(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) **1-0048272**

(51)[2020.01] **H04N 19/91**; H04N 19/137; H04N 19/44; H04N 19/70; H04N 19/132; H04N 19/176

(13) **B**

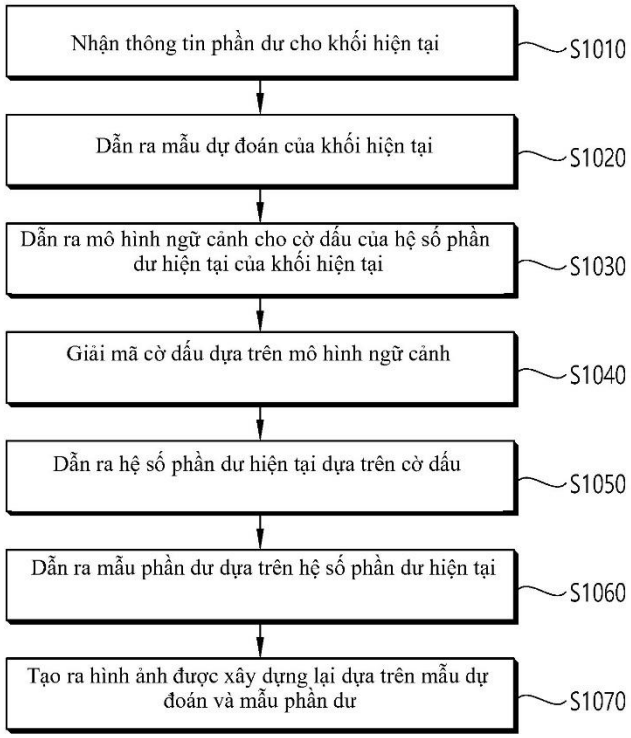
(21) 1-2021-05755
(22) 04/03/2020
(86) PCT/KR2020/003409 04/03/2020
(87) WO2021/180102 10/09/2020
(30) 62/813,662 04/03/2019 US
(45) 25/07/2025 448
(43) 25/11/2021 404A
(73) LG ELECTRONICS INC. (KR)
128, Yeoui-daero, Yeongdeungpo-Gu Seoul 07336, Korea
(72) YOO, Sunmi (KR); HEO, Jin (KR); CHOI, Jungah (KR); KIM, Seunghwan (KR).
(74) Công ty Luật TNHH T&G (TGVN)

(54) PHƯƠNG PHÁP GIẢI MÃ VÀ MÃ HÓA ẢNH, PHƯƠNG PHÁP TRUYỀN DỮ LIỆU CHO ẢNH VÀ PHƯƠNG TIỆN LƯU TRỮ PHI CHUYỂN TIẾP ĐỌC ĐƯỢC BẰNG MÁY TÍNH

(21) 1-2021-05755

(57) Sáng chế đề cập đến phương pháp giải mã ảnh được thực hiện bởi thiết bị giải mã, phương pháp mã hóa ảnh được thực hiện bởi thiết bị mã hóa, phương pháp truyền dữ liệu cho ảnh và phương tiện lưu trữ phi chuyển tiếp đọc được bằng máy tính. Phương pháp giải mã ảnh theo sáng chế bao gồm các bước: dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại của khối hiện tại; và giải mã cờ dấu trên cơ sở mô hình ngữ cảnh, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra trên cơ sở cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại.

FIG. 10



Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập đến kỹ thuật lập mã ảnh, và cụ thể hơn là đến phương pháp giải mã ảnh trong đó mô hình ngữ cảnh của cờ dấu chỉ ra dấu của hệ số phần dư được dẫn ra dựa trên cờ dấu của hệ số phần dư được lập mã trước đó trong hệ thống lập mã ảnh và cờ dấu này được lập mã dựa trên mô hình ngữ cảnh được dẫn ra này, và thiết bị cho phương pháp này.

Tình trạng kỹ thuật của sáng chế

Gần đây, nhu cầu đối với các ảnh độ phân giải cao, chất lượng cao, chẳng hạn các ảnh độ phân giải cao (High Definition, HD) và các ảnh độ phân giải siêu cao (Ultra High Definition, UHD), đang ngày càng gia tăng trong các lĩnh vực khác nhau. Khi dữ liệu ảnh có độ phân giải cao và chất lượng cao, lượng thông tin hoặc bit cần được truyền tăng lên so với dữ liệu ảnh di sản để lại. Do đó, khi dữ liệu ảnh được truyền nhờ sử dụng phương tiện chẳng hạn như đường băng rộng có dây/không dây thông thường hoặc dữ liệu ảnh được lưu trữ nhờ sử dụng phương tiện lưu trữ hiện có, chi phí truyền và chi phí lưu trữ dữ liệu ảnh bị tăng lên.

Theo đó, cần có kỹ thuật nén ảnh hiệu quả cao để truyền, lưu trữ, và tái tạo một cách hiệu quả thông tin về các ảnh có độ phân giải cao và chất lượng cao.

Bản chất kỹ thuật của sáng chế

Sáng chế đề xuất phương pháp và thiết bị để cải thiện hiệu suất lập mã ảnh.

Sáng chế cũng đề xuất phương pháp và thiết bị để cải thiện hiệu suất lập mã phần dư.

Sáng chế cũng đề xuất phương pháp và thiết bị để thực hiện việc lập mã bằng cách dẫn ra mô hình ngữ cảnh của cờ dấu chỉ ra dấu của hệ số phần dư, dựa trên cờ dấu của hệ số phần dư lân cận trước đó được lập mã trước hệ số phần dư này, khi thông tin phần

dư được lập mã.

Theo một phương án, sáng chế đề xuất phương pháp giải mã ảnh được thực hiện bởi thiết bị giải mã. Phương pháp này bao gồm các bước dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại của khối hiện tại, giải mã cờ dấu dựa trên mô hình ngữ cảnh, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại.

Theo một phương án khác, sáng chế đề xuất thiết bị giải mã để thực hiện việc giải mã ảnh. Thiết bị giải mã này bao gồm bộ giải mã entropy để dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại của khối hiện tại, giải mã cờ dấu dựa trên mô hình ngữ cảnh, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại.

Theo một phương án khác, sáng chế đề xuất phương pháp mã hóa ảnh được thực hiện bởi thiết bị mã hóa. Phương pháp này bao gồm các bước dẫn ra hệ số phần dư hiện tại của khối hiện tại, dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại, và mã hóa cờ dấu dựa trên mô hình ngữ cảnh, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

Theo một phương án khác, sáng chế đề xuất thiết bị mã hóa viđeo. Thiết bị mã hóa này bao gồm bộ mã hóa entropy để dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại, mã hóa cờ dấu dựa trên mô hình ngữ cảnh, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

Theo sáng chế này, hiệu suất lập mã phần dư có thể được cải thiện.

Theo sáng chế này, cờ dấu chỉ ra dấu của hệ số phần dư có thể được lập mã dựa trên mô hình ngữ cảnh, qua đó tiết kiệm lượng bit được gán cho cờ dấu cho hệ số phần dư và cải thiện hiệu suất lập mã phần dư tổng thể.

Theo sáng chế này, mô hình ngữ cảnh cho cờ dấu chỉ ra dấu của hệ số phần dư được dẫn ra dựa trên dấu của hệ số phần dư lân cận liền kề hệ số phần dư này, thông qua điều này, cờ dấu được lập mã có xem xét đến mối tương quan giữa các các hệ số phần

dư liền kề, lượng bit được cấp phát cho cờ dấu có thể được giảm xuống, và hiệu suất lập mã phần dư tổng thể có thể được cải thiện.

Theo sáng chế này, mô hình ngữ cảnh cho cờ dấu chỉ ra dấu của hệ số phần dư được dẫn ra dựa trên dấu của hệ số phần dư lân cận được giải mã trước hệ số phần dư này, và do đó cờ dấu được lập mã qua việc xem xét mối tương quan giữa các hệ số phần dư liền kề, qua đó tiết kiệm lượng bit được gán cho cờ dấu và cải thiện hiệu suất lập mã phần dư tổng thể.

Mô tả vắn tắt các hình vẽ

Fig.1 minh họa ngắn gọn ví dụ về thiết bị lập mã viđeo/ảnh mà các phương án của sáng chế có thể áp dụng được với nó.

Fig.2 là sơ đồ giản lược minh họa cấu hình của thiết bị mã hóa viđeo/ảnh mà phương án (các phương án) của sáng chế có thể được áp dụng với nó.

Fig.3 là sơ đồ giản lược minh họa cấu hình của thiết bị giải mã viđeo/ảnh mà phương án (các phương án) của sáng chế có thể được áp dụng với nó.

Fig.4 thể hiện làm ví dụ việc lập mã số học nhị phân thích ứng ngữ cảnh (context-adaptive binary arithmetic coding, CABAC) để mã hóa phần tử cú pháp.

Fig.5 là sơ đồ thể hiện các hệ số biến đổi làm ví dụ bên trong khối 4x4.

Fig.6 minh họa ví dụ về việc xác định mô hình ngữ cảnh của cờ dấu của hệ số phần dư hiện tại theo phương án hiện tại.

Fig.7 thể hiện làm ví dụ thiết bị mã hóa và thiết bị giải mã để lập mã cờ dấu của hệ số phần dư hiện tại dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không và dấu của hệ số phần dư được lập mã trước đó.

Fig.8 minh họa ngắn gọn phương pháp mã hóa ảnh được thực hiện bởi thiết bị mã hóa theo sáng chế này.

Fig.9 minh họa ngắn gọn thiết bị mã hóa để thực hiện phương pháp mã hóa ảnh theo sáng chế này.

Fig.10 minh họa ngắn gọn phương pháp giải mã ảnh được thực hiện bởi thiết bị

giải mã theo sáng chế này.

Fig.11 minh họa ngắn gọn thiết bị giải mã để thực hiện phương pháp giải mã ảnh theo sáng chế này.

Fig.12 minh họa sơ đồ cấu trúc của hệ thống tạo luồng các nội dung mà sáng chế được áp dụng cho hệ thống này.

Mô tả chi tiết sáng chế

Sáng chế có thể được cải biến dưới các dạng khác nhau, và các phương án cụ thể của chúng sẽ được mô tả và được minh họa trên các hình vẽ. Tuy nhiên, các phương án này không nhằm giới hạn sáng chế. Các thuật ngữ được sử dụng trong phần mô tả sau đây được sử dụng đơn thuần để mô tả các phương án cụ thể mà không nhằm giới hạn sáng chế. Cách nói số ít bao gồm cả các cách nói số nhiều, chừng nào nó rõ ràng vẫn được đọc theo cách khác đi. Các thuật ngữ chẳng hạn như "bao gồm" và "có" được dự tính là để chỉ ra rằng các dấu hiệu, số lượng, bước, hoạt động, phần tử, thành phần, hoặc các sự kết hợp của chúng được sử dụng trong phần mô tả sau đây là có hiện hữu và do đó cần phải hiểu rằng khả năng tồn tại hoặc bổ sung một hoặc nhiều dấu hiệu, số lượng, bước, hoạt động, phần tử, thành phần khác, hoặc các sự kết hợp của chúng là không bị loại trừ.

Trong khi đó, các phần tử trên các hình vẽ được mô tả trong sáng chế được vẽ một cách độc lập để thuận tiện cho việc giải thích các chức năng cụ thể khác nhau, và không nghĩa là các phần tử này được thực hiện bởi phần cứng độc lập hoặc phần mềm độc lập. Ví dụ, hai hay hơn hai phần tử trong số các phần tử này có thể được kết hợp để tạo ra phần tử đơn, hoặc một phần tử có thể được phân chia thành nhiều phần tử. Các phương án trong đó các phần tử được kết hợp và/hoặc được phân chia này thuộc về sáng chế mà không tách rời khỏi nguyên lý của sáng chế.

Ở dưới đây, các phương án của sáng chế này sẽ được mô tả chi tiết với tham chiếu đến các hình vẽ kèm theo. Thêm vào đó, các số tham chiếu giống nhau được sử dụng để chỉ các phần tử giống nhau xuyên suốt các hình vẽ, và phần mô tả giống nhau về các phần tử giống nhau này sẽ được lược bỏ.

Fig.1 minh họa ngắn gọn ví dụ về thiết bị lập mã viđeo/ảnh mà các phương án của

sáng chế có thể áp dụng được với nó.

Tham chiếu đến Fig.1, hệ thống lập mã viđeo/ảnh có thể bao gồm thiết bị thứ nhất (thiết bị nguồn) và thiết bị thứ hai (thiết bị nhận). Thiết bị nguồn có thể phân phối thông tin hoặc dữ liệu viđeo/ảnh được mã hóa dưới dạng tệp tin hoặc tạo luồng tới thiết bị nhận qua phương tiện lưu trữ kỹ thuật số hoặc mạng.

Thiết bị nguồn có thể bao gồm nguồn viđeo, thiết bị mã hóa, và bộ truyền. Thiết bị nhận có thể bao gồm bộ nhận, thiết bị giải mã, và bộ kết xuất. Thiết bị mã hóa có thể được gọi là thiết bị mã hóa viđeo/ảnh, và thiết bị giải mã có thể được gọi là thiết bị giải mã viđeo/ảnh. Bộ truyền có thể được chứa trong thiết bị mã hóa. Bộ nhận có thể được chứa trong thiết bị giải mã. Bộ kết xuất có thể bao gồm bộ hiển thị và bộ hiển thị này có thể được tạo cấu hình như thiết bị tách biệt hoặc thành phần bên ngoài.

Nguồn viđeo có thể thu nhận viđeo/ảnh thông qua quy trình chụp, tổng hợp, hoặc tạo ra viđeo/ảnh. Nguồn viđeo có thể bao gồm thiết bị chụp viđeo/ảnh và/hoặc thiết bị tạo ra viđeo/ảnh. Thiết bị chụp viđeo/ảnh có thể bao gồm, ví dụ, một hoặc nhiều camera, phần lưu trữ viđeo/ảnh chứa các viđeo/ảnh đã được chụp trước đó, và dạng tương tự. Thiết bị tạo ra viđeo/ảnh có thể bao gồm, ví dụ, máy tính, máy tính bảng, và điện thoại thông minh, và có thể tạo ra các viđeo/ảnh (theo cách điện tử). Ví dụ, viđeo/ảnh ảo có thể được tạo ra qua máy tính hoặc dạng tương tự. Trong trường hợp này, quy trình chụp viđeo/ảnh có thể được thay thế bởi quy trình tạo ra dữ liệu liên quan.

Thiết bị mã hóa có thể mã hóa ảnh đầu vào/ảnh. Thiết bị mã hóa có thể thực hiện chuỗi các thủ tục chẳng hạn như dự đoán, biến đổi, và lượng tử hóa để đạt được hiệu quả nén và lập mã. Dữ liệu được mã hóa (thông tin viđeo/ảnh được mã hóa) có thể được xuất ra dưới dạng luồng bit.

Bộ truyền có thể truyền ảnh/thông tin hoặc dữ liệu ảnh được mã hóa được xuất ra dưới dạng luồng bit đến bộ nhận của thiết bị nhận thông qua phương tiện lưu trữ kỹ thuật số hoặc mạng dưới dạng tệp tin hoặc tạo luồng. Phương tiện lưu trữ kỹ thuật số có thể bao gồm các phương tiện lưu trữ khác nhau như USB, SD, CD, DVD, Blu-ray, HDD, SSD, và loại tương tự. Bộ truyền có thể bao gồm phần tử để tạo ra tệp tin phương tiện thông qua định dạng tệp tin được xác định trước và có thể bao gồm phần tử để truyền

qua mạng phát quảng bá/truyền thông. Bộ nhận có thể nhận/trích xuất luồng bit và truyền luồng bit nhận được đến thiết bị giải mã.

Thiết bị giải mã có thể giải mã viđeo/ảnh bằng cách thực hiện chuỗi các thủ tục chẳng hạn như giải lượng tử hóa, biến đổi ngược, và dự đoán tương ứng với hoạt động của thiết bị mã hóa.

Bộ kết xuất có thể kết xuất viđeo/ảnh được giải mã. Viđeo/ảnh được kết xuất có thể được hiển thị qua bộ hiển thị.

Sáng chế đề cập đến việc lập mã viđeo/ảnh. Ví dụ, các phương pháp/các phương án được bộc lộ trong sáng chế có thể được áp dụng cho phương pháp được bộc lộ trong việc lập mã viđeo vạn năng (versatile video coding, VVC), tiêu chuẩn EVC (essential video coding, lập mã viđeo thiết yếu), tiêu chuẩn viđeo AOMedia 1 (AOMedia Video 1, AV1), thế hệ thứ hai của tiêu chuẩn lập mã viđeo auđio (2nd generation of audio video coding standard, AVS2), hoặc tiêu chuẩn lập mã viđeo/ảnh thế hệ tiếp theo (ví dụ, H.267 hoặc H.268, v.v.).

Sáng chế thể hiện các phương án khác nhau của việc lập mã viđeo/ảnh, và các phương án này có thể được thực hiện kết hợp với nhau trừ khi được đề cập theo cách khác.

Trong sáng chế này, viđeo có thể chỉ một chuỗi các ảnh theo thời gian. Hình ảnh nói chung đề cập đến đơn vị thể hiện một ảnh trong vùng thời gian cụ thể, và hình ảnh con/lát/ô xếp là đơn vị cấu thành nên một phần của hình ảnh trong việc lập mã. Hình ảnh con/lát/ô xếp có thể bao gồm một hoặc nhiều đơn vị cây lập mã (Coding Tree Unit, CTU). Một hình ảnh có thể bao gồm một hoặc nhiều hình ảnh con/lát/ô xếp. Một hình ảnh có thể bao gồm một hoặc nhiều nhóm ô xếp. Một nhóm ô xếp có thể bao gồm một hoặc nhiều ô xếp. Viên gạch có thể đại diện cho vùng hình chữ nhật gồm các hàng CTU trong ô xếp trong hình ảnh. Ô xếp có thể được phân vùng thành nhiều viên gạch, mỗi viên gạch trong số đó bao gồm một hoặc nhiều hàng CTU bên trong ô xếp này. Ô xếp không được phân vùng thành nhiều viên gạch cũng có thể được gọi là viên gạch. Việc quét viên gạch là việc sắp xếp thứ tự lần lượt cụ thể các CTU đang phân vùng một hình ảnh trong đó các CTU này được sắp xếp thứ tự liên tiếp theo việc quét mành CTU trong

một viên gạch, các viên gạch trong ô xếp được sắp xếp thứ tự liên tiếp theo việc quét mành các viên gạch của ô xếp này, và các ô xếp trong hình ảnh được sắp xếp thứ tự liên tiếp theo việc quét mành các ô xếp của hình ảnh này. Ngoài ra, hình ảnh con có thể thể hiện vùng hình chữ nhật gồm một hoặc nhiều lát trong hình ảnh. Tức là, hình ảnh con có thể chứa một hoặc nhiều lát mà cùng nhau chúng bao trùm một vùng hình chữ nhật của một hình ảnh. Ô xếp là vùng hình chữ nhật gồm các CTU trong cột ô xếp cụ thể và hàng ô xếp cụ thể trong hình ảnh. Cột ô xếp là vùng hình chữ nhật gồm các CTU có chiều cao bằng với chiều cao của hình ảnh và chiều rộng được chỉ rõ bởi các phần tử cú pháp trong tập tham số hình ảnh. Hàng ô xếp là vùng hình chữ nhật gồm các CTU có chiều cao được chỉ rõ bởi các phần tử cú pháp trong tập tham số hình ảnh và chiều rộng bằng với chiều rộng của hình ảnh. Việc quét ô xếp là việc sắp xếp thứ tự lần lượt cụ thể các CTU đang phân vùng một hình ảnh trong đó các CTU này được sắp xếp thứ tự liên tiếp theo việc quét mành CTU trong một ô xếp, trong khi các ô xếp trong hình ảnh được sắp xếp thứ tự liên tiếp theo việc quét mành các ô xếp của hình ảnh này. Lát bao gồm một số lượng nguyên các viên gạch của hình ảnh mà nó có thể được chứa riêng trong một đơn vị NAL đơn lẻ. Lát có thể bao gồm hoặc là một số lượng ô xếp hoàn chỉnh hoặc là chỉ một chuỗi các viên gạch hoàn chỉnh liên tiếp của một ô xếp. Các nhóm ô xếp và các lát có thể được sử dụng thay thể cho nhau trong sáng chế này. Ví dụ, trong sáng chế này, nhóm ô xếp/tiêu đề nhóm ô xếp có thể được gọi là lát/tiêu đề lát.

Điểm ảnh (pixel) hoặc pel có thể nghĩa là đơn vị nhỏ nhất cấu thành nên một hình ảnh (hoặc ảnh). Ngoài ra, 'mẫu' có thể được sử dụng như là thuật ngữ tương ứng với điểm ảnh. Mẫu nhìn chung có thể biểu diễn điểm ảnh hoặc giá trị của điểm ảnh, và có thể biểu diễn chỉ điểm ảnh/giá trị điểm ảnh của thành phần độ chói hoặc chỉ điểm ảnh/giá trị điểm ảnh của thành phần sắc độ.

Đơn vị có thể biểu diễn đơn vị cơ sở của việc xử lý ảnh. Đơn vị có thể bao gồm ít nhất một thành phần trong số vùng cụ thể của hình ảnh và thông tin liên quan tới vùng này. Một đơn vị có thể chứa một khối độ chói và hai khối sắc độ (ví dụ cb, cr). Đơn vị có thể được sử dụng thay thế được cho nhau với các thuật ngữ chẳng hạn như khối hoặc khu vực trong một số trường hợp. Trong trường hợp chung, khối M×N có thể bao gồm các mẫu (hoặc các mảng mẫu) hoặc tập hợp (hoặc mảng) các hệ số biến đổi gồm M cột

và N hàng.

Trong bản mô tả này, "A hoặc B" có thể nghĩa là "chỉ A", "chỉ B" hoặc "cả A và B". Nói cách khác, trong bản mô tả này, "A hoặc B" có thể được diễn dịch là "A và/hoặc B". Ví dụ, trong bản mô tả này, "A, B hoặc C" có thể nghĩa là "chỉ A", "chỉ B", "chỉ C" hoặc "sự kết hợp bất kỳ của A, B và C".

Dấu gạch chéo (/) hoặc dấu phẩy được sử dụng trong bản mô tả này có thể nghĩa là "và/hoặc". Ví dụ, "A/B" có thể nghĩa là "A và/hoặc B". Theo đó, "A/B" có thể nghĩa là "chỉ A", "chỉ B" hoặc "cả A và B". Ví dụ, "A, B, C" có thể nghĩa là "A, B hoặc C".

Trong bản mô tả này, "ít nhất một thành phần trong số A và B" có thể nghĩa là "chỉ A", "chỉ B" hoặc "cả A và B". Ngoài ra, trong bản mô tả này, cách nói "ít nhất một thành phần trong số A hoặc B" hoặc "ít nhất một thành phần trong số A và/hoặc B" có thể được diễn dịch giống với "ít nhất một thành phần trong số A và B".

Ngoài ra, trong bản mô tả này, "ít nhất một thành phần trong số A, B và C" có thể nghĩa là "chỉ A", "chỉ B", "chỉ C" hoặc "sự kết hợp bất kỳ của A, B và C". Ngoài ra, "ít nhất một thành phần trong số A, B, hoặc C" hoặc "ít nhất một thành phần trong số A, B và/hoặc C" có thể có nghĩa là "ít nhất một thành phần trong số A, B, và C".

Thêm vào đó, dấu ngoặc được sử dụng trong bản mô tả này có thể nghĩa là "ví dụ". Cụ thể, khi "việc dự đoán (việc dự đoán nội)" được chỉ ra, "việc dự đoán nội" có thể được đề xuất như ví dụ về "việc dự đoán". Nói cách khác, "việc dự đoán" trong bản mô tả này có thể không bị giới hạn vào "việc dự đoán nội", và "việc dự đoán nội" có thể được đề xuất như ví dụ về "việc dự đoán". Ngoài ra, kể cả khi "việc dự đoán (cụ thể là việc dự đoán nội)" được chỉ ra, "việc dự đoán nội" có thể được đề xuất như ví dụ về "việc dự đoán".

Các dấu hiệu kỹ thuật mà chúng được mô tả riêng lẻ trên một hình vẽ trong bản mô tả này có thể được thi hành một cách riêng lẻ hoặc có thể được thi hành cùng lúc.

Các hình vẽ sau đây được tạo ra nhằm giải thích một ví dụ cụ thể của bản mô tả này. Vì các tên của các thiết bị cụ thể được mô tả trên các hình vẽ hoặc các tên của các tín hiệu/các tin nhắn/các trường cụ thể được thể hiện dưới dạng ví dụ, các dấu hiệu kỹ thuật của bản mô tả này không bị giới hạn ở các tên cụ thể được sử dụng trên các hình

vẽ sau đây.

Fig.2 là sơ đồ giản lược minh họa cấu hình của thiết bị mã hóa viđeo/ảnh mà phương án (các phương án) của sáng chế có thể được áp dụng với nó. Ở dưới đây, thiết bị mã hóa viđeo có thể bao gồm thiết bị mã hóa ảnh.

Tham chiếu đến Fig.2, thiết bị mã hóa 200 bao gồm bộ phân vùng ảnh 210, bộ dự đoán 220, bộ xử lý phần dư 230, bộ mã hóa entropy 240, bộ cộng 250, bộ lọc 260, và bộ nhớ 270. Bộ dự đoán 220 có thể bao gồm bộ dự đoán liên 221 và bộ dự đoán nội 222. Bộ xử lý phần dư 230 có thể bao gồm bộ biến đổi 232, bộ lượng tử hóa 233, bộ giải lượng tử hóa 234, và bộ biến đổi ngược 235. Bộ xử lý phần dư 230 có thể còn bao gồm bộ trừ 231. Bộ cộng 250 có thể được gọi là bộ xây dựng lại hoặc bộ tạo ra khối được xây dựng lại. Bộ phân vùng ảnh 210, bộ dự đoán 220, bộ xử lý phần dư 230, bộ mã hóa entropy 240, bộ cộng 250, và bộ lọc 260 có thể được tạo cấu hình bởi ít nhất một thành phần phần cứng (ví dụ, bộ chip bộ mã hóa hoặc bộ xử lý) theo một phương án. Thêm vào đó, bộ nhớ 270 có thể bao gồm bộ đệm hình ảnh được giải mã (decoded picture buffer, DPB) hoặc có thể được tạo cấu hình bằng phương tiện lưu trữ kỹ thuật số. Thành phần phần cứng có thể còn bao gồm bộ nhớ 270 như thành phần trong/ngoài.

Bộ phân vùng ảnh 210 có thể phân vùng ảnh (hoặc hình ảnh hoặc khung) đầu vào được nhập vào thiết bị mã hóa 200 thành một hoặc nhiều bộ xử lý. Ví dụ, bộ xử lý này có thể được gọi là đơn vị lập mã (coding unit, CU). Trong trường hợp này, đơn vị lập mã có thể được phân vùng theo cách đệ quy theo cấu trúc cây tứ phân cây nhị phân cây tam phân (quad-tree binary-tree ternary-tree, QTBTTT) từ đơn vị cây lập mã (CTU) hoặc đơn vị lập mã lớn nhất (largest coding unit, LCU). Ví dụ, một đơn vị lập mã có thể được phân vùng thành nhiều đơn vị lập mã có độ sâu sâu hơn dựa trên cấu trúc cây tứ phân, cấu trúc cây nhị phân, và/hoặc cấu trúc tam phân. Trong trường hợp này, ví dụ, cấu trúc cây tứ phân có thể được áp dụng trước tiên và cấu trúc cây nhị phân và/hoặc cấu trúc tam phân có thể được áp dụng sau. Theo cách khác, cấu trúc cây nhị phân có thể được áp dụng trước tiên. Thủ tục lập mã theo sáng chế có thể được thực hiện dựa trên đơn vị lập mã cuối cùng mà nó không được phân vùng thêm nữa. Trong trường hợp này, đơn vị lập mã lớn nhất có thể được sử dụng làm đơn vị lập mã cuối cùng dựa trên hiệu suất lập mã theo các đặc điểm của ảnh, hoặc nếu cần thiết, đơn vị lập mã có thể

được phân vùng theo cách đệ quy thành các đơn vị lập mã có độ sâu sâu hơn và đơn vị lập mã có kích thước tối ưu có thể được sử dụng làm đơn vị lập mã cuối cùng. Ở đây, thủ tục lập mã có thể bao gồm thủ tục dự đoán, biến đổi, và xây dựng lại, mà chúng sẽ được mô tả sau. Như là một ví dụ khác, bộ xử lý có thể còn bao gồm đơn vị dự đoán (prediction unit, PU) hoặc đơn vị biến đổi (transform unit, TU). Trong trường hợp này, đơn vị dự đoán và đơn vị biến đổi có thể được phân chia hoặc được phân vùng từ đơn vị lập mã cuối cùng nêu trên. Đơn vị dự đoán có thể là đơn vị của việc dự đoán mẫu, và đơn vị biến đổi có thể là đơn vị để dẫn ra hệ số biến đổi và/hoặc đơn vị để dẫn ra tín hiệu phần dư từ hệ số biến đổi này.

Đơn vị có thể được sử dụng thay thế được cho nhau với các thuật ngữ chẳng hạn như khối hoặc khu vực trong một số trường hợp. Trong trường hợp chung, khối M×N có thể thể hiện tập các mẫu hoặc các hệ số biến đổi gồm M cột và N hàng. Mẫu nhìn chung có thể biểu diễn điểm ảnh hoặc giá trị của điểm ảnh, có thể biểu diễn chỉ điểm ảnh/giá trị điểm ảnh của thành phần độ chói hoặc biểu diễn chỉ điểm ảnh/giá trị điểm ảnh của thành phần sắc độ. Mẫu có thể được sử dụng như thuật ngữ tương ứng với một hình ảnh (hoặc ảnh) cho điểm ảnh hoặc pel.

Trong thiết bị mã hóa 200, tín hiệu dự đoán (khối được dự đoán, mảng mẫu dự đoán) được xuất ra từ bộ dự đoán liên 221 hoặc bộ dự đoán nội 222 được trừ khỏi tín hiệu ảnh đầu vào (khối gốc, mảng mẫu gốc) để tạo ra tín hiệu phần dư (khối phần dư, mảng mẫu phần dư), và tín hiệu phần dư được tạo ra này được truyền đến bộ biến đổi 232. Trong trường hợp này, như được minh họa, đơn vị để trừ tín hiệu dự đoán (khối được dự đoán, mảng mẫu dự đoán) khỏi tín hiệu ảnh đầu vào (khối gốc, mảng mẫu gốc) trong bộ mã hoá 200 có thể được gọi là bộ trừ 231. Bộ dự đoán có thể thực hiện việc dự đoán về khối cần được xử lý (ở dưới đây được gọi là khối hiện tại), và tạo ra khối được dự đoán bao gồm các mẫu dự đoán đối với khối hiện tại này. Bộ dự đoán có thể xác định xem liệu việc dự đoán nội hay việc dự đoán liên được áp dụng trên cơ sở khối hiện tại hay CU. Như sẽ được mô tả sau đây trong phần mô tả của mỗi chế độ dự đoán, bộ dự đoán có thể tạo ra các thông tin khác nhau liên quan đến việc dự đoán, chẳng hạn như thông tin chế độ dự đoán, và truyền thông tin được tạo ra này đến bộ mã hoá entropy 240. Thông tin về việc dự đoán có thể được mã hóa trong bộ mã hóa entropy 240 và

được xuất ra dưới dạng luồng bit.

Bộ dự đoán nội 222 có thể dự đoán khối hiện tại bằng cách tham chiếu đến các mẫu trong hình ảnh hiện tại. Các mẫu được tham chiếu có thể được đặt ở lân cận của khối hiện tại, hoặc có thể được đặt tách ra tùy theo chế độ dự đoán. Trong việc dự đoán nội, các chế độ dự đoán có thể bao gồm nhiều chế độ không hướng và nhiều chế độ có hướng. Các chế độ không hướng có thể bao gồm, ví dụ, chế độ DC và chế độ phẳng. Chế độ có hướng có thể bao gồm, ví dụ, 33 chế độ dự đoán có hướng hoặc 65 chế độ dự đoán có hướng tùy theo mức độ chi tiết của hướng dự đoán. Tuy nhiên, điều này chỉ đơn thuần là ví dụ, nhiều hoặc ít chế độ dự đoán có hướng hơn có thể được sử dụng phụ thuộc vào việc cài đặt. Bộ dự đoán nội 222 có thể xác định chế độ dự đoán được áp dụng cho khối hiện tại bằng cách sử dụng chế độ dự đoán được áp dụng cho khối lân cận.

