWidthC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n]/SubWidthC, (8-1247)
pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosX[n]/SubWidthC − xCtbC − x lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n]/SubWidthC − 1, xCtbC + x + i) (8-1248)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i) (8-1249)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

h_{x+i} = Clip3(SubPicLeftBoundaryPos/SubWidthC, SubPicRightBoundaryPos/SubWidthC, h_{x+i}) (8-1184)]]

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** yCtbC + y − PpsVirtualBoundariesPosY[n]/SubHeightC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n]/SubHeightC , (8-1250)
pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosY[n]/SubHeightC − yCtbC − y lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n]/SubHeightC − 1, yCtbC + y + j) (8-1251)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j) (8-1252)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

v_{y+j} = Clip3(SubPicTopBoundaryPos/SubWidthC, SubPicBotBoundaryPos/SubWidthC, v_{y+j}) (8-1184)

– Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh con và loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.

– Ngược lại, applyVirtualBoundary được gán bằng 1.]]

– ***Biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí biên ngang đối với các mẫu độ sáng như được nêu trong mệnh đề 8.8.5.5 với (xCtbC* SubWidthC, yCtbC*SubWidthC) và (x*SubWidthC, y*SubWidthC) làm các đầu vào.***

– ***Biến clipTopPos được gán bằng clipLeftPos/SubWidthC.***

– ***Biến clipBottomPos được gán bằng clipRightPos/SubWidthC.***

– ***Biến clipTopPos được gán bằng clipTopPos/SubWidthC.***

– ***Biến clipBottomPos được gán bằng clipBottomPos/SubWidthC.***

– Các độ lệch mẫu được tái tạo r1 và r2 được xác định trong Bảng 8-27 theo vị trí mẫu độ sáng ngang y, ***clipTopPos và clipBottomPos*** [[và applyVirtualBoundary]].

– ***Các độ lệch mẫu được tái tạo c1 và c2 được xác định trong Bảng8-28 theo vị trí mẫu độ sáng dọc x, clipLeftPos và clipRightPos.***

– Biến curr được dẫn xuất như sau:

$$curr = recPicture[h_x, v_y] \quad (8\text{-}1253)$$

– Mảng của các hệ số lọc sắc độ f[j] và mảng của các giá trị rút gọn sắc độ c[j] được dẫn xuất như sau với j = 0..5:

$$f[j] = AlfCoeff_C[slice_alf_aps_id_chroma][j] \quad (8\text{-}1254)$$

$$c[j] = AlfClip_C[slice_alf_aps_id_chroma][j] \quad (8\text{-}1255)$$

– Biến sum được dẫn xuất như sau:

$$sum = f[0] * (Clip3(-c[0], c[0], recPicture[h_x, v_{y+r2}] - curr) + Clip3(-c[0], c[0], recPicture[h_x, v_{y-r2}] - curr)) +$$

f[1] * (Clip3(−c[1], c[1], recPicture[h_{x+c1}, v_{y+r1}] − curr) +
Clip3(−c[1], c[1], recPicture[h_{x-c1}, v_{y-r1}] − curr)) +
f[2] * (Clip3(−c[2], c[2], recPicture[h_x, v_{y+r1}] − curr) +
Clip3(−c[2], c[2], recPicture[h_x, v_{y-r1}] − curr)) + (8-1256)
f[3] * (Clip3(−c[3], c[3], recPicture[h_{x-c1}, v_{y+r1}] − curr) +
Clip3(−c[3], c[3], recPicture[h_{x+c1}, v_{y-r1}] − curr)) +
f[4] * (Clip3(−c[4], c[4], recPicture[h_{x+c2}, v_y] − curr) +
Clip3(−c[4], c[4], recPicture[h_{x-c2}, v_y] − curr)) +
f[5] * (Clip3(−c[5], c[5], recPicture[h_{x+c1}, v_y] − curr) +
Clip3(−c[5], c[5], recPicture[h_{x-c1}, v_y] − curr))
sum = curr + (sum + 64) >> 7) (8-1257)

– Mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture[xCtbC + x][yCtbC + y] được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[(xCtbC + x) * SubWidthC][(yCtbC + y) * SubHeightC] đều bằng 1, phương trình sau áp dụng:

alfPicture[xCtbC + x][yCtbC + y] = $recPicture_L[h_x, v_y]$ (8-1258)

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 or pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

alfPicture[xCtbC + x][yCtbC + y] = Clip3(0, (1 << $BitDepth_C$) − 1, sum) (8-1259)

Bảng 8-27 – Mô tả r1 và r2 theo vị trí mẫu độ sáng ngang y, *clipTopPos và clipBottomPos* [[và applyVirtualBoundary]]

[[điều kiện	**r1**	**r2**
(y = = ctbHeightC − 2 || y = = ctbHeightC − 3) && (applyVirtualBoundary = = 1)	0	0
(y = = ctbHeightC − 1 || y = = ctbHeightC − 4) && (applyVirtualBoundary = = 1)	1	1
ngược lại	1	2]]

Điều kiên	r1	r2
(y = = clipBottomPos − 1 || y = = clipTopPos)	0	0
(y == clipBottomPos − 2 || y == clipTopPos + 1) && (clipBottomPos != clipTopPos + 2)	1	1
(y = = clipTopPos + 1) && (clipBottomPos = = clipTopPos + 2)	0	0
ngược lai	1	2

Bảng 8-28 – Mô tả c1 và c2 theo vị trí mẫu sáng dọc x, clipLeftPos và clipRightPos

Điều kiên	c1	c2
(xCtbC + x == clipLeftPos || xCtbC + x == clipRightPos − 1)	0	0
(xCtbC + x == clipLeftPos + 1 || xCtbC + x == clipRightPos − 2)	1	1
ngược lai	1	2

8.8.5.5 Quá trình dẫn xuất vị trí biên ALF cho các mẫu

Các đầu vào của quá trình này là:

- *vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.*
- *vị trí độ sáng (x, y) xác định mẫu hiện tại so với mẫu trên bên trái của CTB độ sáng hiện tại.*

Đầu ra của quá trình này là:

- *vị trí biên dọc trái clipLeftPos.*
- *vị trí biên dọc phải clipRightPos.*
- *vị trí biên ngang trên clipTopPos.*
- *vị trí biên ngang dưới clipBottomPos.*

Biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được gán bằng −128.

Biến clipTopPos được cải biến như sau:

- *Nếu đường biên dưới của CTB hiện tại không phải là đường biên dưới của ảnh và y – (CtbSizeY – 4) lớn hơn hoặc bằng 0, biến clipTopPos được gán bằng yCtb + CtbSizeY – 4.*

– *Ngược lại, nếu pps loop filter across virtual boundaries disabled flag bằng 1, PpsVirtualBoundariesPosY[n] % CtbSizeY bằng 0, PpsVirtualBoundariesPosY[n] không bằng pic height in luma samples – 1 hoặc 0, và yCtb + y – PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps num hor virtual boundaries – 1, phương trình sau áp dụng:*

clipTopPos = PpsVirtualBoundariesPosY[n] (8-1201)

– *Ngược lại, nếu y nhỏ hơn 3, và đường biên trên của CTB hiện tại không phải là đường biên trên của ảnh, và một điều kiện sau là đúng, biến clipTopPos được gán bằng yCtb:*

– *Nếu đường biên trên của CTB hiện tại là đường biên trên của khối viên, loop filter across bricks enabled flag bằng 0.*

– *Nếu đường biên trên của CTB hiện tại là đường biên trên của lát, loop filter across slices enabled flag bằng 0.*

– *Nếu đường biên trên của CTB hiện tại là đường biên trên của ảnh con loop filter across sub pic enabled flag bằng 0.*

Biến clipBottomPos được cải biến như sau:

– *Nếu đường biên dưới của CTB hiện tại không phải là đường biên dưới của ảnh và CtbSizeY – 4 – y lớn hơn 0 và nhỏ hơn 4, biến clipBottomPos được gán bằng yCtb + CtbSizeY – 4.*

– *Ngược lại, nếu pps loop filter across virtual boundaries disabled flag bằng 1, PpsVirtualBoundariesPosY[n] % CtbSizeY bằng 0, PpsVirtualBoundariesPosY[n] không bằng pic height in luma samples – 1 hoặc 0, và PpsVirtualBoundariesPosY[n] – yCtb – y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps num hor virtual boundaries – 1, phương trình sau áp dụng:*

clipBottomPos = PpsVirtualBoundariesPosY[n] (8-1201)

– *Ngược lại, nếu CtbSizeY – y nhỏ hơn 4, và đường biên dưới của CTB hiện tại không phải là đường biên dưới của ảnh, và một điều kiện sau là đúng, biến clipBottomPos được gán bằng yCtb + CtbSizeY.*

– *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của khối viên, và loop_filter_across_bricks_enabled_flag bằng 0.*

– *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của lát, và loop_filter_across_slices_enabled_flag bằng 0.*

– *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của ảnh con, và loop_filter_across_sub_pic_enabled_flag bằng 0.*

Biến clipLeftPos được cải biến như sau:

– *Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, PpsVirtualBoundariesPosX[n] % CtbSizeY bằng 0, PpsVirtualBoundariesPosX[n] không bằng pic_width_in_luma_samples – 1 hoặc 0, và xCtb + x – PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries – 1, phương trình sau áp dụng:*

clipLeftPos = PpsVirtualBoundariesPosX[n] (8-1201)

– *Ngược lại, nếu x nhỏ hơn 3, đường biên trái của CTB hiện tại không phải là đường biên trái của ảnh và một điều kiện sau là đúng, biến clipLeftPos được gán bằng xCtb*

– *Nếu đường biên trái của CTB hiện tại là đường biên trái của khối viên, và loop_filter_across_bricks_enabled_flag bằng 0.*

– *Nếu đường biên trái của CTB hiện tại là đường biên trái của lát, và loop_filter_across_slices_enabled_flag bằng 0.*

– *Nếu đường biên trái của CTB hiện tại là đường biên trái của ảnh con, và loop_filter_across_sub_pic_enabled_flag bằng 0.*

Biến clipRightPos được cải biến như sau:

– *Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, PpsVirtualBoundariesPosX[n] % CtbSizeY bằng 0, PpsVirtualBoundariesPosX[n] không bằng*

pic_width_in_luma_samples – 1 hoặc 0, và PpsVirtualBoundariesPosX[n]– xCtb – x lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries – 1, phương trình sau áp dụng:

clipRightPos = PpsVirtualBoundariesPosX[n] (8-1201)

- *Ngược lại, nếu CtbSizeY – x nhỏ hơn 4, và đường biên phải của CTB hiện tại không phải là đường biên phải của ảnh, và one điều kiện sau là đúng, biến clipRightPos được gán bằng xCtb + CtbSizeY*
  - *Nếu đường biên phải của CTB hiện tại là đường biên phải của khối viên, và loop_filter_across_bricks_enabled_flag bằng 0.*
  - *Nếu đường biên phải của CTB hiện tại là đường biên phải của lát, và loop_filter_across_slices_enabled_flag bằng 0.*
  - *Nếu đường biên phải của CTB hiện tại là đường biên phải của ảnh con, và loop_filter_across_sub_pic_enabled_flag bằng 0.*

Giá trị cụ thể -128 được sử dụng theo phương án thực hiện nêu trên có thể được thay bằng các giá trị khác, chẳng hạn -K trong đó chẳng hạn, K lớn hơn hoặc không nhỏ hơn số lượng dòng được dịch chuyển từ đường biên dưới CTU (chẳng hạn, K = -5).

Theo cách khác, kiểm tra có điều kiện của "PpsVirtualBoundariesPosY[n] không bằng pic_height_in_luma_samples – 1 hoặc 0" có thể còn được loại bỏ dựa trên việc PpsVirtualBoundariesPosY[n] nằm trong khoảng từ 1 đến Ceil(pic_height_in_luma_samples ÷ 8) – 1 bao gồm hai điểm mút.

Theo cách khác, một cờ có thể được sử dụng để đánh dấu liệu mỗi mẫu cần được xử lý theo cách khác nếu nó ở được đặt ở các đường biên đơn vị viđeo.

5.9 Phương án thực hiện #9

Theo phương án thực hiện, các ý tưởng chính sau đây được áp dụng:

Khi kích hoạt các đường biên ảo ALF:

- Đối với các CTU không được đặt trong hàng CTU cuối cùng trong ảnh (chẳng hạn, đường biên dưới của các CTU không phải là đường biên dưới của ảnh hoặc vượt quá đường biên dưới của ảnh), đường biên ảo ALF được kích hoạt, tức là, một CTU có thể được tách thành hai hoặc nhiều phần, và các mẫu

trong một phần không được phép sử dụng các mẫu trong phần khác.

- Đối với các CTU được đặt trong hàng CTU cuối cùng trong ảnh (chẳng hạn, đường biên dưới của các CTU là đường biên dưới của ảnh hoặc vượt quá đường biên dưới của ảnh), đường biên ảo ALF được kích hoạt, tức là, một CTU có thể được tách thành hai hoặc nhiều phần, và các mẫu trong một phần không được phép sử dụng các mẫu trong phần khác.

Khi đệm các đường biên (bao gồm các đường biên ảo ALF, các đường biên lát/ô/khối viên/ảnh con, "các đường biên ảo 360") trong quá trình phân loại:

Đối với mẫu ở một (hoặc nhiều loại) đường biên, khi các mẫu lân cận giao với đường biên không được phép được sử dụng, đệm 1 phía được thực hiện để đệm các mẫu lân cận này.

Khi đệm các đường biên (bao gồm các đường biên ảo ALF, các đường biên lát/ô/khối viên/ảnh con, "các đường biên ảo 360") trong quá trình lọc ALF:

- Đối với mẫu ở một (hoặc nhiều loại) đường biên vốn là lát/ô/khối viên/đường biên ảnh con hoặc "đường biên ảo 360" trùng với đường biên CTU, khi các mẫu lân cận giao với đường biên không được phép được sử dụng, đệm 2 phía được thực hiện để đệm các mẫu lân cận này.

- Đối với mẫu ở một (hoặc nhiều loại) đường biên vốn là đường biên ảnh hoặc "đường biên ảo 360" không trùng với đường biên CTU, khi các mẫu lân cận giao với đường biên không được phép được sử dụng, đệm 1 phía được thực hiện để đệm các mẫu lân cận này.

8.8.5.2 Quá trình lọc CTB đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ trước quá trình ALF,

– mảng mẫu ảnh độ sáng được tái tạo được lọc $alfPicture_L$,

– vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa $alfPicture_L$.

Mệnh đề của quá trình dẫn xuất chỉ số lọc 8.8.5.3 được gọi với vị trí (xCtb, yCtb) và mảng mẫu ảnh độ sáng được tái tạo $recPicture_L$ làm các đầu vào, và filtIdx[x][y] và transposeIdx[x][y] với x, y = 0..CtbSizeY − 1 làm đầu ra.

Để dẫn xuất các mẫu độ sáng được tái tạo được lọc $alfPicture_L$[x][y], mỗi mẫu độ sáng được tái tạo bên trong CTB độ sáng hiện tại $recPicture_L$[x][y] được lọc như sau với x, y = 0..CtbSizeY − 1:

– Mảng của các hệ số lọc độ sáng f[j] và mảng của các giá trị rút gọn độ sáng c[j] tương ứng với bộ lọc được xác định bởi filtIdx[x][y] được dẫn xuất như sau với j = 0..11:

– Nếu AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] nhỏ hơn 16, phương trình sau áp dụng:

i = AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] (8-1187)

f[j] = AlfFixFiltCoeff[AlfClassToFiltMap[i][filtIdx[x][y]]][j] (8-1188)

$c[j] = 2^{BitdepthY}$ (8-1189)

– Ngược lại (AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] lớn hơn hoặc bằng 16, phương trình sau áp dụng:

i = slice_alf_aps_id_luma[AlfCtbFiltSetIdxY[xCtb >> Log2CtbSize][yCtb >> Log2CtbSize] − 6] (8-1190)

$f[j] = AlfCoeff_L[i][filtIdx[x][y]][j]$ (8-1191)

$c[j] = AlfClip_L[i][filtIdx[x][y]][j]$ (8-1192)

– Các hệ số lọc độ sáng và chỉ số của các giá trị rút gọn idx được dẫn xuất tùy thuộc vào transposeIdx[x][y] như sau:

– Nếu transposeIndex[x][y] bằng 1, phương trình sau áp dụng:

idx[] = {9, 4, 10, 8, 1, 5, 11, 7, 3, 0, 2, 6} (8-1193)

– Ngược lại, nếu transposeIndex[x][y] bằng 2, phương trình sau áp dụng:

idx[] = {0, 3, 2, 1, 8, 7, 6, 5, 4, 9, 10, 11} (8-1194)

– Ngược lại, nếu transposeIndex[x][y] bằng 3, phương trình sau áp dụng:

idx[] = {9, 8, 10, 4, 3, 7, 11, 5, 1, 0, 2, 6} (8-1195)

– Ngược lại, phương trình sau áp dụng:

idx[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11} (8-1196)

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −3..3 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i) (8-1197)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosX[n]− xCtb − x lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i) (8-1198)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1199)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

h_{x+i} = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, h_{x+i}) (8-1184)]]

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j) (8-1200)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1201)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1202)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

v_{y+j} = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, v_{y+j}) (8-1184)

– Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh con và loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.

– Ngược lại, applyVirtualBoundary được gán bằng 1.]]

– ***Các biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên ALF như được nêu trong mệnh đề 8.8.5.5 với (xCtb, yCtb) và (x, y) làm các đầu vào.***

– Các độ lệch mẫu được tái tạo r1, r2 và r3 được xác định trong Bảng 8-24 theo ***vị trí mẫu sáng dọc*** ~~[[ngang]]~~ y, ***clipTopPos và clipBottomPos*** ~~[[applyVirtualBoundary]]~~.

– ***Các độ lệch vị trí mẫu c1, c2 và c3 được xác định trong Bảng 8-25 theo vị trí mẫu độ sáng ngang x, clipLeftPos và clipRightPos.***

– Biến curr được dẫn xuất như sau:

$$curr = recPicture_L[h_x, v_y] \quad (8\text{-}1203)$$

– Biến sum được dẫn xuất như sau:

– $sum = f[idx[0]] * (Clip3(-c[idx[0]], c[idx[0]], recPicture_L[h_x, v_{y+r3}] - curr) +$
$Clip3(-c[idx[0]], c[idx[0]], recPicture_L[h_x, v_{y-r3}] - curr)) +$
$f[idx[1]] * (Clip3(-c[idx[1]], c[idx[1]], recPicture_L[h_{x+c1}, v_{y+r2}] - curr) +$
$Clip3(-c[idx[1]], c[idx[1]], recPicture_L[h_{x-c1}, v_{y-r2}] - curr)) +$
$f[idx[2]] * (Clip3(-c[idx[2]], c[idx[2]], recPicture_L[h_x, v_{y+r2}] - curr) +$
$Clip3(-c[idx[2]], c[idx[2]], recPicture_L[h_x, v_{y-r2}] - curr)) +$
$f[idx[3]] * (Clip3(-c[idx[3]], c[idx[3]], recPicture_L[h_{x-c1}, v_{y+r2}] - curr) +$
$Clip3(-c[idx[3]], c[idx[3]], recPicture_L[h_{x+c1}, v_{y-r2}] - curr)) +$
$f[idx[4]] * (Clip3(-c[idx[4]], c[idx[4]], recPicture_L[h_{x+c2}, v_{y+r1}] - curr) +$
$Clip3(-c[idx[4]], c[idx[4]], recPicture_L[h_{x-c2}, v_{y-r1}] - curr)) +$
$f[idx[5]] * (Clip3(-c[idx[5]], c[idx[5]], recPicture_L[h_{x+c1}, v_{y+r1}] - curr) +$
$Clip3(-c[idx[5]], c[idx[5]], recPicture_L[h_{x-c1}, v_{y-r1}] - curr)) +$
$f[idx[6]] * (Clip3(-c[idx[6]], c[idx[6]], recPicture_L[h_x, v_{y+r1}] - curr) +$
$Clip3(-c[idx[6]], c[idx[6]], recPicture_L[h_x, v_{y-r1}] - curr)) +$ (8-1204)
$f[idx[7]] * (Clip3(-c[idx[7]], c[idx[7]], recPicture_L[h_{x-c1}, v_{y+r1}] - curr) +$
$Clip3(-c[idx[7]], c[idx[7]], recPicture_L[h_{x+c1}, v_{y-r1}] - curr)) +$
$f[idx[8]] * (Clip3(-c[idx[8]], c[idx[8]], recPicture_L[h_{x-c2}, v_{y+r1}] - curr) +$
$Clip3(-c[idx[8]], c[idx[8]], recPicture_L[h_{x+c2}, v_{y-r1}] - curr)) +$
$f[idx[9]] * (Clip3(-c[idx[9]], c[idx[9]], recPicture_L[h_{x+c3}, v_y] - curr) +$
$Clip3(-c[idx[9]], c[idx[9]], recPicture_L[h_{x-c3}, v_y] - curr)) +$
$f[idx[10]] * (Clip3(-c[idx[10]], c[idx[10]], recPicture_L[h_{x+c2}, v_y] - curr) +$
$Clip3(-c[idx[10]], c[idx[10]], recPicture_L[h_{x-c2}, v_y] - curr)) +$
$f[idx[11]] * (Clip3(-c[idx[11]], c[idx[11]], recPicture_L[h_{x+c1}, v_y] - curr) +$
$Clip3(-c[idx[11]], c[idx[11]], recPicture_L[h_{x-c1}, v_y] - curr))$

$$sum = curr + ((sum + 64) >> 7) \quad (8\text{-}1205)$$

– Mẫu ảnh độ sáng được tái tạo được lọc được chỉnh sửa $alfPicture_L[xCtb + x][yCtb + y]$ được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[xCtb+ x][yCtb + y] đều bằng 1, phương trình sau áp dụng:

alfPicture$_L$[xCtb + x][yCtb + y] = recPicture$_L$[h_x, v_y] (8-1206)

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 or pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

alfPicture$_L$[xCtb + x][yCtb + y] = Clip3(0, (1 << BitDepth$_Y$) − 1, sum)(8-1207)

Bảng 8-24 – Mô tả r1, r2, và r3 theo *vị trí mẫu sáng dọc* [[ngang]] y, *clipTopPos và clipBottomPos* [[và applyVirtualBoundary]]

[[Điều kiện	r1	r2	r3
(y == CtbSizeY − 5 || y == CtbSizeY − 4) && (applyVirtualBoundary == 1)	0	0	0
(y == CtbSizeY − 6 || y == CtbSizeY − 3) && (applyVirtualBoundary == 1)	1	1	1
(y == CtbSizeY − 7 || y == CtbSizeY − 2) && (applyVirtualBoundary == 1)	1	2	2
ngược lại	1	2	3]]

Điều kiện	*r1*	*r2*	*r3*
(yCtb + y == clipBottomPos − 1 || yCtb + y == clipTopPos)	*0*	*0*	*0*
(yCtb + y == clipBottomPos − 2 || yCtb + y == clipTopPos + 1)	*1*	*1*	*1*
(yCtb + y == clipBottomPos − 3 || yCtb + y == clipTopPos + 2) && (clipBottomPos != clipTopPos + 4)	*1*	*2*	*2*
(yCtb + y == clipTopPos + 2) && (clipBottomPos == clipTopPos + 4)	*1*	*1*	*1*
Ngược lại	*1*	*2*	*3*

Bảng 8-25 – Mô tả c1, c2, và c3 theo vị trí mẫu độ sáng ngang x, clipLeftPos và clipRightPos

Điều kiện	*c1*	*c2*	*c3*
(xCtb + x == clipLeftPos || xCtb + x == clipRightPos − 1)	*0*	*0*	*0*
(xCtb + x == clipLeftPos +1 || xCtb + x == clipRightPos − 2)	*1*	*1*	*1*
(xCtb + x == clipLeftPos +2 || xCtb + x == clipRightPos − 3)	*1*	*2*	*2*
Ngược lại	*1*	*2*	*3*

8.8.5.3 Quá trình dẫn xuất đối với chỉ số lọc và chuyển vị ALF đối với các mẫu độ sáng

Các đầu vào của quá trình này là:

– vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại,

– mảng mẫu ảnh độ sáng được tái tạo recPicture$_L$ trước quá trình ALF.