Bộ dự đoán liên 221 có thể dẫn ra khối được dự đoán cho khối hiện tại dựa trên khối tham chiếu (mảng mẫu tham chiếu) được chỉ rõ bởi vectơ chuyển động trên hình ảnh tham chiếu. Trong trường hợp này, để làm giảm lượng thông tin chuyển động được truyền trong chế độ dự đoán liên, thông tin chuyển động có thể được dự đoán theo các đơn vị là các khối, các khối con, hoặc các mẫu dựa trên sự tương quan của thông tin chuyển động giữa khối lân cận và khối hiện tại. Thông tin chuyển động có thể bao gồm vectơ chuyển động và chỉ số hình ảnh tham chiếu. Thông tin chuyển động có thể còn bao gồm thông tin hướng dự đoán liên (dự đoán L0, dự đoán L1, song dự đoán (Bi prediction), v.v.). Trong trường hợp dự đoán liên, khối lân cận có thể bao gồm khối lân cận theo không gian có mặt trong hình ảnh hiện tại và khối lân cận theo thời gian có mặt trong hình ảnh tham chiếu. Hình ảnh tham chiếu chứa khối tham chiếu và hình ảnh tham chiếu chứa khối lân cận theo thời gian có thể là giống nhau hoặc khác nhau. Khối lân cận theo thời gian có thể được gọi là khối tham chiếu được đặt cùng vị trí, CU được đặt cùng chỗ (co-located CU, colCU), hoặc dạng tương tự, và hình ảnh tham chiếu chứa khối lân cận theo thời gian có thể được gọi là hình ảnh được đặt cùng vị trí (collocated picture, colPic). Ví dụ, bộ dự đoán liên 221 có thể tạo cấu hình danh sách ứng viên thông tin chuyển động dựa trên các khối lân cận và tạo ra thông tin chỉ ra ứng viên nào được sử dụng để dẫn ra vectơ chuyển động và/hoặc chỉ số hình ảnh tham chiếu của khối hiện tại. Việc dự đoán liên có thể được thực hiện dựa trên các chế độ dự đoán khác nhau. Ví

dụ, trong trường hợp của chế độ bỏ qua và chế độ hợp nhất, bộ dự đoán liên 221 có thể sử dụng thông tin chuyển động của khối lân cận làm thông tin chuyển động của khối hiện tại. Trong chế độ bỏ qua, không giống chế độ hợp nhất, tín hiệu phần dư có thể không được truyền. Trong trường hợp chế độ dự đoán vectơ chuyển động (motion vector prediction, MVP), vectơ chuyển động của khối lân cận có thể được sử dụng làm bộ dự đoán vectơ chuyển động và vectơ chuyển động của khối hiện tại có thể được chỉ ra bằng cách báo hiệu chênh lệch vectơ chuyển động.

Bộ dự đoán 220 có thể tạo ra tín hiệu dự đoán dựa trên các phương pháp dự đoán khác nhau được mô tả dưới đây. Ví dụ, bộ dự đoán có thể không chỉ áp dụng việc dự đoán nội hoặc dự đoán liên để dự đoán một khối, mà còn áp dụng đồng thời cả việc dự đoán nội lẫn việc dự đoán liên. Điều này có thể được gọi là việc dự đoán liên và nội kết hợp (combined inter and intra prediction, CIIP). Thêm vào đó, bộ dự đoán có thể được dựa trên chế độ dự đoán sao chép khối nội (intra block copy, IBC) hoặc chế độ bảng màu (palette) để dự đoán về khối. Chế độ dự đoán IBC hoặc chế độ bảng màu có thể được sử dụng cho việc lập mã ảnh/viđeo nội dung của trò chơi hoặc dạng tương tự, chẳng hạn như việc lập mã nội dung màn (screen content coding, SCC). IBC về cơ bản thực hiện việc dự đoán trong hình ảnh hiện tại, nhưng có thể được thực hiện tương tự với việc dự đoán liên ở chỗ khối tham chiếu được dẫn ra trong hình ảnh hiện tại. Tức là, IBC có thể sử dụng ít nhất một trong các kỹ thuật dự đoán liên được mô tả trong sáng chế này. Chế độ bảng màu có thể được xem như là ví dụ của việc lập mã nội hoặc việc dự đoán nội. Khi chế độ bảng màu được áp dụng, giá trị mẫu trong hình ảnh có thể được báo hiệu dựa trên thông tin về bảng của bảng màu và chỉ số bảng màu.

Tín hiệu dự đoán được tạo ra bởi bộ dự đoán (bao gồm bộ dự đoán liên 221 và/hoặc bộ dự đoán nội 222) có thể được sử dụng để tạo ra tín hiệu được xây dựng lại hoặc để tạo ra tín hiệu phần dư. Bộ biến đổi 232 có thể tạo ra các hệ số biến đổi bằng cách áp dụng kỹ thuật biến đổi cho tín hiệu phần dư. Ví dụ, kỹ thuật biến đổi có thể bao gồm ít nhất một biến đổi trong số biến đổi cosin rời rạc (discrete cosine transform, DCT), biến đổi sin rời rạc (discrete sine transform, DST), biến đổi Karhunen-loève (Karhunen-loève transform, KLT), biến đổi dựa trên đồ thị (graph-based transform, GBT), hoặc biến đổi phi tuyến có điều kiện (conditionally non-linear transform, CNT). Ở đây, GBT nghĩa là

việc biến đổi thu được từ đồ thị khi thông tin mối quan hệ giữa các điểm ảnh được biểu diễn bằng đồ thị. CNT chỉ việc biến đổi được tạo ra dựa trên tín hiệu dự đoán được tạo ra nhờ sử dụng tất cả các điểm ảnh được xây dựng lại trước đó. Thêm vào đó, quy trình biến đổi có thể được áp dụng cho các khối điểm ảnh hình vuông có cùng kích thước hoặc có thể được áp dụng cho các khối có kích thước biến thiên thay vì hình vuông.

Bộ lượng tử hoá 233 có thể lượng tử hoá các hệ số biến đổi và truyền chúng đến bộ mã hóa entropy 240 và bộ mã hóa entropy 240 có thể mã hoá tín hiệu được lượng tử hoá (thông tin về các hệ số biến đổi được lượng tử hoá này) và xuất ra luồng bit. Thông tin về các hệ số biến đổi được lượng tử hóa có thể được gọi là thông tin phần dư. Bộ lượng tử hóa 233 có thể sắp xếp lại các hệ số biến đổi được lượng tử hóa ở dạng khối thành dạng vectơ một chiều dựa trên thứ tự quét hệ số và tạo ra thông tin về các hệ số biến đổi được lượng tử hóa dựa trên các hệ số biến đổi được lượng tử hóa ở dạng vectơ một chiều. Thông tin về các hệ số biến đổi có thể được tạo ra. Bộ mã hóa entropy 240 có thể thực hiện các phương pháp mã hóa khác nhau ví dụ chẳng hạn như Golomb hàm số mũ, lập mã chiều dài biến đổi thích ứng ngữ cảnh (context-adaptive variable length coding, CAVLC), lập mã số học nhị phân thích ứng ngữ cảnh (context-adaptive binary arithmetic coding, CABAC), và dạng tương tự. Bộ mã hóa entropy 240 có thể mã hóa thông tin cần thiết cho việc xây dựng lại viđeo/ảnh ngoài các hệ số biến đổi được lượng tử hóa (ví dụ, các giá trị của các phần tử cú pháp, v.v.) cùng nhau hoặc tách biệt. Thông tin được mã hóa (ví dụ, thông tin viđeo/ảnh được mã hóa) có thể được truyền hoặc được lưu trữ trong các đơn vị là các NAL (network abstraction layer, lớp trừu tượng mạng) dưới dạng luồng bit. Thông tin viđeo/ảnh có thể còn bao gồm thông tin về các tập tham số khác nhau như tập tham số thích ứng (adaptation parameter set, APS), tập tham số hình ảnh (picture parameter set, PPS), tập tham số trình tự (sequence parameter set, SPS), hoặc tập tham số viđeo (video parameter set, VPS). Thêm vào đó, thông tin viđeo/ảnh có thể còn bao gồm thông tin ràng buộc chung. Trong sáng chế này, thông tin và/hoặc các phần tử cú pháp được truyền/được báo hiệu từ thiết bị mã hóa đến thiết bị giải mã có thể được đưa vào trong thông tin viđeo/hình ảnh. Thông tin viđeo/ảnh có thể được mã hóa qua thủ tục mã hóa được mô tả trên đây và được đưa vào trong luồng bit. Luồng bit có thể được truyền qua mạng hoặc có thể được lưu trữ trong phương tiện lưu trữ kỹ

thuật số. Mạng này có thể bao gồm mạng phát quảng bá và/hoặc mạng truyền thông, và phương tiện lưu trữ kỹ thuật số này có thể bao gồm các phương tiện lưu trữ khác nhau như USB, SD, CD, DVD, Blu-ray, HDD, SSD, và dạng tương tự. Bộ truyền (không được thể hiện) truyền tín hiệu được xuất ra từ bộ mã hóa entropy 240 và/hoặc đơn vị lưu trữ (không được thể hiện) lưu trữ tín hiệu này có thể được đưa vào dưới dạng phần tử bên trong/bên ngoài của thiết bị mã hóa 200, và theo cách khác, bộ truyền có thể được đưa vào trong bộ mã hóa entropy 240.

Các hệ số biến đổi được lượng tử hóa được xuất ra từ bộ lượng tử hóa 233 có thể được sử dụng để tạo ra tín hiệu dự đoán. Ví dụ, tín hiệu phần dư (khối phần dư hoặc các mẫu phần dư) có thể được xây dựng lại bằng cách áp dụng việc giải lượng tử hóa và biến đổi ngược đối với các hệ số biến đổi được lượng tử hóa qua bộ giải lượng tử hóa 234 và bộ biến đổi ngược 235. Bộ cộng 250 cộng tín hiệu phần dư được xây dựng lại vào tín hiệu dự đoán được xuất ra từ bộ dự đoán liên 221 hoặc bộ dự đoán nội 222 để tạo ra tín hiệu được xây dựng lại (hình ảnh được xây dựng lại, khối được xây dựng lại, mảng mẫu được xây dựng lại). Nếu không có phần dư nào cho khối cần được xử lý, như trường hợp mà trong đó chế độ bỏ qua được áp dụng, thì khối được dự đoán có thể được sử dụng làm khối được xây dựng lại. Bộ cộng 250 có thể được gọi là bộ xây dựng lại hoặc bộ tạo ra khối được xây dựng lại. Tín hiệu được xây dựng lại được tạo ra có thể được sử dụng cho việc dự đoán nội khối tiếp theo cần được xử lý trong hình ảnh hiện tại và có thể được sử dụng cho việc dự đoán liên hình ảnh tiếp theo qua việc lọc như được mô tả bên dưới.

Trong khi đó, việc ánh xạ độ chói với việc định tỷ lệ sắc độ (luma mapping with chroma scaling, LMCS) có thể được áp dụng trong khi mã hóa và/hoặc xây dựng lại hình ảnh.

Bộ lọc 260 có thể cải thiện chất lượng ảnh chủ quan/khách quan bằng cách áp dụng việc lọc cho tín hiệu được xây dựng lại. Ví dụ, bộ lọc 260 có thể tạo ra hình ảnh được xây dựng lại được cải biến bằng cách áp dụng các phương pháp lọc khác nhau đối với hình ảnh được xây dựng lại và lưu trữ hình ảnh được xây dựng lại được cải biến trong bộ nhớ 270, cụ thể, DPB của bộ nhớ 270. Các phương pháp lọc khác nhau này có thể bao gồm, ví dụ, lọc khử khối, phần bù tương thích mẫu, bộ lọc vòng lặp tương thích, bộ

lọc song phương, và dạng tương tự. Bộ lọc 260 có thể tạo ra thông tin khác nhau liên quan tới việc lọc này và truyền thông tin được tạo ra đến bộ mã hóa entropy 240 như sẽ được mô tả sau trong phần mô tả về mỗi phương pháp lọc. Thông tin liên quan tới việc lọc có thể được mã hóa bởi bộ mã hóa entropy 240 và xuất ra dưới dạng luồng bit.

Hình ảnh được xây dựng lại được cải biến được truyền đến bộ nhớ 270 có thể được sử dụng làm hình ảnh tham chiếu trong bộ dự đoán liên 221. Khi việc dự đoán liên được áp dụng thông qua thiết bị mã hóa, việc không khớp dự đoán giữa thiết bị mã hóa 200 và thiết bị giải mã 300 có thể tránh được và hiệu suất mã hóa có thể được cải thiện.

DPB của bộ nhớ 270 có thể lưu trữ hình ảnh được xây dựng lại được cải biến để sử dụng làm hình ảnh tham chiếu trong bộ dự đoán liên 221. Bộ nhớ 270 có thể lưu trữ thông tin chuyển động của khối mà từ đó thông tin chuyển động trong hình ảnh hiện tại được dẫn ra (hoặc được mã hóa) và/hoặc thông tin chuyển động của các khối trong hình ảnh đã được xây dựng lại. Thông tin chuyển động được lưu trữ có thể được truyền đến bộ dự đoán liên 221 và được sử dụng làm thông tin chuyển động của khối lân cận theo không gian hoặc thông tin chuyển động của khối lân cận theo thời gian. Bộ nhớ 270 có thể lưu trữ các mẫu được xây dựng lại của các khối được xây dựng lại trong hình ảnh hiện tại và có thể chuyển các mẫu được xây dựng lại đến bộ dự đoán nội 222.

Fig.3 là sơ đồ giản lược minh họa cấu hình của thiết bị giải mã viđeo/ảnh mà phương án (các phương án) của sáng chế có thể được áp dụng với nó.

Tham chiếu đến Fig.3, thiết bị giải mã 300 có thể bao gồm bộ giải mã entropy 310, bộ xử lý phần dư 320, bộ dự đoán 330, bộ cộng 340, bộ lọc 350, và bộ nhớ 360. Bộ dự đoán 330 có thể bao gồm bộ dự đoán liên 331 và bộ dự đoán nội 332. Bộ xử lý phần dư 320 có thể bao gồm bộ giải lượng tử hóa 321 và bộ biến đổi ngược 322. Bộ giải mã entropy 310, bộ xử lý phần dư 320, bộ dự đoán 330, bộ cộng 340, và bộ lọc 350 có thể được tạo cấu hình bởi thành phần phần cứng (ví dụ, bộ chip bộ giải mã hoặc bộ xử lý) theo một phương án. Thêm vào đó, bộ nhớ 360 có thể bao gồm bộ đệm hình ảnh được giải mã (decoded picture buffer, DPB) hoặc có thể được tạo cấu hình bằng phương tiện lưu trữ kỹ thuật số. Thành phần phần cứng có thể còn bao gồm bộ nhớ 360 như thành phần trong/ngoài.

Khi luồng bit bao gồm thông tin viđeo/ảnh được nhập vào, thiết bị giải mã 300 có thể xây dựng lại ảnh tương ứng với quy trình trong đó thông tin viđeo/ảnh được xử lý trong thiết bị mã hóa trên Fig.2. Ví dụ, thiết bị giải mã 300 có thể dẫn ra các đơn vị/khối dựa trên thông tin liên quan đến việc phân vùng khối thu được từ luồng bit. Thiết bị giải mã 300 có thể thực hiện việc giải mã nhờ sử dụng bộ xử lý được áp dụng trong thiết bị mã hóa. Vì vậy, ví dụ, bộ xử lý của việc giải mã có thể là đơn vị lập mã, và đơn vị lập mã này có thể được phân vùng theo cấu trúc cây tứ phân, cấu trúc cây nhị phân và/hoặc cấu trúc cây tam phân từ đơn vị cây lập mã hoặc đơn vị lập mã lớn nhất. Một hoặc nhiều đơn vị biến đổi có thể được dẫn ra từ đơn vị lập mã. Tín hiệu ảnh được xây dựng lại được giải mã và được xuất ra thông qua thiết bị giải mã 300 có thể được tái tạo thông qua thiết bị tái tạo.

Thiết bị giải mã 300 có thể nhận tín hiệu được xuất ra từ thiết bị mã hoá trên Fig.2 dưới dạng luồng bit, và tín hiệu nhận được này có thể được giải mã qua bộ giải mã entropy 310. Ví dụ, bộ giải mã entropy 310 có thể phân tích cú pháp luồng bit để dẫn ra thông tin (ví dụ, thông tin viđeo/ảnh) cần thiết cho việc xây dựng lại ảnh (hoặc xây dựng lại hình ảnh). Thông tin viđeo/ảnh có thể còn bao gồm thông tin về các tập tham số khác nhau như tập tham số thích ứng (adaptation parameter set, APS), tập tham số hình ảnh (picture parameter set, PPS), tập tham số trình tự (sequence parameter set, SPS), hoặc tập tham số viđeo (video parameter set, VPS). Thêm vào đó, thông tin viđeo/ảnh có thể còn bao gồm thông tin ràng buộc chung. Thiết bị giải mã có thể còn giải mã hình ảnh dựa trên thông tin về tập tham số và/hoặc thông tin ràng buộc chung. Thông tin và/hoặc các phần tử cú pháp được báo hiệu/nhận được được mô tả sau trong sáng chế này có thể được giải mã qua thủ tục giải mã và được thu từ luồng bit. Ví dụ, bộ giải mã entropy 310 giải mã thông tin trong luồng bit dựa trên phương pháp lập mã như lập mã Golomb hàm số mũ, CAVLC hoặc CABAC, và các phần tử cú pháp được xuất ra cần thiết cho việc xây dựng lại ảnh và các giá trị được lượng tử hóa của các hệ số biến đổi cho phần dư. Cụ thể hơn, phương pháp giải mã entropy CABAC có thể nhận ngăn (bin) tương ứng với mỗi phần tử cú pháp trong luồng bit, xác định mô hình ngữ cảnh nhờ sử dụng thông tin về phần tử cú pháp mục tiêu giải mã, thông tin giải mã của khối mục tiêu giải mã hoặc thông tin của ký hiệu/ngăn được giải mã trong pha trước, và thực hiện việc giải

mã số học trên ngăn này bằng cách dự đoán xác suất xuất hiện ngăn theo mô hình ngữ cảnh được xác định, và tạo ra ký hiệu tương ứng với giá trị của mỗi phần tử cú pháp. Trong trường hợp này, phương pháp giải mã entropy CABAC có thể cập nhật mô hình ngữ cảnh bằng cách sử dụng thông tin về ký hiệu/ngăn được giải mã cho mô hình ngữ cảnh của ký hiệu/ngăn tiếp theo sau khi xác định mô hình ngữ cảnh. Thông tin liên quan đến việc dự đoán trong số thông tin được giải mã bởi bộ giải mã entropy 310 có thể được cung cấp cho bộ dự đoán (bộ dự đoán liên 332 và bộ dự đoán nội 331), và giá trị phần dư mà việc giải mã entropy được thực hiện trên đó trong bộ giải mã entropy 310, tức là, các hệ số biến đổi được lượng tử hóa và thông tin tham số liên quan, có thể được cung cấp cho bộ xử lý phần dư 320. Bộ xử lý phần dư 320 có thể dẫn ra tín hiệu phần dư (khối phần dư, các mẫu phần dư, mảng mẫu phần dư). Thêm vào đó, thông tin về việc lọc trong số thông tin được giải mã bởi bộ giải mã entropy 310 có thể được cung cấp cho bộ lọc 350. Trong khi đó, bộ nhận (không được thể hiện) để nhận tín hiệu được xuất ra từ thiết bị mã hóa có thể còn được tạo cấu hình như phần tử bên trong/bên ngoài của thiết bị giải mã 300, hoặc bộ nhận có thể là thành phần của bộ giải mã entropy 310. Trong khi đó, thiết bị giải mã theo sáng chế có thể được gọi là thiết bị giải mã viđeo/ảnh/hình ảnh, và thiết bị giải mã này có thể được phân loại thành bộ giải mã thông tin (bộ giải mã thông tin viđeo/ảnh/hình ảnh) và bộ giải mã mẫu (bộ giải mã mẫu viđeo/ảnh/hình ảnh). Bộ giải mã thông tin có thể bao gồm bộ giải mã entropy 310, và bộ giải mã mẫu có thể bao gồm ít nhất một thành phần trong số bộ giải lượng tử hóa 321, bộ biến đổi ngược 322, bộ cộng 340, bộ lọc 350, bộ nhớ 360, bộ dự đoán liên 332, và bộ dự đoán nội 331.

Bộ giải lượng tử hóa 321 có thể giải lượng tử hóa các hệ số biến đổi được lượng tử hóa và xuất ra các hệ số biến đổi. Bộ giải lượng tử hóa 321 có thể sắp xếp lại các hệ số biến đổi được lượng tử hóa dưới dạng khối hai chiều. Trong trường hợp này, việc sắp xếp lại có thể được thực hiện dựa trên thứ tự quét hệ số được thực hiện trong thiết bị mã hóa. Bộ giải lượng tử hóa 321 có thể thực hiện việc giải lượng tử hóa trên các hệ số biến đổi được lượng tử hóa bằng cách sử dụng tham số lượng tử hóa (ví dụ, thông tin kích thước bước lượng tử hóa) và thu được các hệ số biến đổi.

Bộ biến đổi ngược 322 biến đổi ngược các hệ số biến đổi để thu được tín hiệu phần

dư (khối phần dư, mảng mẫu phần dư).

Bộ dự đoán có thể thực hiện việc dự đoán trên khối hiện tại và tạo ra khối được dự đoán bao gồm các mẫu dự đoán cho khối hiện tại. Bộ dự đoán có thể xác định xem liệu việc dự đoán nội hay việc dự đoán liên được áp dụng cho khối hiện tại dựa trên thông tin về việc dự đoán được xuất ra từ bộ giải mã entropy 310 và có thể xác định chế độ dự đoán nội/liên cụ thể.

Bộ dự đoán 320 có thể tạo ra tín hiệu dự đoán dựa trên các phương pháp dự đoán khác nhau được mô tả dưới đây. Ví dụ, bộ dự đoán có thể không chỉ áp dụng việc dự đoán nội hoặc dự đoán liên để dự đoán một khối, mà còn áp dụng đồng thời việc dự đoán nội và việc dự đoán liên. Điều này có thể được gọi là việc dự đoán liên và nội kết hợp (combined inter and intra prediction, CIIP). Thêm vào đó, bộ dự đoán có thể được dựa trên chế độ dự đoán sao chép khối nội (intra block copy, IBC) hoặc chế độ bảng màu (palette) để dự đoán về khối. Chế độ dự đoán IBC hoặc chế độ bảng màu có thể được sử dụng cho việc lập mã ảnh/viđeo nội dung của trò chơi hoặc dạng tương tự, chẳng hạn như việc lập mã nội dung màn (screen content coding, SCC). IBC về cơ bản thực hiện việc dự đoán trong hình ảnh hiện tại, nhưng có thể được thực hiện tương tự với việc dự đoán liên ở chỗ khối tham chiếu được dẫn ra trong hình ảnh hiện tại. Tức là, IBC có thể sử dụng ít nhất một trong các kỹ thuật dự đoán liên được mô tả trong sáng chế này. Chế độ bảng màu có thể được xem như là ví dụ của việc lập mã nội hoặc việc dự đoán nội. Khi chế độ bảng màu được áp dụng, giá trị mẫu trong hình ảnh có thể được báo hiệu dựa trên thông tin về bảng của bảng màu và chỉ số bảng màu.

Bộ dự đoán nội 331 có thể dự đoán khối hiện tại bằng cách tham chiếu đến các mẫu trong hình ảnh hiện tại. Các mẫu được tham chiếu có thể được đặt ở lân cận của khối hiện tại, hoặc có thể được đặt tách ra tùy theo chế độ dự đoán. Trong việc dự đoán nội, các chế độ dự đoán có thể bao gồm nhiều chế độ không hướng và nhiều chế độ có hướng. Bộ dự đoán nội 331 có thể xác định chế độ dự đoán được áp dụng cho khối hiện tại bằng cách sử dụng chế độ dự đoán được áp dụng cho khối lân cận.

Bộ dự đoán liên 332 có thể dẫn ra khối được dự đoán cho khối hiện tại dựa trên khối tham chiếu (mảng mẫu tham chiếu) được chỉ rõ bởi vectơ chuyển động trên hình ảnh tham chiếu. Trong trường hợp này, để làm giảm lượng thông tin chuyển động được

truyền trong chế độ dự đoán liên, thông tin chuyển động có thể được dự đoán trong các đơn vị là các khối, các khối con, hoặc các mẫu dựa trên sự tương quan của thông tin chuyển động giữa khối lân cận và khối hiện tại. Thông tin chuyển động có thể bao gồm vectơ chuyển động và chỉ số hình ảnh tham chiếu. Thông tin chuyển động có thể còn bao gồm thông tin hướng dự đoán liên (dự đoán L0, dự đoán L1, song dự đoán (Bi prediction), v.v.). Trong trường hợp dự đoán liên, khối lân cận có thể bao gồm khối lân cận theo không gian có mặt trong hình ảnh hiện tại và khối lân cận theo thời gian có mặt trong hình ảnh tham chiếu. Ví dụ, bộ dự đoán liên 332 có thể tạo cấu hình danh sách ứng viên thông tin chuyển động dựa trên các khối lân cận và dẫn ra vectơ chuyển động của khối hiện tại và/hoặc chỉ số hình ảnh tham chiếu dựa trên thông tin chọn ứng viên nhận được. Việc dự đoán liên có thể được thực hiện dựa trên các chế độ dự đoán khác nhau, và thông tin về việc dự đoán có thể bao gồm thông tin chỉ ra chế độ dự đoán liên cho khối hiện tại.

Bộ cộng 340 có thể tạo ra tín hiệu được xây dựng lại (hình ảnh được xây dựng lại, khối được xây dựng lại, mảng mẫu được xây dựng lại) bằng cách cộng tín hiệu phần dư thu được vào tín hiệu dự đoán (khối được dự đoán, mảng mẫu được dự đoán) được xuất ra từ bộ dự đoán (bao gồm bộ dự đoán liên 332 và/hoặc bộ dự đoán nội 331). Nếu không có phần dư nào cho khối cần được xử lý, như khi chế độ bỏ qua được áp dụng, thì khối được dự đoán có thể được sử dụng làm khối được xây dựng lại.

Bộ cộng 340 có thể được gọi là bộ xây dựng lại hoặc bộ tạo ra khối được xây dựng lại. Tín hiệu được xây dựng lại được tạo ra có thể được sử dụng cho việc dự đoán nội của khối tiếp theo cần được xử lý trong hình ảnh hiện tại, có thể được xuất ra thông qua việc lọc như được mô tả dưới đây, hoặc có thể được sử dụng cho việc dự đoán liên của ảnh tiếp theo.

Trong khi đó, việc ánh xạ độ chói với việc định tỷ lệ sắc độ (LMCS) có thể được áp dụng trong quy trình giải mã ảnh.

Bộ lọc 350 có thể cải thiện chất lượng ảnh chủ quan/khách quan bằng cách áp dụng việc lọc cho tín hiệu được xây dựng lại. Ví dụ, bộ lọc 350 có thể tạo ra hình ảnh được xây dựng lại được cải biến bằng cách áp dụng các phương pháp lọc khác nhau đối với hình ảnh được xây dựng lại và lưu trữ hình ảnh được xây dựng lại được cải biến trong

bộ nhớ 360, cụ thể, DPB của bộ nhớ 360. Các phương pháp lọc khác nhau này có thể bao gồm, ví dụ, lọc khử khối, phần bù tương thích mẫu, bộ lọc vòng lặp tương thích, bộ lọc song phương, và dạng tương tự.

Hình ảnh được xây dựng lại (được cải biến) được lưu trữ trong DPB của bộ nhớ 360 có thể được sử dụng làm hình ảnh tham chiếu trong bộ dự đoán liên 332. Bộ nhớ 360 có thể lưu trữ thông tin chuyển động của khối mà thông tin chuyển động trong hình ảnh hiện tại được dẫn ra (hoặc được giải mã) từ đó và/hoặc thông tin chuyển động của các khối trong hình ảnh đã được xây dựng lại sẵn. Thông tin chuyển động được lưu trữ có thể được truyền đến bộ dự đoán liên 260 sao cho nó được sử dụng làm thông tin chuyển động của khối lân cận theo không gian hoặc thông tin chuyển động của khối lân cận theo thời gian. Bộ nhớ 360 có thể lưu trữ các mẫu được xây dựng lại của các khối được xây dựng lại trong hình ảnh hiện tại và chuyển các mẫu được xây dựng lại này đến bộ dự đoán nội 331.

Trong sáng chế này, các phương án được mô tả trong bộ lọc 260, bộ dự đoán liên 221, và bộ dự đoán nội 222 của thiết bị mã hóa 200 có thể giống như hoặc lần lượt được áp dụng để tương ứng với bộ lọc 350, bộ dự đoán liên 332, và bộ dự đoán nội 331 của thiết bị giải mã 300. Điều tương tự cũng có thể áp dụng cho đơn vị 332 và bộ dự đoán nội 331.

Như được mô tả trên đây, thiết bị mã hóa có thể thực hiện các phương pháp mã hóa khác nhau chẳng hạn như Golomb hàm số mũ, lập mã chiều dài biến đổi thích ứng ngữ cảnh (CAVLC), và lập mã số học nhị phân thích ứng ngữ cảnh (CABAC). Thêm vào đó, thiết bị giải mã có thể giải mã thông tin trong luồng bit dựa trên phương pháp lập mã chẳng hạn như lập mã Golomb hàm số mũ, CAVLC hoặc CABAC, và xuất ra giá trị của phần tử cú pháp cần thiết cho việc xây dựng lại ảnh và giá trị được lượng tử hóa của các hệ số biến đổi liên quan đến các phần dư.

Ví dụ, các phương pháp lập mã được mô tả trên đây có thể được thực hiện như được mô tả dưới đây.

Fig.4 thể hiện làm ví dụ việc lập mã số học nhị phân thích ứng ngữ cảnh (context-adaptive binary arithmetic coding, CABAC) để mã hóa phần tử cú pháp. Ví dụ, trong

quy trình mã hóa CABAC, khi tín hiệu đầu vào là phần tử cú pháp thay vì giá trị nhị phân, thiết bị mã hóa có thể chuyển đổi tín hiệu đầu vào thành giá trị nhị phân bằng cách nhị phân hóa giá trị của tín hiệu đầu vào. Thêm vào đó, khi tín hiệu đầu vào đã là giá trị nhị phân (cụ thể là, khi giá trị của tín hiệu đầu vào là giá trị nhị phân), việc nhị phân hóa có thể không được thực hiện và có thể được đi vòng qua. Ở đây, mỗi số nhị phân 0 hoặc 1 đang cấu thành nên giá trị nhị phân có thể được gọi là một ngăn (bin). Ví dụ, nếu chuỗi nhị phân sau khi nhị phân hóa là 110, thì mỗi số 1, 1, và 0 được gọi là một ngăn. Ngăn (các ngăn) cho một phần tử cú pháp có thể chỉ ra giá trị của phần tử cú pháp này.

Sau đó, các ngăn được nhị phân hóa của phần tử cú pháp có thể là đầu vào cho cỗ máy (engine) lập mã thông thường hoặc cỗ máy lập mã đi vòng (bypass). Cỗ máy lập mã thông thường của thiết bị mã hóa có thể cấp phát mô hình ngữ cảnh phản ánh giá trị xác suất cho ngăn tương ứng, và có thể mã hóa ngăn tương ứng dựa trên mô hình ngữ cảnh được cấp phát. Cỗ máy lập mã thông thường của thiết bị mã hóa có thể cập nhật mô hình ngữ cảnh của mỗi ngăn sau khi thực hiện việc mã hóa trên mỗi ngăn. Ngăn được mã hóa như được mô tả trên đây có thể được gọi là ngăn được lập mã ngữ cảnh.

Trong khi đó, khi các ngăn được nhị phân hóa của phần tử cú pháp là đầu vào cho cỗ máy lập mã đi vòng, chúng có thể được lập mã như sau. Ví dụ, cỗ máy lập mã đi vòng của thiết bị mã hóa lược bỏ thủ tục ước lượng xác suất đối với ngăn đầu vào và thủ tục cập nhật mô hình xác suất được áp dụng cho ngăn sau khi mã hóa. Khi việc mã hóa đi vòng được áp dụng, thiết bị mã hóa có thể mã hóa ngăn đầu vào bằng cách áp dụng phân phối xác suất đều thay vì cấp phát mô hình ngữ cảnh, qua đó cải thiện tốc độ mã hóa. Ngăn được mã hóa như được mô tả trên đây có thể được gọi là ngăn đi vòng.

Việc giải mã entropy có thể thể hiện quy trình thực hiện quy trình giống như việc mã hóa entropy được mô tả trên đây theo thứ tự đảo ngược.

Ví dụ, khi phần tử cú pháp được giải mã dựa trên mô hình ngữ cảnh, thiết bị giải mã có thể nhận ngăn tương ứng với phần tử cú pháp này thông qua luồng bit, xác định mô hình ngữ cảnh nhờ sử dụng phần tử cú pháp này và giải mã thông tin về khối mục tiêu giải mã hoặc khối lân cận hoặc thông tin về ký hiệu/ngăn được giải mã ở giai đoạn trước đó, dự đoán xác suất xảy ra của ngăn nhận được theo mô hình ngữ cảnh xác định được, và thực hiện việc giải mã số học trên ngăn này để dẫn ra giá trị của phần tử cú

pháp này. Sau đó, mô hình ngữ cảnh của ngăn mà nó được giải mã tiếp theo có thể được cập nhật bằng mô hình ngữ cảnh xác định được.

Ngoài ra, ví dụ, khi phần tử cú pháp được giải mã đi vòng, thiết bị giải mã có thể nhận ngăn tương ứng với phần tử cú pháp này thông qua luồng bit, và giải mã ngăn đầu vào này bằng cách áp dụng phân phối xác suất đều. Trong trường hợp này, thủ tục của thiết bị giải mã để dẫn ra mô hình ngữ cảnh của phần tử cú pháp và thủ tục cập nhật mô hình ngữ cảnh được áp dụng cho ngăn sau khi giải mã có thể được lược bỏ.

Như được mô tả trên đây, các mẫu phần dư có thể được dẫn ra là các hệ số biến đổi được lượng tử hóa thông qua các quy trình biến đổi và lượng tử hóa. Các hệ số biến đổi được lượng tử hóa này cũng có thể được gọi là các hệ số biến đổi. Trong trường hợp này, các hệ số biến đổi trong khối có thể được báo hiệu dưới dạng thông tin phần dư. Thông tin phần dư này có thể bao gồm cú pháp lập mã phần dư. Tức là, thiết bị mã hóa có thể tạo cấu hình cú pháp lập mã phần dư bằng thông tin phần dư, mã hóa cú pháp này, và xuất nó ra dưới dạng luồng bit, và thiết bị giải mã có thể giải mã cú pháp lập mã phần dư này từ luồng bit và dẫn ra các hệ số biến đổi (được lượng tử hóa) phần dư. Cú pháp lập mã phần dư này có thể bao gồm các phần tử cú pháp thể hiện liệu việc biến đổi có đã được áp dụng cho khối tương ứng hay không, vị trí của hệ số biến đổi có hiệu lực cuối cùng trong khối, liệu hệ số biến đổi có hiệu lực có tồn tại trong khối con hay không, kích thước/dấu của hệ số biến đổi có hiệu lực này, và dạng tương tự, như sẽ được mô tả sau đây.

Ví dụ, các hệ số biến đổi (được lượng tử hóa) (cụ thể là thông tin phần dư) có thể được mã hóa và/hoặc giải mã dựa trên các phần tử cú pháp như transform_skip_flag, last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, coded_sub_block_flag, sig_coeff_flag, par_level_flag, abs_level_gt1_flag, abs_level_gt3_flag, abs_remainder, coeff_sign_flag, dec_abs_level, mts_idx. Các phần tử cú pháp liên quan đến việc mã hóa/giải mã dữ liệu phần dư có thể được biểu diễn như được thể hiện trong bảng sau đây.

Bảng 1

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if(transform_skip_enabled_flag && (cIdx != 0 || tu_mts_flag[x0][y0] == 0) && (log2TbWidth <= 2) && (log2TbHeight <= 2))	
transform_skip_flag[x0][y0][cIdx]	ae(v)
last_sig_coeff_x_prefix	ae(v)
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2SbSize = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
numSbCoeff = 1 << (log2SbSize << 1)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − 2 * log2SbSize)) − 1	
do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock− −	
}	
lastScanPos− −	
xS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize] [lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize] [lastSubBlock][1]	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][lastScanPos][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
numSigCoeff = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i− −) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize] [lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize] [lastSubBlock][1]	
inferSbDcSigCoeffFlag = 0	

if((i < lastSubBlock) && (i > 0)) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
remBinsPass1 = (log2SbSize < 2 ? 6 : 28)	
remBinsPass2 = (log2SbSize < 2 ? 2 : 4)	
firstPosMode0 = (i == lastSubBlock ? lastScanPos − 1 : numSbCoeff − 1)	
firstPosMode1 = −1	
firstPosMode2 = −1	
for(n = (i == firstPosMode0; n >= 0 && remBinsPass1 >= 3; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
numSigCoeff++	
abs_level_gt1_flag[n]	ae(v)
remBinsPass1− −	
if(abs_level_gt1_flag[n]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
if(remBinsPass2 > 0) {	
remBinsPass2− −	
if(remBinsPass2 == 0)	
firstPosMode1 = n − 1	
}	
}	

if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gt1_flag[n]	
if(dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
if(remBinsPass1 < 3)	
firstPosMode2 = n − 1	
}	
if(firstPosMode1 < firstPosMode2)	
firstPosMode1 = firstPosMode2	
for(n = numSbCoeff − 1; n >= firstPosMode2; n− −)	
if(abs_level_gt1_flag[n])	
abs_level_gt3_flag[n]	ae(v)
for(n = numSbCoeff − 1; n >= firstPosMode1; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(abs_level_gt3_flag[n])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] + 2 * (abs_level_gt3_flag[n] + abs_remainder[n])	
}	
for(n = firstPosMode1; n > firstPosMode2; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(abs_level_gt1_flag[n])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] + 2 * abs_remainder[n]	
}	
for(n = firstPosMode2; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
dec_abs_level[n]	ae(v)
if(AbsLevel[xC][yC] > 0)	
firstSigScanPosSb = n	
if(dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(dep_quant_enabled_flag || !sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(sig_coeff_flag[xC][yC] && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	

if(dep_quant_enabled_flag) {	
QState = startQStateSb	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(sig_coeff_flag[xC][yC])	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][par_level_flag[n]]	
} else {	
sumAbsLevel = 0	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(sig_coeff_flag[xC][yC]) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
if(tu_mts_flag[x0][y0] && (cIdx == 0))	
mts_idx[x0][y0][cIdx]	ae(v)
}	

transform_skip_flag chỉ ra liệu việc biến đổi có được bỏ qua trong khối liên quan hay không. transform_skip_flag có thể là phần tử cú pháp của cờ bỏ qua biến đổi. Khối liên quan có thể là khối lập mã (coding block, CB) hoặc khối biến đổi (transform block, TB). Liên quan đến các thủ tục biến đổi (và lượng tử hóa) và lập mã phần dư, CB và TB có thể được sử dụng theo cách thay thế được cho nhau. Ví dụ, như được mô tả trên đây, các mẫu phần dư có thể được dẫn ra cho CB, và các hệ số biến đổi (được lượng tử hóa) có thể được dẫn ra thông qua việc biến đổi và lượng tử hóa đối với các mẫu phần dư, và thông qua thủ tục lập mã phần dư, thông tin (ví dụ, các phần tử cú pháp) chỉ ra một cách hiệu quả vị trí, độ lớn, dấu, v.v. của các hệ số biến đổi (được lượng tử hóa) có thể được tạo ra và được báo hiệu. Các hệ số biến đổi được lượng tử hóa có thể được gọi đơn giản là các hệ số biến đổi. Nói chung, khi CB không lớn hơn TB lớn nhất, kích thước của CB có thể giống như kích thước của TB, và trong trường hợp này, khối mục tiêu cần được biến đổi (và được lượng tử hóa) và phần dư được lập mã có thể được gọi là CB hoặc TB.

Trong khi đó, khi CB lớn hơn TB lớn nhất, khối mục tiêu cần được biến đổi (và được lượng tử hóa) và phần dư được lập mã có thể được gọi là TB. Ở dưới đây, việc các phần tử cú pháp liên quan đến việc lập mã phần dư được báo hiệu theo các đơn vị là các khối biến đổi (TB) sẽ được mô tả nhưng đây là ví dụ và TB có thể được sử dụng theo cách thay thế được cho nhau với các khối lập mã (CB) như được mô tả trên đây.

Trong khi đó, các phần tử cú pháp mà chúng được báo hiệu sau khi cờ bỏ qua biến đổi được báo hiệu có thể là giống như các phần tử cú pháp được bộc lộ trong bảng 6 dưới đây, và các phần mô tả chi tiết về các phần tử cú pháp này được mô tả dưới đây.

Trong khi đó, phương pháp báo hiệu tu_mts_idx có thể được đề xuất, không giống các phương án nêu trên trong đó các phần tử cú pháp được truyền.

Cụ thể, phương pháp báo hiệu tu_mts_idx trong bản thảo VVC 3 thông dụng có thể được so sánh với phương pháp báo hiệu tu_mts_idx được đề xuất này như sau.

Bảng 2

Bản thảo VVC 3	Được đề xuất

transform_unit( )
tu_cbf_luma
...
if(... tu_cbf_luma &&
(tbWidth <= 32) &&
(tbHeight <= 32) ...)
tu_mts_flag
residual_coding(cIdx)
if((cIdx ! = 0 || !tu_mts_flag) &&
(log2TbWidth <= 2) &&
(log2TbHeight <= 2))
transform_skip_flag[cIdx]
... /* coefficient parsing */ ...
if(tu_mts_flag && cIdx = = 0)
mts_idx
	transform_unit( )
	tu_cbf_luma
	if(... tu_cbf_luma &&
	(tbWidth <= 32) &&
	(tbHeight <= 32) ...)
	tu_mts_idx

Như được thể hiện trong bảng 2, theo phương pháp thông dụng, cờ bỏ qua biến đổi có thể được phân tích cú pháp sau khi cờ MTS cho khối hiện tại được phân tích cú pháp, và sau đó chỉ số MTS có thể được lập mã. Ở đây, việc lập mã trên chỉ số MTS có thể được thực hiện qua việc nhị phân hóa chiều dài cố định, và chiều dài bit cố định cho chỉ số MTS có thể là 2.

Mặt khác, theo phương pháp được đề xuất, chỉ số MTS có thể được lập mã mà không phải phân tích cú pháp thêm cờ bỏ qua biến đổi và cờ MTS, và việc nhị phân hóa một ngôi bị cắt cụt có thể được sử dụng trong việc lập mã chỉ số MTS. Ở đây, chỉ số

MTS có thể chỉ ra liệu việc biến đổi có được áp dụng cho thông tin phần dư của khối hiện tại hay không, và có thể chỉ ra liệu MTS có được áp dụng hay không. Tức là, phương pháp được đề xuất có thể là phương pháp trong đó cờ bỏ qua biến đổi, cờ MTS, và chỉ số MTS được báo hiệu dưới dạng một phần tử cú pháp. Trong phương pháp được đề xuất, ngăn thứ nhất của chỉ số MTS có thể chỉ ra liệu việc biến đổi có được áp dụng cho thông tin phần dư của khối hiện tại hay không, và ngăn thứ hai của chỉ số MTS có thể chỉ ra liệu MTS có được áp dụng hay không hoặc có thể chỉ ra nhân biến đổi cần được áp dụng.

Trong phương pháp được đề xuất, ngữ nghĩa được chỉ ra bởi giá trị của chỉ số MTS và giá trị nhị phân hóa của nó có thể như được thể hiện trong bảng sau đây.

Bảng 3

tu_mts_idx	loại biến đổi		việc nhị phân hóa		
	ngang	dọc	MTS & TS được cho phép	MTS được cho phép	TS được cho phép
0	DCT-II	DCT-II	0	0	0
1	SKIP	SKIP	10	-	1
2	DST-VII	DST-VII	110	10	-
3	DCT-VIII	DST-VII	1110	110	-
4	DST-VII	DCT-VIII	11110	1110	-
5	DCT-VIII	DCT-VIII	11111	1111	-

Ví dụ, nếu giá trị của chỉ số MTS là 0 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại được áp dụng, MTS không được áp dụng, và loại nhân biến đổi ngang và loại nhân biến đổi dọc là DCT-2. Thêm vào đó, nếu giá trị của chỉ số MTS là 1 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại không được áp dụng (cụ thể là MTS không được áp dụng, và loại nhân biến đổi không được chỉ ra). Thêm vào đó, nếu giá trị của chỉ số MTS là 2 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, và loại nhân biến đổi ngang và loại nhân biến đổi dọc là DST-7. Thêm vào đó, nếu giá trị của chỉ số MTS là 3 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, loại nhân biến đổi ngang là DCT-8, và loại nhân biến đổi dọc là DST-7. Thêm vào đó, nếu giá trị của chỉ số MTS là 4 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, loại nhân biến đổi ngang là DST-7, và loại nhân biến đổi dọc là DCT-8. Thêm vào đó, nếu giá trị của chỉ số MTS là 5 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, và loại nhân biến đổi ngang và loại nhân biến đổi dọc là DCT-8.

Theo cách khác, một ví dụ khác về ngữ nghĩa được chỉ ra bởi giá trị của chỉ số MTS và giá trị nhị phân hóa của nó có thể như được thể hiện trong bảng sau đây.

Bảng 4

tu_mts_idx	loại biến đổi		việc nhị phân hóa		
	ngang	dọc	MTS & TS được cho phép	MTS được cho phép	TS được cho phép
0	SKIP	SKIP	0	-	0
1	DCT-II	DCT-II	10	0	1
2	DST-VII	DST-VII	110	10	-
3	DCT-VIII	DST-VII	1110	110	-
4	DST-VII	DCT-VIII	11110	1110	-
5	DCT-VIII	DCT-VIII	11111	1111	-

Ví dụ, nếu giá trị của chỉ số MTS là 0 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại không được áp dụng (cụ thể là MTS không được áp dụng, và loại nhân biến đổi không được chỉ ra). Thêm vào đó, nếu giá trị của chỉ số MTS là 1 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại được áp dụng, MTS không được áp dụng, và loại nhân biến đổi ngang và loại nhân biến đổi dọc là DCT-2. Thêm vào đó, nếu giá trị của chỉ số MTS là 2 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, và loại nhân biến đổi ngang và loại nhân biến đổi dọc là DST-7. Thêm vào đó, nếu giá trị của chỉ số MTS là 3 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, loại nhân biến đổi ngang là DCT-8, và loại nhân biến đổi dọc là DST-7. Thêm vào đó, nếu giá trị của chỉ số MTS là 4 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, loại nhân biến đổi ngang là DST-7, và loại nhân biến đổi dọc là DCT-8. Thêm vào đó, nếu giá trị của chỉ số MTS là 5 thì chỉ số MTS có thể chỉ ra rằng việc biến đổi cho khối hiện tại và MTS được áp dụng, và loại nhân biến đổi ngang và loại nhân biến đổi dọc là DCT-8.

Trong khi đó, số lượng mô hình ngữ cảnh có thể không được thay đổi, và phương pháp chỉ rõ lượng gia (increment) chỉ số ngữ cảnh ctxInc cho mỗi ngăn của tu_mts_idx có thể như được thể hiện trong bảng sau đây.

Bảng 5

Phần tử cú pháp	binIdx					
	0	1	2	3	4	>= 5
tu_mts_idx (MTS & TS)	0	1...6 (1 + cqtDepth)	7	8	9	na
tu_mts_idx (MTS)	1...6 (1 + cqtDepth)	7	8	9	na	na
tu_mts_idx (TS)	0	na	na	na	na	na

Chỉ số MTS được đề xuất cũng có thể được thể hiện bởi chỉ số MTS được hợp nhất.

Các phần tử cú pháp liên quan đến việc mã hóa/giải mã dữ liệu phần dư bao gồm chỉ số MTS được hợp nhất này có thể như được thể hiện trong bảng sau đây.

Bảng 6

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if((mts_idx[x0][y0] > 0 || (cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6)) && cIdx = = 0 && log2TbWidth > 4)	
log2TbWidth = 4	
else	
log2TbWidth = Min(log2TbWidth, 5)	
if(mts_idx[x0][y0] > 0 || (cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6)) && cIdx = = 0 && log2TbHeight > 4)	
log2TbHeight = 4	
else	
log2TbHeight = Min(log2TbHeight, 5)	
last_sig_coeff_x_prefix	ae(v)
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2SbSize = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
numSbCoeff = 1 << (log2SbSize << 1)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − 2 * log2SbSize)) − 1	
do {	
if(lastScanPos = = 0) {	
lastScanPos = numSbCoeff	
lastSubBlock− −	
}	
lastScanPos− −	
xS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize][lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize][lastSubBlock][1]	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][lastScanPos][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][lastScanPos][1]	

} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
numSigCoeff = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i− −) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize][lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize][lastSubBlock][1]	
inferSbDcSigCoeffFlag = 0	
if((i < lastSubBlock) && (i > 0)) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
remBinsPass1 = (log2SbSize < 2 ? 6 : 28)	
remBinsPass2 = (log2SbSize < 2 ? 2 : 4)	
firstPosMode0 = (i == lastSubBlock ? lastScanPos − 1 : numSbCoeff − 1)	
firstPosMode1 = −1	
firstPosMode2 = −1	
for(n = (i == firstPosMode0; n >= 0 && remBinsPass1 >= 3; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
numSigCoeff++	
abs_level_gt1_flag[n]	ae(v)
remBinsPass1− −	
if(abs_level_gt1_flag[n]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gt3_flag[n]	ae(v)
remBinsPass1− −	
}	

if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gt1_flag[n] + 2 * abs_level_gt3_flag[n]	
if(dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
if(remBinsPass1 < 3)	
firstPosMode2 = n − 1	
}	
for(n = numSbCoeff − 1; n >= firstPosMode2; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(abs_level_gt3_flag[n])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] + 2 * abs_remainder[n]	
}	
for(n = firstPosMode2; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
dec_abs_level[n]	ae(v)
if(AbsLevel[xC][yC] > 0)	
firstSigScanPosSb = n	
if(dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(dep_quant_enabled_flag || !sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(sig_coeff_flag[xC][yC] && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	

```
if( dep_quant_enabled_flag ) {
  QState = startQStateSb
  for( n = numSbCoeff − 1; n >= 0; n− − ) {
    xC = ( xS << log2SbSize ) +
        DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ]
    yC = ( yS << log2SbSize ) +
        DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ]
    if( sig_coeff_flag[ xC ][ yC ] )
      TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =
          ( 2 * AbsLevel[ xC ][ yC ] − ( QState > 1 ? 1 : 0 ) ) *
          ( 1 − 2 * coeff_sign_flag[ n ] )
    QState = QStateTransTable[ QState ][ par_level_flag[ n ] ]
  } else {
    sumAbsLevel = 0
    for( n = numSbCoeff − 1; n >= 0; n− − ) {
      xC = ( xS << log2SbSize ) +
          DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ]
      yC = ( yS << log2SbSize ) +
          DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ]
      if( sig_coeff_flag[ xC ][ yC ] ) {
        TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =
            AbsLevel[ xC ][ yC ] * ( 1 − 2 * coeff_sign_flag[ n ] )
        if( signHidden ) {
          sumAbsLevel += AbsLevel[ xC ][ yC ]
          if( ( n == firstSigScanPosSb ) && ( sumAbsLevel % 2 ) == 1 ) )
            TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =
                −TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ]
        }
      }
    }
  }
}
```

Tham chiếu đến bảng 6 trên đây, last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, coded_sub_block_flag, sig_coeff_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag, abs_remainder, dec_abs_level, và/hoặc coeff_sign_flag có thể được mã hóa/giải mã.

Trong một phương án, thiết bị mã hóa có thể mã hóa thông tin vị trí (x, y) của hệ số biến đổi khác không cuối cùng trong khối biến đổi dựa trên các phần tử cú pháp last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, và last_sig_coeff_y_suffix. Cụ thể hơn, last_sig_coeff_x_prefix thể hiện tiền tố của vị trí cột của hệ số quan trọng cuối cùng theo thứ tự quét bên trong khối biến đổi, last_sig_coeff_y_prefix thể hiện tiền tố của vị trí hàng của hệ số quan trọng cuối cùng theo thứ tự quét bên trong khối biến đổi, last_sig_coeff_x_suffix thể hiện hậu tố của vị

trí cột của hệ số quan trọng cuối cùng theo thứ tự quét bên trong khối biến đổi, và last_sig_coeff_y_suffix thể hiện hậu tố của vị trí hàng của hệ số quan trọng cuối cùng theo thứ tự quét bên trong khối biến đổi. Ở đây, hệ số quan trọng có thể thể hiện hệ số khác không (0). Thêm vào đó, thứ tự quét có thể là thứ tự quét theo đường chéo bên phải. Theo cách khác, thứ tự quét có thể là thứ tự quét ngang hoặc thứ tự quét dọc. Thứ tự quét có thể được xác định dựa trên liệu việc dự đoán nội/liên có được áp dụng cho khối mục tiêu (CB hoặc CB bao gồm TB) hay không, và/hoặc chế độ dự đoán nội/liên cụ thể.

Sau đó, thiết bị mã hóa có thể chia khối biến đổi thành các khối con 4x4, và sau đó chỉ ra liệu có hệ số khác không trong khối con hiện tại hay không nhờ sử dụng phần tử cú pháp 1 bit coded_sub_block_flag cho mỗi khối con 4x4.

Nếu giá trị của coded_sub_block_flag là 0 thì không có thêm thông tin cần được truyền, và do đó, thiết bị mã hóa có thể chấm dứt quy trình mã hóa trên khối con hiện tại. Ngược lại, nếu giá trị của coded_sub_block_flag là 1 thì thiết bị mã hóa có thể tiếp tục thực hiện quy trình mã hóa trên sig_coeff_flag. Vì khối con bao gồm hệ số khác không cuối cùng không đòi hỏi việc mã hóa cho coded_sub_block_flag và khối con bao gồm thông tin DC của khối biến đổi có xác suất cao là bao gồm hệ số khác không, coded_sub_block_flag có thể không được lập mã và giá trị của nó có thể được giả định là 1.

Nếu giá trị của coded_sub_block_flag là 1 và do đó xác định được rằng hệ số khác không có tồn tại trong khối con hiện tại thì thiết bị mã hóa có thể mã hóa sig_coeff_flag có giá trị nhị phân theo thứ tự quét ngược. Thiết bị mã hóa có thể mã hóa phần tử cú pháp1 bit sig_coeff_flag cho mỗi hệ số biến đổi tùy theo thứ tự quét. Nếu giá trị của hệ số biến đổi tại vị trí quét hiện tại không phải là 0 thì giá trị của sig_coeff_flag có thể là 1. Ở đây, trong trường hợp của khối con bao gồm hệ số khác không cuối cùng, sig_coeff_flag không cần được mã hóa cho hệ số khác không cuối cùng này, vì vậy quy trình lập mã cho khối con có thể được lược bỏ. Việc lập mã thông tin mức có thể được thực hiện chỉ khi sig_coeff_flag là 1, và bốn phần tử cú pháp có thể được sử dụng trong quy trình mã hóa thông tin mức này. Cụ thể hơn, mỗi sig_coeff_flag[xC][yC] có thể chỉ ra liệu mức (giá trị) của hệ số biến đổi tương ứng tại mỗi vị trí hệ số biến đổi (xC, yC)

trong TB hiện tại có khác không hay không. Trong một phương án, sig_coeff_flag có thể tương ứng với ví dụ về phần tử cú pháp của cờ hệ số quan trọng chỉ ra liệu hệ số biến đổi được lượng tử hóa có là hệ số quan trọng khác không hay không.

Giá trị mức còn lại sau khi mã hóa cho sig_coeff_flag có thể được dẫn ra như được thể hiện trong phương trình sau đây. Tức là, phần tử cú pháp remAbsLevel chỉ ra giá trị mức cần được mã hóa có thể được dẫn ra từ phương trình sau đây.

Phương trình 1

$$remAbsLevel = |coeff| - 1$$

Ở đây, coeff nghĩa là giá trị hệ số biến đổi thực sự.

Thêm vào đó, abs_level_gt1_flag có thể chỉ ra liệu remAbsLevel' của vị trí quét tương ứng (n) có lớn hơn 1 hay không. Ví dụ, khi giá trị của abs_level_gt1_flag là 0, giá trị tuyệt đối của hệ số biến đổi của vị trí tương ứng có thể là 1. Thêm vào đó, khi giá trị của abs_level_gt1_flag là 1, remAbsLevel chỉ ra giá trị mức cần được mã hóa sau có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 2

$$remAbsLevel = remAbsLevel - 1$$

Thêm vào đó, giá trị hệ số ít quan trọng nhất (LSB) của remAbsLevel được mô tả trong Phương trình 2 được mô tả trên đây có thể được mã hóa như trong Phương trình 3 dưới đây thông qua par_level_flag.

Phương trình 3

$$par\_level\_flag = |coeff| \,\&\, 1$$

Ở đây, par_level_flag[n] có thể chỉ ra số chẵn lẻ của (giá trị) mức hệ số biến đổi tại vị trí quét n.

Giá trị mức hệ số biến đổi remAbsLevel mà nó cần được mã hóa sau khi thực hiện việc mã hóa par_level_flag có thể được cập nhật như được thể hiện dưới đây trong phương trình sau đây.

Phương trình 4

remAbsLevel = remAbsLevel >> 1

abs_level_gt3_flag có thể chỉ ra liệu remAbsLevel' của vị trí quét tương ứng (n) có lớn hơn 3 hay không. Việc mã hóa cho abs_remainder có thể được thực hiện chỉ trong trường hợp khi rem_abs_gt3_flag bằng 1. Mối quan hệ giữa giá trị hệ số biến đổi thực sự coeff và mỗi phần tử cú pháp có thể như được thể hiện dưới đây trong phương trình sau đây.

Phương trình 5

| coeff | = sig_coeff_flag + abs_level_gt1_flag + par_level_flag + 2 * (abs_level_gt3_flag + abs_remainder)

Thêm vào đó, bảng sau đây chỉ ra các ví dụ liên quan đến phương trình 5 được mô tả trên đây.

Bảng 7

|coeff|	sig_coeff_flag	abs_level_gt1_flag	par_level_flag	abs_level_gt3_flag	abs_remainder
0	0				
1	1	0			
2	1	1	0		
3	1	1	1	0	
4	1	1	0	1	0
5	1	1	1	1	0
6	1	1	0	1	1
7	1	1	1	1	1
8	1	1	0	1	2
9	1	1	1	1	2
10	1	1	0	1	3
11	1	1	1	1	3
...	...	...	...		

Ở đây, | coeff | chỉ ra (giá trị) mức hệ số biến đổi và cũng có thể được chỉ ra là AbsLevel cho hệ số biến đổi. Thêm vào đó, dấu của mỗi hệ số có thể được mã hóa bằng cách sử dụng coeff_sign_flag, mà nó là ký hiệu 1 bit.

Trong khi đó, để làm ví dụ khác với ví dụ trong phương án nêu trên về việc truyền các phần tử cú pháp, sơ đồ để lập mã các phần dư khác nhau tùy theo liệu việc bỏ qua biến đổi có được áp dụng cho việc lập mã phần dư hay không, tức là, phương án về việc truyền các phần tử cú pháp phần dư khác nhau tùy theo liệu việc bỏ qua biến đổi có được áp dụng hay không có thể được đề xuất.

Các phần tử cú pháp cho việc lập mã phần dư theo ví dụ nêu trên có thể như được

thể hiện trong các bảng sau đây.

Bảng 8

transform_unit(x0, y0, tbWidth, tbHeight, treeType, subTuIndex, chType) {	**Descriptor**
transform_skip_flag[x0][y0][0]	ae(v)
if(!transform_skip_flag[x0][y0][0] || slice_ts_residual_coding_disabled_flag)	
residual_coding(x0, y0, Log2(tbWidth), Log2(tbHeight), 0)	
else	
residual_ts_coding(x0, y0, Log2(tbWidth), Log2(tbHeight), 0)	
}	
if(tu_cbf_cb[xC][yC] && treeType != DUAL_TREE_LUMA) {	
if(sps_transform_skip_enabled_flag && !BdpcmFlag[x0][y0][1] && wC <= MaxTsSize && hC <= MaxTsSize && !cu_sbt_flag)	
transform_skip_flag[xC][yC][1]	ae(v)
if(!transform_skip_flag[xC][yC][1] || slice_ts_residual_coding_disabled_flag)	
residual_coding(xC, yC, Log2(wC), Log2(hC), 1)	
else	
residual_ts_coding(xC, yC, Log2(wC), Log2(hC), 1)	
}	
if(tu_cbf_cr[xC][yC] && treeType != DUAL_TREE_LUMA && !(tu_cbf_cb[xC][yC] && tu_joint_cbcr_residual_flag[xC][yC])) {	
if(sps_transform_skip_enabled_flag && !BdpcmFlag[x0][y0][2] && wC <= MaxTsSize && hC <= MaxTsSize && !cu_sbt_flag)	
transform_skip_flag[xC][yC][2]	ae(v)
if(!transform_skip_flag[xC][yC][2] || slice_ts_residual_coding_disabled_flag)	
residual_coding(xC, yC, Log2(wC), Log2(hC), 2)	
else	
residual_ts_coding(xC, yC, Log2(wC), Log2(hC), 2)	
}	
}	

Bảng 9

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx = = 0 && log2TbWidth = = 5 && log2TbHeight < 6)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx = = 0 && log2TbWidth < 6 && log2TbHeight = = 5)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	
do {	
if(lastScanPos = = 0) {	
lastScanPos = numSbCoeff	
lastSubBlock− −	
}	
lastScanPos− −	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH] [lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH] [lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	

} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
if(lastSubBlock = = 0 && log2TbWidth >= 2 && log2TbHeight >= 2 && !transform_skip_flag[x0][y0][cIdx] && lastScanPos > 0)	
LfnstDcOnly = 0	
if((lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2) || (lastScanPos > 7 && (log2TbWidth = = 2 || log2TbWidth = = 3) && log2TbWidth = = log2TbHeight))	
LfnstZeroOutSigCoeffFlag = 0	
if((lastSubBlock > 0 || lastScanPos > 0) && cIdx = = 0)	
MtsDcOnly = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i− −) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][1]	
inferSbDcSigCoeffFlag = 0	
if(i < lastSubBlock && i > 0) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
if(coded_sub_block_flag[xS][yS] && (xS > 3 || yS > 3) && cIdx = = 0)	
MtsZeroOutSigCoeffFlag = 0	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i = = lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = firstPosMode0	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != LastSignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb = = −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	

}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n][1]	
if(ph_dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
firstPosMode1 = n − 1	
}	
for(n = firstPosMode0; n > firstPosMode1; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(abs_level_gtx_flag[n][1])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] +2 * abs_remainder[n]	
}	
for(n = firstPosMode1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS])	
dec_abs_level[n]	ae(v)
if(AbsLevel[xC][yC] > 0) {	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
if(ph_dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(ph_dep_quant_enabled_flag || !pic_sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	
if(ph_dep_quant_enabled_flag) {	
QState = startQStateSb	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0)	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
} else {	

sumAbsLevel = 0	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n = = firstSigScanPosSb) && (sumAbsLevel % 2) = = 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
}	

Bảng 10

residual_ts_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	**Descriptor**
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	
inferSbCbf = 1	
RemCcbs = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
for(i =0; i <= lastSubBlock; i++) {	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][1]	
if(i != lastSubBlock || !inferSbCbf)	
coded_sub_block_flag[xS][yS]	ae(v)
if(coded_sub_block_flag[xS][yS] && i < lastSubBlock)	
inferSbCbf = 0	
/* First scan pass */	
inferSbSigCoeffFlag = 1	
lastScanPosPass1 = −1	
for(n = 0; n <= numSbCoeff − 1 && RemCcbs >= 4; n++) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n != numSbCoeff − 1 || !inferSbSigCoeffFlag)) {	

sig_coeff_flag[xC][yC]	ae(v)
RemCcbs− −	
if(sig_coeff_flag[xC][yC])	
inferSbSigCoeffFlag = 0	
}	
CoeffSignLevel[xC][yC] = 0	
if(sig_coeff_flag[xC][yC] {	
coeff_sign_flag[n]	ae(v)
RemCcbs− −	
CoeffSignLevel[xC][yC] = (coeff_sign_flag[n] > 0 ? −1 : 1)	
abs_level_gtx_flag[n][0]	ae(v)
RemCcbs− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
RemCcbs− −	
}	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0]	
lastScanPosPass1 = n	
}	
/* Greater than X scan pass (numGtXFlags=5) */	
lastScanPosPass2 = −1	
for(n = 0; n <= numSbCoeff − 1 && RemCcbs >= 4; n++) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
AbsLevelPass2[xC][yC] = AbsLevelPass1[xC][yC]	
for(j = 1; j < 5; j++) {	
if(abs_level_gtx_flag[n][j − 1]) {	
abs_level_gtx_flag[n][j]	ae(v)
RemCcbs− −	
}	
AbsLevelPass2[xC][yC] + = 2 * abs_level_gtx_flag[n][j]	
}	
lastScanPosPass2 = n	
}	
/* remainder scan pass */	
for(n = 0; n <= numSbCoeff − 1; n++) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((n <= lastScanPosPass2 && AbsLevelPass2[xC][yC] >= 10) || (n > lastScanPosPass2 && n <= lastScanPosPass1 && AbsLevelPass1[xC][yC] >= 2) || (n > lastScanPosPass1 && coded_sub_block_flag[xS][yS]))	
abs_remainder[n]	ae(v)
if(n <= lastScanPosPass2)	
AbsLevel[xC][yC] = AbsLevelPass2[xC][yC] + 2 * abs_remainder[n]	
else if(n <= lastScanPosPass1)	
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] + 2 * abs_remainder[n]	
else { /* bypass */	

AbsLevel[xC][yC] = abs_remainder[n]	
if(abs_remainder[n])	
coeff_sign_flag[n]	ae(v)
}	
if(BdpcmFlag[x0][y0][cIdx] = = 0 && n <= lastScanPosPass1) {	
absLeftCoeff = xC > 0 ? AbsLevel[xC − 1][yC]) : 0	
absAboveCoeff = yC > 0 ? AbsLevel[xC][yC − 1]) : 0	
predCoeff = Max(absLeftCoeff, absAboveCoeff)	
if(AbsLevel[xC][yC] = = 1 && predCoeff > 0)	
AbsLevel[xC][yC] = predCoeff	
else if(AbsLevel[xC][yC] > 0 && AbsLevel[xC][yC] <= predCoeff)	
AbsLevel[xC][yC]− −	
}	
}	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (1 − 2 * coeff_sign_flag[n]) * AbsLevel[xC][yC]	
}	
}	

Theo phương án này, như được thể hiện trong bảng 8, việc lập mã phần dư có thể được chia theo giá trị của phần tử cú pháp transform_skip_flag của cờ bỏ qua biến đổi. Tức là, phần tử cú pháp khác nhau có thể được sử dụng cho việc lập mã phần dư dựa trên giá trị của cờ bỏ qua biến đổi (dựa trên liệu việc biến đổi có được bỏ qua hay không). Việc lập mã phần dư được sử dụng khi việc bỏ qua biến đổi không được áp dụng (tức là, khi việc biến đổi được áp dụng) có thể được gọi là việc lập mã phần dư thông thường (regular residual coding, RRC), và việc lập mã phần dư được sử dụng khi việc bỏ qua biến đổi không được áp dụng (tức là, khi việc biến đổi không được áp dụng) có thể được gọi là việc lập mã phần dư bỏ qua biến đổi (transform skip residual coding, TSRC). Bảng 9 trên đây có thể thể hiện phần tử cú pháp của việc lập mã phần dư khi giá trị của transform_skip_flag là 0, tức là, khi việc biến đổi được áp dụng, và bảng 20 trên đây có thể thể hiện phần tử cú pháp của việc lập mã phần dư khi giá trị của transform_skip_flag là 1, tức là, khi việc biến đổi không được áp dụng.