Các đầu ra của quá trình này là

– mảng chỉ số lọc phân loại filtIdx[x][y] với x, y = 0..CtbSizeY − 1,

– mảng chỉ số chuyển vị transposeIdx[x][y] với x, y = 0..CtbSizeY − 1.

Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng có i, j = −2..5 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n], pic_width_in_luma_samples − 1, xCtb + x + i) (8-1208)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosX[n] − xCtb − x lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n] − 1, xCtb + x + i) (8-1209)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x + i) (8-1210)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

h_{x+i} = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, h_{x+i}) (8-1184)]]

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n], pic_height_in_luma_samples − 1, yCtb + y + j) (8-1211)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 6 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n] − 1, yCtb + y + j) (8-1212)

– Ngược lại, phương trình sau áp dụng:

– [[Nếu yCtb + CtbSizeY lớn hơn hoặc bằng pic_height_in_luma_samples, phương trình sau áp dụng:]]

v_{y+j} = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1213)

– [[Ngược lại, nếu y nhỏ hơn CtbSizeY − 4, phương trình sau áp dụng:

v_{y+j} = Clip3(0, yCtb + CtbSizeY − 5, yCtb + y + j) (8-1214)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(yCtb + CtbSizeY − 4, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1215)

– Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

v_{y+j} = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, v_{y+j}) (8-1184)]]

– ***Các biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí đường biên ALF với (xCtb, yCtb) và (x, y).***

– ***Nếu clipTopPos không bằng −128, phương trình sau áp dụng:***

v_{y+j} = Clip3(clipTopPos, pic_height_in_luma_samples − 1, yCtb + y + j) (8-1213)

– ***Nếu clipBottomPos không bằng −128, phương trình sau áp dụng:***

v_{y+j} = Clip3(0, clipBottomPos − 1, yCtb + y + j) (8-1213)

– ***Nếu clipLeftPos không bằng −128, phương trình sau áp dụng:***

h_{x+i} = Clip3(clipLeftPos,

pic_width_in_luma_samples − 1, xCtb + x + i) (8-1213)

– ***Nếu clipRightPos không bằng −128, phương trình sau áp dụng:***

h_{x+i} = Clip3(0, clipRightPos − 1, xCtb + x + i) (8-1213)

Mảng chỉ số lọc phân loại filtIdx và mảng chỉ số chuyển vị transposeIdx được dẫn xuất bằng các bước có thứ tự sau:

1. Các biến filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] với x, y = − 2..CtbSizeY + 1 được dẫn xuất như sau:
   – Nếu cả x và y là các số chẵn hoặc cả x lẫn y là các số không chẵn, phương trình sau áp dụng:

$$filtH[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x-1}, v_y] - recPicture[h_{x+1}, v_y]) \quad (8\text{-}1216)$$

$$filtV[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_x, v_{y-1}] - recPicture[h_x, v_{y+1}]) \quad (8\text{-}1217)$$

$$filtD0[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x-1}, v_{y-1}] - recPicture[h_{x+1}, v_{y+1}]) \quad (8\text{-}1218)$$

$$filtD1[x][y] = Abs((recPicture[h_x, v_y] << 1) - recPicture[h_{x+1}, v_{y-1}] - recPicture[h_{x-1}, v_{y+1}]) \quad (8\text{-}1219)$$

   – Ngược lại, filtH[x][y], filtV[x][y], filtD0[x][y] và filtD1[x][y] được gán bằng 0.

2. [[Các biến minY, maxY và ac được dẫn xuất như sau:

– Nếu (y << 2) bằng (CtbSizeY − 8) và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples − 1, minY được gán bằng −2, maxY được gán bằng 3 và ac được gán bằng 96.

– Ngược lại, nếu (y << 2) bằng (CtbSizeY − 4) và (yCtb + CtbSizeY) nhỏ hơn pic_height_in_luma_samples − 1, minY được gán bằng 0, maxY được gán bằng 5 và ac được gán bằng 96.]]

3. ***Các biến sumH[x][y], sumV[x][y], sumD0[x][y], sumD1[x][y] và sumOfHV[x][y] with x, y = 0..(CtbSizeY − 1) >> 2 được dẫn xuất như sau:***

   – ***Các biến minY và maxY được dẫn xuất như sau:***

- *Nếu clipTopPos không bằng −128 và clipBottomPos bằng −128, phương trình sau áp dụng:*
  - *Nếu (y << 2) bằng clipTopPos, minY được gán bằng 0, maxY được gán bằng 5.*
- *Ngược lại, nếu clipTopPos bằng −128 và clipBottomPos không bằng −128, phương trình sau áp dụng:*
  - *Nếu (y << 2) bằng (clipBottomPos − 4), minY được gán bằng −2, maxY được gán bằng 3.*
- *Ngược lại, nếu clipTopPos không bằng −128 và clipBottomPos không bằng −128, phương trình sau áp dụng:*
  - *minY được gán bằng 0, maxY được gán bằng 3.*
- *Ngược lại, minY được gán bằng −2 and maxY được gán bằng 5 và ac được gán bằng 64.*

- *Các biến minX, maxX và ac được dẫn xuất như sau:*
  - *Nếu clipLeftPos không bằng −128, phương trình sau áp dụng:*
    - *Nếu (x << 2) bằng clipLeftPos, minX được gán bằng 0, maxX được gán bằng 5.*
  - *Ngược lại, nếu clipRightPos không bằng −128, phương trình sau áp dụng:*
    - *Nếu (x << 2) bằng (clipRightPos – 4), minX được gán bằng −2, maxX được gán bằng 3.*
  - *Ngược lại, minX được gán bằng −2 và maxX được gán bằng 5.*
- *Biến ac được dẫn xuất như sau:*

Table 8-26 – Specification of ac theo minX, minY, maxX, maxY

*(maxY – minY + 1) * (maxX – minX + 1)*	*8*8*	*8*6*	*8*4*	*6*6*	*6*4*
Ac	*64*	*96*	*128*	*112*	*192*

- [[Các biến sumH[x][y], sumV[x][y], sumD0[x][y], sumD1[x][y] và sumOfHV[x][y] với x, y = 0..(CtbSizeY − 1) >> 2 được dẫn xuất như sau:]]

$$sumH[x][y] = \Sigma_i\Sigma_j\ filtH[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb] \text{ với } i = minX..maxX [[-2..5]], j = minY..maxY \quad (8\text{-}1220)$$

$sumV[x][y] = \Sigma_i\Sigma_j\ filtV[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = \underline{\boldsymbol{minX..maxX}}$ [[−2..5]], j = minY..maxY (8-1221)

$sumD0[x][y] = \Sigma_i\Sigma_j\ filtD0[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = \underline{\boldsymbol{minX..maxX}}$ [[−2..5]], j = minY..maxY (8-1222)

$sumD1[x][y] = \Sigma_i\Sigma_j\ filtD1[h_{(x << 2)+i} - xCtb][v_{(y << 2)+j} - yCtb]$ với $i = \underline{\boldsymbol{minX..maxX}}$ [[−2..5]], j = minY..maxY (8-1223)

sumOfHV[x][y] = sumH[x][y] + sumV[x][y](8-1224)

4. Các biến dir1[x][y], dir2[x][y] và dirS[x][y] với x, y = 0..CtbSizeY − 1 được dẫn xuất như sau:
   - Các biến hv1, hv0 và dirHV được dẫn xuất như sau:
     - Nếu sumV[x >> 2][y >> 2] lớn hơn sumH[x >> 2][y >> 2], phương trình sau áp dụng:

       hv1 = sumV[x >> 2][y >> 2] (8-1225)

       hv0 = sumH[x >> 2][y >> 2] (8-1226)

       dirHV = 1 (8-1227)
     - Ngược lại, phương trình sau áp dụng:

       hv1 = sumH[x >> 2][y >> 2] (8-1228)

       hv0 = sumV[x >> 2][y >> 2] (8-1229)

       dirHV = 3 (8-1230)
   - Các biến d1, d0 và dirD được dẫn xuất như sau:
     - Nếu sumD0[x >> 2][y >> 2] lớn hơn sumD1[x >> 2][y >> 2], phương trình sau áp dụng:

       d1 = sumD0[x >> 2][y >> 2] (8-1231)

       d0 = sumD1[x >> 2][y >> 2] (8-1232)

       dirD = 0 (8-1233)
     - Ngược lại, phương trình sau áp dụng:

d1 = sumD1[x >> 2][y >> 2]

(8-1234)

d0 = sumD0[x >> 2][y >> 2]

(8-1235)

dirD = 2 (8-1236)

– Các biến hvd1, hvd0, được dẫn xuất như sau:

hvd1 = (d1 * hv0 > hv1 * d0) ? d1 : hv1 (8-1237)

hvd0 = (d1 * hv0 > hv1 * d0) ? d0 : hv0 (8-1238)

– Các biến dirS[x][y], dir1[x][y] và dir2[x][y] được dẫn xuất như sau:

dir1[x][y] = (d1 * hv0 > hv1 * d0) ? dirD : dirHV (8-1239)

dir2[x][y] = (d1 * hv0 > hv1 * d0) ? dirHV : dirD (8-1240)

dirS[x][y] = (hvd1 > 2 * hvd0) ? 1 : ((hvd1 * 2 > 9 * hvd0) ? 2 : 0) (8-1241)

5. Biến avgVar[x][y] với x, y = 0..CtbSizeY − 1 được dẫn xuất như sau:

varTab[] = {0, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4} (8-1242)

avgVar[x][y] = varTab[Clip3(0, 15, (sumOfHV[x >> 2][y >> 2] * ac) >> (3 + BitDepth$_Y$))] (8-1243)

6. Mảng chỉ số lọc phân loại filtIdx[x][y] và mảng chỉ số chuyển vị transposeIdx[x][y] với x = y = 0..CtbSizeY − 1 được dẫn xuất như sau:

transposeTable[] = {0, 1, 0, 2, 2, 3, 1, 3}

transposeIdx[x][y] = transposeTable[dir1[x][y] * 2 + (dir2[x][y] >> 1)]

filtIdx[x][y] = avgVar[x][y]

Khi dirS[x][y] không bằng 0, filtIdx[x][y] được cải biến như sau:

filtIdx[x][y] += (((dir1[x][y] & 0x1) << 1) + dirS[x][y]) * 5 (8-1244)

8.8.5.4 Quá trình lọc CTB đối với các mẫu sắc độ

Các đầu vào của quá trình này là:

– mảng mẫu ảnh sắc độ được tái tạo recPicture trước quá trình ALF,

– mảng mẫu ảnh sắc độ được tái tạo được lọc alfPicture,

– vị trí sắc độ (xCtbC, yCtbC) xác định mẫu trên bên trái của CTB sắc độ hiện tại so với mẫu trên bên trái của ảnh hiện tại.

Đầu ra của quá trình này là mảng mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture.

Chiều rộng và chiều cao của CTB sắc độ hiện tại ctbWidthC và ctbHeightC được dẫn xuất như sau:

ctbWidthC = CtbSizeY/SubWidthC (8-1245)

ctbHeightC = CtbSizeY/SubHeightC (8-1246)

Để dẫn xuất các mẫu sắc độ được tái tạo được lọc alfPicture[x][y], mỗi mẫu sắc độ được tái tạo bên trong CTB sắc độ hiện tại recPicture[x][y] được lọc như sau với x = 0..ctbWidthC − 1, y = 0..ctbHeightC − 1:

– Các vị trí (h_{x+i}, v_{y+j}) đối với mỗi mẫu trong các mẫu sắc độ tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu sắc độ với i, j = −2..2 được dẫn xuất như sau:

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** xCtbC + x − PpsVirtualBoundariesPosX[n]/SubWidthC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(PpsVirtualBoundariesPosX[n]/SubWidthC, (8-1247)
pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosX[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosX[n]/SubWidthC − xCtbC − x lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_ver_virtual_boundaries − 1, phương trình sau áp dụng:

h_{x+i} = Clip3(0, PpsVirtualBoundariesPosX[n]/SubWidthC − 1, xCtbC + x + i) (8-1248)

– Ngược lại, phương trình sau áp dụng:

h_{x+i} = Clip3(0,
pic_width_in_luma_samples/SubWidthC − 1, xCtbC + x + i) (8-1249)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

h_{x+i} = Clip3(SubPicLeftBoundaryPos/SubWidthC, SubPicRightBoundaryPos/SubWidthC, h_{x+i}) (8-1184)]]

– Nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** yCtbC + y − PpsVirtualBoundariesPosY[n]/SubHeightC lớn hơn hoặc bằng 0 và nhỏ hơn 2 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(PpsVirtualBoundariesPosY[n]/SubHeightC , (8-1250)
pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j)

– Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, ***và PpsVirtualBoundariesPosY[n] % CtbSizeY không bằng 0, và*** PpsVirtualBoundariesPosY[n]/SubHeightC − yCtbC − y lớn hơn 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:

v_{y+j} = Clip3(0, PpsVirtualBoundariesPosY[n]/SubHeightC − 1, yCtbC + y + j) (8-1251)

– Ngược lại, phương trình sau áp dụng:

v_{y+j} = Clip3(0, pic_height_in_luma_samples/SubHeightC − 1, yCtbC + y + j) (8-1252)

– [[Khi loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0, phương trình sau áp dụng:

v_{y+j} = Clip3(SubPicTopBoundaryPos/SubWidthC, SubPicBotBoundaryPos/SubWidthC, v_{y+j}) (8-1184)

– Biến applyVirtualBoundary được dẫn xuất như sau:

– Nếu một hoặc nhiều điều kiện sau là đúng, applyVirtualBoundary được gán bằng 0:

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh.

– Đường biên dưới của CTB hiện tại là đường biên dưới của khối viên và loop_filter_across_bricks_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của lát và loop_filter_across_slices_enabled_flag bằng 0.

– Đường biên dưới của CTB hiện tại là đường biên dưới của ảnh con và loop_filter_across_sub_pic_enabled_flag[SubPicIdx] đối với ảnh con chứa mẫu độ sáng ở vị trí (h_x, v_y) bằng 0.

– Đường biên dưới của CTB hiện tại là một trong các đường biên ảo dưới của ảnh và pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1.

– Ngược lại, applyVirtualBoundary được gán bằng 1.]]

– ***Các biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được dẫn xuất bằng cách gọi quá trình dẫn xuất vị trí biên ngang đối với các mẫu độ sáng với (xCtbC * SubWidthC, yCtbC * SubHeightC) và (x * SubWidthC, y *SubHeightC) làm các đầu vào.***

– ***Biến clipLeftPos được gán bằng clipLeftPos/SubWidthC.***

– ***Biến clipRightPos được gán bằng clipRightPos/SubWidthC.***

– ***Biến clipTopPos được gán bằng clipTopPos/SubHeightC.***

– ***Biến clipBottomPos được gán bằng clipBottomPos/SubHeightC.***

– Các độ lệch mẫu được tái tạo r1 và r2 được xác định trong Bảng 8-27 theo vị trí ***mẫu sắc độ dọc*** [[~~độ sáng ngang~~]] y, ***clipTopPos và clipBottomPos*** [[~~và applyVirtualBoundary~~]].

– ***Các độ lệch vị trí mẫu c1 và c2 được xác định trong Bảng 8-28 theo vị trí mẫu sắc độ ngang x, clipLeftPos và clipRightPos.***

– Biến curr được dẫn xuất như sau:

$$curr = recPicture[h_x, v_y] \quad (8\text{-}1253)$$

– Mảng của các hệ số lọc sắc độ f[j] và mảng của các giá trị rút gọn sắc độ c[j] được dẫn xuất như sau với j = 0..5:

$$f[j] = AlfCoeff_C[slice_alf_aps_id_chroma][j] \quad (8\text{-}1254)$$

$$c[j] = AlfClip_C[slice_alf_aps_id_chroma][j] \quad (8\text{-}1255)$$

– Biến sum được dẫn xuất như sau:

$$sum = f[0] * (Clip3(-c[0], c[0], recPicture[h_x, v_{y+r2}] - curr) + Clip3(-c[0], c[0], recPicture[h_x, v_{y-r2}] - curr)) +$$

f[1] * (Clip3(−c[1], c[1], recPicture[h_{x+c1}, v_{y+r1}] − curr) +
Clip3(−c[1], c[1], recPicture[h_{x-c1}, v_{y-r1}] − curr)) +
f[2] * (Clip3(−c[2], c[2], recPicture[h_x, v_{y+r1}] − curr) +
Clip3(−c[2], c[2], recPicture[h_x, v_{y-r1}] − curr)) + (8-1256)
f[3] * (Clip3(−c[3], c[3], recPicture[h_{x-c1}, v_{y+r1}] − curr) +
Clip3(−c[3], c[3], recPicture[h_{x+c1}, v_{y-r1}] − curr)) +
f[4] * (Clip3(−c[4], c[4], recPicture[h_{x+c2}, v_y] − curr) +
Clip3(−c[4], c[4], recPicture[h_{x-c2}, v_y] − curr)) +
f[5] * (Clip3(−c[5], c[5], recPicture[h_{x+c1}, v_y] − curr) +
Clip3(−c[5], c[5], recPicture[h_{x-c1}, v_y] − curr))
sum = curr + (sum + 64) >> 7) (8-1257)

– Mẫu ảnh sắc độ được tái tạo được lọc được chỉnh sửa alfPicture[xCtbC + x][yCtbC + y] được dẫn xuất như sau:

– Nếu pcm_loop_filter_disabled_flag và pcm_flag[(xCtbC + x) * SubWidthC][(yCtbC + y) * SubHeightC] đều bằng 1, phương trình sau áp dụng:

alfPicture[xCtbC + x][yCtbC + y] = recPicture$_L$[h_x, v_y] (8-1258)

– Ngược lại (pcm_loop_filter_disabled_flag bằng 0 or pcm_flag[x][y] bằng 0), phương trình sau áp dụng:

alfPicture[xCtbC + x][yCtbC + y] = Clip3(0, (1 << BitDepth$_C$) − 1, sum) (8-1259)

Bảng 8-27 – Mô tả r1 và r2 theo *vị trí mẫu sắc độ dọc* [[độ sáng ngang]] y_2 *clipTopPos và clipBottomPos* [[và applyVirtualBoundary]]

[[điều kiện	**r1**	**r2**
(y = = ctbHeightC − 2 || y = = ctbHeightC − 3) && (applyVirtualBoundary = = 1)	0	0
(y = = ctbHeightC − 1 || y = = ctbHeightC − 4) && (applyVirtualBoundary = = 1)	1	1
ngược lại	1	2]]

Điều kiện	***r1***	***r2***
(yCtbC + y = = clipBottomPos − 1 || yCtbC + y = = clipTopPos)	*0*	*0*
(yCtbC + y = = clipBottomPos − 2 || yCtbC + y = = clipTopPos + 1) && (clipBottomPos != clipTopPos + 2)	*1*	*1*
(yCtbC + y = = clipTopPos + 1) && (clipBottomPos = = clipTopPos + 2)	*0*	*0*
Ngược lại	*1*	*2*

Bảng 8-28 – Mô tả c1 và c2 theo vị trí mẫu sắc độ ngang x, clipLeftPos và clipRightPos

Điều kiện	c1	c2
(xCtbC + x == clipLeftPos || xCtbC + x == clipRightPos – 1)	0	0
(xCtbC + x == clipLeftPos + 1 || xCtbC + x == clipRightPos – 2)	1	1
Ngược lại	1	2

8.8.5.5 Quá trình dẫn xuất vị trí đường biên ALF

Các đầu vào của quá trình này là:

- *vị trí độ sáng (xCtb, yCtb) xác định mẫu trên bên trái của CTB độ sáng hiện tại so với mẫu trên bên trái của ảnh hiện tại.*
- *vị trí độ sáng (x, y) xác định mẫu hiện tại so với mẫu trên bên trái của CTB độ sáng hiện tại.*

Đầu ra của quá trình này là:

- *vị trí biên dọc trái clipLeftPos.*
- *vị trí biên dọc phải clipRightPos.*
- *vị trí biên ngang trên clipTopPos.*
- *vị trí biên ngang dưới clipBottomPos.*

Các biến clipLeftPos, clipRightPos, clipTopPos và clipBottomPos được gán bằng −128.

Biến clipTopPos được cải biến như sau:

- *Nếu đường biên dưới của CTB hiện tại không phải là đường biên dưới của ảnh và y – (CtbSizeY – 4) lớn hơn hoặc bằng 0, biến clipTopPos được gán bằng yCtb + CtbSizeY – 4.*
- *Ngược lại, nếu pps loop filter across virtual boundaries disabled flag bằng 1, và PpsVirtualBoundariesPosY[n] % CtbSizeY bằng 0, và yCtb + y − PpsVirtualBoundariesPosY[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps num hor virtual boundaries − 1, phương trình sau áp dụng:*

  clipTopPos = PpsVirtualBoundariesPosY[n] (8-1201)

– *Ngược lại, nếu y nhỏ hơn 3, và đường biên trên của CTB hiện tại không phải là đường biên trên của ảnh, và một hoặc nhiều điều kiện sau là đúng, biến clipTopPos được gán bằng yCtb:*
  – *Nếu đường biên trên của CTB hiện tại là đường biên trên của khối viên, và loop_filter_across_bricks_enabled_flag bằng 0.*
  – *Nếu đường biên trên của CTB hiện tại là đường biên trên của lát, và loop_filter_across_slices_enabled_flag bằng 0.*
  – *Nếu đường biên trên của CTB hiện tại là đường biên trên của the ảnh con, và loop_filter_across_sub_pic_enabled_flag bằng 0.*

Biến clipBottomPos được cải biến như sau:

– *Nếu đường biên dưới của CTB hiện tại không phải là đường biên dưới của ảnh và CtbSizeY – 4 – y lớn hơn 0 và nhỏ hơn 4, biến clipBottomPos được gán bằng yCtb + CtbSizeY – 4.*
– *Ngược lại, nếu pps_loop_filter_across_virtual_boundaries_disabled_flag bằng 1, PpsVirtualBoundariesPosY[n] % CtbSizeY bằng 0, và PpsVirtualBoundariesPosY[n] − yCtb − y lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps_num_hor_virtual_boundaries − 1, phương trình sau áp dụng:*

$$clipBottomPos = PpsVirtualBoundariesPosY[n] \quad (8\text{-}1201)$$

– *Ngược lại, nếu CtbSizeY – y nhỏ hơn 4, và đường biên dưới của CTB hiện tại không phải là đường biên dưới của ảnh, và một hoặc nhiều điều kiện sau là đúng, biến clipBottomPos được gán bằng yCtb + CtbSizeY.*
  – *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của khối viên, và loop_filter_across_bricks_enabled_flag bằng 0.*
  – *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của lát, và loop_filter_across_slices_enabled_flag bằng 0.*
  – *Nếu đường biên dưới của CTB hiện tại là đường biên dưới của ảnh con, và loop_filter_across_sub_pic_enabled_flag bằng 0.*

Biến clipLeftPos được cải biến như sau:

– *Nếu pps loop filter across virtual boundaries disabled flag bằng 1, và PpsVirtualBoundariesPosX[n] % CtbSizeY bằng 0, và xCtb + x − PpsVirtualBoundariesPosX[n] lớn hơn hoặc bằng 0 và nhỏ hơn 3 đối với n bất kỳ = 0..pps num ver virtual boundaries − 1, phương trình sau áp dụng:*

clipLeftPos = PpsVirtualBoundariesPosX[n] (8-1201)

– *Ngược lại, nếu x nhỏ hơn 3, đường biên trái của CTB hiện tại không phải là đường biên trái của ảnh và một hoặc nhiều điều kiện sau là đúng, biến clipLeftPos được gán bằng xCtb*

– *Nếu đường biên trái của CTB hiện tại là đường biên trái của khối viên, và loop filter across bricks enabled flag bằng 0.*

– *Nếu đường biên trái của CTB hiện tại là đường biên trái của lát, và loop filter across slices enabled flag bằng 0.*

– *Nếu đường biên trái của CTB hiện tại là đường biên trái của ảnh con, và loop filter across sub pic enabled flag bằng 0.*

Biến clipRightPos được cải biến như sau:

– *Nếu pps loop filter across virtual boundaries disabled flag bằng 1, và PpsVirtualBoundariesPosX[n] % CtbSizeY bằng 0, và PpsVirtualBoundariesPosX[n]− xCtb − x lớn hơn 0 và nhỏ hơn 4 đối với n bất kỳ = 0..pps num ver virtual boundaries − 1, phương trình sau áp dụng:*

clipRightPos = PpsVirtualBoundariesPosX[n] (8-1201)

– *Ngược lại, nếu CtbSizeY – x nhỏ hơn 4, và đường biên phải của CTB hiện tại không phải là đường biên phải của ảnh, và một hoặc nhiều điều kiện sau là đúng, biến clipRightPos được gán bằng xCtb + CtbSizeY*

– *Nếu đường biên phải của CTB hiện tại là đường biên phải của khối viên, và loop filter across bricks enabled flag bằng 0.*

– *Nếu đường biên phải của CTB hiện tại là đường biên phải của lát, và loop filter across slices enabled flag bằng 0.*

– *Nếu đường biên phải của CTB hiện tại là đường biên phải của ảnh con, và loop filter across sub pic enabled flag bằng 0.*

Giá trị cụ thể -128 được sử dụng theo phương án thực hiện nêu trên có thể được thay thế bằng các giá trị khác, chẳng hạn –K, trong đó, chẳng hạn, K lớn hơn hoặc không nhỏ hơn số lượng dòng được dịch từ đường biên dưới CTU (chẳng hạn, K = -5).