Cụ thể, ví dụ, cờ bỏ qua biến đổi chỉ ra việc liệu có bỏ qua việc biến đổi khối biến đổi hay không có thể được phân tích cú pháp, và việc liệu cờ bỏ qua biến đổi có là 1 hay không có thể được xác định. Nếu giá trị của cờ bỏ qua biến đổi là 1 thì, như được thể hiện trong bảng 10, các phần tử cú pháp sig_coeff_flag, coeff_sign_flag, abs_level_gtx_flag, và/hoặc abs_remainder cho hệ số phần dư của khối biến đổi có thể được phân tích cú pháp, và hệ số phần dư có thể được dẫn ra dựa trên các phần tử cú pháp này. Trong trường hợp này, các phần tử cú pháp có thể được phân tích cú pháp theo cách tuần tự, và thứ tự phân tích cú pháp có thể được thay đổi. Thêm vào đó,

abs_level_gtx_flag có thể thể hiện abs_level_gt1_flag, abs_level_gt3_flag, abs_level_gt5_flag, abs_level_gt7_flag, và/hoặc abs_level_gt9_flag. Ví dụ, abs_level_gtx_flag[n][j] có thể là cờ chỉ ra liệu giá trị tuyệt đối của mức hệ số biến đổi (hoặc giá trị thu được bằng cách dịch chuyển mức hệ số biến đổi đi 1 sang bên phải) tại vị trí quét n có lớn hơn (j<<1)+1 hay không. Điều kiện (j<<1)+1 có thể được thay thế theo cách tùy chọn bằng ngưỡng cụ thể chẳng hạn như ngưỡng thứ nhất, ngưỡng thứ hai, hoặc dạng tương tự.

Thêm vào đó, nếu giá trị của cờ bỏ qua biến đổi là 0, như được thể hiện trong bảng 9, thì các phần tử cú pháp cho hệ số phần dư của khối biến đổi, cụ thể là sig_coeff_flag, abs_level_gtx_flag, par_level_flag, abs_remainder, dec_abs_level, và coeff_sign_flag, có thể được phân tích cú pháp, và hệ số phần dư có thể được dẫn ra dựa trên các phần tử cú pháp này. Trong trường hợp này, các phần tử cú pháp có thể được phân tích cú pháp theo cách tuần tự, và thứ tự phân tích cú pháp có thể được thay đổi. Thêm vào đó, abs_level_gtx_flag có thể thể hiện abs_level_gt1_flag và/hoặc abs_level_gt3_flag. Ví dụ, abs_level_gtx_flag[n][0] có thể là ví dụ về cờ mức hệ số biến đổi thứ nhất abs_level_gt1_flag, và abs_level_gtx_flag[n][1] có thể là ví dụ về cờ mức hệ số biến đổi thứ hai abs_level_gt3_flag.

Như được mô tả trên đây, khi so sánh các phần tử cú pháp cho hệ số phần dư khi việc biến đổi không được áp dụng và các phần tử cú pháp cho hệ số phần dư khi việc biến đổi được áp dụng, phần tử cú pháp par_level_flag có thể không được mã hóa và giải mã. Khi hệ số phần dư có giá trị mức lớn, có khả năng cao là thông tin được truyền theo cách bị trùng lặp khi các phần tử cú pháp như sig_coeff_flag, par_level_flag, abs_level_gtx_flag, hoặc dạng tương tự cho tất cả các hệ số phần dư được lập mã, so với việc truyền trong đó giá trị mức của hệ số phần dư được trực tiếp trải qua việc nhị phân hóa. Do đó, trong phương án này, hiệu suất lập mã có thể được cải thiện bằng cách lược bỏ phần tử cú pháp par_level_flag khi áp dụng việc bỏ qua biến đổi có khả năng rằng giá trị mức của hệ số phần dư là cao.

Trong khi đó, CABAC đem lại hiệu quả hoạt động cao nhưng có nhược điểm về hiệu quả hoạt động thông lượng kém. Điều này bị gây ra bởi cỗ máy lập mã thông thường của CABAC. Việc mã hóa thông thường (cụ thể là việc lập mã qua cỗ máy lập mã thông

thường của CABAC) thể hiện sự phụ thuộc dữ liệu cao vì nó sử dụng trạng thái và khoảng xác suất được cập nhật thông qua việc lập mã ngăn trước đó, và có thể mất nhiều thời gian để đọc khoảng xác suất và xác định trạng thái hiện tại. Vấn đề về thông lượng của CABAC có thể được giải quyết bằng cách giới hạn số lượng ngăn được lập mã ngữ cảnh. Ví dụ, như được thể hiện trong bảng 1, bảng 6, bảng 9, hoặc bảng 10 được mô tả trên đây, tổng của các ngăn được sử dụng để biểu diễn sig_coeff_flag, abs_level_gt1_flag, par_level_flag, và abs_level_gt3_flag có thể được giới hạn đến số lượng ngăn phụ thuộc vào kích thước của khối tương ứng. Ví dụ, nếu khối tương ứng là khối có kích thước là 4x4, thì tổng của các ngăn cho sig_coeff_flag, abs_level_gt1_flag, par_level_flag, và abs_level_gt3_flag có thể được giới hạn là 32, và nếu khối tương ứng là khối có kích thước là 2x2, thì tổng của các ngăn cho sig_coeff_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag có thể được giới hạn là 8. Số lượng ngăn được giới hạn này có thể được thể hiện bởi remBinsPass1. Hoặc, ví dụ, để có thông lượng CABAC cao hơn, số lượng ngăn được lập mã ngữ cảnh có thể được giới hạn cho khối (CB hoặc TB) bao gồm CG mục tiêu lập mã. Nói cách khác, số lượng ngăn được lập mã ngữ cảnh có thể được giới hạn theo các đơn vị là các khối (CB hoặc TB). Ví dụ, khi kích thước của khối hiện tại là 16x16, số lượng ngăn được lập mã ngữ cảnh cho khối hiện tại có thể được giới hạn đến 1,75 lần số lượng điểm ảnh của khối hiện tại, tức là, 448, bất kể CG hiện tại.

Trong trường hợp này, nếu tất cả các ngăn được lập mã ngữ cảnh mà số lượng của chúng được giới hạn đều được sử dụng khi phần tử ngữ cảnh được lập mã thì thiết bị mã hóa có thể nhị phân hóa các hệ số còn lại thông qua phương pháp nhị phân hóa hệ số như được mô tả dưới đây, thay vì sử dụng CABAC, và có thể thực hiện việc mã hóa đi vòng. Nói cách khác, ví dụ, nếu số lượng ngăn được lập mã ngữ cảnh được lập mã cho CG 4x4 là 32, hoặc nếu số lượng ngăn được lập mã ngữ cảnh được lập mã cho CG 2x2 là 8, thì sig_coeff_flag, abs_level_gt1_flag, par_level_flag, và abs_level_gt3_flag mà chúng được lập mã bằng ngăn được lập mã ngữ cảnh có thể không còn được mã hóa, và có thể được mã hóa trực tiếp thành dec_abs_level như được thể hiện trong bảng 8 được mô tả dưới đây. Hoặc, ví dụ, khi số lượng ngăn được lập mã ngữ cảnh được lập mã cho khối 4x4 là 1,75 lần số lượng điểm ảnh của toàn bộ khối, tức là, khi được giới hạn đến

28, sig_coeff_flag, abs_level_gt1_flag, par_level_flag, và abs_level_gt3_flag được lập mã thành các ngăn được lập mã ngữ cảnh có thể không được mã hóa nữa, và có thể được mã hóa trực tiếp thành dec_abs_level như được thể hiện trong bảng 8 dưới đây.

Bảng 11

|coeff|	dec_abs_level
0	0
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
...	...

Giá trị |coeff| có thể được dẫn ra dựa trên dec_abs_level. Trong trường hợp này, giá trị hệ số biến đổi, cụ thể là |coeff|, có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 6

$$|\,coeff\,| = dec\_abs\_level$$

Thêm vào đó, coeff_sign_flag có thể chỉ ra dấu của mức hệ số biến đổi tại vị trí quét tương ứng n. Tức là, coeff_sign_flag có thể chỉ ra dấu của hệ số biến đổi tại vị trí quét tương ứng n. Thêm vào đó, mts_idx có thể chỉ ra các nhân biến đổi được áp dụng theo hướng ngang và hướng dọc cho các mẫu phần dư trong khối biến đổi hiện tại.

Fig.5 thể hiện ví dụ về các hệ số biến đổi trong khối 4x4.

Khối 4x4 trên Fig.5 thể hiện ví dụ về các hệ số được lượng tử hóa. Khối trên Fig.5 có thể là khối biến đổi 4x4, hoặc khối con 4x4 của khối biến đổi 8x8, 16x16, 32x32, hoặc 64x64. Khối 4x4 trên Fig.5 có thể thể hiện khối độ chói hoặc khối sắc độ.

Ví dụ, kết quả mã hóa cho các hệ số được quét theo đường chéo ngược trên Fig.5 có thể như được thể hiện trong bảng sau đây.

Bảng 12

scan_pos	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
coefficients	0	0	0	0	1	-1	0	2	0	3	-2	-3	4	6	-7	10
sig_coeff_flag	0	0	0	0	1	1	0	1	0	1	1	1	1	1		
abs_level_gt1_flag					0	0		1		1	1	1	1	1		
par_level_flag								0		1	0	1	0	0		
abs_level_gt3_flag													1	1		
abs_remainder													0	1		
dec_abs_level															7	10
coeff_sign_flag	0	0	0	0	0	1	0	0	0	0	1	1	0	0	1	0

Trong bảng 12 trên đây, scan_pos chỉ ra vị trí của các hệ số dựa trên việc quét theo đường chéo ngược. scan_pos 15 có thể là hệ số biến đổi cần được quét đầu tiên trong khối 4x4, cụ thể là tại góc dưới cùng bên phải, và scan_pos 0 có thể là hệ số biến đổi cần được quét cuối cùng, cụ thể là tại góc trên cùng bên trái. Trong khi đó, trong một phương án, scan_pos có thể được gọi là vị trí quét. Ví dụ, scan_pos 0 sẽ được gọi là vị trí quét 0.

Trong khi đó, như được mô tả trên đây, khi tín hiệu đầu vào không phải giá trị nhị phân mà là phần tử cú pháp, thiết bị mã hóa có thể biến đổi tín hiệu đầu vào thành giá trị nhị phân bằng cách nhị phân hóa giá trị của tín hiệu đầu vào. Thêm vào đó, thiết bị giải mã có thể giải mã phần tử cú pháp để dẫn ra giá trị được nhị phân hóa (tức là, ngăn được nhị phân hóa) của phần tử cú pháp này, và có thể giải nhị phân hóa giá trị được nhị phân hóa này để dẫn ra giá trị của phần tử cú pháp này. Quy trình nhị phân hóa có thể được thực hiện dưới dạng quy trình nhị phân hóa rice được cắt cụt (truncated rice, TR), quy trình nhị phân hóa Exp-Golomb (EGk) thứ tự thứ k, Exp-Golomb thứ tự thứ k được giới hạn (limited Exp-Golomb, limited EGk), quy trình nhị phân hóa chiều dài cố định (fixed-length, FL), hoặc dạng tương tự. Thêm vào đó, quy trình giải nhị phân hóa có thể thể hiện quy trình được thực hiện dựa trên quy trình nhị phân hóa TR, quy trình nhị phân hóa EGk, hoặc quy trình nhị phân hóa FL để dẫn ra giá trị của phần tử cú pháp.

Ví dụ, quy trình nhị phân hóa TR có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa TR có thể là cMax và cRiceParam cho phần tử cú pháp và yêu cầu đối với việc nhị phân hóa TR. Thêm vào đó, đầu ra của quy trình nhị phân hóa TR có thể là việc nhị phân hóa TR cho symbolVal mà nó là giá trị tương ứng với chuỗi ngăn.

Cụ thể, ví dụ, khi có mặt chuỗi ngăn hậu tố cho phần tử cú pháp, chuỗi ngăn TR

cho phần tử cú pháp này có thể là việc nối chuỗi ngăn tiền tố và chuỗi ngăn hậu tố này, và khi vắng mặt chuỗi ngăn hậu tố, chuỗi ngăn TR cho phần tử cú pháp có thể là chuỗi ngăn tiền tố này. Ví dụ, chuỗi ngăn tiền tố có thể được dẫn ra như được mô tả dưới đây.

Giá trị tiền tố của symbolVal cho phần tử cú pháp có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 7

$$prefixVal = symbolVal >> cRiceParam$$

Ở đây, prefixVal có thể biểu thị giá trị tiền tố của symbolVal. Tiền tố (cụ thể là chuỗi ngăn tiền tố) của chuỗi ngăn TR của phần tử cú pháp có thể được dẫn ra như được mô tả dưới đây.

Ví dụ, nếu prefixVal nhỏ hơn cMax >> cRiceParam thì chuỗi ngăn tiền tố có thể là chuỗi bit có chiều dài prefixVal+1, được đánh chỉ số bởi binIdx. Tức là, nếu prefixVal nhỏ hơn cMax >> cRiceParam thì chuỗi ngăn tiền tố có thể là chuỗi bit mà số lượng bit của nó là prefixVal+1, được chỉ ra bởi binIdx. Ngăn dành cho binIdx nhỏ hơn prefixVal có thể là bằng 1. Thêm vào đó, ngăn cho binIdx giống như prefixVal có thể là bằng 0.

Ví dụ, chuỗi ngăn được dẫn ra thông qua việc nhị phân hóa một ngôi cho prefixVal có thể như được thể hiện trong bảng sau đây.

Bảng 13

prefixVal	**Chuỗi ngăn**					
0	0					
1	1	0				
2	1	1	0			
3	1	1	1	0		
4	1	1	1	1	0	
5	1	1	1	1	1	0
...						
binIdx	0	1	2	3	4	5

Trong khi đó, nếu prefixVal không nhỏ hơn cMax >> cRiceParam thì chuỗi ngăn tiền tố có thể là chuỗi bit trong đó chiều dài là cMax >> cRiceParam và tất cả các bit là 1.

Thêm vào đó, nếu cMax lớn hơn symbolVal và nếu cRiceParam lớn hơn 0 thì chuỗi ngăn hậu tố ngăn của chuỗi ngăn TR có thể có mặt. Ví dụ, chuỗi ngăn hậu tố có thể được

dẫn ra như được mô tả dưới đây.

Giá trị hậu tố của symbolVal cho phần tử cú pháp có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 8

$$suffixVal = symbolVal - ((prefixVal) << cRiceParam)$$

Ở đây, suffixVal có thể biểu thị giá trị hậu tố của symbolVal.

Hậu tố của chuỗi ngăn TR (cụ thể là chuỗi ngăn hậu tố) có thể được dẫn ra dựa trên quy trình nhị phân hóa FL cho suffixVal mà giá trị cMax của nó là (1 << cRiceParam)-1.

Trong khi đó, nếu giá trị của tham số đầu vào, cụ thể là cRiceParam, là 0 thì việc nhị phân hóa TR có thể là nhị phân hóa một ngôi bị cắt cụt một cách chuẩn xác, và có thể luôn sử dụng giá trị cMax giống như giá trị lớn nhất có thể của phần tử cú pháp cần được giải mã.

Thêm vào đó, ví dụ, quy trình nhị phân hóa EGk có thể được thực hiện như sau. Phần tử cú pháp được lập mã bằng ue(v) có thể là phần tử cú pháp được trải qua việc lập mã Exp-Golomb.

Ví dụ, quy trình nhị phân hóa Exp-Golomb thứ tự thứ 0 (0-th order Exp-Golomb, EG0) có thể được thực hiện như sau.

Quy trình phân tích cú pháp cho phần tử cú pháp có thể bắt đầu bằng việc đọc bit bao gồm bit khác không thứ nhất bắt đầu tại vị trí hiện tại của luồng bit và đếm số lượng bit dẫn đầu mà chúng bằng 0. Quy trình này có thể được biểu diễn như được thể hiện trong bảng sau đây.

Bảng 14

```
leadingZeroBits = −1
for( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 )
```

Thêm vào đó, biến số ‘codeNum’ có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 9

$$codeNum = 2^{leadingZeroBits} - 1 + read\_bits(leadingZeroBits)$$

Ở đây, giá trị được trả lại từ read_bits(leadingZeroBits), tức là, giá trị được chỉ ra bởi read_bits(leadingZeroBits), có thể được diễn dịch như sự biểu diễn nhị phân của số nguyên không dấu cho bit quan trọng nhất được ghi nhận đầu tiên.

Cấu trúc của mã Exp-Golomb trong đó chuỗi bit được chia thành bit "tiền tố" và bit "hậu tố" có thể được biểu diễn như được thể hiện trong bảng sau đây.

Bảng 15

Dạng chuỗi bit	Khoảng của codeNum
1	0
0 1 x_0	1..2
0 0 1 x_1 x_0	3..6
0 0 0 1 x_2 x_1 x_0	7..14
0 0 0 0 1 x_3 x_2 x_1 x_0	15..30
0 0 0 0 0 1 x_4 x_3 x_2 x_1 x_0	31..62
...	...

Bit "tiền tố" có thể là bit được phân tích cú pháp như được mô tả trên đây để tính leadingZeroBits, và có thể được thể hiện bởi 0 hoặc 1 của chuỗi bit trong bảng 15. Tức là, chuỗi bit được bộc lộ bởi 0 hoặc 1 trong bảng 15 trên đây có thể thể hiện chuỗi bit tiền tố. Bit "hậu tố" có thể là bit được phân tích cú pháp khi tính codeNum, và có thể được thể hiện bởi xi trong bảng 15 trên đây. Tức là, chuỗi bit được bộc lộ dưới dạng xi trong bảng 15 trên đây có thể thể hiện chuỗi bit hậu tố. Ở đây, i có thể là giá trị trong phạm vi LeadingZeroBits-1. Thêm vào đó, mỗi xi có thể bằng 0 hoặc 1.

Chuỗi bit được gán cho codeNum có thể như được thể hiện trong bảng sau đây.

Bảng 16

Chuỗi bit	codeNum
1	0
0 1 0	1
0 1 1	2
0 0 1 0 0	3
0 0 1 0 1	4
0 0 1 1 0	5
0 0 1 1 1	6
0 0 0 1 0 0 0	7
0 0 0 1 0 0 1	8
0 0 0 1 0 1 0	9
...	...

Nếu bộ mô tả của phần tử cú pháp là ue(v), tức là, nếu phần tử cú pháp được lập mã bằng ue(v), thì giá trị của phần tử cú pháp này có thể là bằng codeNum.

Thêm vào đó, ví dụ, quy trình nhị phân hóa EGk có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa EGk có thể là yêu cầu đối với việc nhị phân hóa EGk. Thêm vào đó, đầu ra của quy trình nhị phân hóa EGk có thể là việc nhị phân hóa EGk cho symbolVal, cụ thể là giá trị tương ứng với chuỗi ngăn.

Chuỗi bit của quy trình nhị phân hóa EGk cho symbolVal có thể được dẫn ra như sau.

Bảng 17

```
absV = Abs( symbolVal )
stopLoop = 0
do
    if( absV >= ( 1 << k ) ) {
        put( 1 )
        absV = absV − ( 1 << k )
        k++
    } else {
        put( 0 )
        while( k− − )
            put( ( absV >> k ) & 1 )
        stopLoop = 1
    }
while( !stopLoop )
```

Tham chiếu đến bảng 17 trên đây, giá trị nhị phân X có thể được thêm vào cuối của chuỗi ngăn thông qua mỗi lần gọi put(X). Ở đây, X có thể là 0 hoặc 1.

Thêm vào đó, ví dụ, quy trình nhị phân hóa EGk được giới hạn có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa EGk được giới hạn có thể là yêu cầu đối với việc nhị phân hóa EGk được giới hạn, tham số Rice riceParam, log2TransformRange để làm biến số thể hiện logarit nhị phân của giá trị tối đa, và maxPreExtLen để làm biến số thể hiện chiều dài mở rộng tiền tố tối đa. Thêm vào đó, đầu ra của quy trình nhị phân hóa EGk được giới hạn có thể là việc nhị phân hóa EGk được giới hạn cho symbolVal

dưới dạng giá trị tương ứng với chuỗi rỗng.

Chuỗi bit của quy trình nhị phân hóa EGk được giới hạn cho symbolVal có thể được dẫn ra như sau.

Bảng 18

```
codeValue = symbolVal >> riceParam
PrefixExtensionLength = 0
while( ( PrefixExtensionLength < maxPrefixExtensionLength ) &&
          ( codeValue > ( ( 2 << PrefixExtensionLength ) − 2 ) ) ) {
     PrefixExtensionLength++
     put( 1 )
}
if( PrefixExtensionLength == maxPrefixExtensionLength )
     escapeLength = log2TransformRange
else {
     escapeLength = PrefixExtensionLength + riceParam
     put( 0 )
}
symbolVal = symbolVal − ( ( ( 1 << PrefixExtensionLength ) − 1 ) << riceParam )
while( ( escapeLength− − ) > 0 )
     put( ( symbolVal >> escapeLength ) & 1 )
```

Thêm vào đó, ví dụ, quy trình nhị phân hóa FL có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa FL có thể là yêu cầu đối với việc nhị phân hóa FL và cMax cho phần tử cú pháp. Thêm vào đó, đầu ra của quy trình nhị phân hóa FL có thể là việc nhị phân hóa FL cho symbolVal dưới dạng giá trị tương ứng với chuỗi ngăn.

Việc nhị phân hóa FL có thể được tạo cấu hình bằng cách sử dụng chuỗi bit mà số lượng bit của nó có chiều dài cố định là symbolVal. Ở đây, bit có chiều dài cố định có thể là chuỗi bit số nguyên không dấu. Tức là, chuỗi bit cho symbolVal dưới dạng giá trị ký hiệu có thể được dẫn ra thông qua việc nhị phân hóa FL, và chiều dài bit (cụ thể là số lượng bit) của chuỗi bit này có thể là chiều dài cố định.

Ví dụ, chiều dài cố định này có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 10

$$fixedLength = Ceil(Log2(cMax + 1))$$

Việc đánh chỉ số các ngăn cho việc nhị phân hóa FL có thể là phương pháp sử dụng giá trị mà nó tăng lên theo thứ tự từ bit quan trọng nhất đến bit ít quan trọng nhất. Ví dụ, chỉ số ngăn liên quan đến bit quan trọng nhất có thể là binIdx = 0.

Trong khi đó, ví dụ, quy trình nhị phân hóa cho phần tử cú pháp abs_remainder trong thông tin phần dư có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa cho abs_remainder có thể là yêu cầu đối với việc nhị phân hóa phần tử cú pháp abs_remainder[n], thành phần màu cIdx, và vị trí độ chói (x0, y0). Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói trên cùng bên trái của hình ảnh.

Đầu ra của quy trình nhị phân hóa cho abs_remainder có thể là việc nhị phân hóa của abs_remainder (cụ thể là chuỗi ngăn được nhị phân hóa của abs_remainder). Các chuỗi ngăn khả dụng cho abs_remainder có thể được dẫn ra thông qua quy trình nhị phân hóa này.

Đầu tiên, lastAbsRemainder và lastRiceParam cho abs_remainder[n] có thể được dẫn ra như sau. Ở đây, lastAbsRemainder có thể biểu thị giá trị của abs_remainder được dẫn ra trước abs_remainder[n], và lastRiceParam có thể biểu thị tham số rice cRiceParam cho abs_remainder được dẫn ra trước abs_remainder[n].

Ví dụ, nếu quy trình dẫn ra lastAbsRemainder và lastRiceParam cho abs_remainder[n] được gọi ra đầu tiên cho khối con hiện tại, tức là, nếu quy trình của abs_remainder[n] cho hệ số biến đổi có thứ tự thứ nhất trên thứ tự quét được thực hiện trong số các hệ số biến đổi của khối con hiện tại, thì cả lastAbsRemainder và lastRiceParam có thể được thiết lập là 0.

Thêm vào đó, nếu không phải trường hợp trên đây, tức là, nếu quy trình này không được gọi ra đầu tiên cho khối con hiện tại, thì lastAbsRemainder và lastRiceParam có thể được thiết lập lần lượt giống như abs_remainder[n] và cRiceParam được dẫn ra trong lần gọi ra cuối cùng. Tức là, lastAbsRemainder có thể được dẫn ra là giá trị giống như abs_remainder[n] được lập mã trước abs_remainder[n] hiện đang cần được lập mã, và lastRiceParam có thể được dẫn ra là giá trị giống như cRiceParam cho abs_remainder[n] được lập mã trước abs_remainder[n].

Sau đó, tham số rice cRiceParam cho abs_remainder[n] hiện đang cần được lập mã có thể được dẫn ra dựa trên lastAbsRemainder và lastRiceParam. Ví dụ, tham số rice cRiceParam cho abs_remainder[n] hiện đang cần được lập mã có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 11

$$cRiceParam = Min(lastRiceParam + ((lastAbsRemainder > (3 * (1 << lastRiceParam))) ? 1 : 0), 3)$$

Thêm vào đó, ví dụ, cMax cho abs_remainder[n] hiện đang cần được lập mã có thể được dẫn ra dựa trên tham số rice cRiceParam. cMax có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 12

$$cMax = 6 << cRiceParam$$

Theo cách khác, ví dụ, tham số rice cRiceParam có thể được xác định dựa trên liệu việc bỏ qua biến đổi có được áp dụng cho khối hiện tại hay không. Tức là, nếu việc biến đổi không được áp dụng cho TB hiện tại bao gồm CG hiện tại, nói cách khác, nếu việc bỏ qua biến đổi được áp dụng cho TB hiện tại bao gồm CG hiện tại này, thì tham số rice cRiceParam có thể được dẫn ra là 1. Theo cách khác, nếu việc biến đổi được áp dụng cho TB hiện tại bao gồm CG hiện tại này, nói cách khác, nếu việc bỏ qua biến đổi không được áp dụng cho TB hiện tại bao gồm CG hiện tại này, thì tham số rice cRiceParam cho abs_remainder[n] hiện đang cần được lập mã như được mô tả trên đây có thể được dẫn ra là giá trị giống như cRiceParam cho abs_remainder[n] được lập mã trước đó.

Trong khi đó, việc nhị phân hóa cho abs_remainder, tức là, chuỗi ngăn cho abs_remainder, có thể là việc nối chuỗi ngăn tiền tố và chuỗi ngăn hậu tố khi có mặt chuỗi ngăn hậu tố này. Thêm vào đó, khi không có mặt chuỗi ngăn hậu tố, chuỗi ngăn cho abs_remainder có thể là chuỗi ngăn tiền tố.

Ví dụ, chuỗi ngăn tiền tố có thể được dẫn ra như được mô tả dưới đây.

Giá trị tiền tố prefixVal của abs_remainder[n] có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 13

prefixVal = Min(cMax, abs_remainder[n])

Tiền tố của chuỗi ngăn (cụ thể là chuỗi ngăn tiền tố) của abs_remainder[n] có thể được dẫn ra thông qua quy trình nhị phân hóa TR cho prefixVal, trong đó cMax và cRiceParam được sử dụng làm đầu vào.

Nếu chuỗi ngăn tiền tố là giống hệt như chuỗi bit trong đó tất cả các bit đều là 1 và chiều dài bit là 6 thì chuỗi ngăn hậu tố của chuỗi ngăn của abs_remainder[n] có thể tồn tại, và có thể được dẫn ra như được mô tả dưới đây.

Giá trị hậu tố suffixVal của abs_remainder có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 14

suffixVal = abs_remainder[n] − cMax

Chuỗi ngăn hậu tố của chuỗi ngăn của abs_remainder có thể được dẫn ra thông qua quy trình nhị phân hóa EGk được giới hạn cho suffixVal trong đó k được thiết lập là cRiceParam+1, riceParam được thiết lập là cRiceParam, và log2TransformRange được thiết lập là 15, và maxPreExtLen được thiết lập là 11.

Trong khi đó, ví dụ, quy trình nhị phân hóa cho phần tử cú pháp dec_abs_level trong thông tin phần dư có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa cho dec_abs_level có thể là yêu cầu đối với việc nhị phân hóa phần tử cú pháp dec_abs_level[n], thành phần màu cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như là logarit nhị phân của chiều rộng của khối biến đổi, và log2TbHeight như là logarit nhị phân của chiều cao của khối biến đổi. Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói trên cùng bên trái của hình ảnh.

Đầu ra của quy trình nhị phân hóa cho dec_abs_level có thể là việc nhị phân hóa dec_abs_level (cụ thể là chuỗi ngăn được nhị phân hóa của dec_abs_level). Các chuỗi ngăn khả dụng cho dec_abs_level có thể được dẫn ra thông qua quy trình nhị phân hóa này.

Tham số rice cRiceParam cho dec_abs_level[n] có thể được dẫn ra thông qua quy

trình dẫn ra tham số rice được thực hiện với đầu vào là thành phần màu cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như là logarit nhị phân của chiều rộng của khối biến đổi, và log2TbHeight như là logarit nhị phân của chiều cao của khối biến đổi. Quy trình dẫn ra tham số rice sẽ được mô tả chi tiết dưới đây.

Thêm vào đó, ví dụ, cMax cho dec_abs_level[n] có thể được dẫn ra dựa trên tham số rice cRiceParam. cMax có thể được dẫn ra như được thể hiện trong bảng sau đây.

Phương trình 15

$$cMax = 6 << cRiceParam$$

Trong khi đó, việc nhị phân hóa cho dec_abs_level[n], tức là, chuỗi ngăn cho dec_abs_level[n], có thể là việc nối chuỗi ngăn tiền tố và chuỗi ngăn hậu tố khi có mặt chuỗi ngăn hậu tố này. Thêm vào đó, khi không có mặt chuỗi ngăn hậu tố, chuỗi ngăn cho dec_abs_level[n] có thể là chuỗi ngăn tiền tố.

Ví dụ, chuỗi ngăn tiền tố có thể được dẫn ra như được mô tả dưới đây.

Giá trị tiền tố prefixVal của dec_abs_level[n] có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 16

$$prefixVal = Min(cMax, dec\_abs\_level[n])$$

Tiền tố của chuỗi ngăn (cụ thể là chuỗi ngăn tiền tố) của dec_abs_level[n] có thể được dẫn ra thông qua quy trình nhị phân hóa TR cho prefixVal, trong đó cMax và cRiceParam được sử dụng làm đầu vào.

Nếu chuỗi ngăn tiền tố là giống hệt như chuỗi bit trong đó tất cả các bit đều là 1 và chiều dài bit là 6 thì chuỗi ngăn hậu tố của chuỗi ngăn của dec_abs_level[n] có thể tồn tại, và có thể được dẫn ra như được mô tả dưới đây.

Quy trình dẫn ra tham số rice cho dec_abs_level[n] có thể là như sau.

Đầu vào của quy trình dẫn ra tham số rice có thể là chỉ số thành phần màu cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như là logarit nhị phân của chiều rộng của khối biến đổi, và log2TbHeight như là logarit nhị phân của

chiều cao của khối biến đổi. Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói trên cùng bên trái của hình ảnh. Thêm vào đó, đầu ra của quy trình dẫn ra tham số rice có thể là tham số rice cRiceParam.

Ví dụ, biến số locSumAbs có thể được dẫn ra tương tự với mã giả được bộc lộ trong bảng sau đây, dựa trên các phần tử cú pháp đã cho sig_coeff_flag[x][y] và mảng AbsLevel[x][y] cho khối biến đổi có chỉ số thành phần cIdx và vị trí độ chói trên cùng bên trái (x0, y0).