Theo cách khác, một cờ có thể được sử dụng để đánh dấu liệu mỗi mẫu cần được xử lý theo cách khác nếu nó ở được đặt ở các đường biên của đơn vị viđeo.

Fig.22 là sơ đồ khối của thiết bị xử lý viđeo 2200. Thiết bị 2200 có thể được sử dụng để thực hiện một hoặc nhiều phương pháp nêu trên. Thiết bị 2200 có thể được triển khai trong điện thoại thông minh, máy tính bảng, máy tính, bộ thu Internet vạn vật (IoT), và v.v.. Thiết bị 2200 có thể bao gồm một hoặc nhiều bộ xử lý 2202, một hoặc nhiều bộ nhớ 2204 và phần cứng xử lý viđeo 2206. (các) bộ xử lý 2202 có thể được tạo cấu hình để thực hiện một hoặc nhiều phương pháp được nêu trong bản mô tả. Bộ nhớ (các bộ nhớ) 2204 có thể được sử dụng để lưu trữ dữ liệu và mã được sử dụng để triển khai các phương pháp và các kỹ thuật nêu trên. Phần cứng xử lý viđeo 2206 có thể được sử dụng để thực hiện, trong hệ mạch phần cứng, một số kỹ thuật được nêu trong bản mô tả. Theo một số phương án thực hiện, phần cứng xử lý viđeo 2206 có thể nằm bên trong hoặc nằm bên trong một phần với bộ xử lý 2202 (chẳng hạn, khối xử lý đồ họa).

Theo một số phương án thực hiện, các phương pháp mã hóa viđeo có thể được triển khai nhờ sử dụng thiết bị được triển khai trên nền tảng phần cứng như được mô tả dựa vào Fig.22.

Fig.23 là lưu đồ của phương pháp ví dụ 2300 xử lý viđeo. Phương pháp bao gồm xác định (2302), đối với phép biến đổi giữa khối viđeo hiện tại của viđeo và biểu diễn dòng bit của khối viđeo hiện tại, một hoặc nhiều bộ lọc nội suy để sử dụng trong khi biến đổi, trong đó the một hoặc nhiều bộ lọc nội suy từ nhiều bộ lọc nội suy đối với viđeo và thực hiện (2304) biến đổi nhờ sử dụng một hoặc nhiều bộ lọc nội suy.

Các giải pháp khác nhau và các phương án thực hiện được mô tả theo sáng chế còn được mô tả nhờ sử dụng danh sách giải pháp.

Phần 4, mục 1 nêu các ví dụ bổ sung của các giải pháp sau.

1. Phương pháp xử lý viđeo bao gồm các bước: thực hiện biến đổi giữa các khối

viđeo của ảnh viđeo và biểu diễn dòng bit của nó, trong đó các khối viđeo được xử lý nhờ sử dụng các nhóm lôgic của các khối cây mã, trong đó các CTB được xử lý dựa trên liệu đường biên dưới của khối cây mã dưới nằm ngoài đường biên dưới của ảnh viđeo.

2. Phương pháp theo giải pháp 1, trong đó việc xử lý CTB bao gồm thực hiện lọc vòng thích ứng các giá trị mẫu của CTB nhờ sử dụng các mẫu trong CTB.

3. Phương pháp theo giải pháp 1, trong đó việc xử lý CTB bao gồm thực hiện lọc vòng thích ứng các giá trị mẫu của CTB bằng cách vô hiệu hóa tách CTB thành hai phần theo các đường biên ảo.

Phần 4, mục 2 nêu các ví dụ bổ sung của các giải pháp sau.

4. Phương pháp xử lý viđeo bao gồm bước: xác định, dựa trên điều kiện của khối cây mã của khối viđeo hiện tại, tình trạng sử dụng của các mẫu ảo trong khi lọc trong vòng; và thực hiện biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo nhất quán với tình trạng sử dụng của các mẫu ảo.

5. Phương pháp theo giải pháp 4, trong đó, giá trị true lôgic của tình trạng sử dụng chỉ báo rằng khối viđeo hiện tại được tách thành ít nhất hai phần bằng đường biên ảo và lọc các mẫu ở một phần không được phép sử dụng thông tin từ phần khác.

6. Phương pháp theo giải pháp 4, trong đó, giá trị true lôgic của tình trạng sử dụng chỉ báo các mẫu ảo được sử dụng trong khi lọc trong vòng, và trong đó lọc trong vòng được thực hiện nhờ sử dụng các giá trị được chỉnh sửa của các mẫu được tái tạo của khối viđeo hiện tại.

7. Phương pháp theo giải pháp 4, trong đó giá trị false lôgic của tình trạng sử dụng chỉ báo rằng việc lọc các mẫu trong khối được phép sử dụng thông tin trong cùng khối.

8. Phương pháp theo giải pháp 4, trong đó, giá trị true lôgic của tình trạng sử dụng chỉ báo lọc trong vòng được thực hiện trên các mẫu được tái tạo của khối viđeo hiện tại mà không chỉnh sửa thêm các mẫu được tái tạo.

9. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 4 đến 8, trong đó điều kiện xác định để thiết lập tình trạng sử dụng bằng giá trị false lôgic do CTB có kích thước cụ thể.

10. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 4 đến 8, trong đó điều kiện xác định để thiết lập tình trạng sử dụng bằng giá trị false lôgic do CTB có kích thước lớn hơn kích thước cụ thể.

11. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 4 đến 8, khối cây mã có kích thước nhỏ hơn kích thước cụ thể.

Phần 4, mục 3 nêu các ví dụ bổ sung của các giải pháp sau.

12. Phương pháp theo giải pháp 5, trong đó điều kiện phụ thuộc liệu đường biên dưới của khối viđeo hiện tại là đường biên dưới của đơn vị viđeo mà nhỏ hơn ảnh viđeo hoặc đường biên dưới của khối viđeo hiện tại là đường biên ảo.

13. Phương pháp theo giải pháp 12, trong đó điều kiện phụ thuộc liệu đường biên dưới của khối viđeo hiện tại là đường biên dưới của lát hoặc ô hoặc đường biên khối viên.

14. Phương pháp theo giải pháp 12, trong đó điều kiện xác định để thiết lập tình trạng sử dụng bằng giá trị true lôgic khi đường biên dưới của khối viđeo hiện tại là đường biên dưới của lát hoặc ô hoặc đường biên khối viên.

15. Phương pháp theo giải pháp từ 4 đến 12, trong đó điều kiện xác định để thiết lập tình trạng sử dụng bằng giá trị false lôgic khi đường biên dưới của khối viđeo hiện tại là đường biên dưới của đường biên ảnh hoặc ngoài đường biên dưới của đường biên ảnh.

Phần 4, mục 4 nêu các ví dụ bổ sung của các giải pháp sau.

16. Phương pháp xử lý viđeo bao gồm các bước: xác định, trong khi biến đổi giữa ảnh viđeo được tạo nhóm lôgic thành một hoặc nhiều lát viđeo hoặc khối viên viđeo, và biểu diễn dòng bit của ảnh viđeo, để vô hiệu hóa sử dụng các mẫu trong lát hoặc khối viên khác trong quá trình ALF; và thực hiện biến đổi nhất quán với việc xác định.

Phần 4, mục 5 nêu các ví dụ bổ sung của các giải pháp sau.

17. Phương pháp xử lý viđeo bao gồm: xác định, trong khi biến đổi giữa khối viđeo hiện tại của ảnh viđeo và biểu diễn dòng bit của khối viđeo hiện tại, rằng khối viđeo hiện tại bao gồm các mẫu được đặt ở đường biên của đơn vị viđeo của ảnh viđeo; và thực hiện biến đổi dựa trên việc xác định, trong đó the thực hiện biến đổi bao gồm tạo các mẫu ảo cho quá trình lọc trong vòng nhờ sử dụng

phương pháp thống nhất giống như đối với tất cả các loại đường biên trong ảnh viđeo.

18. Phương pháp theo giải pháp 17, trong đó đơn vị viđeo là lát hoặc ô hoặc viđeo 360°.

19. Phương pháp theo giải pháp 17, trong đó bước lọc trong vòng bao gồm lọc vòng thích ứng.

20. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 17 đến 19, trong đó phương pháp thống nhất là phương pháp đệm hai phía.

21. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 17 đến 20, trong đó phương pháp thống nhất là khi truy nhập các mẫu dưới dòng thứ nhất không được phép và việc đệm được sử dụng để tạo các mẫu ảo cho các mẫu đó dưới dòng thứ nhất, sau đó việc truy nhập các mẫu phía trên dòng thứ hai cũng được thiết lập là không được phép và việc đệm được sử dụng để tạo các mẫu ảo cho các mẫu đó phía trên dòng thứ hai.

22. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 17 đến 20, trong đó phương pháp thống nhất là khi truy nhập các mẫu phía trên dòng thứ nhất không được phép và việc đệm được sử dụng để tạo các mẫu ảo cho các mẫu đó phía trên dòng thứ nhất, sau đó việc truy nhập các mẫu dưới dòng thứ hai cũng được thiết lập là không được phép và việc đệm được sử dụng để tạo các mẫu ảo cho các mẫu đó dưới dòng thứ hai.

23. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 21 đến 22, trong đó khoảng cách giữa dòng thứ nhất và dòng hiện tại trong đó mẫu hiện tại cần được lọc được đặt và khoảng cách giữa dòng thứ hai và dòng thứ nhất bằng.

Phần 4, mục 6 nêu các ví dụ bổ sung của các giải pháp sau.

24. Phương pháp xử lý viđeo bao gồm các bước: xác định để áp dụng, trong khi biến đổi giữa khối viđeo hiện tại của ảnh viđeo và biểu diễn dòng bit của nó, một trong nhiều phương pháp chọn mẫu lọc trong vòng (ALF) khả dụng cho ảnh viđeo trong khi biến đổi; và thực hiện biến đổi bằng cách áp dụng một trong nhiều phương pháp chọn mẫu ALF.

25. Phương pháp theo giải pháp 24, trong đó nhiều phương pháp chọn mẫu ALF bao gồm phương pháp thứ nhất trong đó các mẫu được chọn trước bộ lọc trong

vòng được áp dụng cho khối viđeo hiện tại trong khi biến đổi và phương pháp thứ hai trong đó các mẫu được chọn sau bộ lọc trong vòng được áp dụng cho khối viđeo hiện tại trong khi biến đổi.

Phần 4, mục 7 nêu các ví dụ bổ sung của các giải pháp sau.

26. Phương pháp xử lý viđeo bao gồm các bước: thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên vô hiệu hóa việc sử dụng các mẫu giao cắt khối dữ liệu đường ống ảo (VPDU) của ảnh viđeo, và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

27. Phương pháp theo giải pháp 26, trong đó VPDU tương ứng với vùng của ảnh viđeo có kích thước cố định.

28. Phương pháp theo giải pháp 26 hoặc 27, trong đó quy tắc đường biên còn xác định để sử dụng các mẫu ảo để lọc trong vòng thay cho các mẫu bị vô hiệu hóa.

29. Phương pháp theo giải pháp 28, trong đó các mẫu ảo được tạo bằng cách đệm.

Phần 4, mục 8 nêu các ví dụ bổ sung của các giải pháp sau.

30. Phương pháp xử lý viđeo bao gồm các bước: thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên xác định để sử dụng, đối với các vị trí của khối viđeo hiện tại đi qua đường biên đơn vị viđeo, các mẫu được tạo mà không sử dụng đệm; và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

31. Phương pháp theo giải pháp 30, trong đó các mẫu được tạo nhờ sử dụng kỹ thuật đệm hai phía.

32. Phương pháp theo giải pháp 30, trong đó hoạt động lọc trong vòng bao gồm việc sử dụng kỹ thuật tạo mẫu ảo tương tự đối với các mẫu được đặt đối xứng trong khi hoạt động lọc trong vòng.

33. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 30 đến 32, trong đó hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại bao gồm thực

hiện tạo dạng lại các mẫu của khối viđeo hiện tại trước khi áp dụng lọc trong vòng.

Phần 4, mục 9 nêu các ví dụ bổ sung của các giải pháp sau.

34. Phương pháp xử lý viđeo bao gồm các bước: thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên xác định lựa chọn, đối với hoạt động lọc trong vòng, bộ lọc có các kích thước sao cho các mẫu của khối viđeo hiện tại được sử dụng trong khi lọc trong vòng không giao với đường biên của đơn vị viđeo của ảnh viđeo; và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

Phần 4, mục 10 nêu các ví dụ bổ sung của các giải pháp sau.

35. Phương pháp xử lý viđeo, bao gồm: thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên xác định lựa chọn, đối với hoạt động lọc trong vòng, các tham số xén hoặc các hệ số lọc dựa trên liệu các mẫu được đệm có cần cho lọc trong vòng hay không; và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

36. Phương pháp theo giải pháp 35, trong đó các tham số xén hoặc các hệ số lọc được bao gồm trong biểu diễn dòng bit.

Phần 4, mục 11 nêu các ví dụ bổ sung của các giải pháp sau.

37. Phương pháp xử lý viđeo, bao gồm các bước: thực hiện, dựa trên quy tắc đường biên, hoạt động lọc trong vòng trên các mẫu của khối viđeo hiện tại của ảnh viđeo trong khi biến đổi giữa khối viđeo hiện tại và biểu diễn dòng bit của khối viđeo hiện tại; trong đó quy tắc đường biên tùy thuộc vào định danh thành phần màu của khối viđeo hiện tại; và thực hiện biến đổi nhờ sử dụng kết quả của hoạt động lọc trong vòng.

38. Phương pháp theo giải pháp 37, trong đó quy tắc đường biên là khác nhau đối với độ sáng và/hoặc các thành phần màu khác nhau.

39. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 1 đến 38, trong đó

phép biến đổi bao gồm bước mã hóa khối viđeo hiện tại thành biểu diễn dòng bit.

40. Phương pháp theo giải pháp bất kỳ trong các giải pháp từ 1 đến 38, trong đó phép biến đổi bao gồm giải mã biểu diễn dòng bit để tạo các giá trị mẫu của khối viđeo hiện tại.

41. Thiết bị mã hóa viđeo bao gồm bộ xử lý được tạo cấu hình để thực hiện phương pháp được nêu theo một hoặc nhiều giải pháp từ 1 đến 38.

42. Thiết bị giải mã viđeo bao gồm bộ xử lý được tạo cấu hình để thực hiện phương pháp được nêu theo một hoặc nhiều giải pháp từ 1 đến 38.

43. Vật ghi máy tính đọc được có mã được lưu trữ trên đó, mã, khi thực thi bằng bộ xử lý, khiến bộ xử lý để thực hiện phương pháp được nêu theo một hoặc nhiều giải pháp từ 1 đến 38.

Fig.31 là sơ đồ khối thể hiện hệ thống xử lý viđeo lấy làm ví dụ 3100 trong đó các kỹ thuật khác nhau được bộc lộ ở đây có thể được triển khai. Các cách triển khai khác nhau có thể bao gồm một số hoặc tất cả thành phần của hệ thống 3100. Hệ thống 3100 có thể bao gồm đầu vào 3102 để nhận nội dung viđeo. Nội dung viđeo có thể được nhận ở định dạng thô hoặc không được nén, chẳng hạn, các giá trị pixel nhiều thành phần 8 hoặc 10 bit, hoặc có thể ở định dạng nén hoặc được mã hóa. Đầu vào 3102 có thể là giao diện mạng, giao diện buýt ngoại vi, hoặc giao diện lưu trữ. Các ví dụ về giao diện mạng bao gồm giao diện hữu tuyến chẳng hạn Ethernet, mạng quang thụ động (PON), v.v., và giao diện không dây chẳng hạn giao diện Wi-Fi hoặc tế bào.

Hệ thống 3100 có thể bao gồm thành phần mã hóa 3104 mà có thể thực hiện phương pháp tạo mã hoặc mã hóa khác nhau được nêu trong bản mô tả. Thành phần mã hóa 3104 có thể giảm tốc độ bit trung bình của viđeo từ đầu vào 3102 đến đầu ra của thành phần mã hóa 3104 để tạo biểu diễn được mã hóa của viđeo. Do vậy, các kỹ thuật mã hóa đôi lúc được gọi các kỹ thuật nén viđeo hoặc chuyển mã viđeo. Đầu ra của thành phần mã hóa 3104 có thể được lưu trữ, hoặc được truyền qua kết nối truyền thông, như được biểu diễn bằng thành phần 3106. Biểu diễn dòng bit được lưu trữ hoặc được truyền (hoặc được mã hóa) của viđeo được nhận ở đầu vào 3102 có thể được sử dụng bởi thành phần 3108 để tạo các giá trị

pixel hoặc viđeo có thể hiển thị được gửi đến giao diện hiển thị 3110. Quá trình tạo viđeo mà người dùng có thể xem từ biểu diễn dòng bit đôi lúc được gọi là giải nén viđeo. Ngoài ra, trong khi các hoạt động xử lý viđeo cụ thể được gọi là các hoạt động hoặc các công cụ "mã hóa", cần hiểu rằng các công cụ hoặc các hoạt động mã hóa được sử dụng ở bộ mã hóa và các công cụ hoặc các hoạt động giải mã tương ứng mà đảo ngược các kết quả mã hóa sẽ được thực hiện bởi bộ giải mã.

Các ví dụ về giao diện buýt ngoại vi hoặc giao diện hiển thị có thể bao gồm buýt nối tiếp đa năng (USB) hoặc giao diện đa phương tiện có độ nét cao (HDMI) hoặc Displayport, và v.v.. Các ví dụ về các giao diện lưu trữ bao gồm SATA (serial advanced technology attachment), PCI, giao diện IDE, và tương tự. Các kỹ thuật được mô tả theo sáng chế có thể được triển khai trong các thiết bị điện tử khác nhau chẳng hạn điện thoại di động, máy tính xách tay, điện thoại thông minh hoặc các thiết bị khác có thể thực hiện xử lý dữ liệu số và/hoặc hiển thị viđeo.

Fig.32 là hình vẽ thể hiện lưu đồ của phương pháp 3200 xử lý viđeo theo sáng chế. Phương pháp 3200 bao gồm, ở hoạt động 3210, xác định, đối với phép biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn dòng bit của viđeo, các građien của tập con của các mẫu trong vùng đối với hoạt động phân loại trong quá trình lọc. Vùng có chiều M×N và khối có chiều K×L, M, N, K, L là các số nguyên dương, và khối được đặt trong vùng. Phương pháp cũng bao gồm, ở hoạt động 3220, thực hiện biến đổi dựa trên việc xác định.

Theo một số phương án thực hiện, tập hợp con của các mẫu bao gồm các mẫu không yêu cầu các mẫu lân cận được đặt giữa các đường biên bất kỳ của nhiều vùng viđeo của ảnh viđeo để xác định các građien. Theo một số phương án thực hiện, tập hợp con của các mẫu bao gồm mẫu hiện tại được đặt ở một hoặc nhiều đường biên của nhiều vùng viđeo của ảnh viđeo. Theo một số phương án thực hiện, các mẫu được đệm được tạo trong trường hợp ít nhất một mẫu lân cận của mẫu hiện tại được đặt giữa một hoặc nhiều đường biên.

Theo một số phương án thực hiện, các mẫu được đệm được tạo nhờ sử dụng các mẫu dọc theo một phía của đường biên. Theo một số phương án thực hiện,

các mẫu được đệm được tạo bằng cách lặp lại các mẫu ở một hoặc nhiều đường biên. Theo một số phương án thực hiện, một hoặc nhiều các dòng của các mẫu được đệm phía trên khối được tạo trong trường hợp mẫu hiện tại ở được đặt ở đường biên trên của đơn vị viđeo. Theo một số phương án thực hiện, một hoặc nhiều các dòng của các mẫu được đệm sang bên trái của khối được tạo trong trường hợp mẫu hiện tại ở được đặt ở đường biên trái của đơn vị viđeo. Theo một số phương án thực hiện, một hoặc nhiều các dòng của các mẫu được đệm sang bên phải của khối được tạo trong trường hợp mẫu hiện tại ở được đặt ở đường biên phải của đơn vị viđeo. Theo một số phương án thực hiện, một hoặc nhiều các dòng của các mẫu được đệm dưới khối được tạo trong trường hợp mẫu hiện tại ở được đặt ở đường biên dưới của đơn vị viđeo. Theo một số phương án thực hiện, đường biên bao gồm đường biên lát, đường biên khối viên, đường biên ô, đường biên ảnh con, đường biên ảo 360° hoặc đường biên ảo lọc vòng thích ứng. Theo một số phương án thực hiện, đường biên ảo lọc vòng thích ứng được kích hoạt cho khối trong trường hợp khối ở đường biên dưới của ảnh viđeo sao cho các mẫu ở phần thứ nhất của khối không được phép được sử dụng để phần thứ hai của khối trong quá trình lọc, các phần thứ nhất và thứ hai được phân chia bởi đường biên ảo lọc vòng thích ứng. Theo một số phương án thực hiện, đường biên ảo lọc vòng thích ứng được kích hoạt cho khối trong trường hợp khối không nằm ở đường biên dưới của ảnh viđeo sao cho các mẫu ở phần thứ nhất của khối không được phép được sử dụng để phần thứ hai của khối trong quá trình lọc, các phần thứ nhất và thứ hai được phân chia bằng đường biên ảo lọc vòng thích ứng.

Theo một số phương án thực hiện, một hoặc nhiều các dòng của các mẫu được đệm bao gồm một dòng hoặc hai dòng của các mẫu được đệm. Theo một số phương án thực hiện, vùng có tâm ở khối. Theo một số phương án thực hiện, $M = N = 8$. Theo một số phương án thực hiện, $K = L = 4$.

Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại ở được đặt ở đường biên trên và đường biên trái của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng $M \times (N + K1)$ dựa trên K1 dòng của các mẫu được đệm phía trên khối, K1 là số nguyên dương; và xác định vùng $(M + K2) \times (N + K1)$ dựa trên K2 dòng cuỷa các mẫu được đệm sang bên trái của khối, K2 là số nguyên

dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại ở được đặt ở đường biên trên và đường biên trái của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng (M + K2) × N dựa trên K2 dòng của các mẫu được đệm sang bên trái của khối, K2 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K1 dòng của các mẫu được đệm phía trên khối, K1 là số nguyên dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại ở được đặt ở đường biên trên và đường biên phải của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng M × (N + K1) dựa trên K1 dòng của các mẫu được đệm phía trên khối, K1 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K2 dòng của các mẫu được đệm sang bên phải của khối, K2 là số nguyên dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại được đặt ở đường biên trên và đường biên phải của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng (M + K2) × N dựa trên K2 dòng của các mẫu được đệm sang bên phải của khối, K2 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K1 dòng của các mẫu được đệm phía trên khối, K1 là số nguyên dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại ở được đặt ở đường biên dưới và đường biên phải của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng M × (N + K1) dựa trên K1 dòng của các mẫu được đệm dưới khối, K1 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K2 dòng của các mẫu được đệm sang bên phải của khối, K2 là số nguyên dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại ở được đặt ở đường biên dưới và đường biên phải của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng (M + K2) × N dựa trên K2 dòng của các mẫu được đệm sang bên phải của khối, K2 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K1 dòng của các mẫu được đệm dưới khối, K1 là số nguyên dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại ở được đặt ở đường biên dưới và đường biên trái của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng M × (N + K1) dựa trên K1 dòng của các mẫu được đệm dưới khối, K1 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K2 dòng của các mẫu được đệm sang bên trái của khối, K2 là số nguyên dương. Theo một số phương án thực hiện, trong trường hợp mẫu hiện tại

được đặt ở đường biên dưới và đường biên trái của đơn vị viđeo, vùng được điều chỉnh dựa trên xác định vùng (M + K2) × N dựa trên K2 dòng của các mẫu được đệm sang bên trái của khối, K2 là số nguyên dương; và xác định vùng (M + K2) × (N + K1) dựa trên K1 dòng của các mẫu được đệm dưới khối, K1 là số nguyên dương. Theo một số phương án thực hiện, các građien được xác định một phần dựa trên các mẫu được đệm. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con M × (N - C1) của vùng trong trường hợp khối ở đường biên trên hoặc dưới của đơn vị viđeo, C1 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng trên hoặc dưới của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1) × N của vùng trong trường hợp khối ở đường biên trái hoặc phải của đơn vị viđeo, C1 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng bên trái hoặc bên phải của các mẫu không được bao gồm đối với hoạt động phân loại.

Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con M × (N - C1 - C2) của vùng trong trường hợp khối nằm ở đường biên trên và dưới của đơn vị viđeo, C1 và C2 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng trên và C2 dòng dưới của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1) × (N - C2) của vùng trong trường hợp khối nằm ở đường biên trên và bên trái của đơn vị viđeo, C1 và C2 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng trên và C2 dòng bên trái của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1) × (N - C2) của vùng trong trường hợp khối ở đường biên trên và phải của đơn vị viđeo, C1 và C2 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng bên trên và C2 dòng bên phải của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1) × (N - C2) của vùng trong trường hợp khối nằm ở đường biên dưới và trái của đơn vị viđeo, C1 và C2 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng dưới và C2 dòng trái của các mẫu không được bao gồm đối

với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1) × (N - C2) của vùng trong trường hợp khối nằm ở đường biên dưới và phải của đơn vị viđeo, C1 và C2 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng dưới và C2 dòng phải của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1 - C2) × N của vùng trong trường hợp khối nằm ở đường biên trái và phải của đơn vị viđeo, C1 và C2 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng trên và C2 dòng dưới của các mẫu không được bao gồm đối với hoạt động phân loại.

Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C3) × (N - C1 - C2) của vùng trong trường hợp khối nằm ở đường biên trên, đường biên dưới, và đường biên trái của đơn vị viđeo, C1, C2 và C3 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng bên trên, C2 dòng dưới, và C3 dòng bên trái của samples không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C3) × (N - C1 - C2) của vùng trong trường hợp khối nằm ở đường biên trên, đường biên dưới, và đường biên phải của đơn vị viđeo, C1, C2 và C3 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng trên, C2 dòng dưới, và C3 dòng bên phải của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1 - C2) × (N - C3) của vùng trong trường hợp khối nằm ở đường biên trái, đường biên phải, và đường biên trên của đơn vị viđeo, C1, C2 và C3 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng trái, C2 dòng phải, và top C3 dòng của các mẫu không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng con (M - C1 - C2) × (N - C3) của vùng trong trường hợp khối nằm ở đường biên trái, đường biên phải, và đường biên dưới của đơn vị viđeo, C1, C2 và C3 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng bên trái, C2 dòng bên phải, và C3 dòng bên trên của các mẫu không được bao gồm đối với hoạt động phân loại.

Theo một số phương án thực hiện, tập hợp con của các mẫu được đặt ở vùng

con (M - C1 - C2) × (N - C3 - C4) của vùng trong trường hợp khối nằm ở đường biên trái, đường biên phải, đường biên trên and đường biên dưới của đơn vị viđeo, C1, C2, C3, và C4 là số nguyên dương. Theo một số phương án thực hiện, C1 dòng bên trái, C2 dòng bên phải, C3 dòng bên trên, và C4 mẫu bên dưới không được bao gồm đối với hoạt động phân loại. Theo một số phương án thực hiện, ít nhất một trong C1, C2, C3, hoặc C4 bằng 2. Theo một số phương án thực hiện, C1, C2, C3, và C4 bằng 2.

Theo một số phương án thực hiện, đơn vị viđeo bao gồm lát, khối viên, hoặc ô. Theo một số phương án thực hiện, tập hợp con của các mẫu bao gồm các mẫu có các mẫu lân cận không được đặt giữa các đường biên bất kỳ của nhiều vùng viđeo của ảnh viđeo.

Fig.33 là hình vẽ thể hiện lưu đồ của phương pháp 3300 xử lý viđeo theo sáng chế. Phương pháp 3300 bao gồm, ở hoạt động 3310, xác định, đối với phép biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn dòng bit của viđeo, hoạt động đệm đồng nhất trong quá trình lọc vòng thích ứng có thể áp dụng cho các mẫu được đặt ở đường biên ảo 360 của nhiều vùng viđeo của ảnh viđeo bất kể vị trí của đường biên ảo 360 trong ảnh viđeo. Phương pháp 3300 cũng bao gồm, ở hoạt động 3320, thực hiện biến đổi dựa trên việc xác định.

Theo một số phương án thực hiện, hoạt động đệm đồng nhất bao gồm bước tạo các mẫu được đệm bước sử dụng các mẫu dọc theo một phía của đường biên ảo 360. Theo một số phương án thực hiện, các mẫu được đệm được tạo bằng cách lặp lại các mẫu ở đường biên ảo 360. Theo một số phương án thực hiện, đường biên ảo 360 không trùng với đường biên thực của khối. Theo một số phương án thực hiện, hoạt động đệm đồng nhất bao gồm tạo các mẫu được đệm nhờ sử dụng các mẫu từ hai phía của the đường biên ảo 360. Theo một số phương án thực hiện, đường biên ảo 360 bao gồm mép bao quanh của viđeo 360° được chiếu trên ảnh viđeo nhờ sử dụng định dạng chiếu đẳng chữ nhật.

Theo một số phương án thực hiện, phép biến đổi bao gồm bước mã hóa viđeo thành biểu diễn dòng bit. Theo một số phương án thực hiện, phép biến đổi bao gồm giải mã biểu diễn dòng bit thành viđeo.

Từ phần trên, cần hiểu rằng các phương án thực hiện cụ thể của sáng chế đã

được nêu trên để minh họa, nhưng các chỉnh sửa khác nhau có thể được thực hiện mà không lệch khỏi phạm vi sáng chế. Do đó, sáng chế không bị giới hạn ngoại trừ như bởi các điểm yêu cầu bảo hộ đi kèm.

Một số phương án thực hiện sáng chế bao gồm ra quyết định hoặc xác định kích hoạt công cụ hoặc chế độ xử lý viđeo. Trong ví dụ, khi công cụ hoặc chế độ xử lý viđeo được kích hoạt, bộ mã hóa sẽ sử dụng hoặc triển khai công cụ hoặc chế độ khi xử lý khối viđeo, nhưng có thể không nhất thiết chỉnh sửa dòng bit thu được dựa trên sử dụng công cụ hoặc chế độ. Tức là, biến đổi từ khối viđeo sang biểu diễn dòng bit của viđeo sẽ sử dụng công cụ hoặc chế độ xử lý viđeo khi nó được kích hoạt dựa trên quyết định hoặc xác định. Trong ví dụ khác, khi công cụ hoặc chế độ xử lý viđeo được kích hoạt, bộ giải mã sẽ xử lý dòng bit với nhận thức rằng dòng bit đã được chỉnh sửa dựa trên công cụ hoặc chế độ xử lý viđeo. Tức là, biến đổi từ biểu diễn dòng bit của viđeo sang khối viđeo sẽ được thực hiện nhờ sử dụng công cụ hoặc chế độ xử lý viđeo đã được kích hoạt dựa trên quyết định hoặc xác định.

Một số phương án thực hiện sáng chế bao gồm ra quyết định hoặc xác định để vô hiệu hóa công cụ hoặc chế độ xử lý viđeo. Trong ví dụ, khi công cụ hoặc chế độ xử lý viđeo bị vô hiệu hóa, bộ mã hóa sẽ không sử dụng công cụ hoặc chế độ trong phép biến đổi của khối viđeo sang biểu diễn dòng bit của viđeo. Trong ví dụ khác, khi công cụ hoặc chế độ xử lý viđeo bị vô hiệu hóa, bộ giải mã sẽ xử lý dòng bit với nhận thức rằng dòng bit không được chỉnh sửa nhờ sử dụng công cụ hoặc chế độ xử lý viđeo đã được kích hoạt dựa trên quyết định hoặc xác định.

Các phương án thực hiện sáng chế và các hoạt động chức năng được mô tả theo sáng chế có thể được triển khai trong các hệ thống khác nhau, hệ mạch điện tử số, hoặc trong phần mềm máy tính, firmware, hoặc phần cứng, bao gồm các cấu trúc được bộc lộ theo sáng chế và các tương đương về cấu trúc, hoặc khi kết hợp của một hoặc nhiều trong số chúng. Các phương án thực hiện sáng chế được mô tả theo sáng chế có thể được triển khai dưới dạng một hoặc nhiều sản phẩm chương trình máy tính, chẳng hạn, một hoặc nhiều môđun lệnh chương trình máy tính được mã hóa trên phương tiện máy tính đọc được hữu hình và bất biến để thực thi bởi, hoặc để điều khiển hoạt động của, thiết bị xử lý dữ liệu. Phương tiện

máy tính đọc được có thể là thiết bị lưu trữ máy đọc được, nền tảng lưu trữ máy đọc được, thiết bị nhớ, hợp chất ảnh hưởng tín hiệu được lan truyền máy đọc được, hoặc kết hợp của một hoặc nhiều trong số chúng. Cụm từ "khối xử lý dữ liệu" hoặc "thiết bị xử lý dữ liệu" bao gồm tất cả thiết bị, bộ phận, và máy để xử lý dữ liệu, bao gồm thông qua ví dụ bộ xử lý lập trình được, máy tính, hoặc nhiều bộ xử lý hoặc máy tính. Thiết bị có thể bao gồm, bao gồm phần cứng, mã mà tạo môi trường thực thi cho chương trình máy tính đang chạy, chẳng hạn, mã tạo firmware bộ xử lý, xếp chồng giao thức, hệ thống quản lý cơ sở dữ liệu, hệ điều hành, hoặc kết hợp của một hoặc nhiều trong số chúng.

Chương trình máy tính (cũng được biết đến như là chương trình, phần mềm, ứng dụng phần mềm, tập lệnh, hoặc mã) có thể được viết ở dạng ngôn ngữ lập trình bất kỳ, bao gồm ngôn ngữ được biên dịch hoặc thông dịch, và có thể được khai triển ở dạng bất kỳ, bao gồm dưới dạng chương trình độc lập hoặc dưới dạng môđun, thành phần, trình con, hoặc đơn vị khác thích hợp để sử dụng trong môi trường tính toán. Chương trình máy tính không nhất thiết tương ứng với tệp tin trong hệ thống tệp tin. Chương trình có thể được lưu trữ trong phần tệp tin chứa các chương trình hoặc dữ liệu khác(chẳng hạn, một hoặc nhiều tập lệnh được lưu trữ trong tài liệu ngôn ngữ đánh dấu), trong một tệp tin dành riêng cho chương trình đang chạy, hoặc trong nhiều tập tin được phối hợp (chẳng hạn, các tệp tin mà lưu trữ một hoặc nhiều môđun, chương trình con, hoặc các phần mã). Chương trình máy tính có thể được khai triển để được thực thi trên một máy tính hoặc trên nhiều máy tính mà được đặt ở một địa điểm hoặc được phân tán giữa nhiều địa điểm và được liên kết bởi mạng truyền thông.

Các tiến trình và các luồng lôgic được mô tả theo sáng chế có thể được thực hiện bởi một hoặc nhiều bộ xử lý lập trình được thực thi một hoặc nhiều các chương trình máy tính thực hiện các chức năng bằng cách hoạt động trên dữ liệu đầu vào và tạo đầu ra. Các tiến trình và các luồng lôgic cũng có thể được thực hiện bởi, và thiết bị cũng có thể được triển khai dưới dạng, hệ mạch lôgic chuyên dụng, chẳng hạn, FPGA (mảng cổng dạng trường lập trình được) hoặc ASIC (mạch tích hợp ứng dụng cụ thể).

Các bộ xử lý thích hợp để thực thi chương trình máy tính bao gồm, thông qua

ví dụ, cả bộ vi xử lý đa năng và chuyên dụng, và một hoặc nhiều bộ xử lý của loại máy tính số bất kỳ. Nói chung, bộ xử lý sẽ nhận lệnh và dữ liệu từ bộ nhớ chỉ đọc hoặc bộ nhớ truy nhập ngẫu nhiên hoặc cả hai. Các phần tử cơ bản của máy tính là bộ xử lý để thực hiện các lệnh và một hoặc nhiều các thiết bị nhớ để lưu trữ các lệnh và dữ liệu. Nói chung, máy tính cũng sẽ bao gồm, hoặc được ghép nối vận hành để nhận dữ liệu từ hoặc truyền dữ liệu đến, hoặc cả hai, một hoặc nhiều thiết bị lưu trữ lớn để lưu trữ dữ liệu, chẳng hạn, đĩa từ, đĩa quang từ, hoặc đĩa quang. Tuy nhiên, máy tính không cần có các thiết bị này. Các phương tiện máy tính đọc được thích hợp để lưu trữ các lệnh chương trình máy tính và dữ liệu bao gồm tất cả các dạng của bộ nhớ bất biến, phương tiện và các thiết bị nhớ, bao gồm thông qua ví dụ các thiết bị nhớ bán dẫn, chẳng hạn, EPROM, EEPROM, và các thiết bị nhớ nhanh. Bộ xử lý và bộ nhớ có thể được bổ sung bởi, hoặc được đưa vào, hệ mạch lôgic chuyên dụng.

Bản mô tả, cùng với các hinh vẽ, sẽ, được xem chỉ lấy ví dụ, trong đó lấy làm ví dụ nghĩa là ví dụ. Như được sử dụng ở đây, việc sử dụng "hoặc" sẽ bao gồm "và/hoặc", trừ khi ngữ cảnh rõ ràng chỉ báo ngược lại.

Trong khi sáng chế chứa nhiều đặc tả, điều này không được hiểu như là giới hạn ở phạm vi của sáng chế hoặc phần có thể được bảo hộ, nhưng thay vì dưới dạng các phần mô tả của các dấu hiệu mà có thể riêng với các phương án thực hiện cụ thể của sáng chế. Các dấu hiệu cụ thể mà được mô tả theo sáng chế trong các phương án thực hiện riêng rẽ cũng có thể được triển khai kết hợp theo một phương án thực hiện. Ngược lại, các dấu hiệu khác nhau mà được mô tả theo ngữ cảnh của một phương án thực hiện cũng có thể được triển khai theo nhiều phương án thực hiện riêng rẽ hoặc theo tổ hợp phụ thích hợp bất kỳ. Ngoài ra, mặc dù các dấu hiệu có thể được nêu trên như là hoạt động theo tổ hợp cụ thể và thậm chí được bảo hộ ban đầu, một hoặc nhiều dấu hiệu từ tổ hợp được bảo hộ trong một số trường hợp có thể được thực thi từ tổ hợp này, và tổ hợp được bảo hộ có thể được dành cho tổ hợp phụ hoặc biến thể của tổ hợp phụ.

Tương tự, trong khi các hoạt động được mô tả trên các hình vẽ theo thứ tự cụ thể, điều này không nên được hiểu như là yêu cầu rằng các hoạt động này được thực hiện theo thứ tự cụ thể được thể hiện hoặc lần lượt, hoặc tất cả các hoạt

động được minh họa được thực hiện, để đạt được các kết quả mong muốn. Ngoài ra, việc tách riêng các thành phần hệ thống khác nhau trong các phương án thực hiện được mô tả theo sáng chế không nên được hiểu như là yêu cầu tách riêng theo tất cả các phương án thực hiện.

Chỉ vài triển khai và ví dụ được mô tả và các triển khai khác, các cải tiến và các biến thể có thể được thực hiện dựa trên phần được mô tả và được minh họa theo sáng chế.

YÊU CẦU BẢO HỘ

1. Phương pháp xử lý dữ liệu viđeo bao gồm các bước:

xác định, để biến đổi giữa ảnh trong viđeo bao gồm một hoặc nhiều khối cây tạo mã và dòng bit của viđeo, các građien của nhóm phụ của các mẫu trong vùng có kích thước 8×8 để dẫn xuất phân loại của khối 4×4 của khối cây tạo mã thứ nhất trong quá trình lọc thứ nhất, trong đó khối 4×4 được đặt trong khối cây tạo mã thứ nhất và khối 4×4 được đặt trong vùng 8×8; và

thực hiện biến đổi dựa trên bước xác định,

trong đó quá trình lọc thứ nhất bao gồm các bước:

xác định tập hợp hệ số lọc đối với mẫu của khối 4×4 dựa trên các građien;

xác định giá trị cắt xén được thiết lập cho mẫu; và

áp dụng bộ lọc trên mẫu dựa trên tập hợp hệ số lọc và tập hợp giá trị cắt xén,

trong đó trong trường hợp không được phép sử dụng các mẫu giữa đường biên của đơn vị viđeo trong quá trình lọc thứ nhất, hai cột cực trái của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên trái của đơn vị viđeo; hoặc hai đường trên cùng của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên trên cùng của đơn vị viđeo; hoặc hai đường cực phải của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên phải của đơn vị viđeo; hoặc hai đường cực dưới của các mẫu được đệm trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên dưới cùng của đơn vị viđeo.

2. Phương pháp theo điểm 1, trong đó đường biên của đơn vị viđeo bao gồm đường biên lát, đường biên ô, đường biên ảnh phụ, đường biên ảo 360° hoặc đường biên ảo lọc vòng thích ứng.

3. Phương pháp theo điểm 2, trong đó đường biên ảo lọc vòng thích ứng được kích hoạt cho khối cây tạo mã thứ nhất trong trường hợp đường biên dưới cùng của khối cây tạo mã thứ nhất không phải ở đường biên dưới cùng của ảnh.

4. Phương pháp theo điểm 2, trong đó đường biên ảo lọc vòng thích ứng được kích hoạt cho khối cây tạo mã thứ nhất trong trường hợp khối cây tạo mã thứ

nhất ơ đường biên dưới cùng của ảnh và khoảng cách dọc giữa mẫu trên bên trái của khối cây tạo mã thứ nhất và đường biên dưới của ảnh không nhỏ hơn giá trị cụ thể.

5. Phương pháp theo điểm 2, trong đó hoạt động đệm một phía được áp dụng cho các mẫu giữa đường biên ảo 360° bất kể vị trí của đường biên ảo 360 độ trong ảnh, trong đó trong hoạt động đệm một phía, nếu một mẫu cần được sử dụng nằm ngoài đường biên ảo 360°, nó được sao chép từ mẫu khả dụng bên trong đường biên ảo 360°.

6. Phương pháp theo điểm 1, trong đó các phần tử cú pháp riêng rẽ chỉ báo liệu các mẫu giữa các loại đường biên khác nhau được phép sử dụng được báo hiệu ở các mức khác nhau, và trong đó các loại đường biên khác nhau bao gồm đường biên lát, đường biên ô và đường biên ảnh phụ.

7. Phương pháp theo điểm 6, trong đó phần tử cú pháp thứ nhất đối với đường biên ô của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên ô được phép sử dụng; phần tử cú pháp thứ hai đối với đường biên lát của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên lát được phép sử dụng; phần tử cú pháp thứ ba đối với đường biên ảnh phụ của các phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi để chỉ báo liệu các mẫu giữa các đường biên ảnh phụ được phép sử dụng.

8. Phương pháp theo điểm 6, trong đó các phần tử cú pháp riêng rẽ còn được áp dụng trong ít nhất một trong quá trình lọc tách khối, quá trình lọc độ lệch mẫu thích ứng (sample adaptive offset, SAO), quá trình lọc hai bên, hoặc quá trình ọc biến đổi Hadamard.

9. Phương pháp theo điểm 1, trong đó vùng 8×8 được đặt trong vùng 10×10, trong trường hợp không được phép sử dụng các mẫu giữa đường biên của đơn vị viđeo trong quá trình lọc thứ nhất, ba cột cực trái của các mẫu trong vùng 10×10 bao gồm hai cột của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên trái của đơn vị viđeo.

10. Phương pháp theo điểm 1, trong đó phép biến đổi bao gồm bước mã hóa viđeo thành dòng bit.

11. Phương pháp theo điểm 1, trong đó phép biến đổi bao gồm bước giải mã viđeo từ dòng bit.

12. Thiết bị xử lý dữ liệu viđeo bao gồm bộ xử lý và bộ nhớ bất biến có các lệnh trên đó, trong đó các lệnh khi thực thi bởi bộ xử lý, khiến bộ xử lý:

xác định, để biến đổi giữa ảnh trong viđeo bao gồm một hoặc nhiều khối cây tạo mã và dòng bit của viđeo, các građien của nhóm phụ của các mẫu trong vùng có kích thước 8×8 để dẫn xuất phân loại của khối 4×4 của khối cây tạo mã thứ nhất trong quá trình lọc thứ nhất, trong đó khối 4×4 được đặt trong khối cây tạo mã thứ nhất và khối 4×4 được đặt trong vùng 8×8; và

thực hiện phép biến đổi dựa trên bước xác định,

trong đó quá trình lọc thứ nhất bao gồm:

xác định tập hợp hệ số lọc đối với mẫu của khối 4×4 dựa trên các građien;

xác định giá trị cắt xén được thiết lập cho mẫu; và

áp dụng bộ lọc trên mẫu dựa trên tập hợp hệ số lọc và tập hợp giá trị cắt xén,

trong đó trong trường hợp không được phép sử dụng các mẫu giữa đường biên của đơn vị viđeo trong quá trình lọc thứ nhất, hai cột cực trái của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên trái của đơn vị viđeo; hoặc hai đường trên cùng của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên trên cùng của đơn vị viđeo; hoặc hai đường cực phải của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên phải của đơn vị viđeo; hoặc hai đường cực dưới của các mẫu được đệm trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên dưới cùng của đơn vị viđeo.

13. Thiết bị theo điểm 12, trong đó đường biên của đơn vị viđeo bao gồm đường biên lát, đường biên ô, đường biên ảnh phụ, đường biên ảo 360° hoặc đường biên ảo lọc vòng thích ứng; và

trong đó đường biên ảo lọc vòng thích ứng được kích hoạt cho khối cây tạo mã thứ nhất trong trường hợp đường biên dưới cùng của khối cây tạo mã thứ nhất không phải ở đường biên dưới cùng của ảnh, hoặc trong trường hợp khối cây tạo mã thứ nhất ở đường biên dưới cùng của ảnh và khoảng cách dọc giữa

mẫu trên bên trái của khối cây tạo mã thứ nhất và đường biên dưới của ảnh không nhỏ hơn giá trị cụ thể;

trong đó hoạt động đệm một phía được áp dụng cho các mẫu giữa đường biên ảo 360° bất kể vị trí của đường biên ảo 360° trong ảnh, trong đó trong hoạt động đệm một phía, nếu một mẫu cần được sử dụng nằm ngoài đường biên ảo 360°, nó được sao chép từ mẫu khả dụng bên trong đường biên ảo 360°;

trong đó các phần tử cú pháp riêng rẽ chỉ báo liệu các mẫu giữa các loại đường biên khác nhau được phép sử dụng được báo hiệu ở các mức khác nhau, và trong đó các loại đường biên khác nhau bao gồm đường biên lát, đường biên ô và đường biên ảnh phụ;

trong đó phần tử cú pháp thứ nhất đối với đường biên ô của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên ô được phép sử dụng; phần tử cú pháp thứ hai đối với đường biên lát của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên lát được phép sử dụng; phần tử cú pháp thứ ba đối với đường biên ảnh phụ của các phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi để chỉ báo liệu các mẫu giữa các đường biên ảnh phụ được phép sử dụng; và

trong đó các phần tử cú pháp riêng rẽ còn được áp dụng trong ít nhất một trong quá trình lọc tách khối, quá trình lọc SAO, quá trình lọc hai bên, ohoặc quá trình lọc biến đổi Hadamard.