Bảng 19

```
locSumAbs = 0
if( xC < (1 << log2TbWidth) − 1 ) {
  locSumAbs += AbsLevel[ xC + 1 ][ yC ] − sig_coeff_flag[ xC + 1 ][ yC ]
  if( xC < (1 << log2TbWidth) − 2 )
    locSumAbs += AbsLevel[ xC + 2 ][ yC ] − sig_coeff_flag[ xC + 2 ][ yC ]
  if( yC < (1 << log2TbHeight) − 1 )
    locSumAbs += AbsLevel[ xC + 1 ][ yC + 1 ] − sig_coeff_flag[ xC + 1 ][ yC + 1 ]
}
if( yC < (1 << log2TbHeight) − 1 ) {
  locSumAbs += AbsLevel[ xC ][ yC + 1 ] − sig_coeff_flag[ xC ][ yC + 1 ]
  if( yC < (1 << log2TbHeight) − 2 )
    locSumAbs += AbsLevelPass1 [ xC ][ yC + 2 ] − sig_coeff_flag[ xC ][ yC + 2 ]
}
if( locSumAbs > 31 )
  locSumAbs = 31
```

Tham số rice cRiceParam có thể được dẫn ra như sau.

Ví dụ, tham số rice cRiceParam có thể được dẫn ra dựa trên biến số được dẫn ra locSumAbs và biến số s. Biến số s có thể được thiết lập là Max(0, QState – 1). Tức là, biến số s có thể được thiết lập là giá trị lớn nhất giữa 0 và QState – 1.

ZeroPos[n] và tham số rice cRiceParam được dẫn ra dựa trên biến số locSumAbs và biến số s có thể là như được thể hiện trong bảng sau đây.

Bảng 20

s	locSumAbs	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	cRiceParam	0	0	0	0	0	0	0	1	1	1	1	1	1	1	2	2
0	ZeroPos[n]	0	0	0	0	0	1	2	2	2	2	2	2	4	4	4	4
1	ZeroPos[n]	1	1	1	1	2	3	4	4	4	6	6	6	8	8	8	8
2	ZeroPos[n]	1	1	2	2	2	3	4	4	4	6	6	6	8	8	8	8
	locSumAbs	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	cRiceParam	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3
0	ZeroPos[n]	4	4	4	4	4	4	4	8	8	8	8	8	16	16	16	16
1	ZeroPos[n]	4	4	12	12	12	12	12	12	12	12	16	16	16	16	16	16
2	ZeroPos[n]	8	8	12	12	12	12	12	12	12	16	16	16	16	16	16	16

Ví dụ, tham chiếu đến bảng 20 trên đây, nếu locSumAbs nhỏ hơn hoặc bằng 6 thì cRiceParam có thể được thiết lập là 0. Nếu locSumAbs lớn hơn hoặc bằng 7 và nhỏ hơn hoặc bằng 13 thì cRiceParam có thể được thiết lập là 1. Nếu locSumAbs lớn hơn hoặc bằng 14 và nhỏ hơn hoặc bằng 27 thì cRiceParam có thể được thiết lập là 2. Nếu locSumAbs lớn hơn hoặc bằng 28 thì cRiceParam có thể được thiết lập là 3.

Thêm vào đó, tham chiếu đến bảng 20 trên đây, nếu s là 0 và locSumAbs nhỏ hơn hoặc bằng 4 thì ZeroPos[n] có thể được thiết lập là 0. Nếu s là 0 và locSumAbs là 5 thì ZeroPos[n] có thể được thiết lập là 1. Nếu s là 0 và locSumAbs lớn hơn hoặc bằng 6 và nhỏ hơn hoặc bằng 11 thì ZeroPos[n] có thể được thiết lập là 2. Nếu s là 0 và locSumAbs lớn hơn hoặc bằng 12 và nhỏ hơn hoặc bằng 22 thì ZeroPos[n] có thể được thiết lập là 4. Nếu s là 0 và locSumAbs lớn hơn hoặc bằng 23 và nhỏ hơn hoặc bằng 27 thì ZeroPos[n] có thể được thiết lập là 8. Nếu s là 0 và locSumAbs lớn hoặc hoặc bằng 28 thì ZeroPos[n] có thể được thiết lập là 16. Thêm vào đó, tham chiếu đến bảng 20 trên đây, nếu locSumAbs nhỏ hơn hoặc bằng 3 thì ZeroPos[n] có thể được thiết lập là 1. Nếu s là 1 và locSumAbs là 4 thì ZeroPos[n] có thể được thiết lập là 2. Nếu s là 1 và locSumAbs là 5 thì ZeroPos[n] có thể được thiết lập là 3. Nếu s là 1 và locSumAbs lớn hơn hoặc bằng 6 và nhỏ hơn hoặc bằng 8 thì ZeroPos[n] có thể được thiết lập là 4. Nếu s là 1 và locSumAbs lớn hơn hoặc bằng 9 và nhỏ hơn hoặc bằng 11 thì ZeroPos[n] có thể được thiết lập là 6. Nếu s là 1 và locSumAbs lớn hơn hoặc bằng 12 và nhỏ hơn hoặc bằng 15 thì ZeroPos[n] có thể được thiết lập là 8. Nếu s là 1 và locSumAbs lớn hơn hoặc bằng 16 và nhỏ hơn hoặc bằng 17 thì ZeroPos[n] có thể được thiết lập là 4. Nếu s là 1 và locSumAbs lớn hơn hoặc bằng 18 và nhỏ hơn hoặc bằng 25 thì ZeroPos[n] có thể được thiết lập là 12. Nếu locSumAbs lớn hơn hoặc bằng 26 và nhỏ hơn hoặc bằng 31 thì ZeroPos[n] có thể được thiết lập là 16. Thêm vào đó, tham chiếu đến bảng 20 trên đây, nếu s là 2 và locSumAbs nhỏ hơn hoặc bằng 1 thì ZeroPos[n] có thể được thiết lập

là 1. Nếu s là 2 và locSumAbs lớn hơn hoặc bằng 2 và nhỏ hơn hoặc bằng 4 thì ZeroPos[n] có thể được thiết lập là 2. Nếu s là 2 và locSumAbs là 5 thì ZeroPos[n] có thể được thiết lập là 3. Nếu s là 2 và locSumAbs lớn hơn hoặc bằng 6 và nhỏ hơn hoặc bằng 8 thì ZeroPos[n] có thể được thiết lập là 4. Nếu s là 2 và locSumAbs lớn hơn hoặc bằng 9 và nhỏ hơn hoặc bằng 11 thì ZeroPos[n] có thể được thiết lập là 6. Nếu s là 2 và locSumAbs lớn hơn hoặc bằng 12 và nhỏ hơn hoặc bằng 17 thì ZeroPos[n] có thể được thiết lập là 8. Nếu s là 2 và locSumAbs lớn hơn hoặc bằng 18 và nhỏ hơn hoặc bằng 24 thì ZeroPos[n] có thể được thiết lập là 12. Nếu s là 2 và locSumAbs lớn hoặc hoặc bằng 25 thì ZeroPos[n] có thể được thiết lập là 16.

Theo cách khác, ví dụ, ZeroPos[n] có thể được dẫn ra như được thể hiện trong phương trình sau đây, trong đó cRiceParam được chứa trong phương trình sau đây có thể được dẫn ra với tham chiếu đến bảng 20.

Phương trình 17

$$ZeroPos[n] = (QState < 2 ? 1 : 2) << cRiceParam$$

Thêm vào đó, giá trị hậu tố suffixVal của dec_abs_level[n] có thể được dẫn ra như được thể hiện trong phương trình sau đây.

Phương trình 18

$$suffixVal = dec\_abs\_level[\ n\ ] - cMax$$

Chuỗi ngăn hậu tố của chuỗi ngăn của dec_abs_level[n] có thể được dẫn ra thông qua quy trình nhị phân hóa EGk được giới hạn cho suffixVal trong đó k được thiết lập là cRiceParam+1, riceParam được thiết lập là cRiceParam, log2TransformRange được thiết lập là 15, và maxPreExtLen được thiết lập là 11.

Thêm vào đó, sáng chế đề xuất phương pháp cải biến nội dung được mô tả dưới đây trong phương pháp lập mã phần dư hiện có sao cho số liệu thống kê và đặc điểm tín hiệu của mức bỏ qua biến đổi (cụ thể là phần dư trong miền không gian) chỉ ra phần dư của việc dự đoán được lượng tử hóa được làm thích ứng với việc lập mã phần dư.

Thứ tự quét: Ví dụ, thứ tự quét cho các khối con trong khối TB và các hệ số phần dư trong các khối con này có thể là thứ tự quét theo đường chéo di chuyển từ dưới cùng

bên phải đến trên cùng bên trái. Tức là, thứ tự quét cho khối con trong khối TB và hệ số phần dư trong khối con có thể là thứ tự quét theo đường chéo để quét từ dưới cùng bên phải đến trên cùng bên trái. Theo cách khác, ví dụ, thứ tự quét cho khối con trong khối TB và hệ số phần dư trong khối con có thể là thứ tự quét theo đường chéo di chuyển từ trên cùng bên trái đến dưới cùng bên phải. Tức là, thứ tự quét cho khối con trong khối TB và hệ số phần dư trong khối con có thể là thứ tự theo quét đường chéo để quét từ trên cùng bên trái đến dưới cùng bên phải.

Không có vị trí của hệ số biến đổi khác không cuối cùng: Sau khi dự đoán, vì phần dư theo không gian được áp dụng và việc nén năng lượng dựa trên việc biến đổi không được thực hiện với việc bỏ qua biến đổi, tín hiệu phần dư (cụ thể là mẫu phần dư) có thể không còn có xác suất cao đối với số không (zero) kế tiếp hoặc mức không quan trọng ở dưới cùng bên phải của khối biến đổi. Do đó, trong trường hợp này, việc báo hiệu thông tin về vị trí quét của hệ số biến đổi khác không cuối cùng có thể được bỏ qua. Thay vào đó, khối con thứ nhất cần được lập mã đầu tiên có thể là khối con trên cùng bên trái trong khối biến đổi. Trong khi đó, hệ số phần dư khác không cũng có thể được gọi là hệ số quan trọng.

Khối con CBF: Bộ phận của việc báo hiệu thông tin về vị trí quét của hệ số biến đổi khác không cuối cùng sẽ cải biến việc báo hiệu CBF của khối con mà việc bỏ qua biến đổi được áp dụng với khối con này và khối con này có coded_sub_block_flag, như được mô tả dưới đây.

Do việc lượng tử hóa, trình tự của mức không quan trọng nêu trên có thể vẫn được tạo ra một cách cục bộ trong khối biến đổi. Do đó, thông tin về vị trí quét của hệ số biến đổi khác không cuối cùng có thể được loại bỏ như được mô tả trên đây, và coded_sub_block_flag có thể được lập mã cho tất cả các khối con.

Thêm vào đó, coded_sub_block_flag cho khối con (khối con trên cùng bên trái) cho vị trí tần số DC có thể thể hiện trường hợp đặc biệt. Ví dụ, trong bản thảo VVC 3, coded_sub_block_flag cho khối con trên cùng bên trái có thể không được báo hiệu và có thể được dẫn ra để luôn bằng 1. Khi vị trí quét của hệ số biến đổi khác không cuối cùng được đặt tại khối con khác khối con trên cùng bên trái, điều này có thể thể hiện rằng có ít nhất một mức quan trọng bên ngoài khối con DC (cụ thể là khối con trên cùng

bên trái). Kết quả là, coded_sub_block_flag cho khối con DC có thể được dẫn ra là 1, nhưng có thể bao gồm chỉ 0/mức không quan trọng. Nếu việc bỏ qua biến đổi được áp dụng cho khối hiện tại và không có thông tin về vị trí quét của hệ số biến đổi khác không cuối cùng như được mô tả trên đây thì coded_sub_block_flag cho mỗi khối con có thể được báo hiệu. Ở đây, coded_sub_block_flag cho khối con DC cũng có thể được đưa vào ngoại trừ trường hợp trong đó coded_sub_block_flag cho tất cả các khối con khác với khối con DC đã là 0. Trong khi đó, ví dụ, khi thứ tự quét theo đường chéo di chuyển từ dưới cùng bên phải đến trên cùng bên trái được áp dụng làm thứ tự quét của khối biến đổi và coded_sub_block_flag cho khối con DC không được báo hiệu, coded_sub_block_flag cho khối con DC có thể được dẫn ra là bằng 1 (inferDcSbCbf = 1). Do đó, vì khối con DC sẽ phải có ít nhất một mức quan trọng, nếu tất cả các sig_coeff_flag khác ngoài sig_coeff_flag cho vị trí thứ nhất (0,0) trong khối con DC là 0 thì sig_coeff_flag cho vị trí thứ nhất (0,0) này có thể không được báo hiệu, và có thể được dẫn ra là bằng 1 (inferSbDcSigCoeffFlag = 1).

Thêm vào đó, có thể có sự thay đổi về việc lập mô hình ngữ cảnh của coded_sub_block_flag. Ví dụ, chỉ số mô hình ngữ cảnh có thể được tính là tổng của coded_sub_block_flag của khối con bên trái của khối con hiện tại và coded_sub_block_flag của khối con trên cùng của khối con hiện tại và sự biệt lập theo logic của các coded_sub_block_flag này.

Việc lập mô hình ngữ cảnh sig_coeff_flag: Khuôn mẫu cục bộ của việc lập mô hình ngữ cảnh sig_coeff_flag có thể được cải biến để bao gồm chỉ vị trí bên trái NB0 và vị trí trên cùng NB1 của vị trí quét hiện tại. Phần bù (offset) mô hình ngữ cảnh có thể được dẫn ra là giá trị, cụ thể là sig_coeff_flag [NB0] + sig_coeff_flag [NB1], của vị trí lân cận quan trọng. Do đó, việc chọn các tập ngữ cảnh khác nhau phụ thuộc vào đường chéo d của khối biến đổi hiện tại có thể được loại bỏ. Kết quả là, ba mô hình ngữ cảnh và mô hình ngữ cảnh đơn lẻ có thể được thiết lập để lập mã cho sig_coeff_flag.

Việc lập mô hình ngữ cảnh abs_level_gt1_flag và par_level_flag: Mô hình ngữ cảnh đơn lẻ có thể được sử dụng cho abs_level_gt1_flag và par_level_flag.

Việc lập mã abs_remainder: Mặc dù phân phối thực nghiệm của mức tuyệt đối phần dư bỏ qua biến đổi vẫn phù hợp với phân phối Laplace hoặc hình học, có thể tồn

tại tính bất ổn lớn hơn so với mức tuyệt đối hệ số biến đổi. Cụ thể, phương sai bên trong cửa sổ của phép thể hiện liên tiếp có thể lớn hơn đối với mức tuyệt đối phần dư này. Theo đó, việc lập mô hình ngữ cảnh và nhị phân hóa abs_remainder có thể được cải biến như sau.

Ví dụ, giá trị ngắt (cutoff) cao hơn có thể được sử dụng trong khi nhị phân hóa abs_remainder. Theo đó, hiệu quả nén cao hơn có thể được cung cấp cho mô hình ngữ cảnh dành riêng cho mỗi vị trí ngăn và điểm chuyển tiếp từ việc lập mã bằng sig_coeff_flag, abs_level_gt1_flag, par_level_flag và abs_level_gt3_flag sang mã rice cho abs_remainder. Việc tăng giá trị ngắt có thể dẫn đến nhiều cờ "lớn hơn X" hơn (ví dụ, abs_level_gt5_flag, abs_level_gt7_flag, v.v.) cho đến khi giá trị ngắt này được đạt đến. Giá trị ngắt này có thể được cố định là 5 (numGtFlags = 5).

Thêm vào đó, khuôn mẫu để dẫn ra tham số rice có thể được cải biến. Tức là, chỉ vị trí lân cận bên phải và vị trí lân cận dưới cùng của vị trí quét hiện tại có thể được xem là khuôn mẫu cục bộ của việc lập mô hình ngữ cảnh sig_coeff_flag.

Việc lập mô hình ngữ cảnh coeff_sign_flag: Kể cả khi phân phối thực nghiệm toàn cục hầu như được phân phối đều do tính bất ổn bên trong trình tự dấu và thực tế rằng phần dư của việc dự đoán thường bị định thiên, thông tin liên quan đến dấu có thể được lập mã nhờ sử dụng mô hình ngữ cảnh. Mô hình ngữ cảnh dành riêng đơn lẻ có thể được sử dụng trong khi lập mã thông tin liên quan đến dấu, và thông tin liên quan đến dấu có thể được phân tích cú pháp sau sig_coeff_flag để giữ tất cả các ngăn được lập mã ngữ cảnh với nhau.

Việc giảm các ngăn được lập mã ngữ cảnh: Việc truyền các phần tử cú pháp cho chặng quét thứ nhất, tức là, sig_coeff_flag, abs_level_gt1_flag và par_level_flag có thể không được cải biến. Tuy nhiên, sự ràng buộc về giá trị tối đa của các ngăn được lập mã ngữ cảnh trên mỗi mẫu (context coded bin, CCB) có thể được loại bỏ và có thể được điều chỉnh khác đi. Việc giảm CCB có thể được dẫn ra bằng cách chỉ rõ chế độ không quan trọng khi CCB> k. Ở đây, k có thể là số nguyên dương. Ví dụ, trong trường hợp của chế độ lập mã mức thông thường, k có thể là 2. Sự ràng buộc nêu trên có thể tương ứng với việc giảm về không gian lượng tử hóa.

Các phần tử cú pháp liên quan đến dữ liệu phần dư được lập mã bằng cách áp dụng các cải biến nêu trên có thể được biểu diễn như được thể hiện trong bảng sau đây.

Bảng 21

residual_ts_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
log2SbSize = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
numSbCoeff = 1 << (log2SbSize << 1)	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − 2 * log2SbSize)) − 1	
/* Loop over subblocks from last to the top-left (DC) subblock */	
inferDcSbCbf = 1	
for(i = lastSubBlock; i >= 0; i− −) {	
xS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize][lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbSize][log2TbHeight − log2SbSize][lastSubBlock][1]	
if((i > 0 || !inferDcSbCbf)	
coded_sub_block_flag[xS][yS]	ae(v)
if(coded_sub_block_flag[xS][yS] && i > 0)	
inferDcSbCbf = 0	
}	
/* First scan pass */	
inferSbDcSigCoeffFlag = 1	
for(n = (i = = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag)) {	
sig_coeff_flag[xC][yC]	ae(v)
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
coeff_sign_flag[n]	ae(v)
abs_level_gtX_flag[n][0]	ae(v)
if(abs_level_gtX_flag[n][0])	
par_level_flag[n]	ae(v)
}	
AbsLevelPassX[xC][yC] =	
sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtX_flag[n][0]	
}	
/* Greater than X scan passes (numGtXFlags=5) */	
for(i = 1; i <= numGtXFlags − 1 && abs_level_gtX_flag[n][i − 1]; i++) {	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
abs_level_gtX_flag[n][i]	ae(v)
AbsLevelPassX[xC][yC] + = 2 * abs_level_gtX_flag[n][i]	
}	
}	
/* remainder scan pass */	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(abs_level_gtX_flag[n][numGtXFlags − 1])	
abs_remainder[n]	ae(v)
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (1 − 2 * coeff_sign_flag[n]) * (AbsLevelPassX[xC][yC] + abs_remainder[n])	
}	
}	
}	

Trong khi đó, chỉ số ngữ cảnh ctxIdx chỉ ra mô hình ngữ cảnh của phần tử cú pháp được lập mã dựa trên ngữ cảnh được chứa trong thông tin phần dư nêu trên có thể được dẫn ra như được mô tả dưới đây.

Ví dụ, đầu vào của quy trình dẫn ra chỉ số ngữ cảnh cho phần tử cú pháp này có

thể là binIdx chỉ ra vị trí của ngăn hiện tại trong chuỗi ngăn cho phần tử cú pháp này, và ctxTable, ctxIdx, và bypassFlag có thể được dẫn ra để làm đầu ra.

Đầu tiên, ctxInc cho ngăn hiện tại cho phần tử cú pháp này có thể được dẫn ra. Tức là, binIdx có thể được dẫn ra dựa trên binIdx chỉ ra vị trí của ngăn hiện tại cho phần tử cú pháp này. ctxInc có thể được biểu diễn bởi tham số lượng gia ngữ cảnh.

ctxInc được dẫn ra dựa trên binIdx cho phần tử cú pháp này có thể như được thể hiện trong bảng sau đây.

Bảng 22

Phần tử cú pháp	binIdx					
	0	1	2	3	4	>= 5
end_of_tile_group_flag	terminate	na	na	na	na	na
alf_ctb_flag[ ][ ][ ]	0..8 (clause 9.5.4.2.2)	na	na	na	na	na
sao_merge_left_flag	0	na	na	na	na	na
sao_merge_up_flag	0	na	na	na	na	na
sao_type_idx_luma	0	bypass	na	na	na	na
sao_type_idx_chroma	0	bypass	na	na	na	na
sao_offset_abs[ ][ ][ ][ ]	bypass	bypass	bypass	bypass	bypass	na
sao_offset_sign[ ][ ][ ][ ]	bypass	na	na	na	na	na
sao_band_position[ ][ ][ ]	bypass	bypass	bypass	bypass	bypass	bypass
sao_eo_class_luma	bypass	bypass	na	na	na	na
sao_eo_class_chroma	bypass	bypass	na	na	na	na
qt_split_cu_flag[ ][ ]	0..5 (clause 9.5.4.2.2)	na	na	na	na	na
mtt_split_cu_flag	0..12 (clause 9.5.4.2.2)	na	na	na	na	na
mtt_split_cu_vertical_flag	(cbWidth = = cbHeight) ? 0 : ((cbWidth > cbHeight) ? 1 : 2)	na	na	na	na	na
mtt_split_cu_binary_flag	0	na	na	na	na	na
cu_skip_flag[ ][ ]	0,1,2 (clause 9.5.4.2.2)	na	na	na	na	na
pred_mode_flag	0	na	na	na	na	na
pcm_flag[ ][ ]	terminate	na	na	na	na	na
intra_luma_ref_idx[ ][ ]	0	1	2	na	na	na
intra_subpartitions_mode_flag	0	na	na	na	na	na
intra_subpartition_split_flag	0	na	na	na	na	na
intra_luma_mpm_flag[ ][ ]	0	na	na	na	na	na
intra_luma_mpm_idx[ ][ ]	bypass	bypass	bypass	bypass	bypass	na
intra_luma_mpm_remainder[ ][ ]	bypass	bypass	bypass	bypass	bypass	bypass
intra_chroma_pred_mode[ ][ ] sps_cclm_enabled_flag = = 0	0	bypass	bypass	na	na	na
intra_chroma_pred_mode[ ][ ] sps_cclm_enabled_flag = = 1 && bin at binIdx equal to 2 = = 0	0	1	2	bypass	bypass	na
intra_chroma_pred_mode[ ][ ] sps_cclm_enabled_flag = = 1 && bin at binIdx equal to 2 = = 1	0	1	2	2	na	na
merge_subblock_flag[ ][ ]	0,1,2 (clause 9.5.4.2.2)	na	na	na	na	na
merge_subblock_idx[ ][ ] sps_sbtmvp_enabled_flag = = 0	0	bypass	bypass	bypass	bypass	bypass
merge_subblock_idx[ ][ ] sps_sbtmvp_enabled_flag = = 1	0	1	2	3	4	4
merge_flag[ ][ ]	0	na	na	na	na	na
mmvd_flag[ ][ ]	0	na	na	na	na	na

mmvd_merge_flag[ ][ ]	0	na	na	na	na	na
mmvd_distance_idx[ ][ ]	0	bypass	bypass	bypass	bypass	bypass
mmvd_direction_idx[ ][ ]	bypass	bypass	na	na	na	na
merge_triangle_flag[ ][ ]	0,1,2 (clause 9.5.4.2.2)	na	na	na	na	na
merge_triangle_idx[ ][ ]	0	bypass	bypass	bypass	bypass	bypass
merge_idx[ ][ ]	0	bypass	bypass	bypass	bypass	na
ciip_flag[ ][ ]	0	na	na	na	na	na
ciip_luma_mpm_flag[ ][ ]	0	na	na	na	na	na
ciip_luma_mpm_idx[ ][ ]	bypass	bypass	na	na	na	na
inter_pred_idc[x0][y0]	(cbWidth + cbHeight) != 8 ? 7 − ((1 + Log2(cbWidth) + Log2(cbHeight)) >> 1) : 4	4	na	na	na	na
inter_affine_flag[ ][ ]	0,1,2 (clause 9.5.4.2.2)	na	na	na	na	na
cu_affine_type_flag[ ][ ]	0	na	na	na	na	na
ref_idx_l0[ ][ ]	0	1	bypass	bypass	bypass	bypass
ref_idx_l1[ ][ ]	0	1	bypass	bypass	bypass	bypass
mvp_l0_flag[ ][ ]	0	na	na	na	na	na
mvp_l1_flag[ ][ ]	0	na	na	na	na	na
amvr_flag[ ][ ]	0,1,2 (clause 9.5.4.2.2)	na	na	na	na	na
amvr_4pel_flag[ ][ ]	0	na	na	na	na	na
gbi_idx[ ][ ] NoBackwardPredFlag == 0	0	1	na	na	na	na
gbi_idx[ ][ ] NoBackwardPredFlag == 1	0	1	2	3	na	na
cu_cbf	0	na	na	na	na	na
cu_sbt_flag	(cbWidth * cbHeight < 256) ? 1 : 0	na	na	na	na	na
cu_sbt_quad_flag	0	na	na	na	na	na
cu_sbt_horizontal_flag	(cbWidth == cbHeight) ? 0 : (cbWidth < cbHeight) ? 1 : 2	na	na	na	na	na
cu_sbt_pos_flag	0	na	na	na	na	na
abs_mvd_greater0_flag[ ]	0	na	na	na	na	na
abs_mvd_greater1_flag[ ]	0	na	na	na	na	na
abs_mvd_minus2[ ]	bypass	bypass	bypass	bypass	bypass	bypass
mvd_sign_flag[ ]	bypass	na	na	na	na	na

tu_cbf_luma[ ][ ][ ]	0,1,2,3 (clause 9.5.4.2.4)	na	na	na	na	na
tu_cbf_cb[ ][ ][ ]	trDepth = = 0 ? 0 : 1	na	na	na	na	na
tu_cbf_cr[ ][ ][ ]	tu_cbf_cb[ ][ ][ ]	na	na	na	na	na
cu_qp_delta_abs	0	1	1	1	1	bypass
cu_qp_delta_sign_flag	bypass	na	na	na	na	na
transform_skip_flag[ ][ ]	0	na	na	na	na	na
tu_mts_idx[ ][ ]	cqtDepth	6	7	8	na	na
last_sig_coeff_x_prefix	0..23 (clause 9.5.4.2.3)					
last_sig_coeff_y_prefix	0..23 (clause 9.5.4.2.3)					
last_sig_coeff_x_suffix	bypass	bypass	bypass	bypass	bypass	bypass
last_sig_coeff_y_suffix	bypass	bypass	bypass	bypass	bypass	bypass
coded_sub_block_flag[ ][ ]	0..3 (clause 9.5.4.2.5)	na	na	na	na	na
sig_coeff_flag[ ][ ]	0..89 (clause 9.5.4.2.7)	na	na	na	na	na
par_level_flag[ ]	0..32 (clause 9.5.4.2.8)	na	na	na	na	na
abs_level_gt1_flag[ ]	0..32 (clause 9.5.4.2.8)	na	na	na	na	na
abs_level_gt3_flag[ ]	0..32 (clause 9.5.4.2.8)	na	na	na	na	na
abs_remainder[ ]	bypass	bypass	bypass	bypass	bypass	bypass
dec_abs_level[ ]	bypass	bypass	bypass	bypass	bypass	bypass
coeff_sign_flag[ ]	bypass	na	na	na	na	na

Trong bảng 22, chỉ số ngữ cảnh của ngăn có giá trị mà nó không phải là "bypass" (đi vòng), "terminate" (chấm dứt), hoặc "na" (không khả dụng) có thể được dẫn ra như sau.

ctxInc cho ngăn hiện tại của phần tử cú pháp có thể được dẫn ra là giá trị được chỉ rõ là một khoản mục cho ngăn hiện tại trong bảng 22. Thêm vào đó, nếu nhiều giá trị được chỉ rõ là khoản mục cho ngăn hiện tại này thì ctxInc có thể được dẫn ra thông qua quy trình của mệnh đề được đưa ra trong dấu ngoặc đơn trong khoản mục này. Mệnh đề trên đây có thể ngụ ý là mệnh đề được lộc lộ trong tiêu chuẩn VVC. Sau đó, biến số ctxIdxOffset có thể được chỉ rõ là giá trị thấp nhất của ctxIdx theo giá trị hiện tại của initType. Thêm vào đó, chỉ số ngữ cảnh ctxIdx cho ngăn hiện tại của phần tử cú pháp này có thể được thiết lập là bằng tổng của ctxInc và ctxIdxOffset. Tức là, chỉ số ngữ cảnh có thể được thiết lập là tổng của ctxInc và ctxIdxOffset. Thêm vào đó, bypassFlag có thể được thiết lập là 0.

Trong khi đó, nếu khoản mục cho ngăn hiện tại là "đi vòng" trong bảng 22 thì chỉ số ngữ cảnh của ngăn có thể được dẫn ra như sau. Ví dụ, ctxTable cho ngăn hiện tại thể được thiết lập là 0. Thêm vào đó, chỉ số ngữ cảnh ctxIdx cho ngăn hiện tại có thể được thiết lập là 0. Thêm vào đó, bypassFlag có thể được thiết lập là 1.

Trong khi đó, nếu khoản mục cho ngăn hiện tại là "chấm dứt" trong bảng 22 thì

chỉ số ngữ cảnh của ngăn này có thể được dẫn ra như sau. Ví dụ, ctxTable cho ngăn hiện tại thể được thiết lập là 0. Thêm vào đó, chỉ số ngữ cảnh ctxIdx cho ngăn hiện tại có thể được thiết lập là 0. Thêm vào đó, bypassFlag có thể được thiết lập là 0.

Trong khi đó, nếu khoản mục cho ngăn hiện tại là "na" trong bảng 22 thì các phần tử cú pháp cho ngăn này, cụ thể là chỉ số ngữ cảnh cho ngăn này, có thể được dẫn ra như sau. Ví dụ, ctxIdx, ctxTable, và/hoặc bypassFlag có thể không xuất hiện đối với ngăn hiện tại.

Các quy trình của các mệnh đề để dẫn ra ctxInc cho ngăn có giá trị khác ngoài "bypass" (đi vòng), "terminate" (chấm dứt), hoặc "na" (không khả dụng) có thể như được mô tả dưới đây.

Ví dụ, quy trình dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.2 có thể như được thể hiện trong bảng sau đây.

Bảng 23

9.5.4.2.2 Quy trình dẫn ra ctxInc nhờ sử dụng các phần tử cú pháp bên trái và bên trên
Đầu vào cho quy trình này là vị trí độ chói (x0, y0) chỉ rõ mẫu độ chói trên cùng bên trái của khối độ chói hiện tại so với mẫu trên cùng bên trái của hình ảnh hiện tại, thành phần màu cIdx, chiều sâu cây tứ phân lập mã hiện tại cqDepth, và chiều rộng và chiều cao của khối lập mã hiện tại tính theo các mẫu độ chói cbWidth và cbHeight. Đầu ra của quy trình này là ctxInc. Vị trí (xNbL, yNbL) được thiết lập bằng (x0 − 1, y0) và biến số availableL, chỉ rõ tính khả dụng của khối được đặt trực tiếp sang bên trái của khối hiện tại, được dẫn ra bằng cách gọi ra quy trình dẫn ra tính khả dụng cho khối theo thứ tự quét z như được chỉ rõ trong tiểu mục 6.4 với vị trí (xCurr, yCurr) được thiết lập bằng (x0, y0) và vị trí lân cận (xNbY, yNbY) được thiết lập bằng (xNbL, yNbL) để làm các đầu vào, và đầu ra được gán cho availableL.

Vị trí (xNbA, yNbA) được thiết lập bằng (x0, y0 − 1) và biến số availableA chỉ rõ tính khả dụng của khối lập mã được đặt trực tiếp bên trên khối hiện tại, được dẫn ra bằng cách gọi ra quy trình dẫn ra tính khả dụng cho khối theo thứ tự quét z như được chỉ rõ trong tiểu mục 6.4 với vị trí (xCurr, yCurr) được thiết lập bằng (x0, y0) và vị trí lân cận (xNbY, yNbY) được thiết lập bằng (xNbA, yNbA) để làm các đầu vào, và đầu ra được gán cho availableA.