14. Vật ghi máy tính đọc được bất biến lưu trữ các lệnh mà khiến bộ xử lý:

xác định, để biến đổi giữa ảnh trong viđeo bao gồm một hoặc nhiều khối cây tạo mã và dòng bit của viđeo, các građien của nhóm phụ của các mẫu trong vùng có kích thước 8×8 để dẫn xuất phân loại của a 4×4 block của khối cây tạo mã thứ nhất trong quá trình lọc thứ nhất, trong đó khối 4×4 được đặt trong khối cây tạo mã thứ nhất và khối 4×4 được đặt trong vùng 8×8; và

thực hiện phép biến đổi dựa trên bước xác định,

trong đó quá trình lọc thứ nhất bao gồm:

xác định tập hợp hệ số lọc đối với mẫu của khối 4×4 dựa trên các građien;

xác định giá trị cắt xén được thiết lập cho mẫu; và

áp dụng bộ lọc trên mẫu dựa trên tập hợp hệ số lọc và tập hợp giá trị cắt xén,

trong đó trong trường hợp không được phép sử dụng các mẫu giữa đường biên của đơn vị viđeo trong quá trình lọc thứ nhất, hai cột cực trái của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên trái của đơn vị viđeo; hoặc hai đường trên cùng của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên trên cùng của đơn vị viđeo; hoặc hai đường cực phải của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên phải của đơn vị viđeo; hoặc hai đường cực dưới của các mẫu được đệm trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên dưới cùng của đơn vị viđeo.

15. Vật ghi máy tính đọc được bất biến theo điểm 14, trong đó đường biên của đơn vị viđeo bao gồm đường biên lát, đường biên ô, đường biên ảnh phụ, đường biên ảo 360° hoặc đường biên ảo lọc vòng thích ứng;

trong đó đường biên ảo lọc vòng thích ứng được kích hoạt cho khối cây tạo mã thứ nhất trong trường hợp đường biên dưới cùng của khối cây tạo mã thứ nhất không phải ở đường biên dưới cùng của ảnh, hoặc trong trường hợp khối cây tạo mã thứ nhất ở đường biên dưới cùng của ảnh và khoảng cách dọc giữa mẫu trên bên trái của khối cây tạo mã thứ nhất và đường biên dưới của ảnh không nhỏ hơn giá trị cụ thể;

trong đó hoạt động đệm một phía được áp dụng cho các mẫu giữa đường biên ảo 360° bất kể vị trí của đường biên ảo 360° trong ảnh, trong đó trong hoạt động đệm một phía, nếu một mẫu cần được sử dụng nằm ngoài đường biên ảo 360°, nó được sao chép từ mẫu khả dụng bên trong đường biên ảo 360°;

trong đó các phần tử cú pháp riêng rẽ chỉ báo liệu các mẫu giữa các loại đường biên khác nhau được phép sử dụng được báo hiệu ở các mức khác nhau, và trong đó các loại đường biên khác nhau bao gồm đường biên lát, đường biên ô và đường biên ảnh phụ;

trong đó phần tử cú pháp thứ nhất đối với đường biên ô của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên ô được phép sử dụng;

phần tử cú pháp thứ hai đối với đường biên lát của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên lát được phép sử dụng; phần tử cú pháp thứ ba đối với đường biên ảnh phụ của các phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi để chỉ báo liệu các mẫu giữa các đường biên ảnh phụ được phép sử dụng; và

trong đó các phần tử cú pháp riêng rẽ còn được áp dụng trong ít nhất một trong quá trình lọc tách khối, quá trình lọc SAO, quá trình lọc hai bên, hoặc quá trình lọc biến đổi Hadamard.

16. Phương pháp lưu trữ dòng bit của viđeo bao gồm các bước:

xác định các građien của nhóm phụ của các mẫu trong vùng có kích thước 8×8 để dẫn xuất phân loại của khối 4×4 của khối cây tạo mã thứ nhất trong ảnh của viđeo trong quá trình lọc thứ nhất, trong đó khối 4×4 được đặt trong khối cây tạo mã thứ nhất và khối 4×4 được đặt trong vùng 8×8;

tạo dòng bit dựa trên việc xác định; và

lưu trữ dòng bit trong vật ghi máy tính đọc được bất biến,

trong đó quá trình lọc thứ nhất bao gồm:

xác định tập hợp hệ số lọc đối với mẫu của khối 4×4 dựa trên các građien;

xác định giá trị cắt xén được thiết lập cho mẫu; và

áp dụng bộ lọc trên mẫu dựa trên tập hợp hệ số lọc và tập hợp giá trị cắt xén,

trong đó trong trường hợp không được phép sử dụng các mẫu giữa đường biên của đơn vị viđeo trong quá trình lọc thứ nhất, hai cột cực trái của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên trái của đơn vị viđeo; hoặc hai đường trên cùng của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên trên cùng của đơn vị viđeo; hoặc hai đường cực phải của các mẫu trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên bên phải của đơn vị viđeo; hoặc hai đường cực dưới của các mẫu được đệm trong vùng 8×8 được đệm khi có mẫu trong khối 4×4 được đặt ở đường biên dưới cùng của đơn vị viđeo.

17. Phương pháp theo điểm 16, trong đó đường biên của đơn vị viđeo bao gồm đường biên lát, đường biên ô, đường biên ảnh phụ, đường biên ảo 360° hoặc đường biên ảo lọc vòng thích ứng;

trong đó đường biên ảo lọc vòng thích ứng được kích hoạt cho khối cây tạo mã thứ nhất trong trường hợp đường biên dưới cùng của khối cây tạo mã thứ nhất không phải ở đường biên dưới cùng của ảnh, hoặc trong trường hợp khối cây tạo mã thứ nhất ở đường biên dưới cùng của ảnh và khoảng cách dọc giữa mẫu trên bên trái của khối cây tạo mã thứ nhất và đường biên dưới của ảnh không nhỏ hơn giá trị cụ thể;

trong đó hoạt động đệm một phía được áp dụng cho các mẫu giữa đường biên ảo 360° bất kể vị trí của đường biên ảo 360° trong ảnh, trong đó trong hoạt động đệm một phía, nếu một mẫu cần được sử dụng nằm ngoài đường biên ảo 360°, nó được sao chép từ mẫu khả dụng bên trong đường biên ảo 360°;

trong đó các phần tử cú pháp riêng rẽ chỉ báo liệu các mẫu giữa các loại đường biên khác nhau được phép sử dụng được báo hiệu ở các mức khác nhau, và trong đó các loại đường biên khác nhau bao gồm đường biên lát, đường biên ô và đường biên ảnh phụ;

trong đó phần tử cú pháp thứ nhất đối với đường biên ô của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên ô được phép sử dụng; phần tử cú pháp thứ hai đối với đường biên lát của các phần tử cú pháp được bao gồm trong tập hợp tham số ảnh để chỉ báo liệu các mẫu giữa các đường biên lát được phép sử dụng; phần tử cú pháp thứ ba đối với đường biên ảnh phụ của các phần tử cú pháp được bao gồm trong tập hợp tham số chuỗi để chỉ báo liệu các mẫu giữa các đường biên ảnh phụ được phép sử dụng; và

trong đó các phần tử cú pháp riêng rẽ còn được áp dụng trong ít nhất một trong quá trình lọc tách khối, quá trình lọc SAO, quá trình lọc hai bên, hoặc quá trình lọc biến đổi Hadamard.

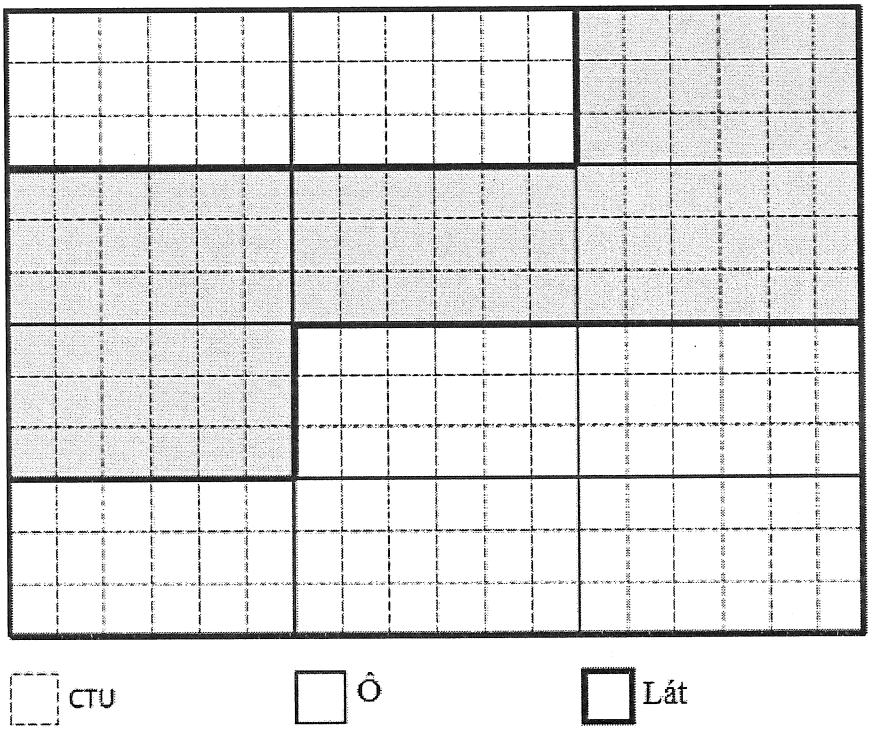


Fig.1

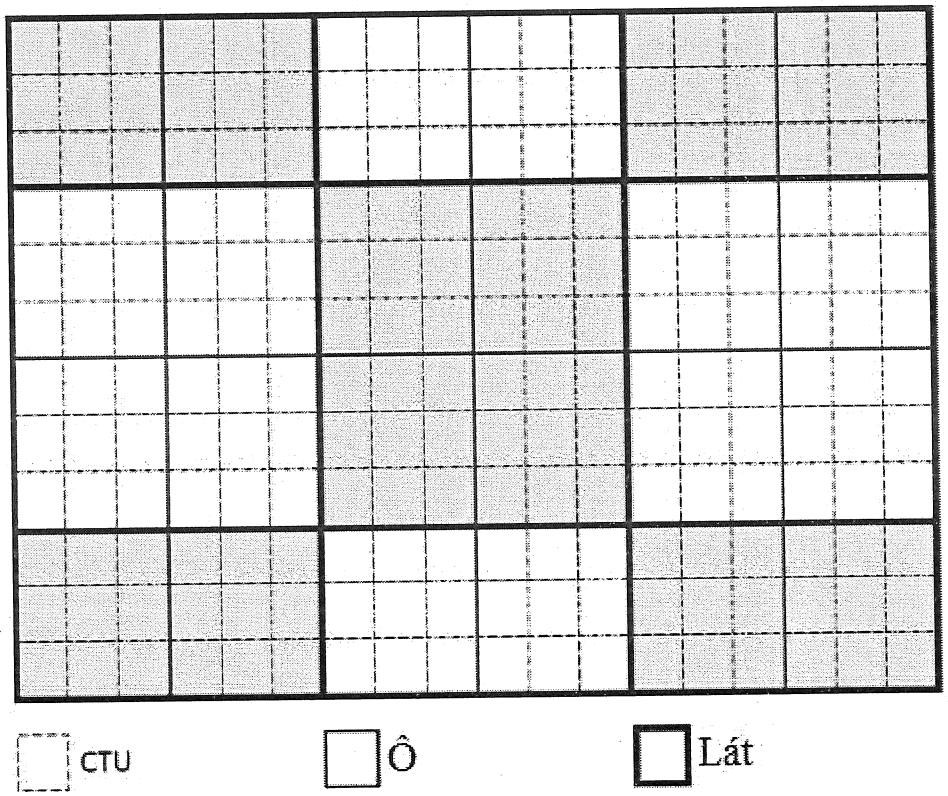


Fig.2

3/36

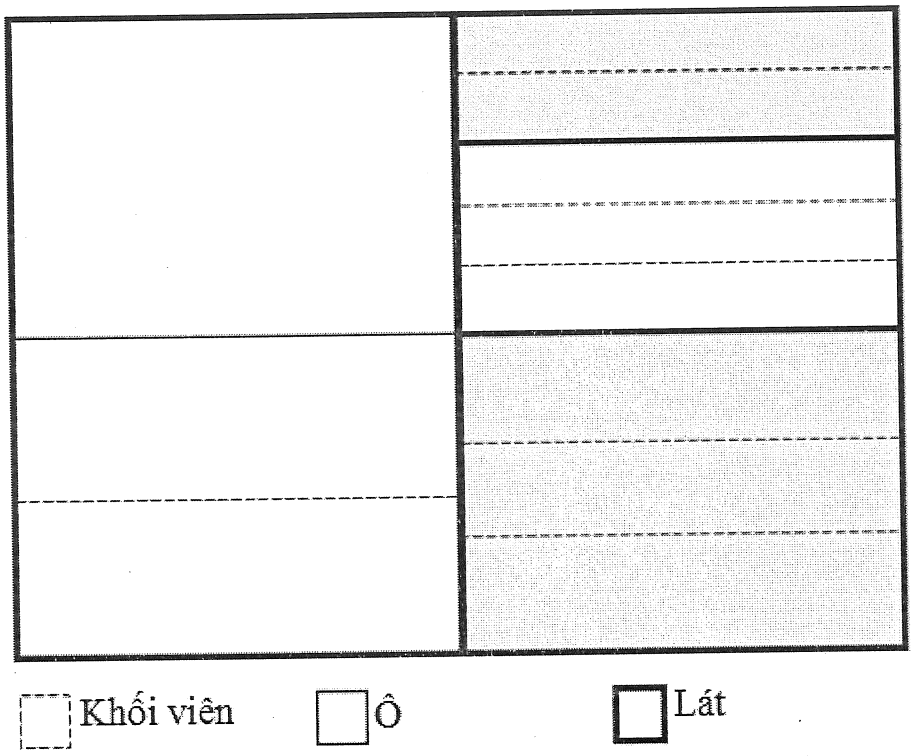


Fig.3

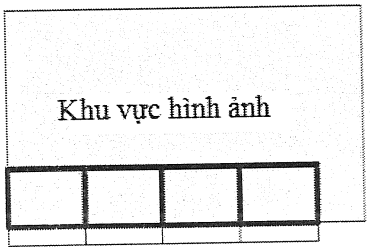


Fig.4A

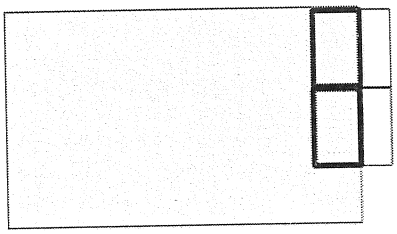


Fig.4B

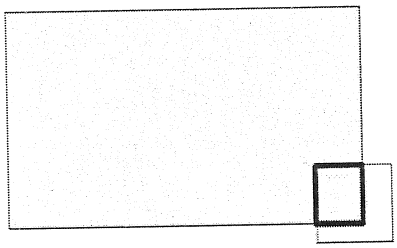


Fig.4C

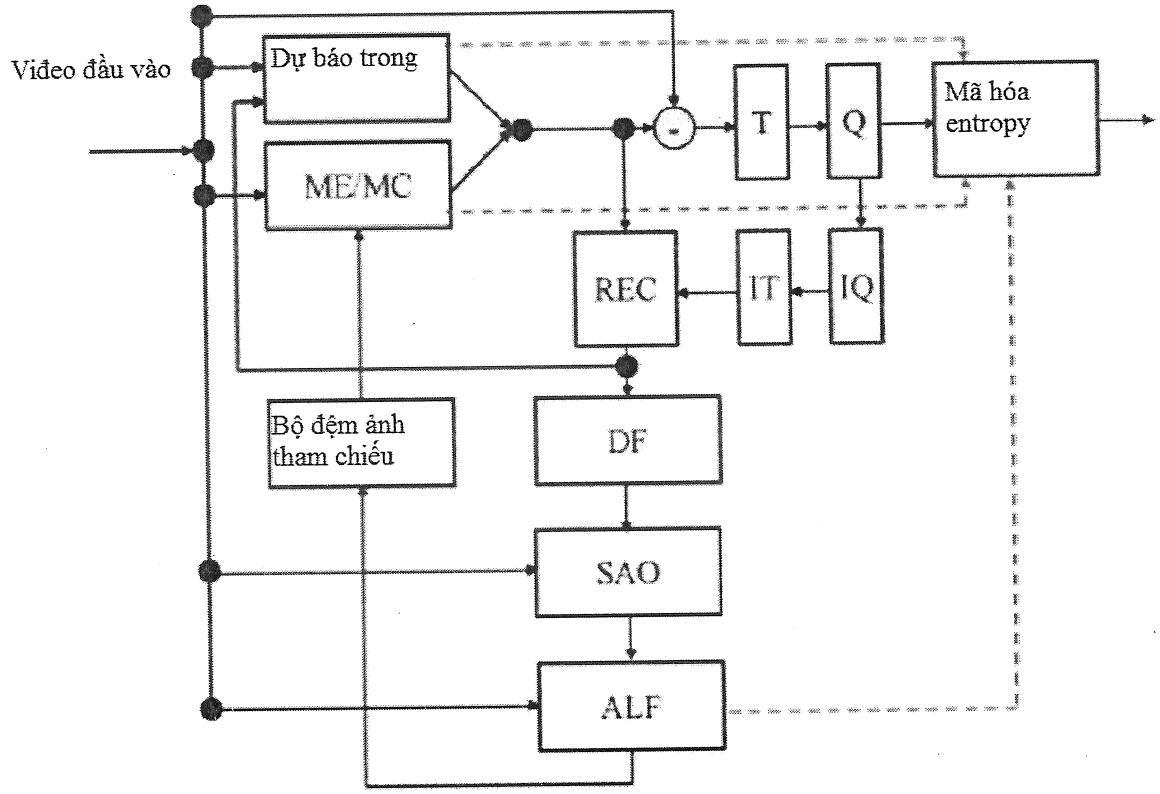

Viđeo đầu vào
Dự báo trong
ME/MC
T
Q
Mã hóa entropy
REC
IT
IQ
Bộ đệm ảnh tham chiếu
DF
SAO
ALF

Fig.5

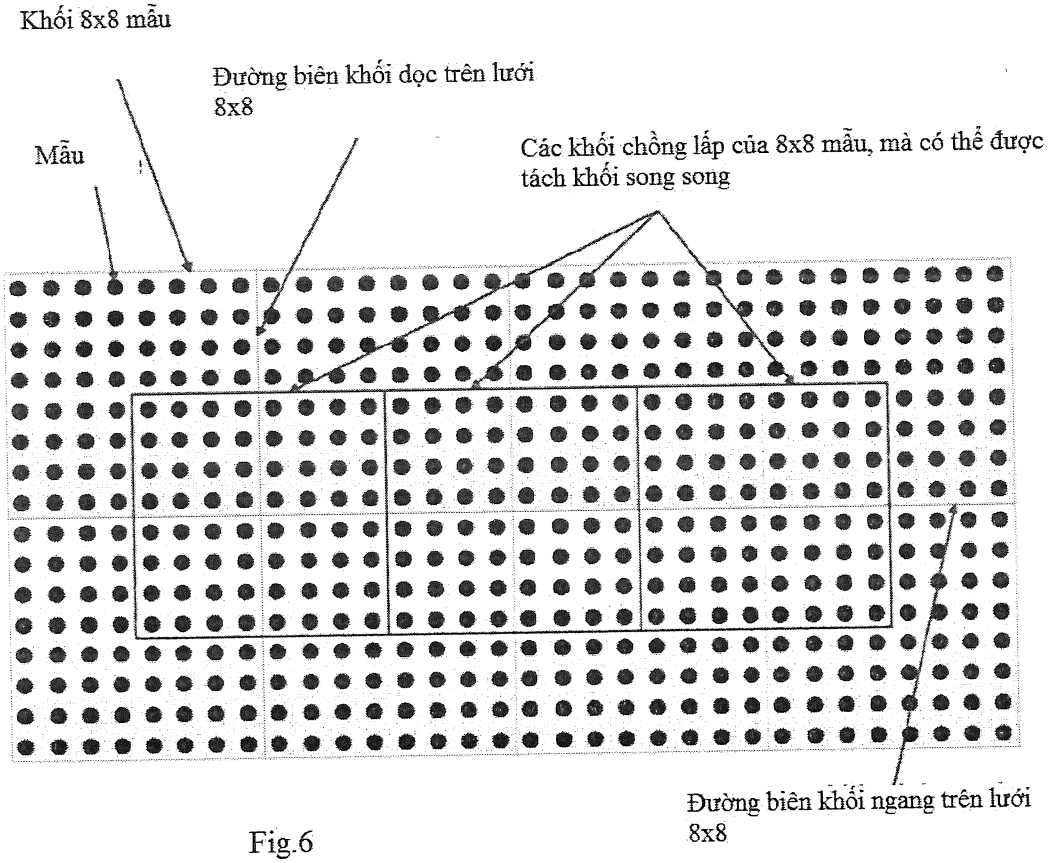

Khối 8x8 mẫu
Đường biên khối dọc trên lưới 8x8
Mẫu
Các khối chồng lấp của 8x8 mẫu, mà có thể được tách khối song song
Đường biên khối ngang trên lưới 8x8

Fig.6

$p3_0$	$p2_0$	$p1_0$	$p0_0$	$q0_0$	$q1_0$	$q2_0$	$q3_0$
$p3_1$	$p2_1$	$p1_1$	$p0_1$	$q0_1$	$q1_1$	$q2_1$	$q3_1$
$p3_2$	$p2_2$	$p1_2$	$p0_2$	$q0_2$	$q1_2$	$q2_2$	$q3_2$
$p3_3$	$p2_3$	$p1_3$	$p0_3$	$q0_3$	$q1_3$	$q2_3$	$q3_3$
$p3_4$	$p2_4$	$p1_4$	$p0_4$	$q0_4$	$q1_4$	$q2_4$	$q3_4$
$p3_5$	$p2_5$	$p1_5$	$p0_5$	$q0_5$	$q1_5$	$q2_5$	$q3_5$
$p3_6$	$p2_6$	$p1_6$	$p0_6$	$q0_6$	$q1_6$	$q2_6$	$q3_6$
$p3_7$	$p2_7$	$p1_7$	$p0_7$	$q0_7$	$q1_7$	$q2_7$	$q3_7$

4 đòng thứ nhất

4 đòng thứ hai

Fig.7

8/36

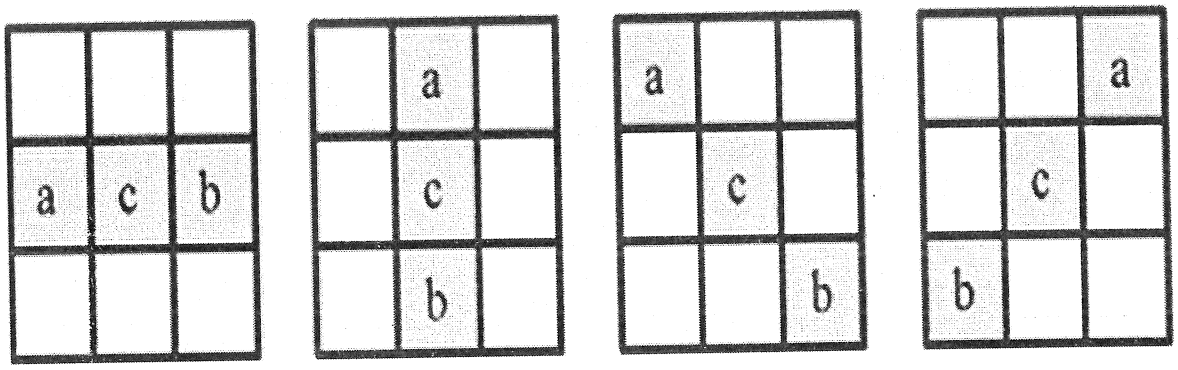


Fig.8

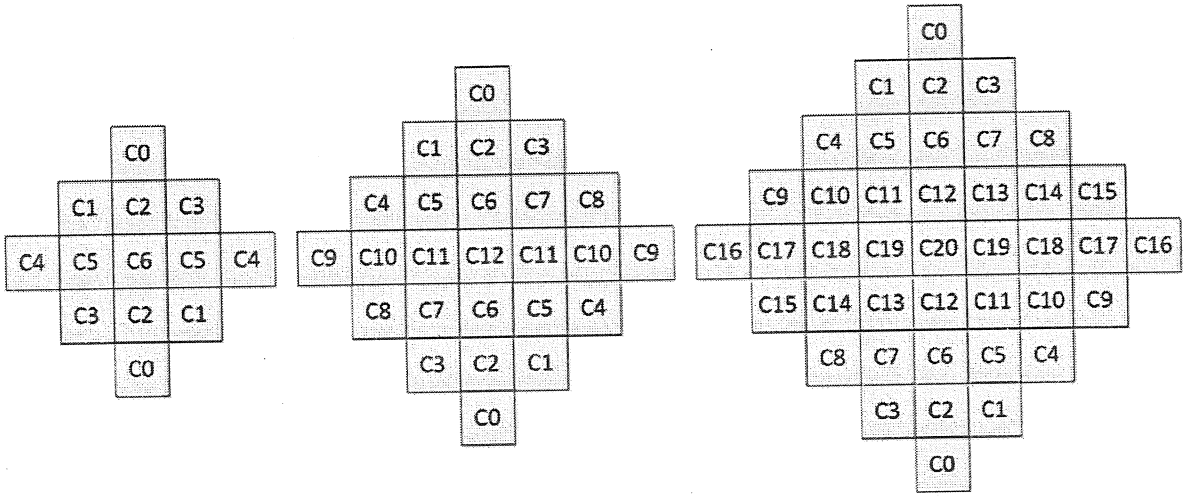

C0
C1 C2 C3
C4 C5 C6 C5 C4
C3 C2 C1
C0
C0
C1 C2 C3
C4 C5 C6 C7 C8
C9 C10 C11 C12 C11 C10 C9
C8 C7 C6 C5 C4
C3 C2 C1
C0
C0
C1 C2 C3
C4 C5 C6 C7 C8
C9 C10 C11 C12 C13 C14 C15
C16 C17 C18 C19 C20 C19 C18 C17 C16
C15 C14 C13 C12 C11 C10 C9
C8 C7 C6 C5 C4
C3 C2 C1
C0

Fig.9

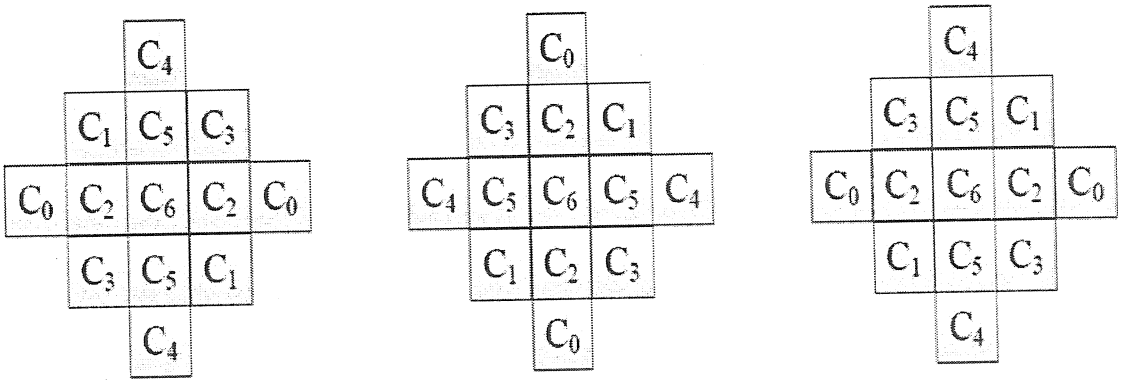

C4
C1 C5 C3
C0 C2 C6 C2 C0
C3 C5 C1
C4
C0
C3 C2 C1
C4 C5 C6 C5 C4
C1 C2 C3
C0
C4
C3 C5 C1
C0 C2 C6 C2 C0
C1 C5 C3
C4

Fig.10

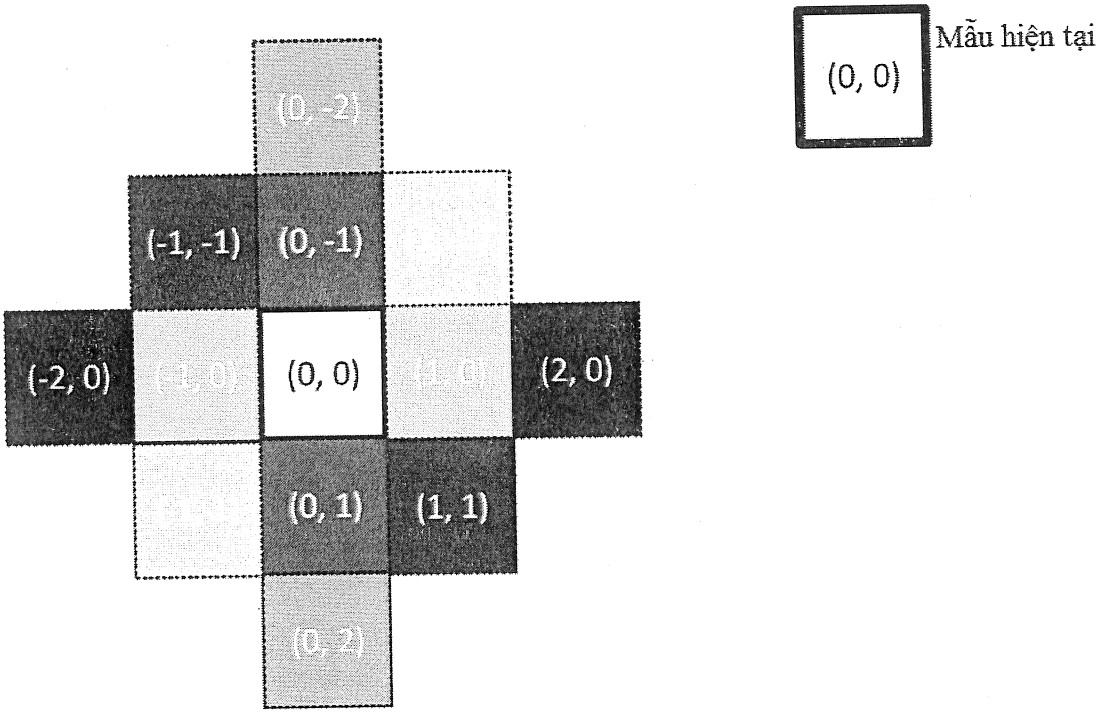


Fig.11

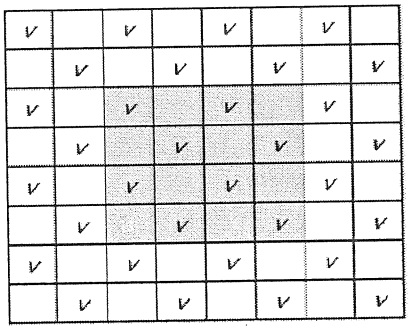


Fig.12A

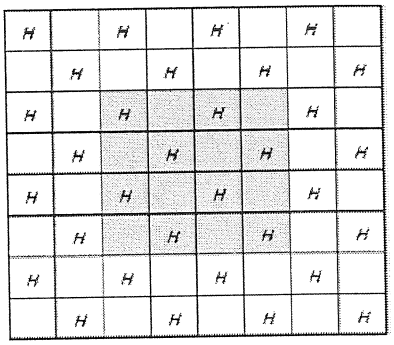


Fig.12B

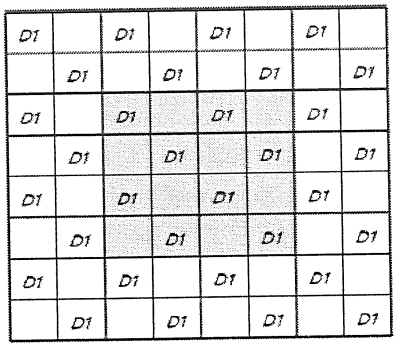


Fig.12C

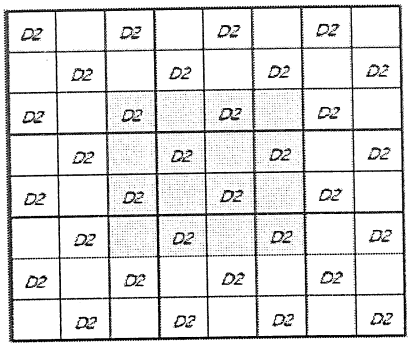


Fig.12D

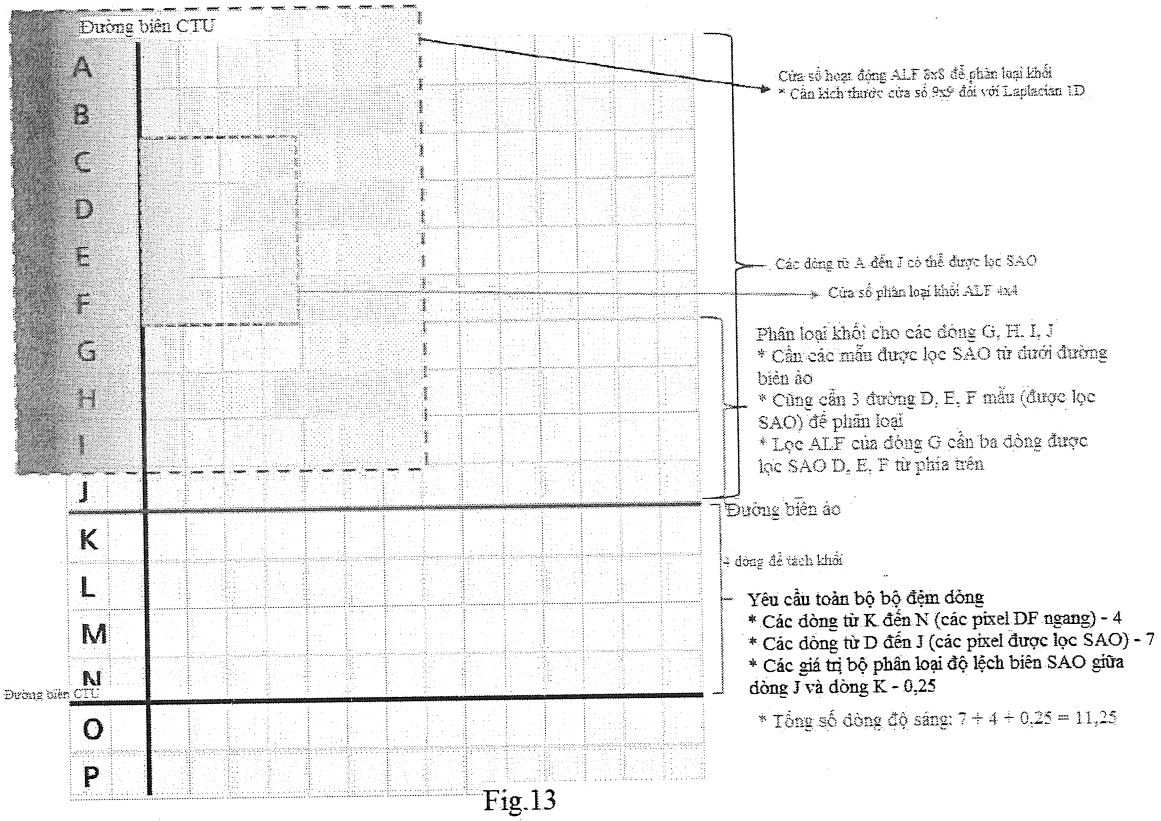

Đường biên CTU
A
B
C
D
E
F
G
H
I
J
K
L
M
N
Đường biên CTU
O
P
Cửa sổ hoạt động ALF 8x8 để phân loại khối
* Cần kích thước cửa sổ 9x9 đối với Laplacian 1D
Các dòng từ A đến J có thể được lọc SAO
Cửa sổ phân loại khối ALF 4x4
Phân loại khối cho các dòng G, H, I, J
* Cần các mẫu được lọc SAO từ dưới đường biên ảo
* Cũng cần 3 đường D, E, F mẫu (được lọc SAO) để phân loại
* Lọc ALF của dòng G cần ba dòng được lọc SAO D, E, F từ phía trên
Đường biên ảo
4 dòng để tách khối
Yêu cầu toàn bộ bộ đệm dòng
* Các dòng từ K đến N (các pixel DF ngang) - 4
* Các dòng từ D đến J (các pixel được lọc SAO) - 7
* Các giá trị bộ phân loại độ lệch biên SAO giữa dòng J và dòng K - 0,25
* Tổng số dòng độ sáng: 7 + 4 + 0,25 = 11,25

Fig.13

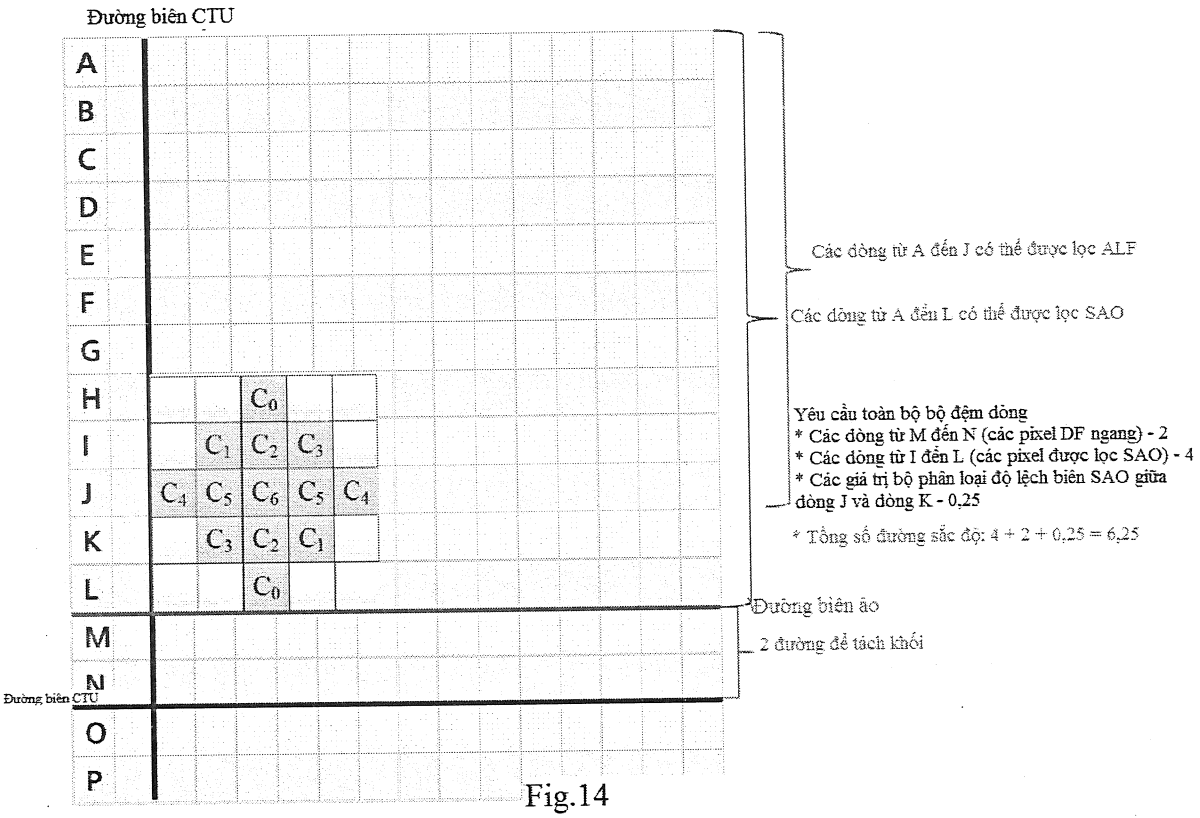

Đường biên CTU
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
C0
C1
C2
C3
C4
C5
C6
Các dòng từ A đến J có thể được lọc ALF
Các dòng từ A đến L có thể được lọc SAO
Yêu cầu toàn bộ bộ đệm dòng
* Các dòng từ M đến N (các pixel DF ngang) - 2
* Các dòng từ I đến L (các pixel được lọc SAO) - 4
* Các giá trị bộ phân loại độ lệch biên SAO giữa dòng J và dòng K - 0,25
* Tổng số đường sắc độ: 4 + 2 + 0,25 = 6,25
Đường biên ảo
2 đường để tách khối
Đường biên CTU

Fig.14

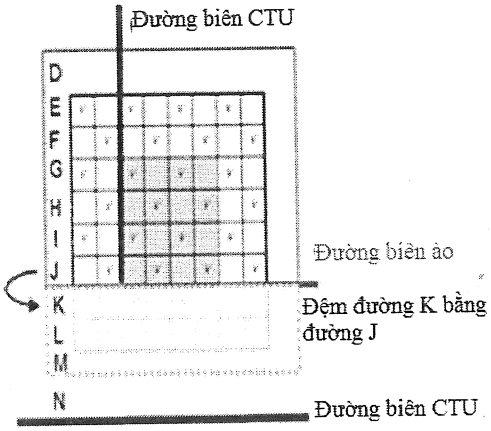


Fig.15A

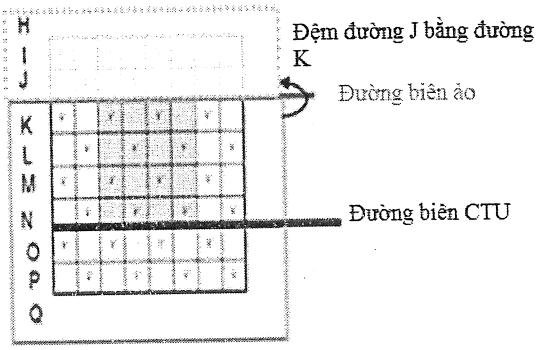


Fig.15B

16/36

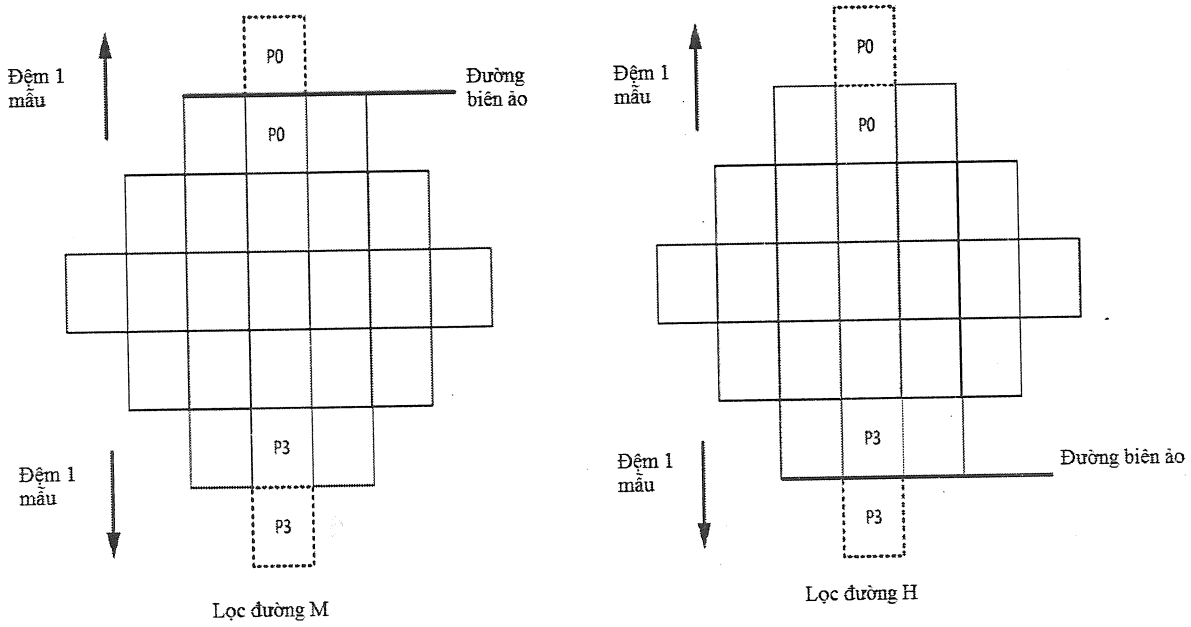

Đệm 1 mẫu
P0
Đường biên ảo
P0
P3
Đệm 1 mẫu
P3
Lọc đường M
Đệm 1 mẫu
P0
P0
P3
Đệm 1 mẫu
Đường biên ảo
P3
Lọc đường H

Fig.16A

17/36

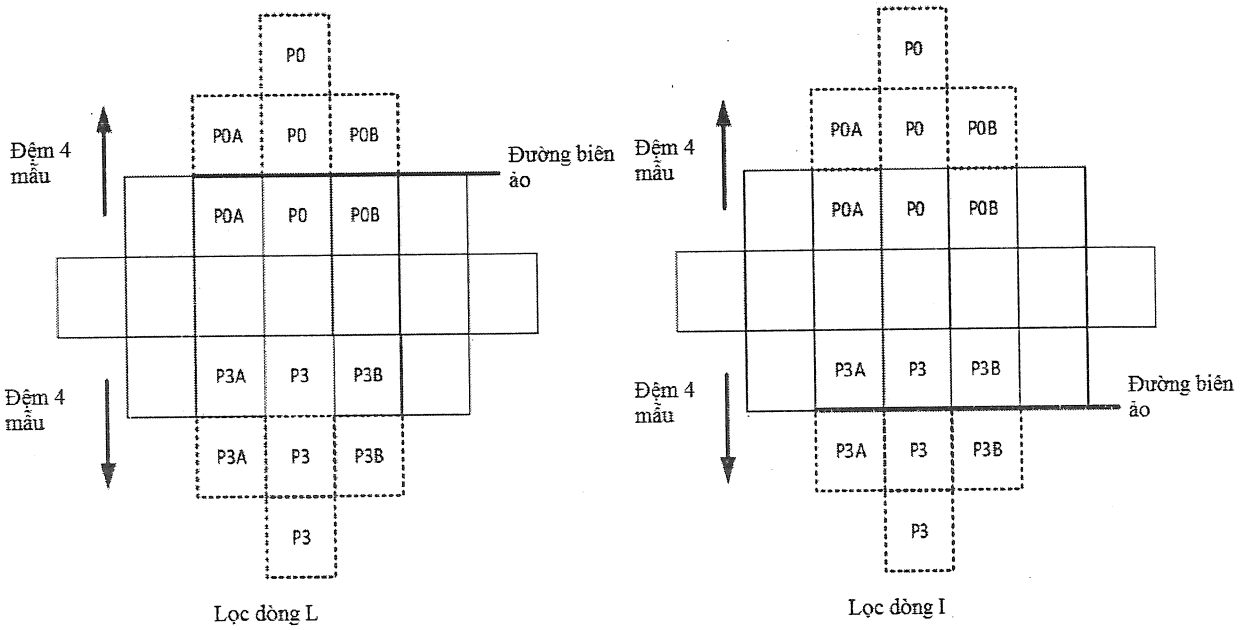

P0
P0A
P0
P0B
Đệm 4 mẫu
Đường biên ảo
P0A
P0
P0B
P3A
P3
P3B
Đệm 4 mẫu
P3A
P3
P3B
P3
Lọc dòng L
P0
P0A
P0
P0B
Đệm 4 mẫu
P0A
P0
P0B
P3A
P3
P3B
Đệm 4 mẫu
Đường biên ảo
P3A
P3
P3B
P3
Lọc dòng I