Các biến số sizeC, sizeTh2 và sizeTh1 được dẫn ra như sau:

sizeTh2 = (MaxBtSizeY = = 128) ? 1024 : ((MaxBtSizeY = = 64) ? 512 : 256) (9 19)

sizeTh1 = (MaxBtSizeY = = 128) ? 128 : 64 (9 20) **sizeC = cbWidth * cbHeight** (9 21)

Việc gán ctxInc được chỉ rõ như sau với condL và condA cho các phần tử cú pháp alf_ctb_flag[x0][y0][cIdx], qt_split_cu_flag[x0][y0], mtt_split_cu_flag[x0][y0], cu_skip_flag[x0][y0], amvr_flag[x0][y0], inter_affine_flag[x0][y0], merge_triangle_flag[x0][y0] và merge_subblock_flag[x0][y0] được chỉ rõ trong bảng 9 11:

ctxInc = (condL && availableL) + (condA && availableA) + ctxSetIdx * 3 (9 22)

Thêm vào đó, ví dụ, quy trình dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.3 có thể như được thể hiện trong bảng sau đây.

Bảng 24

9.5.4.2.3 Quy trình dẫn ra ctxInc cho các phần tử cú pháp last_sig_coeff_x_prefix và last_sig_coeff_y_prefix Các đầu vào cho quy trình này là biến số binIdx, chỉ số thành phần màu cIdx, logarit nhị phân của chiều rộng khối biến đổi log2TbWidth và chiều

cao khối biến đổi log2TbHeight.

Đầu ra của quy trình này là biến số ctxInc.

Biến số log2TbSize được dẫn ra như sau:

– Nếu phần tử cú pháp cần được phân tích cú pháp là last_sig_coeff_x_prefix thì log2TbSize được thiết lập bằng log2TbWidth.

– Nếu không (phần tử cú pháp cần được phân tích cú pháp là last_sig_coeff_y_prefix) thì log2TbSize được thiết lập bằng log2TbHeight.

Các biến số ctxOffset và ctxShift được dẫn ra như sau:

– Nếu cIdx được thiết lập bằng 0 thì ctxOffset được thiết lập bằng 3 * (log2TbSize − 2) + ((log2TbSize − 1) >> 2) và ctxShift được thiết lập bằng (log2TbSize + 1) >> 2.

– Nếu không (cIdx lớn hơn 0) thì ctxOffset được thiết lập bằng 21 và ctxShift được thiết lập bằng Clip3(0, 2, 2log2TbSize >> 3).

Biến số ctxInc được dẫn ra như sau:

ctxInc = (binIdx >> ctxShift) + ctxOffset (9 23)

Thêm vào đó, ví dụ, quy trình dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.4 có thể như được thể hiện trong bảng sau đây.

Bảng 25

9.5.4.2.4 Quy trình dẫn ra ctxInc cho phần tử cú pháp tu_cbf_luma

Các đầu vào cho quy trình này là biến số binIdx, chỉ số thành phần màu cIdx, logarit nhị phân của chiều rộng khối biến đổi log2TbWidth và chiều cao khối biến đổi log2TbHeight và chiều sâu cây biến đổi hiện tại trDepth.

Đầu ra của quy trình này là biến số ctxInc.

Biến số ctxInc được dẫn ra như sau:

– Nếu IntraSubpartitionSplitType bằng ISP_NO_SPLIT hoặc cIdx không

bằng 0 thì áp dụng như sau:

ctxInc = trDepth = = 0 ? 1 : 0 (9 24)

– Nếu không (IntraSubpartitionSplitType không bằng ISP_NO_SPLIT và cIdx bằng 0) thì được áp dụng như sau:

– Biến số prevTuCbfY được dẫn ra như sau:

– Nếu đơn vị biến đổi hiện tại là đơn vị thứ nhất cần được phân tích cú pháp trong đơn vị lập mã thì prevTuCbfY được thiết lập bằng 0.

– Nếu không thì prevTuCbfY được thiết lập bằng giá trị của tu_cbf_luma của đơn vị biến đổi độ chói trước đó trong đơn vị lập mã hiện tại.

– Biến số ctxInc được dẫn ra như sau:

ctxInc = 2 + prevTuCbfY (9 25)

Thêm vào đó, ví dụ, quy trình để dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.5 có thể như được thể hiện trong bảng sau đây.

Bảng 26

9.5.4.2.5 Quy trình dẫn ra ctxInc cho phần tử cú pháp coded_sub_block_flag
Các đầu vào cho quy trình này là chỉ số thành phần màu cIdx, vị trí quét khối con hiện tại (xS, yS), các ngăn được giải mã trước đó của phần tử cú pháp coded_sub_block_flag và logarit nhị phân của chiều rộng khối biến đổi log2TbWidth và chiều cao khối biến đổi log2TbHeight.
Đầu ra của quy trình này là biến số ctxInc.
Biến số csbfCtx được dẫn ra nhờ sử dụng vị trí hiện tại (xS, yS), hai ngăn được giải mã trước đó của phần tử cú pháp coded_sub_block_flag theo thứ tự quét, log2TbWidth và log2TbHeight, như sau:
– Các biến số log2SbWidth và log2SbHeight được dẫn ra như sau:
log2SbWidth = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)

(9 26)

log2SbHeight = log2SbWidth (9 27)

– Các biến số log2SbWidth và log2SbHeight được cải biến như sau:

– Nếu log2TbWidth nhỏ hơn 2 và cIdx bằng 0 thì áp dụng như sau

log2SbWidth = log2TbWidth (9 28)

log2SbHeight = 4 − log2SbWidth (9 29)

– Nếu không, nếu log2TbHeight nhỏ hơn 2 và cIdx bằng 0 thì áp dụng như sau

log2SbHeight = log2TbHeight (9 30)

log2SbWidth = 4 − log2SbHeight (9 31)

– csbfCtx được khởi tạo bằng 0 như sau:

csbfCtx = 0 (9 32)

– Khi xS nhỏ hơn (1 << (log2TbWidth − log2SbWidth)) − 1, csbfCtx được cải biến như sau:

csbfCtx += coded_sub_block_flag[xS + 1][yS] (9 33)

– Khi yS nhỏ hơn (1 << (log2TbHeight − log2SbHeight)) − 1, csbfCtx được cải biến như sau:

csbfCtx += coded_sub_block_flag[xS][yS + 1] (9 34)

Lượng gia chỉ số ngữ cảnh ctxInc được dẫn ra nhờ sử dụng chỉ số thành phần màu cIdx và csbfCtx như sau:

– Nếu cIdx bằng 0 thì ctxInc được dẫn ra như sau:

ctxInc = Min(csbfCtx, 1) (9 35)

– Nếu không (cIdx lớn hơn 0) thì ctxInc được dẫn ra như sau:

ctxInc = 2 + Min(csbfCtx, 1) (9 36)

Thêm vào đó, ví dụ, quy trình dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.6 có thể như

được thể hiện trong bảng sau đây.

Bảng 27

9.5.4.2.6 Quy trình dẫn ra các biến số locNumSig, locSumAbsPass1

Các đầu vào cho quy trình này là chỉ số thành phần màu cIdx, vị trí độ chói (x0, y0) chỉ rõ mẫu trên cùng bên trái của khối biến đổi hiện tại so với mẫu trên cùng bên trái của hình ảnh hiện tại, vị trí quét hệ số hiện tại (xC, yC), logarit nhị phân của chiều rộng khối biến đổi log2TbWidth, và logarit nhị phân của chiều cao khối biến đổi log2TbHeight.

Các đầu ra của quy trình này là các biến số locNumSig và locSumAbsPass1.

Căn cứ vào các phần tử cú pháp sig_coeff_flag[x][y] và mảng AbsLevelPass1[x][C] cho khối biến đổi với chỉ số thành phần cIdx và vị trí độ chói trên cùng bên trái (x0, y0), các biến số locNumSig và locSumAbsPass1 được dẫn ra như được chỉ rõ bởi mã giả sau đây:

```
locNumSig          = 0
locSumAbsPass1 = 0
if( xC < (1 << log2TbWidth) − 1 ) {
    locNumSig          += sig_coeff_flag[ xC + 1 ][ yC ]
    locSumAbsPass1 += AbsLevelPass1[ xC + 1 ][ yC ]
    if( xC < (1 << log2TbWidth) − 2 ) {
        locNumSig          += sig_coeff_flag[ xC + 2 ][ yC ]
        locSumAbsPass1 += AbsLevelPass1[ xC + 2 ][ yC ]
    }
    if( yC < (1 << log2TbHeight) − 1 ) {
        locNumSig          += sig_coeff_flag[ xC + 1 ][ yC + 1 ]
```

```
        (9 37)
            locSumAbsPass1 += AbsLevelPass1[ xC + 1 ][ yC + 1 ]
        }
    }
    if( yC < (1 << log2TbHeight) − 1 ) {
        locNumSig              += sig_coeff_flag[ xC ][ yC + 1 ]
        locSumAbsPass1 += AbsLevelPass1[ xC ][ yC + 1 ]
        if( yC < (1 << log2TbHeight) − 2 ) {
            locNumSig              += sig_coeff_flag[ xC ][ yC + 2 ]
            locSumAbsPass1 += AbsLevelPass1[ xC ][ yC + 2 ]
        }
    }
```

Thêm vào đó, ví dụ, quy trình dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.7 có thể như được thể hiện trong bảng sau đây.

Bảng 28

9.5.4.2.7 Quy trình dẫn ra ctxInc cho phần tử cú pháp sig_coeff_flag Các đầu vào cho quy trình này là chỉ số thành phần màu cIdx, vị trí độ chói (x0, y0) chỉ rõ mẫu trên cùng bên trái của khối biến đổi hiện tại so với mẫu trên cùng bên trái của hình ảnh hiện tại, vị trí quét hệ số hiện tại (xC, yC), logarit nhị phân của chiều rộng khối biến đổi log2TbWidth, và logarit nhị phân của chiều cao khối biến đổi log2TbHeight. Đầu ra của quy trình này là biến số ctxInc. Biến số locSumAbsPass1 được dẫn ra bằng cách gọi ra quy trình dẫn ra các biến số locNumSig và locSumAbsPass1 được chỉ rõ trong mệnh đề 9.5.4.2.6 với chỉ số thành phần màu cIdx, vị trí độ chói (x0, y0), vị trí quét

hệ số hiện tại (xC, yC), logarit nhị phân của chiều rộng khối biến đổi log2TbWidth, và logarit nhị phân của chiều cao khối biến đổi log2TbHeight làm đầu vào.

Biến số d được thiết lập là xC + yC.

Biến số ctxInc được dẫn ra như sau:

– Nếu cIdx bằng 0 thì ctxInc được dẫn ra như sau:

ctxInc = 18 * Max(0, QState − 1) + Min(locSumAbsPass1, 5) + (d < 2 ? 12 : (d < 5 ? 6 : 0)) (9 38)

– Nếu không (cIdx lớn hơn 0) thì ctxInc được dẫn ra như sau:

ctxInc = 54 + 12 * Max(0, QState − 1) + Min(locSumAbsPass1, 5) + (d < 2 ? 6 : 0) (9 39)

Thêm vào đó, ví dụ, quy trình dẫn ra ctxInc dựa trên mệnh đề 9.5.4.2.8 có thể như được thể hiện trong bảng sau đây.

Bảng 29

9.5.4.2.8 Quy trình dẫn ra ctxInc cho các phần tử cú pháp par_level_flag, abs_level_gt1_flag, và abs_level_gt3_flag Các đầu vào cho quy trình này là chỉ số thành phần màu cIdx, vị trí độ chói (x0, y0) chỉ rõ mẫu trên cùng bên trái của khối biến đổi hiện tại so với mẫu trên cùng bên trái của hình ảnh hiện tại, vị trí quét hệ số hiện tại (xC, yC), logarit nhị phân của chiều rộng khối biến đổi log2TbWidth, và logarit nhị phân của chiều cao khối biến đổi log2TbHeight. Đầu ra của quy trình này là biến số ctxInc. Biến số locNumSig và locSumAbsPass1 được dẫn ra bằng cách gọi ra quy trình dẫn ra các biến số locNumSig và locSumAbsPass1 được chỉ rõ trong mệnh đề 9.5.4.2.6 với chỉ số thành phần màu cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), logarit nhị phân của chiều rộng khối biến đổi log2TbWidth, và logarit nhị phân của chiều cao khối biến đổi

log2TbHeight làm đầu vào.

Biến số ctxOffset được thiết lập là Min(locSumAbsPass1 − locNumSig, 4).

Biến số d được thiết lập là xC + yC.

Biến số ctxInc được dẫn ra như sau:

– Nếu xC bằng LastSignificantCoeffX và yC bằng LastSignificantCoeffY thì ctxInc được dẫn ra như sau:

ctxInc = (cIdx == 0 ? 0 : 21) (9 40)

– Nếu không, nếu cIdx bằng 0 thì ctxInc được dẫn ra như sau:

ctxInc = 1 + ctxOffset + (d == 0 ? 15 : (d < 3 ? 10 : (d < 10 ? 5 : 0))) (9 41)

– Nếu không (cIdx lớn hơn 0) thì ctxInc được dẫn ra như sau:

ctxInc = 22 + ctxOffset + (d == 0 ? 5 : 0) (9 42)

Trong khi đó, như được mô tả trên đây, vì khối mà nó không được trải qua việc mã hóa biến đổi, tức là, khối biến đổi bao gồm các hệ số phần dư mà việc biến đổi không được áp dụng cho chúng, có đặc điểm khác so với đặc điểm của khối được trải qua việc mã hóa biến đổi điển hình, cần có phương pháp mã hóa dữ liệu phần dư hiệu quả đối với khối không được trải qua việc mã hóa biến đổi này. Trong khi đó, như được mô tả trên đây, cờ bỏ qua biến đổi chỉ ra liệu việc biến đổi có được áp dụng hay không có thể được truyền trên cơ sở khối biến đổi, và kích thước của khối biến đổi này không bị giới hạn trong sáng chế này. Ví dụ, nếu giá trị của cờ bỏ qua biến đổi là 1 thì phương pháp mã hóa/giải mã thông tin phần dư được đề xuất trong sáng chế này có thể được thực hiện, và nếu giá trị của cờ bỏ qua biến đổi là 0 thì phương pháp mã hóa/giải mã thông tin phần dư thông dụng được mô tả trong bảng 1, bảng 6, hoặc bảng 9 trên đây có thể được thực hiện. Theo cách khác, nếu cờ bỏ qua biến đổi chỉ ra rằng việc biến đổi không được áp dụng cho khối hiện tại (việc biến đổi được bỏ qua) thì phương pháp mã hóa/giải mã thông tin phần dư trong chế độ bỏ qua biến đổi được bộc lộ trong bảng 10 hoặc bảng 21 trên đây có thể được thực hiện.

Trong khi đó, nếu việc biến đổi không được áp dụng cho khối hiện tại thì hệ số biến đổi có thể được dẫn ra bằng mẫu phần dư. Do đó, nếu việc biến đổi được bỏ qua thì mẫu phần dư cũng có thể được gọi là hệ số hoặc hệ số phần dư.

Như được mô tả trong bảng 1, bảng 6, hoặc bảng 9 trên đây, đối với việc mã hóa/giải mã phần dư, cờ hệ số quan trọng, cờ mức hệ số biến đổi thứ nhất, cờ mức số chẵn lẻ, và cờ mức biến đổi thứ hai có thể được mã hóa/giải mã, phần tử cú pháp cho giá trị mức còn lại, cụ thể là abs_remainder hoặc dec_abs_level, có thể được mã hóa/giải mã, và sau đó cờ dấu cho mỗi hệ số phần dư có thể được mã hóa/giải mã. Ở đây, cờ hệ số quan trọng có thể là sig_coeff_flag, và cờ mức chẵn lẻ có thể là par_level_flag, cờ mức hệ số biến đổi thứ nhất có thể là abs_level_gt1_flag, và cờ mức hệ số biến đổi thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag. Thêm vào đó, cờ dấu có thể là coeff_sign_flag nêu trên.

Trong khi đó, cờ dấu có thể được mã hóa/giải mã như được mô tả trong phương pháp được thể hiện trong bảng 21 trên đây. Trong trường hợp này, cờ dấu có thể được mã hóa/giải mã dựa trên mô hình ngữ cảnh, không giống như trong bảng 1, bảng 6, hoặc bảng 9 trên đây trong đó cờ dấu được trải qua việc lập mã đi vòng.

Ở đây, trong trường hợp của khối bỏ qua biến đổi (cụ thể là khối mà việc biến đổi không được áp dụng cho nó), vì việc biến đổi không được áp dụng, có khả năng cao là hệ số phần dư trong khối bỏ qua biến đổi có dấu tương tự như dấu của hệ số phần dư lân cận với khối này.

Do đó, phương án này của sáng chế đề xuất phương pháp trong đó tín hiệu phần dư được bỏ qua biến đổi, cụ thể là bảng ngữ cảnh (ctxTable) của cờ dấu của hệ số phần dư của khối được bỏ qua biến đổi, được xác định dựa trên cờ dấu của hệ số phần dư lân cận liền kề hệ số phần dư này và cờ dấu của hệ số phần dư này được lập mã ngữ cảnh dựa trên bảng ngữ cảnh được xác định này. Tức là, phương án này của sáng chế đề xuất phương pháp trong đó tín hiệu phần dư được bỏ qua biến đổi, cụ thể là bảng ngữ cảnh (ctxTable) của cờ dấu của hệ số phần dư của khối được bỏ qua biến đổi, được xác định dựa trên cờ dấu của hệ số phần dư được mã hóa/giải mã ngay trước hệ số phần dư này và cờ dấu của hệ số phần dư này được lập mã ngữ cảnh dựa trên bảng ngữ cảnh được xác định này. Phương án này đề xuất phương pháp trong đó mô hình ngữ cảnh cho cờ

dấu của hệ số phần dư trong khối hiện tại của khối hiện tại được xác định dựa trên cờ dấu của hệ số phần dư được lập mã trước hệ số phần dư này trong khối con hiện tại, và cờ dấu này được lập mã dựa trên mô hình ngữ cảnh được xác định này.

Không giống như trong việc mã hóa/giải mã cờ dấu được bộc lộ trong bảng 1, bảng 6, hoặc bảng 21 trên đây trong nhóm hệ số (coefficient group, CG) của khối biến đổi, cờ dấu có thể được mã hóa/giải mã ngữ ảnh trong phương án này để cải thiện hiệu suất nén. CG cũng có thể được biểu diễn bởi khối con. Thêm vào đó, việc mã hóa/giải mã cờ dấu được bộc lộ trong bảng 21 trên đây sử dụng chỉ một mô hình ngữ cảnh, mà điều này là khác với phương án này trong đó việc lập mã được thực hiện qua việc sử dụng mô hình ngữ cảnh được xác định dựa trên cờ dấu của hệ số phần dư được lập mã trước hệ số phần dư này trong số nhiều mô hình ngữ cảnh.

Theo phương án này, cờ dấu có thể được lập mã bằng cách xem xét mối tương quan cao với hệ số phần dư liền kề. Vì các cờ dấu của các hệ số phần dư liền kề trong khối bỏ qua biến đổi có xu hướng có các giá trị tương tự nhau như được mô tả trên đây, qua việc xem xét khía cạnh nêu trên, phương án này đề xuất phương pháp trong đó hai mô hình ngữ cảnh (hoặc các bảng ngữ cảnh) được chọn dựa trên giá trị (ví dụ, 0 hoặc 1) của cờ dấu của hệ số phần dư ngay trước đó theo thứ tự quét hệ số, khi cờ dấu của hệ số phần dư trong khối hiện tại được mã hóa/giải mã. Trong khi đó, khi hệ số phần dư trong khối hiện tại được lập mã đầu tiên trong số các hệ số phần dư của khối hiện tại, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư này có thể được dẫn ra là mô hình ngữ cảnh định trước (ví dụ, mô hình ngữ cảnh 0). Ngoài ra, ví dụ, khi khối con bao gồm các hệ số phần dư trong khối hiện tại không là khối con được lập mã theo thứ tự thứ nhất, và khối con được lập mã trước đó là có mặt, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư có thể được dẫn ra dựa trên cờ dấu của hệ số phần dư của khối con lân cận được lập mã ngay trước hệ số phần dư này. Hiệu suất nén đối với thông tin phần dư có thể được cải thiện thông qua phương án nêu trên. Trong khi đó, mô hình ngữ cảnh có thể được chỉ ra dựa trên ctxIdx hoặc ctxInc, và ctxIdx có thể được dẫn ra dựa trên dựa trên ctxInc.

Thêm vào đó, để làm một phương án khác, sáng chế này có thể đề xuất phương pháp chọn một mô hình ngữ cảnh trong số M mô hình ngữ cảnh bằng cách sử dụng N (N>=1) cờ dấu bất kỳ được mã hóa/giải mã trước hệ số phần dư trong khối hiện tại. Ví

dụ, mô hình ngữ cảnh cho cờ dấu của phần dư hiện tại có thể được dẫn ra là một mô hình trong số ba mô hình ngữ cảnh, dựa trên các cờ dấu của hai hệ số phần dư được mã hóa/giải mã trước hệ số phần dư hiện tại trong khối hiện tại. Theo cách khác, ví dụ, mô hình ngữ cảnh cho cờ dấu của phần dư hiện tại có thể được dẫn ra là một mô hình trong số sáu mô hình ngữ cảnh, dựa trên các cờ dấu của hai hệ số phần dư được mã hóa/giải mã trước hệ số phần dư hiện tại trong khối hiện tại. Theo cách khác, ví dụ, mô hình ngữ cảnh cho cờ dấu của phần dư hiện tại có thể được dẫn ra là một mô hình trong số ba mô hình ngữ cảnh, dựa trên cờ dấu của hệ số phần dư lân cận bên trái liền kề hệ số phần dư hiện tại và cờ dấu của hệ số phần dư lân cận trên cùng của hệ số phần dư hiện tại. Theo cách khác, ví dụ, mô hình ngữ cảnh cho cờ dấu của phần dư hiện tại có thể được dẫn ra là một mô hình trong số sáu mô hình ngữ cảnh, dựa trên cờ dấu của hệ số phần dư lân cận bên trái liền kề hệ số phần dư hiện tại và cờ dấu của hệ số phần dư lân cận trên cùng của hệ số phần dư hiện tại. Trong khi đó, khi hệ số phần dư trong khối hiện tại được lập mã đầu tiên trong số các hệ số phần dư của khối hiện tại, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư này có thể được dẫn ra là mô hình ngữ cảnh định trước (ví dụ, mô hình ngữ cảnh 0).

Ví dụ, khi khối con hiện tại bao gồm năm hệ số phần dư khác không, nếu giá trị của các cờ dấu của các hệ số phần dư khác không này là (0, 0, 0, 1, 0) thì các giá trị mô hình ngữ cảnh cho các cờ dấu của các hệ số phần dư khác không này có thể được định ra là (0, 0, 0, 0, 1). Ở đây, giá trị mô hình ngữ cảnh có thể chỉ ra mô hình ngữ cảnh. Tức là, nếu giá trị mô hình ngữ cảnh là 0 thì giá trị mô hình ngữ cảnh có thể chỉ ra mô hình ngữ cảnh 0, và nếu giá trị mô hình ngữ cảnh là 1 thì giá trị mô hình ngữ cảnh có thể chỉ ra mô hình ngữ cảnh 1. Trong phương án này, mã giả có thể như được thể hiện trong bảng sau đây.

Bảng 30

```
signPattern = 00010 (Herein, signPattern represents successive values of sign flags)
lastSignFlag = 0
numNonZeroCoeff = 5
shiftNum = numNonZeroCoeff - 1
while (numNonZeroCoeff > 0)
{
 currSignFlag = signPattern & (1 << shiftNum)
 ctxModel = lastSign
encodeBin(currSignFlag, ctxModel)
lastSignFlag = currSignFlag
signPattern <<= 1
numNonZeroCoeff -= 1
}
```

Trong khi đó, các phương án để dẫn ra mô hình ngữ cảnh của cờ dấu được đề xuất trong sáng chế này có thể được định ra như được thể hiện trong bảng sau đây.

Bảng 31

Phần tử cú pháp	binIdx					
	0	1	2	3	4	>= 5
coeff_sign_flag[ ][ ]	0, 1 (Xem quy trình dẫn ra sau đây)	na	na	na	na	na

Quy trình dẫn ra ctxInc được bộc lộ trong bảng 31 trên đây có thể là như sau.

1. Cờ dấu cuối cùng (cờ dấu được mã hóa/giải mã ngay trước cờ dấu cần được lập mã) được nhập vào.

2. Cờ dấu cuối cùng này được xuất ra dưới dạng ctxInc. Tức là, giá trị của cờ dấu cuối cùng được xuất ra dưới dạng ctxInc.

Theo cách khác, các phương án để dẫn ra mô hình ngữ cảnh của cờ dấu được đề xuất trong sáng chế này có thể được định ra như được thể hiện trong bảng sau đây.

Bảng 32

Phần tử cú pháp	binIdx					
	0	1	2	3	4	>= 5
coeff_sign_flag[ ][ ]	Lastcoeffsignflag = =0 ? 0 : 1	na	na	na	na	na

Tham chiếu đến bảng 32 trên đây, mô hình ngữ cảnh cho ngăn của phần tử cú pháp coeff_sign_flag (cụ thể là cờ dấu) cho hệ số phần dư hiện tại có thể được xác định dựa trên lastcoeffsignflag. Cụ thể, ví dụ, mô hình ngữ cảnh cho ngăn của phần tử cú pháp coeff_sign_flag cho hệ số phần dư hiện tại có thể được xác định dựa trên ctxinc, và

ctxinc có thể được dẫn ra là 0 nếu giá trị của lastcoeffsignflag là 0, và có thể được dẫn ra là 1 nếu giá trị của lastcoeffsignflag là 1. Ở đây, lastcoeffsignflag có thể là giống như giá trị của coeff_sign_flag cho hệ số phần dư ngay trước hệ số phần dư hiện tại theo thứ tự quét hệ số trong khối hiện tại (ví dụ, TU). Trong khi đó, nếu hệ số phần dư hiện tại là hệ số phần dư thứ nhất trong khối hiện tại, thì lastcoeffsignflag có thể được thiết lập là 0. Tức là, giá trị ban đầu của lastcoeffsignflag có thể được thiết lập là 0.

Tức là, ví dụ, theo phương án này, mô hình ngữ cảnh cho ngăn của coeff_sign_flag của hệ số phần dư hiện tại có thể được xác định theo cách thích ứng dựa trên giá trị của coeff_sign_flag của hệ số phần dư trước đó của hệ số phần dư hiện tại theo thứ tự quét hệ số trong số các hệ số phần dư trong khối hiện tại. Do đó, ngăn của coeff_sign_flag của hệ số phần dư hiện tại có thể được giải mã, và giá trị của coeff_sign_flag của hệ số phần dư hiện tại có thể được dẫn ra. Theo đó, mối tương quan của các hệ số phần dư liền kề có thể được xem xét, và độ lợi lập mã có thể được tăng lên. Dấu của mẫu phần dư hiện tại có thể được dẫn ra như được mô tả trên đây, bằng cách sử dụng giá trị của coeff_sign_flag của hệ số phần dư hiện tại.

Trong khi đó, sau khi ctxInc cho coeff_sign_flag của hệ số phần dư hiện tại được dẫn ra như được mô tả trên đây, quy trình dẫn ra mô hình ngữ cảnh dựa trên ctxInc có thể được thực hiện như được mô tả trên đây.

Cụ thể, ctxIdxOffset cho cờ dấu có thể được dẫn ra là giá trị nhỏ nhất trong số các giá trị khả dĩ của ctxIdx cho cờ dấu. Ví dụ, trong phần mô tả trên đây, vì ctxIdx (chỉ số mô hình ngữ cảnh chỉ ra mô hình ngữ cảnh) có thể được dẫn ra là 0 (cụ thể là mô hình ngữ cảnh 0) hoặc 1 (cụ thể là mô hình ngữ cảnh 1), ctxIdxOffset có thể được dẫn ra là 0. Sau đó, ctxIdx cho cờ dấu có thể được dẫn ra là tổng của ctxIdxOffset và ctxInc. Vì ctxIdxOffset được dẫn ra là 0, ctxIdx cho cờ dấu có thể là giống như ctxInc. Theo cách khác, ví dụ, ctxIdx (chỉ số mô hình ngữ cảnh thể hiện mô hình ngữ cảnh) cho cờ dấu có thể được dẫn ra là giá trị của một trong số 0 (cụ thể là mô hình ngữ cảnh 0) đến N (cụ thể là mô hình ngữ cảnh N), trong trường hợp này, ctxIdxOffset được dẫn ra là 0. Sau đó, ctxIdx cho cờ dấu có thể được dẫn ra là tổng của ctxIdxOffset và ctxInc. Vì ctxIdxOffset được dẫn ra là 0, ctxIdx cho cờ dấu có thể là giống như ctxInc. Ví dụ, N có thể là 2 hoặc 5.

Trong khi đó, các phương án nêu trên có thể được áp dụng cho tất cả các khối bỏ qua biến đổi. Theo cách khác, vì cờ dấu được mã hóa/giải mã chỉ cho hệ số phần dư khác không, có thể ưu tiên áp dụng các phương án nêu trên chỉ khi số lượng hệ số phần dư khác không cần được trải qua việc mã hóa/giải mã cờ dấu là lớn hơn hoặc bằng một giá trị cụ thể. Theo đó, khi phương pháp lập mã phần dư được mô tả trong bảng 1 đến bảng 6 được sử dụng, số lượng hệ số phần dư khác không được lập mã trước đó có thể được dẫn ra, và việc liệu cờ dấu sẽ được trải qua việc lập mã ngữ cảnh hay lập mã đi vòng có thể được xác định theo số lượng hệ số phần dư khác không này. Tức là, ví dụ, nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng một giá trị cụ thể thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn giá trị cụ thể này thì cờ dấu có thể được trải qua việc lập mã đi vòng. Ví dụ, giá trị cụ thể này có thể là 5. Cụ thể, ví dụ, nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng 5 thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại có thể nhỏ hơn 5 thì cờ dấu có thể được trải qua việc lập mã đi vòng. Trong khi đó, nếu khối con hiện tại là khối có kích thước 4x4 thì số lượng hệ số phần dư khác không có thể là một trong các giá trị trong khoảng từ 0 đến 16, và nếu khối con hiện tại là khối có kích thước 2x2 thì số lượng hệ số phần dư khác không có thể là một trong các giá trị trong khoảng từ 0 đến 4. Sáng chế không giới hạn thứ tự lập mã cờ dấu, và có thể áp dụng cho bước bất kỳ sau khi xác định sự có mặt/vắng mặt của hệ số phần dư khác không của khối con hiện tại.

Theo cách khác, phương án xác định xem liệu sẽ thực hiện việc lập mã ngữ cảnh/lập mã đi vòng trên cờ dấu có thể được đề xuất dựa trên số lượng hệ số phần dư khác không và một giá trị cụ thể được dẫn ra phụ thuộc vào kích thước của khối hiện tại. Ở đây, số lượng hệ số phần dư khác không được sử dụng làm ngưỡng, tức là làm giá trị cụ thể này, có thể là một trong các giá trị trong khoảng từ 0 đến số lượng mẫu của khối hiện tại (khối bao gồm khối con hiện tại), hoặc có thể được kiểm soát trên cơ sở khối con và do đó có thể là một trong các giá trị trong khoảng từ 0 đến 16 trong trường hợp khối con 4x4 và một trong các giá trị trong khoảng từ 0 đến 4 trong trường hợp khối con 2x2. Tức là, giá trị cụ thể này có thể được dẫn ra dựa trên khối hiện tại hoặc kích thước

của khối hiện tại, và nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng giá trị cụ thể này thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn giá trị cụ thể này thì cờ dấu có thể được trải qua việc lập mã đi vòng.

Ví dụ, khi kích thước của khối hiện tại là kích thước 8x8, giá trị cụ thể này có thể được dẫn ra là 5, và nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng 5 thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn 5 thì cờ dấu có thể được trải qua việc lập mã đi vòng. Thêm vào đó, khi kích thước của khối hiện tại là kích thước 4x4, giá trị cụ thể này có thể được dẫn ra là 4, và nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng 4 thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn 4 thì cờ dấu có thể được trải qua việc lập mã đi vòng.

Theo cách khác, phương án để xác định xem liệu cờ dấu sẽ được trải qua việc lập mã ngữ cảnh/lập mã đi vòng có thể được đề xuất dựa trên kích thước của khối hiện tại, giá trị cụ thể được dẫn ra phụ thuộc vào vị trí của khối con hiện tại, và số lượng hệ số phần dư khác không. Ở đây, số lượng hệ số phần dư khác không được sử dụng làm ngưỡng, tức là làm giá trị cụ thể này, có thể là một trong các giá trị trong khoảng từ 0 đến số lượng mẫu của khối hiện tại (khối bao gồm khối con hiện tại), hoặc có thể được kiểm soát trên cơ sở khối con và do đó có thể là một trong các giá trị trong khoảng từ 0 đến 16 trong trường hợp khối con 4x4 và một trong các giá trị trong khoảng từ 0 đến 4 trong trường hợp khối con 2x2. Tức là, giá trị cụ thể này có thể được dẫn ra dựa trên khối hiện tại hoặc kích thước của khối hiện tại và vị trí của khối con hiện tại, và nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng giá trị cụ thể này thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn giá trị cụ thể này thì cờ dấu có thể được trải qua việc lập mã đi vòng.