Fig.16B

18/36

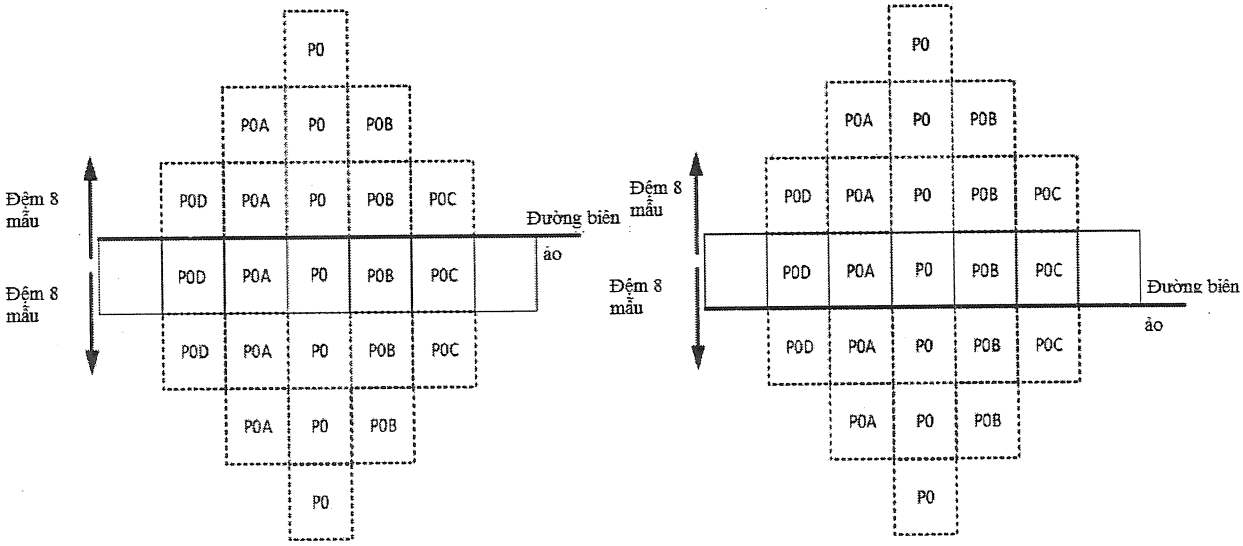

P0
P0A
P0B
P0D
P0C
Đệm 8 mẫu
Đệm 8 mẫu
Đường biên ảo
Lọc dòng J
Lọc dòng K

Fig.16C

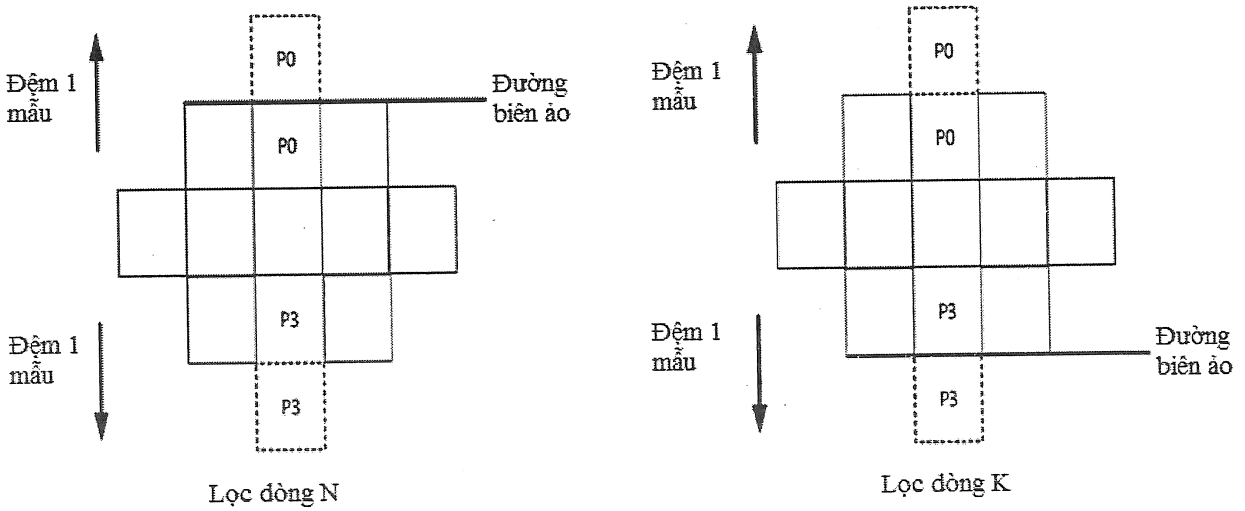


Fig.17A

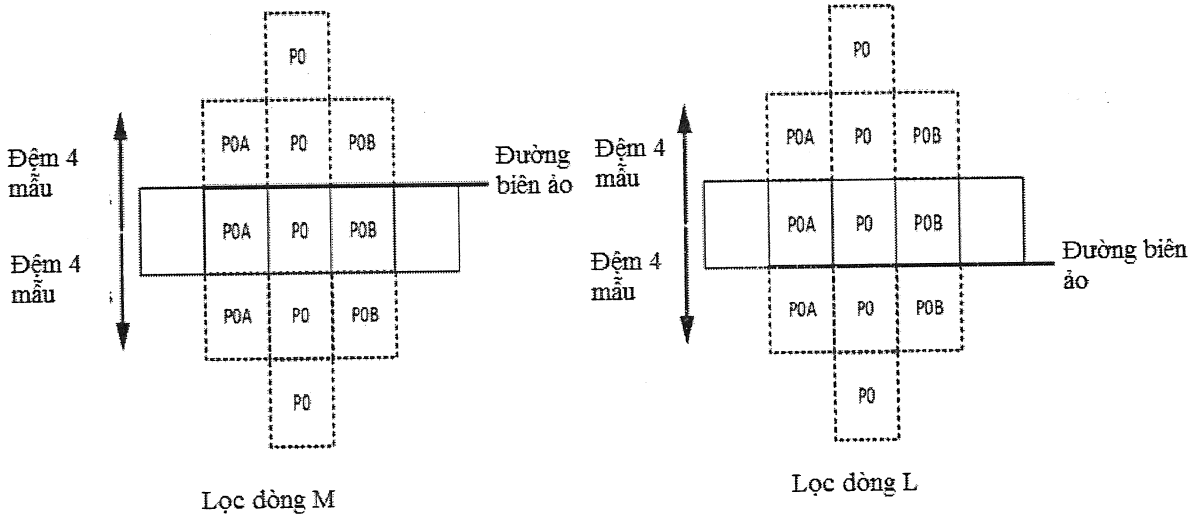
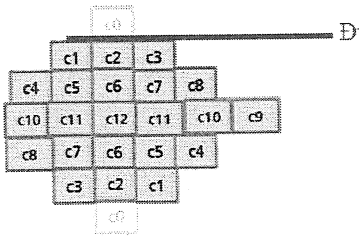
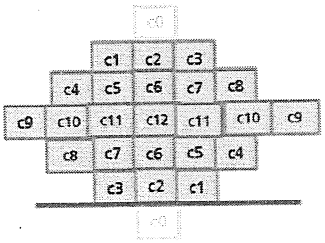


Fig.17B

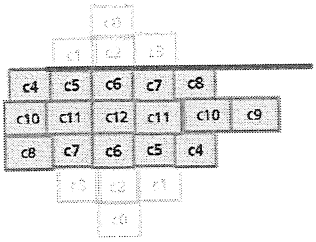


$c2' = c2 + c0$

Lọc dòng M



Lọc dòng H

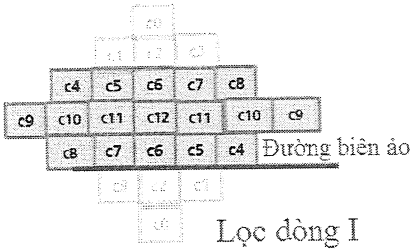


$c5' = c5 + c1$

$c6' = c6 + c2 + c0$

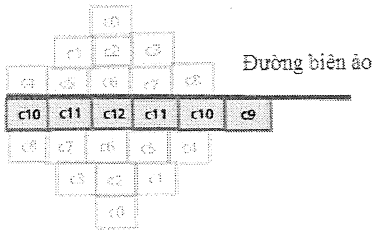
$c7' = c7 + c3$

Lọc dòng L

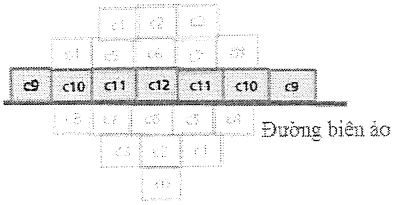


Lọc dòng I

Fig.18A



Lọc dòng J



Lọc dòng K

$$c10' = c10 + c4 + c8$$
$$c11' = c11 + c5 + c7 + c1 + c3$$
$$c12' = c12 + 2 * c6 + 2 * c2 + 2 * c0$$

Fig.18B

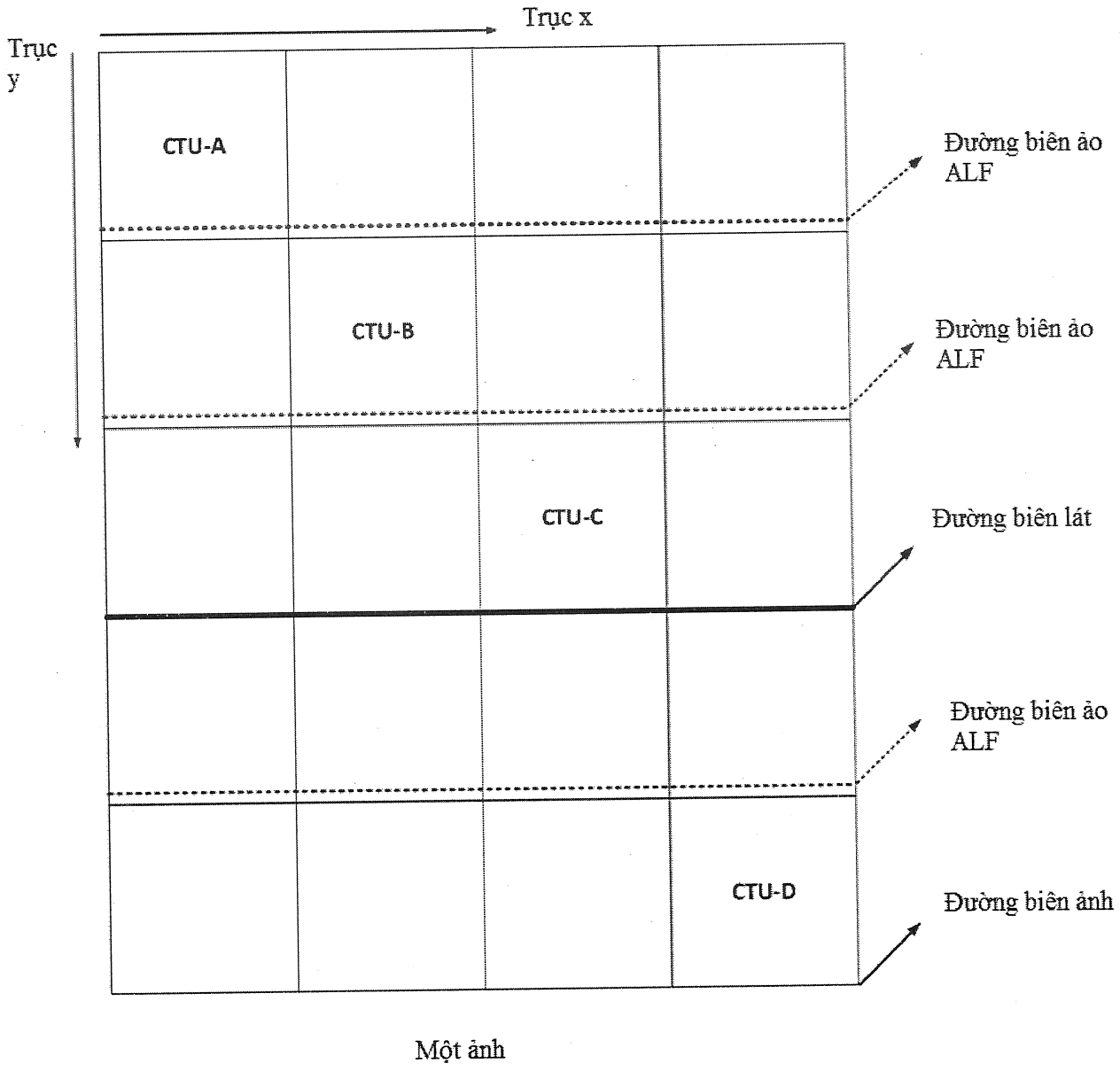

Trục x
Trục y
CTU-A
CTU-B
CTU-C
CTU-D
Đường biên ảo ALF
Đường biên ảo ALF
Đường biên lát
Đường biên ảo ALF
Đường biên ảnh
Một ảnh

Fig.19

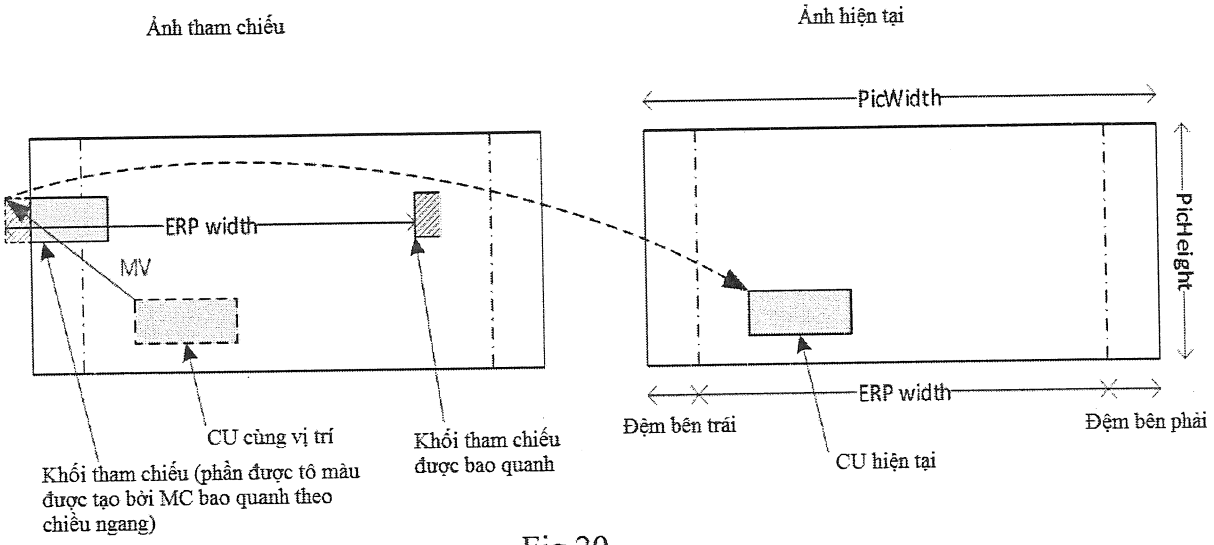

Ảnh tham chiếu
Ảnh hiện tại
PicWidth
PicHeight
ERP width
MV
CU cùng vị trí
Khối tham chiếu (phần được tô màu được tạo bởi MC bao quanh theo chiều ngang)
Khối tham chiếu được bao quanh
Đệm bên trái
ERP width
CU hiện tại
Đệm bên phải

Fig.20

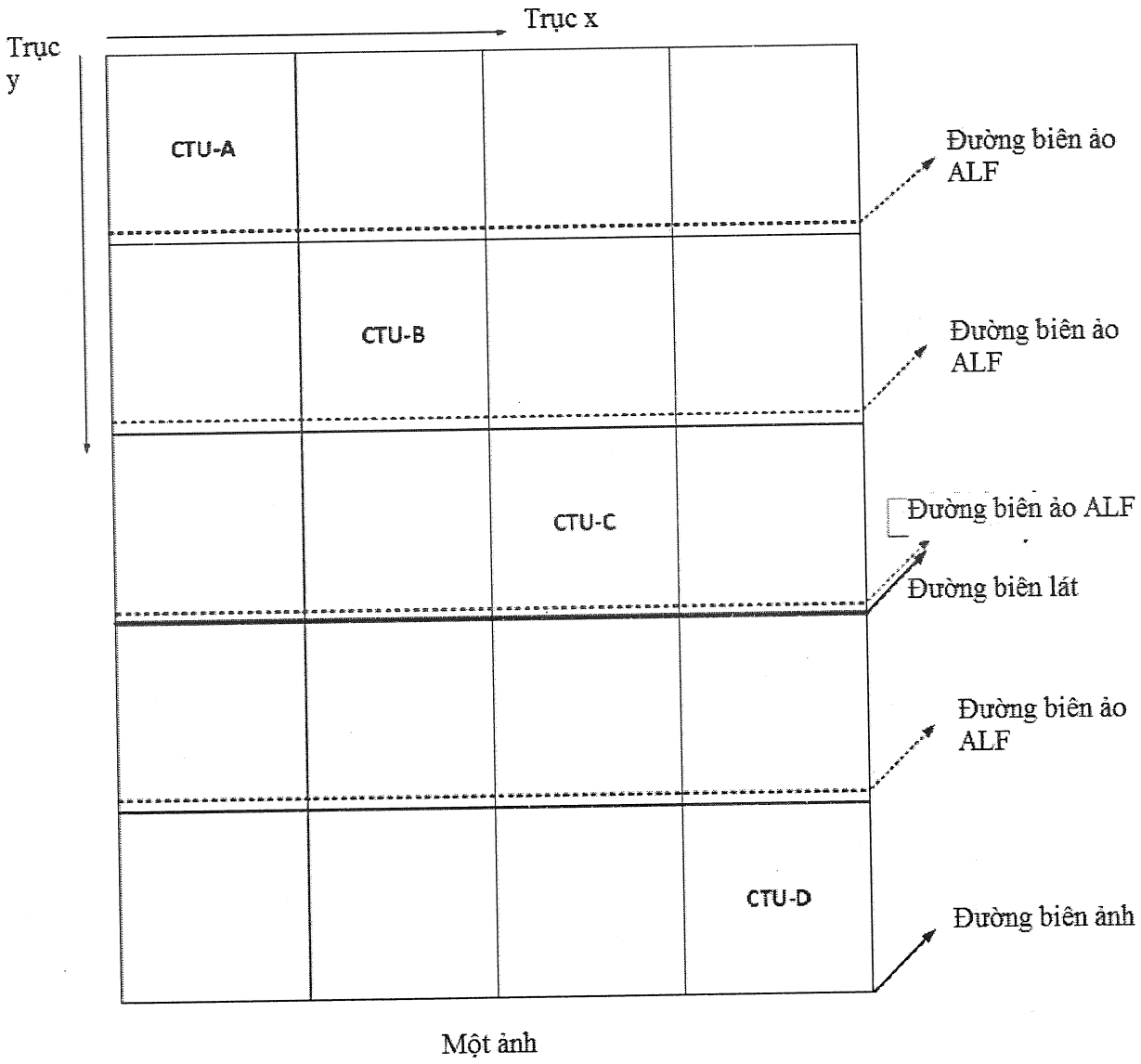

Trục x
Trục y
CTU-A
CTU-B
CTU-C
CTU-D
Đường biên ảo ALF
Đường biên ảo ALF
Đường biên ảo ALF
Đường biên lát
Đường biên ảo ALF
Đường biên ảnh
Một ảnh

Fig.21

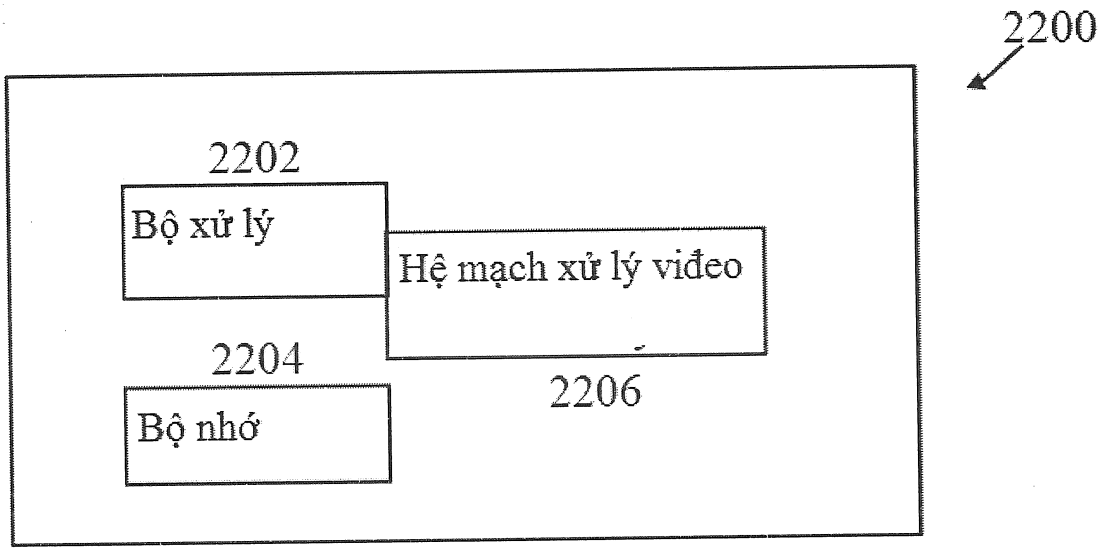


Fig.22

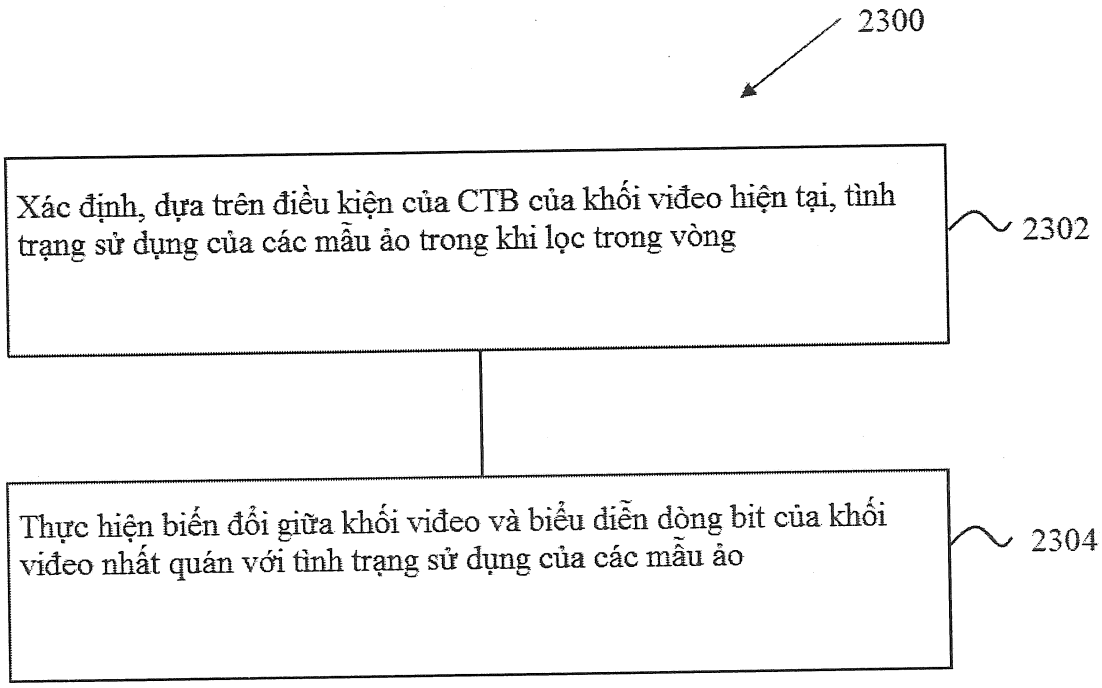

2300
Xác định, dựa trên điều kiện của CTB của khối viđeo hiện tại, tình trạng sử dụng của các mẫu ảo trong khi lọc trong vòng
2302
Thực hiện biến đổi giữa khối viđeo và biểu diễn dòng bit của khối viđeo nhất quán với tình trạng sử dụng của các mẫu ảo
2304

Fig.23

27/36

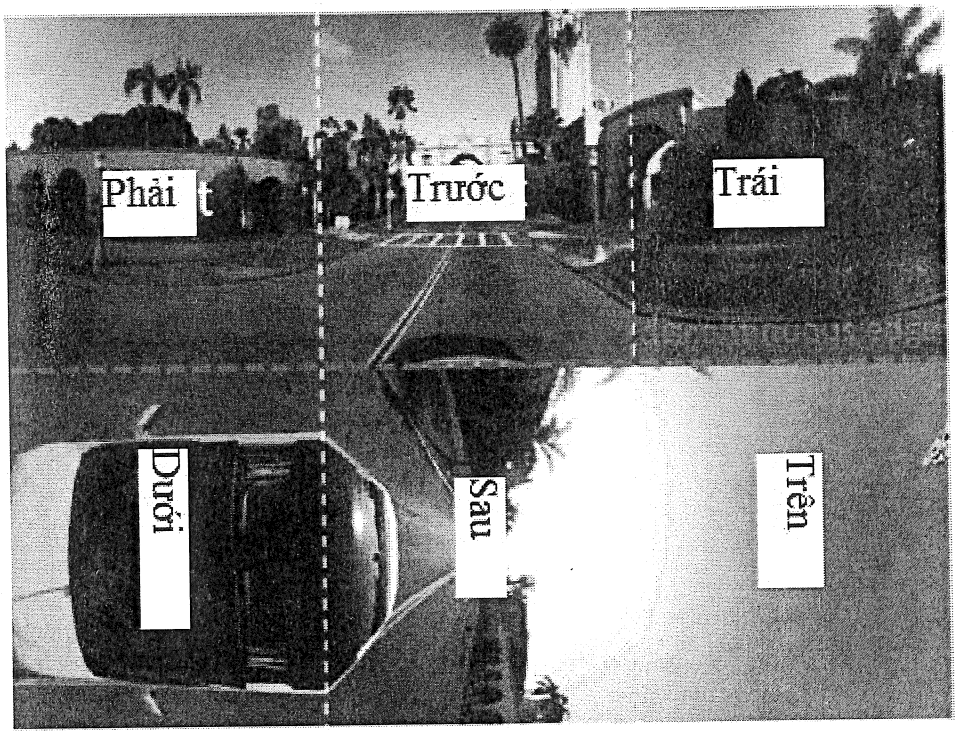


Fig.24

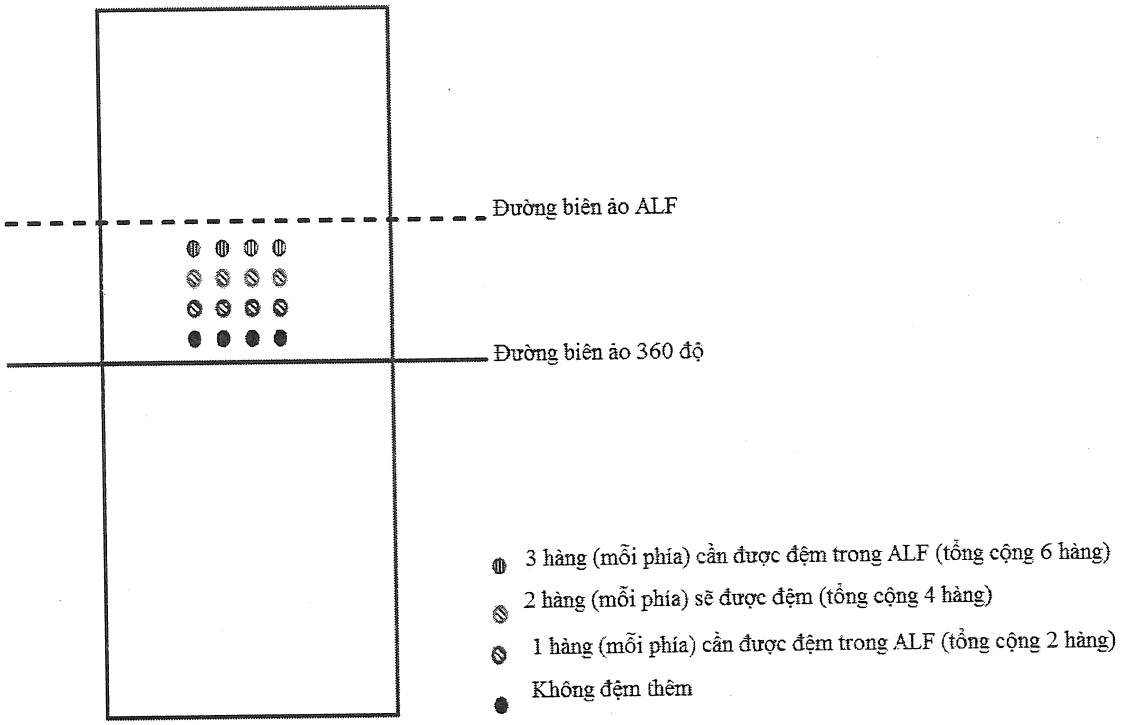
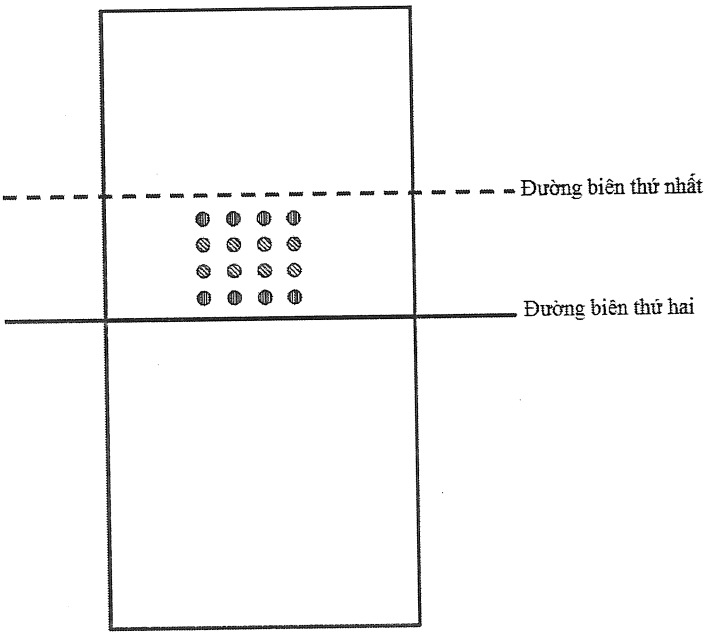
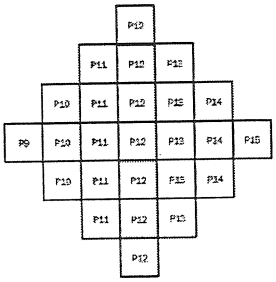

Đường biên ảo ALF
Đường biên ảo 360 độ
3 hàng (mỗi phía) cần được đệm trong ALF (tổng cộng 6 hàng)
2 hàng (mỗi phía) sẽ được đệm (tổng cộng 4 hàng)
1 hàng (mỗi phía) cần được đệm trong ALF (tổng cộng 2 hàng)
Không đệm thêm

Fig.25


Đường biên thứ nhất
Đường biên thứ hai
6 hàng cần được đệm trong ALF
4 hàng cần được đệm trong ALF


6 hàng được đệm trong ALF

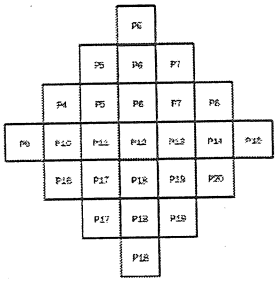

4 hàng được đệm trong ALF

Fig.26

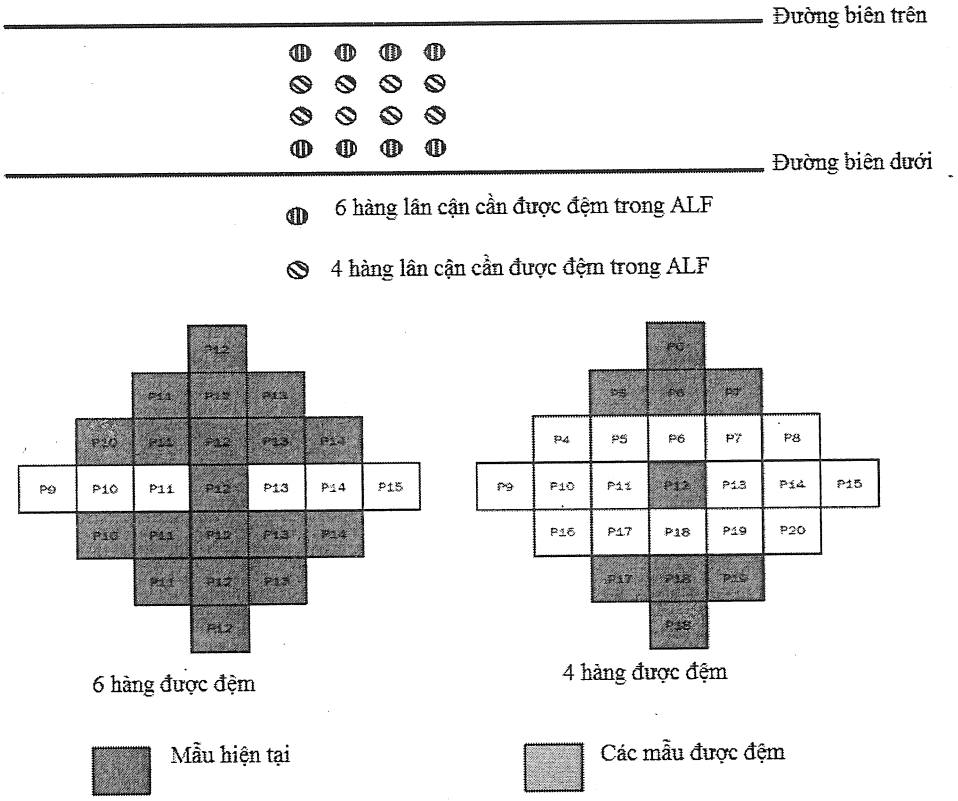

Đường biên trên
Đường biên dưới
6 hàng lân cận cần được đệm trong ALF
4 hàng lân cận cần được đệm trong ALF
6 hàng được đệm
4 hàng được đệm
Mẫu hiện tại
Các mẫu được đệm

Fig.27

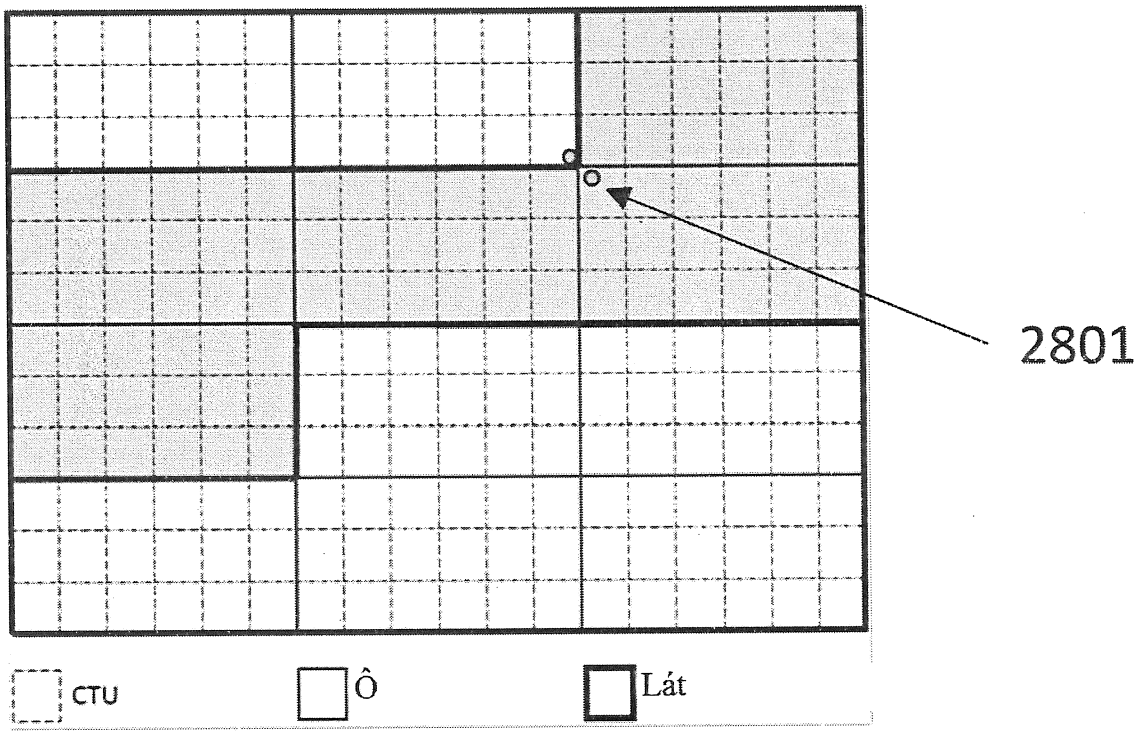


Fig.28

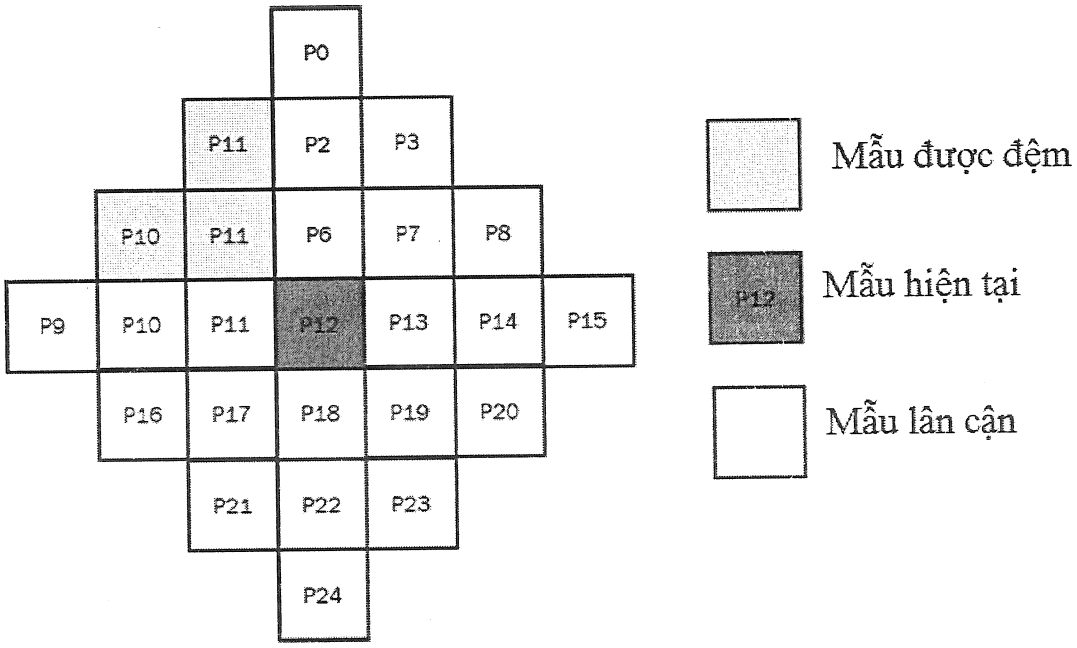

P0
P11
P2
P3
P10
P11
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
P16
P17
P18
P19
P20
P21
P22
P23
P24
Mẫu được đệm
P12
Mẫu hiện tại
Mẫu lân cận

Fig.29

33/36

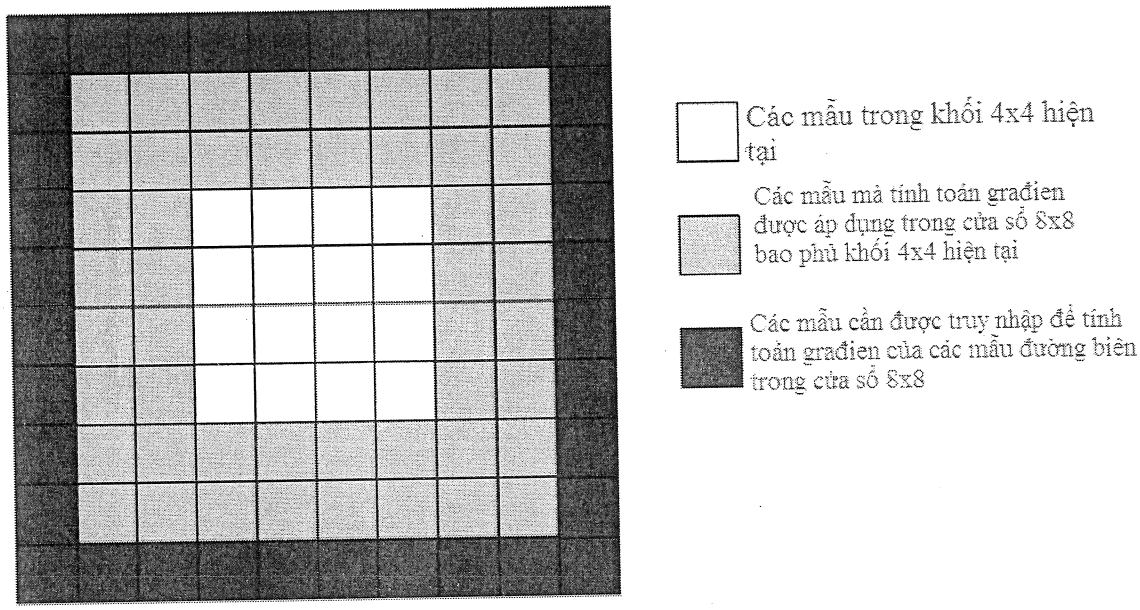

Các mẫu trong khối 4x4 hiện tại
Các mẫu mà tính toán gradien được áp dụng trong cửa sổ 8x8 bao phủ khối 4x4 hiện tại
Các mẫu cần được truy nhập để tính toán gradien của các mẫu đường biên trong cửa sổ 8x8

Fig.30

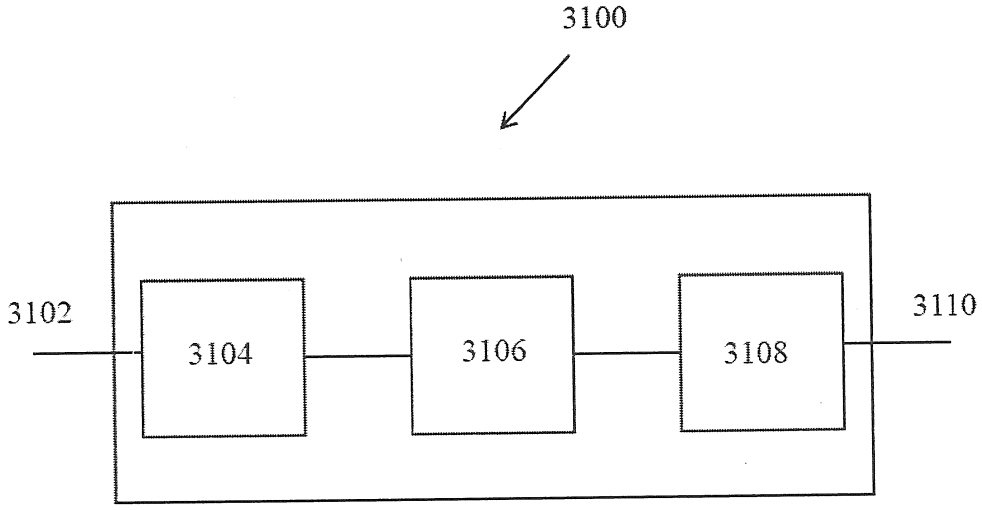


Fig.31

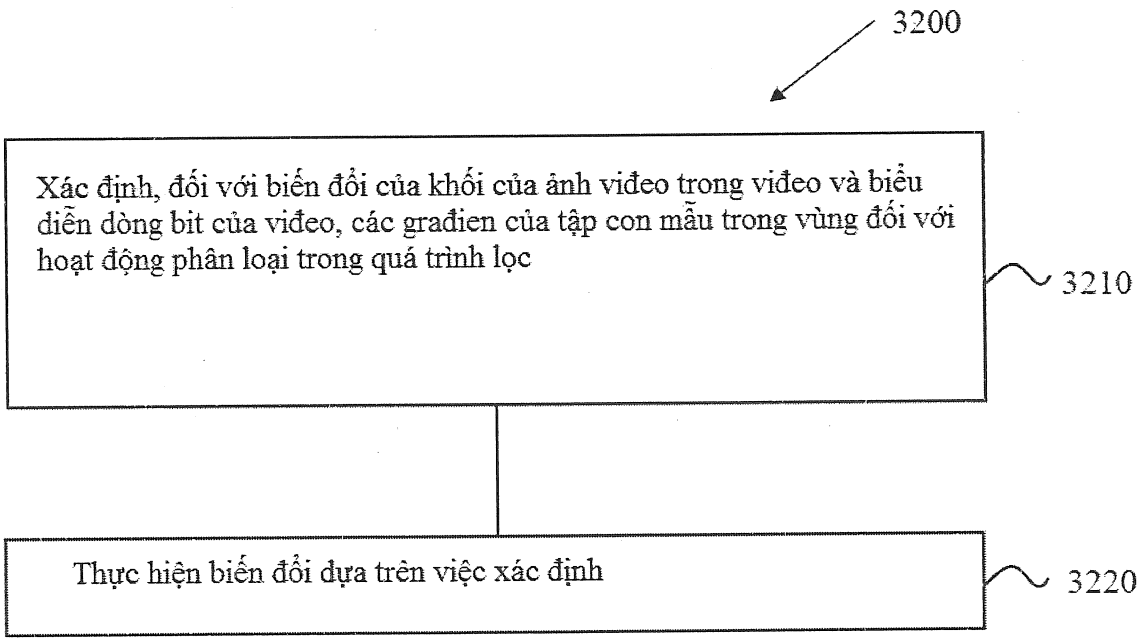

3200
Xác định, đối với biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn dòng bit của viđeo, các gradien của tập con mẫu trong vùng đối với hoạt động phân loại trong quá trình lọc
3210
Thực hiện biến đổi dựa trên việc xác định
3220

Fig.32

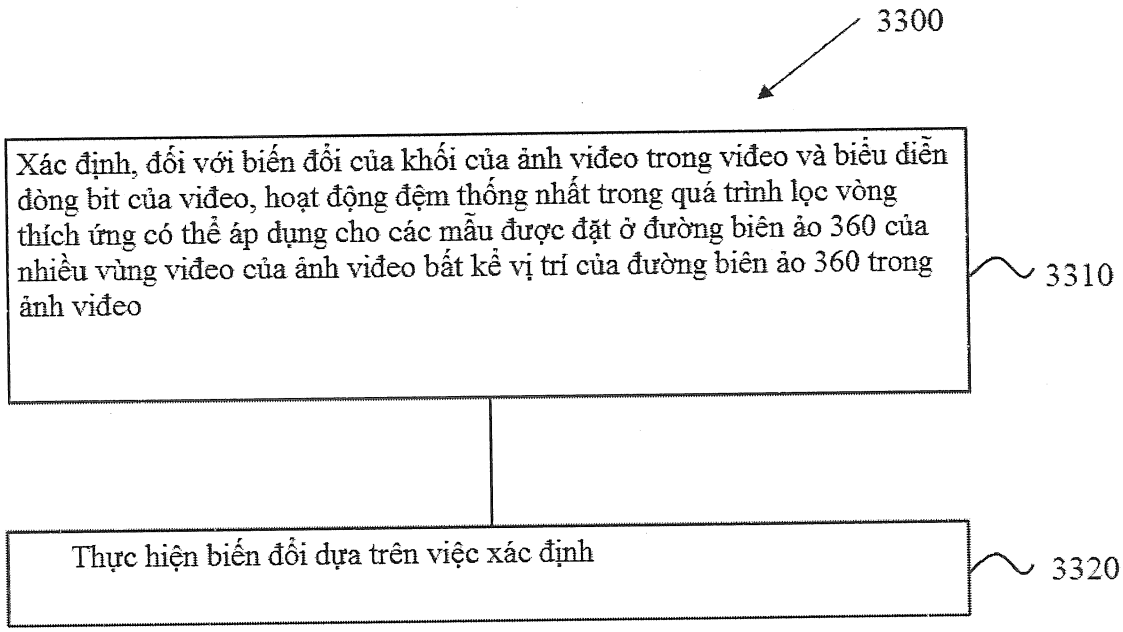

3300
Xác định, đối với biến đổi của khối của ảnh viđeo trong viđeo và biểu diễn đòng bit của viđeo, hoạt động đệm thống nhất trong quá trình lọc vòng thích ứng có thể áp dụng cho các mẫu được đặt ở đường biên ảo 360 của nhiều vùng viđeo của ảnh viđeo bất kể vị trí của đường biên ảo 360 trong ảnh viđeo
3310
Thực hiện biến đổi dựa trên việc xác định
3320

Fig.33