Ví dụ, khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là CG#3 mà nó được mã hóa theo thứ tự thứ nhất trong số các thứ tự được xác định bởi thứ tự quét theo đường chéo, giá trị cụ thể này có thể được dẫn ra là 5, và nếu số lượng

hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng 5 thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn 5 thì cờ dấu có thể được trải qua việc lập mã đi vòng. Ở đây, CG#3 có thể là khối con dưới cùng bên phải. Thêm vào đó, khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là CG#0 trong số các thứ tự được xác định bởi thứ tự quét theo đường chéo, cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh bất kể số lượng hệ số phần dư khác không của khối con hiện tại. Tức là, giá trị cụ thể có thể được dẫn ra là 0. Ở đây, CG#0 có thể là khối con trên cùng bên trái.

Theo cách khác, phương án để xác định xem liệu cờ dấu sẽ được trải qua việc lập mã ngữ cảnh/lập mã đi vòng có thể được đề xuất dựa trên kích thước của khối hiện tại, giá trị cụ thể được dẫn ra phụ thuộc vào vị trí của khối con hiện tại và chế độ dự đoán của khối hiện tại, và số lượng hệ số phần dư khác không. Ở đây, số lượng hệ số phần dư khác không được sử dụng làm ngưỡng, tức là làm giá trị cụ thể này, có thể là một trong các giá trị trong khoảng từ 0 đến số lượng mẫu của khối hiện tại (khối bao gồm khối con hiện tại), hoặc có thể được kiểm soát trên cơ sở khối con và do đó có thể là một trong các giá trị trong khoảng từ 0 đến 16 trong trường hợp khối con 4x4 và một trong các giá trị trong khoảng từ 0 đến 4 trong trường hợp khối con 2x2. Tức là, giá trị cụ thể này có thể được dẫn ra dựa trên khối hiện tại hoặc kích thước của khối hiện tại, và vị trí của khối con hiện tại và chế độ dự đoán của khối hiện tại, và nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng giá trị cụ thể này thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn giá trị cụ thể này thì cờ dấu có thể được trải qua việc lập mã đi vòng.

Ví dụ, khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là CG#3 mà nó được mã hóa theo thứ tự thứ nhất trong số các thứ tự được xác định bởi thứ tự quét theo đường chéo và nếu chế độ dự đoán của khối hiện tại là chế độ dự đoán nội, giá trị cụ thể này có thể được dẫn ra là 5, và nếu số lượng hệ số phần dư khác không của khối con hiện tại lớn hơn hoặc bằng 5 thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh, và nếu số lượng hệ số phần dư khác không của khối con hiện tại nhỏ hơn 5 thì cờ dấu có thể được trải qua việc lập mã đi vòng. Ở đây, CG#3 có thể là khối con

dưới cùng bên phải. Thêm vào đó, nếu khối con hiện tại là CG#0 trong số các thứ tự được xác định bởi thứ tự quét theo đường chéo thì cờ dấu có thể được lập mã dựa trên mô hình ngữ cảnh bất kể số lượng hệ số phần dư khác không của khối con hiện tại. Tức là, giá trị cụ thể có thể được dẫn ra là 0. Ở đây, CG#0 có thể là khối con trên cùng bên trái.

Theo cách khác, ví dụ, việc liệu cờ dấu sẽ được lập mã dựa trên mô hình ngữ cảnh hay sẽ được lập mã đi vòng có thể được xác định dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không.

Fig.6 minh họa ví dụ về việc xác định mô hình ngữ cảnh của cờ dấu của hệ số phần dư hiện tại theo phương án hiện tại.

Tham chiếu đến Fig.6, thiết bị mã hóa/thiết bị giải mã có thể xác định xem liệu hệ số phần dư hiện tại có là hệ số phần dư thứ nhất của khối hiện tại được mã hóa/giải mã theo thứ tự quét hệ số trong khối hiện tại hay không (S610). Ở đây, hệ số phần dư hiện tại có thể thể hiện hệ số phần dư mà nó là mục tiêu mã hóa/giải mã hiện tại trong số các hệ số phần dư trong khối hiện tại. Ngoài ra, ví dụ, khối hiện tại có thể là đơn vị biến đổi (TU).

Khi hệ số phần dư hiện tại là hệ số phần dư thứ nhất được mã hóa/giải mã trong khối hiện tại, thiết bị mã hóa/thiết bị giải mã có thể dẫn ra biến số lastcoeffsignflag cho cờ dấu của hệ số phần dư hiện tại là 0 (S620), có thể thiết lập ctxInc cho cờ dấu của hệ số phần dư hiện tại là lastcoeffsignflag (S630). Ví dụ, tham chiếu đến bảng 32 trên đây, khi giá trị của lastcoeffsignflag là 0, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 0. Cụ thể, ví dụ, khi giá trị của lastcoeffsignflag là 0, ctxIdc cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 0, vì ctxIdxOffset cho cờ dấu được dẫn ra là 0, ctxIdx cho cờ dấu có thể được dẫn ra là 0 (cụ thể là mô hình ngữ cảnh 0).

Theo cách khác, khi hệ số phần dư hiện tại không là hệ số phần dư được mã hóa/giải mã thứ nhất trong khối hiện tại, thiết bị mã hóa/thiết bị giải mã có thể xác định xem liệu giá trị (lastcoeffsignflag) của cờ dấu của hệ số phần dư được mã hóa/giải mã trước hệ số phần dư hiện tại có là 0 hay không (S640). Ví dụ, hệ số phần dư được mã

hóa/giải mã trước có thể là hệ số phần dư liền kề. Ví dụ, hệ số phần dư được mã hóa/giải mã trước đó có thể bao gồm ít nhất một hệ số trong số hệ số phần dư lân cận bên trái và hệ số phần dư lân cận trên cùng của hệ số phần dư hiện tại. Theo cách khác, ví dụ, hệ số phần dư lân cận có thể bao gồm ít nhất một hệ số trong số hệ số phần dư lân cận bên phải và hệ số phần dư lân cận phía dưới của hệ số phần dư hiện tại. Khi giá trị của cờ dấu của hệ số phần dư được mã hóa/giải mã trước đó là 0, thiết bị mã hóa/thiết bị giải mã có thể dẫn ra biến số lastcoeffsignflag cho cờ dấu của hệ số phần dư hiện tại là 0 (S620), có thể thiết lập ctxInc cho cờ dấu của hệ số phần dư hiện tại là lastcoeffsignflag (S630).

Thêm vào đó, khi giá trị của cờ dấu của hệ số phần dư được mã hóa/giải mã trước đó không là 0, thiết bị mã hóa/thiết bị giải mã có thể dẫn ra biến số lastcoeffsignflag cho cờ dấu của hệ số phần dư hiện tại là 1 (S650), có thể thiết lập ctxInc cho cờ dấu của hệ số phần dư hiện tại là lastcoeffsignflag (S630).

Fig.7 thể hiện làm ví dụ thiết bị mã hóa và thiết bị giải mã để lập mã cờ dấu của hệ số phần dư hiện tại dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không và dấu của hệ số phần dư được lập mã trước đó.

Phần (a) của Fig.7 minh họa làm ví dụ thiết bị mã hóa để mã hóa cờ dấu của hệ số phần dư hiện tại dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không và dấu của hệ số phần dư được lập mã trước hệ số phần dư hiện tại này. Ví dụ, tham chiếu đến phần (a) của Fig.7, thiết bị mã hóa có thể xác định xem liệu việc biến đổi có không được áp dụng cho khối hiện tại hay không, và khi việc biến đổi không được áp dụng cho khối hiện tại, thiết bị mã hóa có thể dẫn ra mô hình ngữ cảnh của cờ dấu của hệ số phần dư hiện tại dựa trên dấu của hệ số phần dư được lập mã trước hệ số phần dư hiện tại cần được mã hóa này. Sau đó, thiết bị mã hóa có thể mã hóa ngữ cảnh cờ dấu của hệ số phần dư hiện tại dựa trên mô hình ngữ cảnh được xác định, và có thể truyền luồng bit bao gồm cờ dấu được mã hóa này.

Thêm vào đó, phần (b) của Fig.7 minh họa làm ví dụ thiết bị giải mã để giải mã cờ dấu của hệ số phần dư hiện tại dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không và dấu của hệ số phần dư được lập mã trước hệ số phần dư hiện tại này. Tham chiếu đến phần (b) của Fig.7, thiết bị giải mã có thể xác định xem liệu việc biến

đổi có không được áp dụng cho khối hiện tại hay không dựa trên cờ bỏ qua biến đổi của khối hiện tại, và khi việc biến đổi không được áp dụng cho khối hiện tại, thiết bị giải mã có thể dẫn ra mô hình ngữ cảnh của cờ dấu của hệ số phần dư hiện tại dựa trên dấu của hệ số phần dư được lập mã trước hệ số phần dư hiện tại cần được giải mã này. Sau đó, thiết bị giải mã có thể giải mã ngữ cảnh cờ dấu của hệ số phần dư hiện tại dựa trên mô hình ngữ cảnh được xác định, và có thể dẫn ra hệ số phần dư hiện tại dựa trên cờ dấu được giải mã này.

Fig.8 minh họa ngắn gọn phương pháp mã hóa ảnh được thực hiện bởi thiết bị mã hóa theo sáng chế này. Phương pháp được bộc lộ trên Fig.8 có thể được thực hiện bởi thiết bị mã hóa được bộc lộ trên Fig.2. Cụ thể, ví dụ, S810 đến S820 trên Fig.8 có thể được thực hiện bởi bộ xử lý phần dư của thiết bị mã hóa, và S830 đến S850 có thể được thực hiện bởi bộ mã hóa entropy của thiết bị mã hóa. Thêm vào đó, mặc dù không được thể hiện, quy trình dẫn ra mẫu dự đoán có thể được thực hiện bởi bộ dự đoán của thiết bị mã hóa này, và quy trình dẫn ra mẫu phần dư cho khối hiện tại, dựa trên mẫu gốc và mẫu dự đoán cho khối hiện tại, có thể được thực hiện bởi bộ trừ của thiết bị mã hóa này, và quy trình tạo ra mẫu được xây dựng lại và hình ảnh được xây dựng lại cho khối hiện tại, dựa trên mẫu phần dư và mẫu dự đoán cho khối hiện tại, có thể được thực hiện bởi bộ cộng của thiết bị mã hóa này.

Thiết bị mã hóa dẫn ra mẫu phần dư của khối hiện tại (S810). Ví dụ, thiết bị mã hóa có thể xác định xem liệu sẽ thực hiện việc dự đoán liên hay việc dự đoán nội trong khối hiện tại, và có thể xác định chế độ dự đoán liên cụ thể hoặc chế độ dự đoán nội cụ thể, dựa trên chi phí RD. Thiết bị mã hóa có thể dẫn ra mẫu dự đoán cho khối hiện tại, dựa trên chế độ được xác định, và có thể dẫn ra mẫu phần dư thông qua việc trừ mẫu dự đoán và mẫu gốc cho khối hiện tại.

Thiết bị mã hóa dẫn ra hệ số phần dư hiện tại của khối hiện tại dựa trên mẫu phần dư (S820).

Sau đó, thiết bị mã hóa có thể xác định xem liệu việc biến đổi có được áp dụng cho khối hiện tại hay không. Tức là, thiết bị mã hóa có thể xác định xem liệu việc biến đổi có được áp dụng cho mẫu phần dư của khối hiện tại hay không. Thiết bị mã hóa có thể xác định xem liệu có áp dụng việc biến đổi cho khối hiện tại hay không qua việc xem

xét hiệu suất lập mã. Ví dụ, thiết bị mã hóa có thể xác định rằng việc biến đổi không được áp dụng cho khối hiện tại.

Nếu việc biến đổi không được áp dụng cho khối hiện tại, tức là, nếu việc biến đổi không được áp dụng cho mẫu phần dư, thì thiết bị mã hóa có thể dẫn ra mẫu phần dư được dẫn ra là hệ số phần dư hiện tại. Thêm vào đó, nếu việc biến đổi được áp dụng cho khối hiện tại, tức là, nếu việc biến đổi được áp dụng cho mẫu phần dư, thì thiết bị mã hóa có thể dẫn ra hệ số phần dư hiện tại bằng cách thực hiện việc biến đổi trên mẫu phần dư được dẫn ra. Hệ số phần dư hiện tại có thể được chứa trong khối con hiện tại của khối hiện tại. Khối con hiện tại có thể được gọi là nhóm hệ số (CG) hiện tại. Thêm vào đó, kích thước của khối con hiện tại của khối hiện tại có thể là kích thước 4x4 hoặc kích thước 2x2. Tức là, khối con hiện tại của khối hiện tại có thể bao gồm lên đến 16 hệ số phần dư khác không hoặc lên đến 4 hệ số phần dư khác không.

Trong khi đó, thiết bị mã hóa có thể tạo ra và mã hóa cờ bỏ qua biến đổi chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư của khối hiện tại hay không. Thông tin phần dư có thể bao gồm cờ bỏ qua biến đổi này cho khối hiện tại. Cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư của khối hiện tại hay không. Tức là, cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư hay không. Các phần tử cú pháp chỉ ra cờ bỏ qua biến đổi có thể là transform_skip_flag nêu trên.

Thiết bị mã hóa dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại (S830).

Ví dụ, thiết bị mã hóa có thể dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại trong số nhiều mô hình ngữ cảnh.

Ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra dựa trên cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại. Ví dụ, nó có thể được dẫn ra dựa trên cờ dấu của hệ số phần dư lân cận của hệ số phần dư hiện tại trong số các hệ số phần dư được mã hóa (hoặc được quét) trước hệ số phần dư hiện tại theo thứ tự quét hệ số cho khối hiện tại. Ví dụ, khi giá trị của cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại là 0, giá trị của chỉ số ngữ cảnh chỉ ra mô hình ngữ

cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 0, và khi giá trị của cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại là 1, giá trị của chỉ số ngữ cảnh chỉ ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 1. Tức là, khi giá trị của cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại là 0, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 0, và khi giá trị của cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại là 1, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 1. Trong khi đó, ví dụ, khi hệ số phần dư hiện tại là hệ số phần dư được mã hóa đầu tiên trong khối hiện tại, giá trị của chỉ số ngữ cảnh chỉ ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 0. Tức là, ví dụ, khi hệ số phần dư hiện tại là hệ số phần dư được mã hóa đầu tiên trong khối hiện tại, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 0. Ngoài ra, ví dụ, khối con hiện tại trong khối hiện tại bao gồm hệ số phần dư hiện tại không là khối con được mã hóa theo thứ tự thứ nhất, và hệ số phần dư hiện tại là hệ số phần dư được mã hóa đầu tiên trong khối con hiện tại, hệ số phần dư được mã hóa trước hệ số phần dư hiện tại có thể được dẫn ra là hệ số phần dư được mã hóa cuối cùng trong khối con được mã hóa trước khối con hiện tại. Tức là, khi khối con hiện tại trong khối hiện tại không là khối con được mã hóa theo thứ tự thứ nhất, và hệ số phần dư hiện tại là hệ số phần dư được mã hóa đầu tiên trong khối con hiện tại, vì hệ số phần dư hiện tại không được mã hóa đầu tiên trong khối hiện tại, tồn tại hệ số phần dư được mã hóa trước đó, nó có thể được dẫn ra là hệ số phần dư được mã hóa trước đó này.

Theo cách khác, trong một ví dụ khác, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra dựa trên các cờ dấu của nhiều hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại. Mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra là một mô hình trong số nhiều mô hình ngữ cảnh, dựa trên các cờ dấu của nhiều hệ số phần dư được mã hóa trước hệ số phần dư hiện tại này trong khối hiện tại. Ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra là một mô hình trong số ba mô hình ngữ cảnh, dựa trên các cờ dấu của hai hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại. Theo cách khác, ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra

là một mô hình trong số sáu mô hình ngữ cảnh, dựa trên các cờ dấu của hai hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

Trong khi đó, thiết bị mã hóa có thể xác định xem liệu cờ dấu có được mã hóa dựa trên mô hình ngữ cảnh hay không, và có thể dẫn ra mô hình ngữ cảnh cho cờ dấu khi xác định được rằng cờ dấu được mã hóa dựa trên mô hình ngữ cảnh.

Ví dụ, dựa trên cờ bỏ qua biến đổi cho khối hiện tại, thiết bị mã hóa có thể xác định xem liệu cờ dấu có được mã hóa dựa trên mô hình ngữ cảnh hay không. Tức là, dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không, thiết bị mã hóa có thể xác định xem liệu cờ dấu có được mã hóa dựa trên mô hình ngữ cảnh hay không. Cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho khối hiện tại hay không. Tức là, cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư của khối hiện tại hay không. Thông tin phần dư cho khối hiện tại có thể bao gồm cờ bỏ qua biến đổi. Nếu giá trị của cờ bỏ qua biến đổi là 0 thì cờ dấu có thể không được mã hóa dựa trên mô hình ngữ cảnh (tức là, cờ dấu có thể được giải mã đi vòng), và nếu giá trị của cờ bỏ qua biến đổi là 1 thì cờ dấu có thể được mã hóa dựa trên mô hình ngữ cảnh. Tức là, nếu giá trị của cờ bỏ qua biến đổi là 1 thì có thể xác định được rằng cờ dấu được mã hóa dựa trên mô hình ngữ cảnh, và thiết bị mã hóa có thể dẫn ra mô hình ngữ cảnh cho cờ dấu, và có thể mã hóa cờ dấu dựa trên mô hình ngữ cảnh.

Theo cách khác, ví dụ, thiết bị mã hóa có thể xác định xem liệu cờ dấu có được mã hóa dựa trên mô hình ngữ cảnh hay không, bằng cách so sánh giá trị cụ thể và số lượng hệ số phần dư khác không trong khối con hiện tại. Ở đây, khối con hiện tại có thể nghĩa là khối con của khối hiện tại bao gồm hệ số phần dư hiện tại. Khi số lượng hệ số phần dư khác không nhỏ hơn giá trị cụ thể này, cờ dấu có thể không được mã hóa dựa trên mô hình ngữ cảnh (tức là, cờ dấu có thể được giải mã đi vòng), và khi số lượng hệ số phần dư khác không lớn hơn hoặc bằng giá trị cụ thể này, cờ dấu có thể được mã hóa dựa trên mô hình ngữ cảnh. Nói cách khác, khi số lượng hệ số phần dư khác không lớn hơn hoặc bằng giá trị cụ thể này, thiết bị mã hóa có thể xác định rằng cờ dấu được mã hóa dựa trên mô hình ngữ cảnh, và thiết bị mã hóa có thể dẫn ra mô hình ngữ cảnh cho cờ dấu, và có thể mã hóa cờ dấu dựa trên mô hình ngữ cảnh. Giá trị cụ thể này cũng có thể được biểu diễn bởi ngưỡng.

Ở đây, ví dụ, giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến số lượng mẫu của khối hiện tại. Ví dụ, giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến 64. Theo cách khác, ví dụ, giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến số lượng khối con hiện tại. Tức là, ví dụ, nếu kích thước của khối con hiện tại là kích thước 4x4 thì giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến 16, và nếu kích thước của khối con hiện tại là kích thước 2x2 thì giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến 4. Ví dụ, giá trị cụ thể này có thể là 5.

Theo cách khác, ví dụ, giá trị cụ thể này có thể được dẫn ra dựa trên kích thước của khối hiện tại. Ví dụ, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8, và giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 4x4.

Theo cách khác, ví dụ, giá trị cụ thể này có thể được dẫn ra dựa trên kích thước của khối hiện tại và vị trí của khối con hiện tại trong khối hiện tại.

Ví dụ, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là khối con dưới cùng bên phải của khối hiện tại. Ở đây, khối con dưới cùng bên phải có thể là khối con #3 (cụ thể là CG#3) theo thứ tự được xác định bởi thứ tự quét theo đường chéo.

Thêm vào đó, ví dụ, giá trị cụ thể này có thể được dẫn ra là 4 khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là khối trên cùng bên trái của khối hiện tại. Ở đây, khối con trên cùng bên trái có thể có thể là khối con #0 (hoặc CG#0) theo thứ tự được xác định bởi thứ tự quét theo đường chéo.

Theo cách khác, ví dụ, giá trị cụ thể này có thể được dẫn ra dựa trên kích thước của khối hiện tại, vị trí của khối con trong khối hiện tại, và chế độ dự đoán của khối hiện tại.

Ví dụ, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là khối con dưới cùng bên phải của khối hiện tại, và chế độ dự đoán của khối hiện tại là chế độ dự đoán nội. Tức là, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là

khối con dưới cùng bên phải của khối hiện tại, và chế độ dự đoán được áp dụng cho khối hiện tại là chế độ dự đoán nội.

Thêm vào đó, ví dụ, giá trị cụ thể này có thể được dẫn ra là 0 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là khối con trên cùng bên trái của khối hiện tại, và chế độ dự đoán của khối hiện tại là chế độ dự đoán nội. Tức là, giá trị cụ thể này có thể được dẫn ra là 0 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là khối con trên cùng bên trái của khối hiện tại, và chế độ dự đoán được áp dụng cho khối hiện tại là chế độ dự đoán nội. Theo đó, có thể xác định được rằng cờ dấu được mã hóa dựa trên mô hình ngữ cảnh bất kể số lượng hệ số phần dư khác không.

Thiết bị mã hóa mã hóa cờ dấu, dựa trên mô hình ngữ cảnh (S840). Thiết bị mã hóa có thể mã hóa cờ dấu, dựa trên mô hình ngữ cảnh. Tức là, thiết bị mã hóa có thể mã hóa cờ dấu theo cách dựa trên ngữ cảnh, dựa trên mô hình ngữ cảnh. Cờ dấu có thể chỉ ra dấu của hệ số phần dư hiện tại. Khi giá trị của cờ dấu là 0, cờ dấu có thể chỉ ra rằng hệ số phần dư hiện tại là giá trị dương (tức là, cờ dấu có thể thể hiện rằng dấu của hệ số phần dư hiện tại là 1), và khi giá trị của cờ dấu là 1, cờ dấu có thể chỉ ra rằng hệ số phần dư hiện tại là giá trị âm (tức là, cờ dấu có thể thể hiện rằng dấu của hệ số phần dư hiện tại là -1). Tức là, nếu giá trị của cờ dấu là 0 thì hệ số phần dư hiện tại có thể là giá trị dương, và nếu giá trị của cờ dấu là 1 thì hệ số phần dư hiện tại có thể là giá trị âm.

Thêm vào đó, thiết bị mã hóa có thể mã hóa thông tin phần dư cho khối hiện tại.

Thông tin phần dư có thể bao gồm các phần tử cú pháp cho hệ số phần dư của khối hiện tại. Thông tin phần dư có thể bao gồm các phần tử cú pháp cho hệ số phần dư hiện tại của khối hiện tại. Ở đây, các phần tử cú pháp có thể bao gồm các phần tử cú pháp được lập mã dựa trên ngữ cảnh và các phần tử cú pháp mà được lập mã đi vòng (cụ thể là các phần tử cú pháp được lập mã dựa trên phân phối xác suất đều).

Ví dụ, thông tin phần dư có thể bao gồm các phần tử cú pháp chẳng hạn như transform_skip_flag, last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, coded_sub_block_flag, sig_coeff_flag, par_level_flag, abs_level_gt1_flag, abs_level_gtX_flag, abs_remainder, coeff_sign_flag, dec_abs_level, và/hoặc mts_idx.

Cụ thể, ví dụ, thông tin phần dư có thể bao gồm cờ bỏ qua biến đổi cho khối hiện tại. Cờ bỏ qua biến đổi có thể chỉ ra liệu có áp dụng việc biến đổi của khối hiện tại hay không. Tức là, cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư của khối hiện tại hay không. Phần tử cú pháp chỉ ra cờ bỏ qua biến đổi có thể là transform_skip_flag nêu trên.

Thêm vào đó, ví dụ, thông tin phần dư có thể bao gồm thông tin vị trí chỉ ra vị trí của hệ số phần dư khác không cuối cùng trong mảng hệ số phần dư của khối hiện tại. Tức là, thông tin phần dư có thể bao gồm thông tin vị trí chỉ ra vị trí của hệ số phần dư khác không cuối cùng theo thứ tự quét của khối hiện tại. Thông tin vị trí có thể bao gồm thông tin chỉ ra tiền tố của vị trí cột của hệ số phần dư khác không cuối cùng, thông tin chỉ ra tiền tố của vị trí hàng của hệ số phần dư khác không cuối cùng, thông tin chỉ ra hậu tố của vị trí cột của hệ số phần dư khác không cuối cùng, và thông tin chỉ ra hậu tố của vị trí hàng của hệ số phần dư khác không cuối cùng. Các phần tử cú pháp cho thông tin vị trí có thể là last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, và last_sig_coeff_y_suffix. Trong khi đó, hệ số phần dư khác không cũng có thể được gọi là hệ số quan trọng.

Thêm vào đó, ví dụ, thông tin phần dư có thể bao gồm các phần tử cú pháp được lập mã dựa trên ngữ cảnh cho hệ số phần dư hiện tại của khối hiện tại. Các phần tử cú pháp này có thể bao gồm cờ hệ số quan trọng chỉ ra liệu hệ số phần dư hiện tại có là hệ số phần dư khác không hay không, cờ mức chẵn lẻ dành cho độ ưu tiên của mức hệ số cho hệ số phần dư hiện tại, cờ mức hệ số thứ nhất dành cho việc liệu mức hệ số có lớn hơn ngưỡng thứ nhất hay không, và cờ mức hệ số thứ hai dành cho việc liệu mức hệ số của hệ số phần dư hiện tại có lớn hơn ngưỡng thứ hai hay không. Ở đây, cờ hệ số quan trọng có thể là sig_coeff_flag, cờ mức chẵn lẻ có thể là par_level_flag, cờ mức hệ số thứ nhất có thể là abs_level_gt1_flag, và cờ mức hệ số thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag.

Thêm vào đó, ví dụ, các phần tử cú pháp được lập mã dựa trên ngữ cảnh cho hệ số phần dư hiện tại có thể bao gồm cờ dấu chỉ ra dấu của hệ số phần dư hiện tại. Ví dụ, nếu việc biến đổi không được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 1) thì các phần tử cú pháp được lập mã dựa trên ngữ cảnh có thể bao gồm cờ

dấu. Tức là, nếu việc biến đổi không được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 1) thì cờ dấu có thể được mã hóa dựa trên mô hình ngữ cảnh.

Thêm vào đó, ví dụ, thông tin phần dư có thể bao gồm phần tử cú pháp được lập mã dựa trên việc đi vòng đối với hệ số phần dư hiện tại của khối hiện tại. Phần tử cú pháp được lập mã đi vòng này có thể bao gồm thông tin liên quan đến giá trị hệ số dành cho giá trị của hệ số phần dư hiện tại. Thông tin liên quan đến giá trị hệ số này có thể là abs_remainder và/hoặc dec_abs_level. Thêm vào đó, ví dụ, nếu việc biến đổi được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 0) thì phần tử cú pháp được lập mã đi vòng có thể bao gồm cờ dấu. Tức là, nếu việc biến đổi được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 0) thì cờ dấu có thể được mã hóa đi vòng (tức là, cờ dấu có thể được mã hóa dựa trên phân phối xác suất đều).

Thiết bị mã hóa tạo ra luồng bit bao gồm cờ dấu (S850). Ví dụ, thiết bị mã hóa có thể xuất ra thông tin ảnh bao gồm thông tin phần dư bao gồm cờ dấu dưới dạng luồng bit. Luồng bit này có thể bao gồm thông tin phần dư này.

Trong khi đó, luồng bit này có thể bao gồm thông tin dự đoán cho khối hiện tại. Thông tin dự đoán này có thể bao gồm thông tin về chế độ dự đoán liên hoặc chế độ dự đoán nội được thực hiện trong khối hiện tại. Thiết bị mã hóa có thể tạo ra và mã hóa thông tin dự đoán cho khối hiện tại.

Trong khi đó, luồng bit này có thể được truyền tới thiết bị giải mã thông qua mạng hoặc phương tiện lưu trữ (số). Ở đây, mạng có thể bao gồm mạng phát quảng bá và/hoặc mạng truyền thông, và phương tiện lưu trữ số có thể bao gồm các phương tiện lưu trữ khác nhau chẳng hạn như bus nối tiếp vạn năng (universal serial bus, USB), SD, đĩa compăc (compact disc, CD), đĩa đa năng kỹ thuật số (digital versatile disc, DVD), Blu-ray, ổ đĩa cứng (hard disk drive, HDD), ổ đĩa trạng thái rắn (solid state drive, SSD), hoặc dạng tương tự.

Fig.9 minh họa ngắn gọn thiết bị mã hóa để thực hiện phương pháp mã hóa ảnh theo sáng chế này. Phương pháp được bộc lộ trên Fig.8 có thể được thực hiện bởi thiết bị mã hóa được bộc lộ trên Fig.9. Cụ thể, ví dụ, bộ xử lý phần dư của thiết bị mã hóa trên Fig.9 có thể thực hiện S810 đến S820 trên Fig.8, và bộ mã hóa entropy của thiết bị

mã hóa trên Fig.9 có thể thực hiện S830 đến S850 trên Fig.8. Thêm vào đó, mặc dù không được thể hiện, quy trình dẫn ra mẫu dự đoán có thể được thực hiện bởi bộ dự đoán của thiết bị mã hóa này, và quy trình dẫn ra mẫu được xây dựng lại cho khối hiện tại, dựa trên mẫu dự đoán này và mẫu phần dư cho khối hiện tại có thể được thực hiện bởi bộ cộng của thiết bị mã hóa này, và quy trình mã hóa thông tin dự đoán cho khối hiện tại có thể được thực hiện bởi bộ mã hóa entropy của thiết bị mã hóa này.

Fig.10 minh họa ngắn gọn phương pháp giải mã ảnh được thực hiện bởi thiết bị giải mã theo sáng chế này. Phương pháp được bộc lộ trên Fig.10 có thể được thực hiện bởi thiết bị giải mã được bộc lộ trên Fig.3. Cụ thể, ví dụ, S1000, S1030 đến S1050 trên Fig.10 có thể được thực hiện bởi bộ giải mã entropy của thiết bị giải mã này, S1020 trên Fig.10 có thể được thực hiện bởi bộ dự đoán của thiết bị giải mã này, S1060 có thể được thực hiện bởi bộ xử lý phần dư của thiết bị giải mã này, và S1070 có thể được thực hiện bởi bộ cộng của thiết bị giải mã này.

Thiết bị giải mã nhận thông tin phần dư cho khối hiện tại (S1010). Thiết bị giải mã có thể nhận thông tin ảnh bao gồm thông tin phần dư cho khối hiện tại thông qua luồng bit. Ở đây, khối hiện tại có thể là khối lập mã (CB) hoặc khối biến đổi (TB). Thông tin phần dư có thể bao gồm các phần tử cú pháp cho hệ số phần dư của khối hiện tại. Ở đây, các phần tử cú pháp có thể bao gồm các phần tử cú pháp được lập mã dựa trên ngữ cảnh và các phần tử cú pháp mà được lập mã đi vòng (cụ thể là các phần tử cú pháp được lập mã dựa trên phân phối xác suất đều).

Ví dụ, thông tin phần dư có thể bao gồm các phần tử cú pháp chẳng hạn như transform_skip_flag, last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, coded_sub_block_flag, sig_coeff_flag, par_level_flag, abs_level_gt1_flag, abs_level_gtX_flag, abs_remainder, coeff_sign_flag, dec_abs_level, và/hoặc mts_idx.

Cụ thể, ví dụ, thông tin phần dư có thể bao gồm cờ bỏ qua biến đổi cho khối hiện tại. Cờ bỏ qua biến đổi có thể chỉ ra liệu có áp dụng việc biến đổi của khối hiện tại hay không. Tức là, cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư của khối hiện tại hay không. Phần tử cú pháp chỉ ra cờ bỏ qua biến đổi có thể là transform_skip_flag nêu trên.

Thêm vào đó, ví dụ, thông tin phần dư có thể bao gồm thông tin vị trí chỉ ra vị trí của hệ số phần dư khác không cuối cùng trong mảng hệ số phần dư của khối hiện tại. Tức là, thông tin phần dư có thể bao gồm thông tin vị trí chỉ ra vị trí của hệ số phần dư khác không cuối cùng theo thứ tự quét của khối hiện tại. Thông tin vị trí có thể bao gồm thông tin chỉ ra tiền tố của vị trí cột của hệ số phần dư khác không cuối cùng, thông tin chỉ ra tiền tố của vị trí hàng của hệ số phần dư khác không cuối cùng, thông tin chỉ ra hậu tố của vị trí cột của hệ số phần dư khác không cuối cùng, và thông tin chỉ ra hậu tố của vị trí hàng của hệ số phần dư khác không cuối cùng. Các phần tử cú pháp cho thông tin vị trí có thể là last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, và last_sig_coeff_y_suffix. Trong khi đó, hệ số phần dư khác không cũng có thể được gọi là hệ số quan trọng.

Thêm vào đó, ví dụ, thông tin phần dư có thể bao gồm các phần tử cú pháp được lập mã dựa trên ngữ cảnh cho hệ số phần dư hiện tại của khối hiện tại. Các phần tử cú pháp này có thể bao gồm cờ hệ số quan trọng chỉ ra liệu hệ số phần dư hiện tại có là hệ số phần dư khác không hay không, cờ mức chẵn lẻ dành cho độ ưu tiên của mức hệ số cho hệ số phần dư hiện tại, cờ mức hệ số thứ nhất dành cho việc liệu mức hệ số có lớn hơn ngưỡng thứ nhất hay không, và cờ mức hệ số thứ hai dành cho việc liệu mức hệ số của hệ số phần dư hiện tại có lớn hơn ngưỡng thứ hai hay không. Ở đây, cờ hệ số quan trọng có thể là sig_coeff_flag, cờ mức chẵn lẻ có thể là par_level_flag, cờ mức hệ số thứ nhất có thể là abs_level_gt1_flag, và cờ mức hệ số thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag.

Thêm vào đó, ví dụ, các phần tử cú pháp được lập mã dựa trên ngữ cảnh cho hệ số phần dư hiện tại có thể bao gồm cờ dấu chỉ ra dấu của hệ số phần dư hiện tại. Ví dụ, nếu việc biến đổi không được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 1) thì các phần tử cú pháp được lập mã dựa trên ngữ cảnh có thể bao gồm cờ dấu. Tức là, nếu việc biến đổi không được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 1) thì cờ dấu có thể được giải mã dựa trên mô hình ngữ cảnh.

Thêm vào đó, ví dụ, thông tin phần dư có thể bao gồm phần tử cú pháp được lập mã dựa trên việc đi vòng đối với hệ số phần dư hiện tại của khối hiện tại. Phần tử cú pháp được lập mã đi vòng này có thể bao gồm thông tin liên quan đến giá trị hệ số dành

cho giá trị của hệ số phần dư hiện tại. Thông tin liên quan đến giá trị hệ số này có thể là abs_remainder và/hoặc dec_abs_level. Thêm vào đó, ví dụ, nếu việc biến đổi được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 0) thì phần tử cú pháp được lập mã đi vòng có thể bao gồm cờ dấu. Tức là, nếu việc biến đổi được áp dụng cho khối hiện tại (tức là, nếu giá trị của cờ bỏ qua biến đổi là 0) thì cờ dấu có thể được giải mã đi vòng (tức là, cờ dấu có thể được giải mã dựa trên phân phối xác suất đều).

Trong khi đó, luồng bit này có thể bao gồm thông tin dự đoán cho khối hiện tại. Thông tin dự đoán này có thể bao gồm thông tin về chế độ dự đoán liên hoặc chế độ dự đoán nội được thực hiện trong khối hiện tại.

Thiết bị giải mã dẫn ra mẫu dự đoán của khối hiện tại (S1020). Ví dụ, thiết bị giải mã có thể thực hiện việc dự đoán liên hoặc việc dự đoán nội trên khối hiện tại dựa trên thông tin dự đoán nhận được qua luồng bit, và có thể dẫn ra mẫu dự đoán của khối hiện tại. Ví dụ, thiết bị giải mã có thể dẫn ra chế độ dự đoán được áp dụng cho khối hiện tại dựa trên thông tin dự đoán. Ví dụ, khi việc dự đoán liên được áp dụng cho khối hiện tại, thiết bị giải mã có thể dẫn ra thông tin chuyển động của khối hiện tại dựa trên thông tin dự đoán, và dẫn ra mẫu dự đoán của khối hiện tại dựa trên thông tin chuyển động này. Ngoài ra, ví dụ, khi việc dự đoán nội được áp dụng cho khối hiện tại, thiết bị giải mã có thể dẫn ra mẫu tham chiếu dựa trên mẫu lân cận của khối hiện tại, và dẫn ra mẫu dự đoán của khối hiện tại dựa trên mẫu tham chiếu này và chế độ dự đoán nội của khối hiện tại.

Thiết bị giải mã dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại của khối hiện tại (S1030).

Ví dụ, thiết bị giải mã có thể dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại trong số nhiều mô hình ngữ cảnh.

Ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra dựa trên cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại. Ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra dựa trên cờ dấu của hệ số phần dư lân cận của hệ số phần dư hiện tại được giải mã trước hệ số phần dư hiện tại trong khối hiện tại. Tức là, nó có thể được dẫn ra dựa trên cờ dấu của hệ số phần dư được giải mã (hoặc được

quét) trước hệ số phần dư hiện tại theo thứ tự quét hệ số cho khối hiện tại. Ví dụ, khi giá trị của cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại là 0, giá trị của chỉ số ngữ cảnh chỉ ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 0, và khi giá trị của cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại là 1, giá trị của chỉ số ngữ cảnh chỉ ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 1. Tức là, khi giá trị của cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại là 0, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 0, và khi giá trị của cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại là 1, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 1. Trong khi đó, ví dụ, khi hệ số phần dư hiện tại là hệ số phần dư được giải mã đầu tiên trong khối hiện tại, giá trị của chỉ số ngữ cảnh chỉ ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là 0. Tức là, ví dụ, khi hệ số phần dư hiện tại là hệ số phần dư được giải mã đầu tiên trong khối hiện tại, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại có thể được dẫn ra là mô hình ngữ cảnh 0. Ngoài ra, ví dụ, khối con hiện tại trong khối hiện tại bao gồm hệ số phần dư hiện tại không là khối con được giải mã theo thứ tự thứ nhất, và hệ số phần dư hiện tại là hệ số phần dư được giải mã đầu tiên trong khối con hiện tại, hệ số phần dư được giải mã trước hệ số phần dư hiện tại có thể được dẫn ra là hệ số phần dư được giải mã cuối cùng trong khối con được giải mã trước khối con hiện tại. Tức là, khi khối con hiện tại trong khối hiện tại không là khối con được giải mã theo thứ tự thứ nhất, và hệ số phần dư hiện tại là hệ số phần dư được giải mã đầu tiên trong khối con hiện tại, vì hệ số phần dư hiện tại không được giải mã đầu tiên trong khối hiện tại, tồn tại hệ số phần dư được giải mã trước đó, nó có thể được dẫn ra là hệ số phần dư được giải mã trước đó này.

Theo cách khác, trong một ví dụ khác, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra dựa trên các cờ dấu của nhiều hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại. Mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra là một mô hình trong số nhiều mô hình ngữ cảnh, dựa trên các cờ dấu của nhiều hệ số phần dư được giải mã trước hệ số phần dư hiện tại này trong khối hiện tại. Ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra là một mô hình trong số ba mô hình ngữ cảnh, dựa

trên các cờ dấu của hai hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại. Theo cách khác, ví dụ, mô hình ngữ cảnh cho cờ dấu có thể được dẫn ra là một mô hình trong số sáu mô hình ngữ cảnh, dựa trên các cờ dấu của hai hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại.

Trong khi đó, thiết bị giải mã có thể xác định xem liệu cờ dấu có được giải mã dựa trên mô hình ngữ cảnh hay không, và có thể dẫn ra mô hình ngữ cảnh cho cờ dấu khi xác định được rằng cờ dấu được giải mã dựa trên mô hình ngữ cảnh.

Ví dụ, dựa trên cờ bỏ qua biến đổi cho khối hiện tại, thiết bị giải mã có thể xác định xem liệu cờ dấu có được giải mã dựa trên mô hình ngữ cảnh hay không. Tức là, dựa trên liệu việc biến đổi có được áp dụng cho khối hiện tại hay không, thiết bị giải mã có thể xác định xem liệu cờ dấu có được giải mã dựa trên mô hình ngữ cảnh hay không. Cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho khối hiện tại hay không. Tức là, cờ bỏ qua biến đổi có thể chỉ ra liệu việc biến đổi có được áp dụng cho các hệ số phần dư của khối hiện tại hay không. Thông tin phần dư cho khối hiện tại có thể bao gồm cờ bỏ qua biến đổi. Nếu giá trị của cờ bỏ qua biến đổi là 0 thì cờ dấu có thể không được giải mã dựa trên mô hình ngữ cảnh (tức là, cờ dấu có thể được giải mã đi vòng), và nếu giá trị của cờ bỏ qua biến đổi là 1 thì cờ dấu có thể được giải mã dựa trên mô hình ngữ cảnh. Tức là, nếu giá trị của cờ bỏ qua biến đổi là 1, có thể xác định được rằng cờ dấu được giải mã dựa trên mô hình ngữ cảnh, và thiết bị giải mã có thể dẫn ra mô hình ngữ cảnh cho cờ dấu, và có thể giải mã cờ dấu dựa trên mô hình ngữ cảnh.

Theo cách khác, ví dụ, thiết bị giải mã có thể xác định xem liệu cờ dấu có được giải mã dựa trên mô hình ngữ cảnh hay không, bằng cách so sánh giá trị cụ thể và số lượng hệ số phần dư khác không trong khối con hiện tại. Ở đây, khối con hiện tại có thể nghĩa là khối con của khối hiện tại bao gồm hệ số phần dư hiện tại. Khi số lượng hệ số phần dư khác không nhỏ hơn giá trị cụ thể này, cờ dấu có thể không được giải mã dựa trên mô hình ngữ cảnh (tức là, cờ dấu có thể được giải mã đi vòng), và khi số lượng hệ số phần dư khác không lớn hơn hoặc bằng giá trị cụ thể này, cờ dấu có thể được giải mã dựa trên mô hình ngữ cảnh. Nói cách khác, khi số lượng hệ số phần dư khác không lớn hơn hoặc bằng giá trị cụ thể này, thiết bị giải mã có thể xác định rằng cờ dấu được giải

mã dựa trên mô hình ngữ cảnh, và thiết bị giải mã có thể dẫn ra mô hình ngữ cảnh cho cờ dấu, và có thể giải mã cờ dấu dựa trên mô hình ngữ cảnh. Giá trị cụ thể này cũng có thể được biểu diễn bởi ngưỡng.

Ở đây, ví dụ, giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến số lượng mẫu của khối hiện tại. Ví dụ, giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến 64. Theo cách khác, ví dụ, giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến số lượng khối con hiện tại. Tức là, ví dụ, nếu kích thước của khối con hiện tại là kích thước 4x4 thì giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến 16, và nếu kích thước của khối con hiện tại là kích thước 2x2 thì giá trị cụ thể này có thể là một trong các giá trị trong khoảng từ 0 đến 4. Ví dụ, giá trị cụ thể này có thể là 5.

Theo cách khác, ví dụ, giá trị cụ thể này có thể được dẫn ra dựa trên kích thước của khối hiện tại. Ví dụ, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8, và giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 4x4.

Theo cách khác, ví dụ, giá trị cụ thể này có thể được dẫn ra dựa trên kích thước của khối hiện tại và vị trí của khối con hiện tại trong khối hiện tại.

Ví dụ, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là khối con dưới cùng bên phải của khối hiện tại. Ở đây, khối con dưới cùng bên phải có thể là khối con #3 (cụ thể là CG#3) theo thứ tự được xác định bởi thứ tự quét theo đường chéo.

Thêm vào đó, ví dụ, giá trị cụ thể này có thể được dẫn ra là 4 khi kích thước của khối hiện tại là kích thước 8x8 và khối con hiện tại là khối trên cùng bên trái của khối hiện tại. Ở đây, khối con trên cùng bên trái có thể có thể là khối con #0 (hoặc CG#0) theo thứ tự được xác định bởi thứ tự quét theo đường chéo.

Theo cách khác, ví dụ, giá trị cụ thể này có thể được dẫn ra dựa trên kích thước của khối hiện tại, vị trí của khối con trong khối hiện tại, và chế độ dự đoán của khối hiện tại.

Ví dụ, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là

kích thước 8x8, khối con hiện tại là khối con dưới cùng bên phải của khối hiện tại, và chế độ dự đoán của khối hiện tại là chế độ dự đoán nội. Tức là, giá trị cụ thể này có thể được dẫn ra là 5 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là khối con dưới cùng bên phải của khối hiện tại, và chế độ dự đoán được áp dụng cho khối hiện tại là chế độ dự đoán nội.

Thêm vào đó, ví dụ, giá trị cụ thể này có thể được dẫn ra là 0 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là khối con trên cùng bên trái của khối hiện tại, và chế độ dự đoán của khối hiện tại là chế độ dự đoán nội. Tức là, giá trị cụ thể này có thể được dẫn ra là 0 khi kích thước của khối hiện tại là kích thước 8x8, khối con hiện tại là khối con trên cùng bên trái của khối hiện tại, và chế độ dự đoán được áp dụng cho khối hiện tại là chế độ dự đoán nội. Theo đó, có thể xác định được rằng cờ dấu được giải mã dựa trên mô hình ngữ cảnh bất kể số lượng hệ số phần dư khác không.

Thiết bị giải mã giải mã cờ dấu, dựa trên mô hình ngữ cảnh (S1040). Thiết bị giải mã có thể giải mã cờ dấu, dựa trên mô hình ngữ cảnh. Cờ dấu có thể chỉ ra dấu của hệ số phần dư hiện tại. Khi giá trị của cờ dấu là 0, cờ dấu có thể chỉ ra rằng hệ số phần dư hiện tại là giá trị dương (tức là, cờ dấu có thể thể hiện rằng dấu của hệ số phần dư hiện tại là 1), và khi giá trị của cờ dấu là 1, cờ dấu có thể chỉ ra rằng hệ số phần dư hiện tại là giá trị âm (tức là, cờ dấu có thể thể hiện rằng dấu của hệ số phần dư hiện tại là -1). Tức là, nếu giá trị của cờ dấu là 0 thì hệ số phần dư hiện tại có thể là giá trị dương, và nếu giá trị của cờ dấu là 1 thì hệ số phần dư hiện tại có thể là giá trị âm.

Thiết bị giải mã dẫn ra hệ số phần dư hiện tại, dựa trên cờ dấu (S1050). Thiết bị giải mã có thể dẫn ra kích thước (cụ thể là giá trị mức) của hệ số phần dư hiện tại, dựa trên thông tin phần dư (ví dụ, thông tin liên quan đến kích thước về hệ số phần dư hiện tại), và có thể dẫn ra hệ số phần dư hiện tại của khối hiện tại bằng cách sử dụng kích thước của hệ số phần dư hiện tại và dấu của hệ số phần dư hiện tại được dẫn ra dựa trên cờ dấu. Tức là, thiết bị giải mã có thể dẫn ra hệ số phần dư hiện tại của khối hiện tại, dựa trên thông tin phần dư (ví dụ, các phần tử cú pháp cho hệ số phần dư hiện tại) và cờ dấu cho hệ số phần dư hiện tại.

Thiết bị giải mã dẫn ra mẫu phần dư, dựa trên hệ số phần dư hiện tại (S1060).

Thiết bị giải mã có thể dẫn ra mẫu phần dư của khối hiện tại, dựa trên hệ số phần dư hiện tại. Tức là, thiết bị giải mã có thể dẫn ra mẫu phần dư của khối hiện tại, dựa trên hệ số phần dư hiện tại. Ví dụ, nếu dẫn ra được rằng việc biến đổi không được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, tức là, nếu giá trị của cờ bỏ qua biến đổi là 1, thì thiết bị giải mã có thể dẫn ra hệ số phần dư hiện tại là mẫu phần dư của khối hiện tại. Theo cách khác, ví dụ, nếu dẫn ra được rằng việc biến đổi không được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, tức là, nếu giá trị của cờ bỏ qua biến đổi là 1, thì thiết bị giải mã có thể dẫn ra mẫu phần dư của khối hiện tại bằng cách giải lượng tử hóa hệ số phần dư hiện tại. Theo cách khác, nếu dẫn ra được rằng việc biến đổi được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, tức là, nếu giá trị của cờ bỏ qua biến đổi là 0, thì thiết bị giải mã có thể dẫn ra mẫu phần dư của khối hiện tại bằng cách biến đổi ngược hệ số phần dư hiện tại. Theo cách khác, ví dụ, nếu dẫn ra được rằng việc biến đổi được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, tức là, nếu giá trị của cờ bỏ qua biến đổi là 0, thì thiết bị giải mã có thể dẫn ra mẫu phần dư bằng cách giải lượng tử hóa hệ số phần dư hiện tại và bằng cách biến đổi ngược hệ số được giải lượng tử hóa này.

Thiết bị giải mã tạo ra hình ảnh được xây dựng lại, dựa trên mẫu phần dư và mẫu dự đoán (S1070).

Ví dụ, thiết bị giải mã có thể dẫn ra mẫu dự đoán bằng cách thực hiện chế độ dự đoán liên hoặc chế độ dự đoán nội cho khối hiện tại, dựa trên thông tin dự đoán nhận được qua luồng bit, và có thể tạo ra hình ảnh được xây dựng lại bằng cách cộng mẫu dự đoán và mẫu phần dư. Thêm vào đó, ví dụ, thông tin dự đoán có thể bao gồm thông tin chỉ ra chế độ dự đoán nội của khối hiện tại. Thiết bị giải mã có thể dẫn ra chế độ dự đoán nội của khối hiện tại, dựa trên thông tin chỉ ra chế độ dự đoán nội, và có thể dẫn ra mẫu dự đoán của khối hiện tại, dựa trên chế độ dự đoán nội và các mẫu tham chiếu của khối hiện tại. Các mẫu tham chiếu có thể bao gồm các mẫu tham chiếu trên cùng và các mẫu tham chiếu bên trái của khối hiện tại. Ví dụ, nếu khối hiện tại có kích thước là NxN và thành phần x và thành phần y của vị trí mẫu trên cùng bên trái của khối hiện tại là 0 thì các mẫu tham chiếu bên trái có thể là p[-1][0] đến p[-1][2N-1] và các mẫu tham chiếu trên cùng có thể là p[0][-1] đến p[2N-1][-1].

Sau đó, theo cách tùy chọn, thủ tục lọc trong vòng chẳng hạn như việc lọc khử khối, SAO, và/hoặc các thủ tục ALF có thể được áp dụng cho hình ảnh được xây dựng lại như được mô tả trên đây để cải thiện chất lượng hình ảnh chủ quan/khách quan.

Fig.11 minh họa ngắn gọn thiết bị giải mã để thực hiện phương pháp giải mã ảnh theo sáng chế này. Phương pháp được bộc lộ trên Fig.10 có thể được thực hiện bởi thiết bị giải mã được bộc lộ trên Fig.11. Cụ thể, ví dụ, bộ giải mã entropy của thiết bị giải mã trên Fig.11 có thể thực hiện S1010, S1030 đến S1050 trên Fig.10, bộ dự đoán của thiết bị giải mã trên Fig.11 có thể thực hiện S1020 trên Fig.10, bộ xử lý phần dư của thiết bị giải mã trên Fig.11 có thể thực hiện S1060 trên Fig.10, và bộ cộng của thiết bị giải mã trên Fig.11 có thể thực hiện S1070 trên Fig.10.

Theo nội dung bộc lộ nêu trên, hiệu quả của việc lập mã phần dư có thể được cải thiện.

Thêm vào đó, theo sáng chế này, cờ dấu chỉ ra dấu của hệ số phần dư có thể được lập mã dựa trên mô hình ngữ cảnh, qua đó tiết kiệm lượng bit được gán cho cờ dấu cho hệ số phần dư và cải thiện hiệu suất lập mã phần dư tổng thể.

Thêm vào đó, theo sáng chế này, mô hình ngữ cảnh cho cờ dấu thể hiện dấu của hệ số phần dư có thể được dẫn ra dựa trên dấu của hệ số phần dư lân cận liền kề hệ số phần dư này, thông qua điều này, cờ dấu có thể được lập mã có xem xét đến mối tương quan giữa các các hệ số phần dư liền kề, lượng bit được cấp phát cho cờ dấu có thể được giảm xuống, và hiệu suất lập mã phần dư tổng thể có thể được cải thiện.

Thêm vào đó, theo sáng chế này, mô hình ngữ cảnh cho cờ dấu thể hiện dấu của hệ số phần dư có thể được dẫn ra dựa trên dấu của hệ số phần dư lân cận được giải mã trước hệ số phần dư này, thông qua điều này, cờ dấu có thể được lập mã có xem xét đến mối tương quan giữa các các hệ số phần dư liền kề, lượng bit được cấp phát cho cờ dấu có thể được giảm xuống, và hiệu suất lập mã phần dư tổng thể có thể được cải thiện.

Trong phương án được mô tả trên đây, các phương pháp được mô tả dựa trên lưu đồ có hàng loạt các bước hoặc các khối. Sáng chế không bị giới hạn ở thứ tự của các bước hoặc các khối nêu trên. Một số bước hoặc một số khối có thể xảy ra đồng thời hoặc theo thứ tự khác với các bước hoặc các khối khác như được mô tả trên đây. Hơn nữa,

những người có hiểu biết trong lĩnh vực kỹ thuật tương ứng sẽ hiểu rằng các bước được thể hiện trên các lưu đồ nêu trên là không loại trừ, rằng thêm các bước có thể được đưa vào, hoặc hiểu rằng một hoặc nhiều bước trong lưu đồ này có thể được xóa mà không làm ảnh hưởng đến phạm vi của sáng chế.

Các phương án được mô tả trong bản mô tả này có thể được thực hiện bằng cách được thi hành trên bộ xử lý, bộ vi xử lý, bộ điều khiển hoặc chip. Ví dụ, các đơn vị chức năng được thể hiện trên mỗi hình vẽ có thể được thực hiện bằng cách được thi hành trên máy tính, bộ xử lý, bộ vi xử lý, bộ điều khiển hoặc chip. Trong trường hợp này, thông tin dành cho việc thi hành (ví dụ, thông tin về các lệnh) hoặc các thuật toán có thể được lưu trữ trong phương tiện lưu trữ kỹ thuật số.

Thêm vào đó, thiết bị giải mã và thiết bị mã hóa mà sáng chế được áp dụng với chúng có thể được chứa trong máy truyền/nhận phát quảng bá đa phương tiện, thiết bị đầu cuối truyền thông di động, thiết bị viđeo rạp chiếu phim gia đình, thiết bị viđeo rạp chiếu phim số, camera giám sát, thiết bị trò chuyện viđeo, thiết bị truyền thông thời gian thực chẳng hạn như truyền thông viđeo, thiết bị tạo luồng di động, phương tiện lưu trữ, máy quay viđeo, thiết bị cung cấp dịch vụ VoD, thiết bị viđeo trên cùng (Over the top, OTT), thiết bị cung cấp dịch vụ tạo luồng Internet, thiết bị viđeo ba chiều (three-dimensional, 3D), thiết bị viđeo hội nghị từ xa, thiết bị người dùng giao thông vận tải (ví dụ, thiết bị người dùng xe, thiết bị người dùng máy bay, thiết bị người dùng tàu, v.v.) và thiết bị viđeo y tế và có thể được sử dụng để xử lý các tín hiệu viđeo và các tín hiệu dữ liệu. Ví dụ, thiết bị viđeo trên cùng (OTT) có thể bao gồm bảng điều khiển trò chơi, máy đọc đĩa Blu-ray, ti vi truy cập Internet, hệ thống rạp hát gia đình, điện thoại thông minh, PC bảng, đầu ghi viđeo số (Digital Video Recorder, DVR), hoặc loại tương tự.

Hơn nữa, phương pháp xử lý mà sáng chế được áp dụng với nó có thể được tạo ra dưới dạng chương trình mà nó cần được thực thi bởi máy tính và có thể được lưu trữ trong phương tiện ghi có thể đọc được bằng máy tính. Dữ liệu đa phương tiện có cấu trúc dữ liệu theo sáng chế cũng có thể được lưu trữ trong các phương tiện ghi có thể đọc được bằng máy tính. Phương tiện ghi có thể đọc được bằng máy tính bao gồm tất cả các loại thiết bị lưu trữ mà dữ liệu đọc được bởi hệ thống máy tính được lưu trữ trong đó. Các phương tiện ghi có thể đọc được bằng máy tính, ví dụ, có thể bao gồm BD, bus nối

tiếp vạn năng (Universal Serial Bus, USB), ROM, PROM, EPROM, EEPROM, RAM, CD-ROM, băng từ, đĩa mềm, và thiết bị lưu trữ dữ liệu quang. Hơn nữa, các phương tiện ghi có thể đọc được bằng máy tính bao gồm các phương tiện được thi hành dưới dạng các sóng mang (ví dụ, hoạt động truyền qua Internet). Thêm vào đó, luồng bit được tạo ra bởi phương pháp mã hóa có thể được lưu trữ trong phương tiện ghi có thể đọc được bằng máy tính hoặc có thể được truyền qua các mạng truyền thông có dây/không dây.

Thêm vào đó, các phương án của sáng chế có thể được thi hành với sản phẩm chương trình máy tính theo các mã chương trình, và các mã chương trình này có thể được thực hiện trong máy tính bởi các phương án của sáng chế. Các mã chương trình có thể được lưu trữ trên bộ mang mà nó là có thể đọc được bởi máy tính.

Fig.12 minh họa sơ đồ cấu trúc của hệ thống tạo luồng các nội dung mà sáng chế được áp dụng cho hệ thống này.

Hệ thống tạo luồng nội dung này, mà phương án (các phương án) của sáng chế này được áp dụng với nó, có thể phần lớn bao gồm máy chủ mã hóa, máy chủ tạo luồng, máy chủ web, bộ lưu trữ phương tiện, thiết bị người dùng, và thiết bị đầu vào đa phương tiện.

Máy chủ mã hóa nén nội dung được nhập vào từ các thiết bị đầu vào đa phương tiện như điện thoại thông minh, camera, máy quay viđeo, v.v. thành dữ liệu số để tạo ra luồng bit và truyền luồng bit này tới máy chủ tạo luồng. Như là một ví dụ khác, khi các thiết bị đầu vào đa phương tiện như điện thoại thông minh, camera, máy quay viđeo, v.v. trực tiếp tạo ra luồng bit, máy chủ mã hóa có thể được lược bỏ.

Luồng bit có thể được tạo ra bởi phương pháp mã hóa hoặc phương pháp tạo ra luồng bit mà phương án (các phương án) của sáng chế này được áp dụng với chúng, và máy chủ tạo luồng có thể lưu trữ tạm thời luồng bit trong quy trình truyền hoặc nhận luồng bit.

Máy chủ tạo luồng truyền dữ liệu đa phương tiện tới thiết bị người dùng dựa trên yêu cầu của người dùng qua máy chủ web, và máy chủ web đóng vai trò như là phương tiện để thông báo cho người dùng về dịch vụ. Khi người dùng yêu cầu dịch vụ mong

muốn từ máy chủ web, máy chủ web phân phát nó đến máy chủ tạo luồng, và máy chủ tạo luồng truyền dữ liệu đa phương tiện đến người dùng. Trong trường hợp này, hệ thống tạo luồng nội dung có thể bao gồm máy chủ điều khiển riêng biệt. Trong trường hợp này, máy chủ điều khiển này đóng vai trò điều khiển mệnh lệnh/phản hồi giữa các thiết bị trong hệ thống tạo luồng nội dung.

Máy chủ tạo luồng có thể nhận nội dung từ bộ lưu trữ phương tiện và/hoặc máy chủ mã hóa. Ví dụ, khi nội dung được nhận từ máy chủ mã hóa, nội dung này có thể được nhận theo thời gian thực. Trong trường hợp này, để cung cấp dịch vụ tạo luồng suôn sẻ, máy chủ tạo luồng có thể lưu trữ luồng bit trong thời gian được xác định trước.

Các ví dụ về thiết bị người dùng có thể bao gồm điện thoại di động, điện thoại thông minh, máy tính xách tay, thiết bị đầu cuối phát quảng bá kỹ thuật số, thiết bị trợ giúp số cá nhân (personal digital assistant, PDA), thiết bị trình diễn đa phương tiện di động (portable multimedia player, PMP), thiết bị định vị, PC dạng phiến, các PC dạng bảng, các máy tính siêu di động (ultrabook), các thiết bị đeo được (ví dụ, các đồng hồ thông minh, các kính thông minh, các thiết bị hiển thị gắn trên đầu), các TV kỹ thuật số, các máy tính để bàn, bảng ký hiệu kỹ thuật số, và dạng tương tự. Mỗi máy chủ trong hệ thống tạo luồng nội dung có thể được vận hành như là máy chủ phân tán, mà trong trường hợp đó dữ liệu được nhận từ từng máy chủ có thể được phân tán.

Các điểm yêu cầu bảo hộ được mô tả trong sáng chế có thể được kết hợp theo nhiều cách khác nhau. Ví dụ, các dấu hiệu kỹ thuật của các yêu cầu bảo hộ dạng phương pháp của bản mô tả này có thể được kết hợp để được thi hành dưới dạng một thiết bị, và các dấu hiệu kỹ thuật của các yêu cầu bảo hộ dạng thiết bị của bản mô tả này có thể được kết hợp để được thi hành dưới dạng phương pháp. Thêm vào đó, các dấu hiệu kỹ thuật của yêu cầu bảo hộ dạng phương pháp của bản mô tả này và các dấu hiệu kỹ thuật của yêu cầu bảo hộ dạng thiết bị có thể được kết hợp để được thi hành dưới dạng thiết bị, và các dấu hiệu kỹ thuật của yêu cầu bảo hộ dạng phương pháp của bản mô tả này và các dấu hiệu kỹ thuật của yêu cầu bảo hộ dạng thiết bị có thể được kết hợp để được thi hành dưới dạng phương pháp.

YÊU CẦU BẢO HỘ

1. Phương pháp giải mã ảnh, được thực hiện bởi thiết bị giải mã, phương pháp này bao gồm các bước:

nhận thông tin phần dư cho khối hiện tại;

dẫn ra mẫu dự đoán của khối hiện tại;

dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại của khối hiện tại;

giải mã cờ dấu dựa trên mô hình ngữ cảnh;

dẫn ra hệ số phần dư hiện tại dựa trên cờ dấu;

dẫn ra mẫu phần dư dựa trên hệ số phần dư hiện tại; và

tạo ra hình ảnh được xây dựng lại dựa trên mẫu dự đoán và mẫu phần dư,

trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại.

2. Phương pháp giải mã ảnh theo điểm 1,

trong đó khi hệ số phần dư hiện tại là hệ số phần dư được giải mã đầu tiên trong khối hiện tại bao gồm hệ số phần dư hiện tại, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại được dẫn ra là mô hình ngữ cảnh 0.

3. Phương pháp giải mã ảnh theo điểm 1,

trong đó khi giá trị của cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại là 0, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại được dẫn ra là mô hình ngữ cảnh 0, và

trong đó khi giá trị của cờ dấu của hệ số phần dư được giải mã trước hệ số phần dư hiện tại là 1, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại được dẫn ra là mô hình ngữ cảnh 1.

4. Phương pháp giải mã ảnh theo điểm 1, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra là một mô hình trong số nhiều mô hình ngữ cảnh dựa trên các cờ dấu của nhiều

hệ số phần dư được giải mã trước hệ số phần dư hiện tại trong khối hiện tại.

5. Phương pháp giải mã ảnh theo điểm 1,

trong đó khi khối con hiện tại trong khối hiện tại không là khối con được giải mã theo thứ tự thứ nhất và hệ số phần dư hiện tại là hệ số phần dư được giải mã đầu tiên trong khối con hiện tại, hệ số phần dư được giải mã trước hệ số phần dư hiện tại được dẫn ra là hệ số phần dư được giải mã cuối cùng trong khối con được giải mã trước khối con hiện tại.

6. Phương pháp giải mã ảnh theo điểm 1,

trong đó hệ số phần dư được giải mã trước hệ số phần dư hiện tại là hệ số phần dư liền kề hệ số phần dư hiện tại.

7. Phương pháp giải mã ảnh theo điểm 1, còn bao gồm bước xác định xem liệu cờ dấu có được giải mã dựa trên mô hình ngữ cảnh hay không, dựa trên liệu việc biến đổi khối hiện tại có được áp dụng hay không.

8. Phương pháp giải mã ảnh theo điểm 7,

trong đó khi việc biến đổi được áp dụng cho khối hiện tại, cờ dấu được giải mã đi vòng (bypass),

trong đó khi việc biến đổi không được áp dụng cho khối hiện tại, cờ dấu được giải mã dựa trên mô hình ngữ cảnh.

9. Phương pháp mã hóa ảnh, được thực hiện bởi thiết bị mã hóa, phương pháp này bao gồm các bước:

dẫn ra mẫu phần dư của khối hiện tại;

dẫn ra hệ số phần dư hiện tại của khối hiện tại dựa trên mẫu phần dư;

dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại;

mã hóa cờ dấu dựa trên mô hình ngữ cảnh; và

tạo ra luồng bit bao gồm cờ dấu,

trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

10. Phương pháp mã hóa ảnh theo điểm 9,

trong đó khi hệ số phần dư hiện tại là hệ số phần dư được mã hóa đầu tiên trong khối hiện tại bao gồm hệ số phần dư hiện tại, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại được dẫn ra là mô hình ngữ cảnh 0.

11. Phương pháp mã hóa ảnh theo điểm 9,

trong đó khi giá trị của cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại là 0, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại được dẫn ra là mô hình ngữ cảnh 0, và

trong đó khi giá trị của cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại là 1, mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại được dẫn ra là mô hình ngữ cảnh 1.

12. Phương pháp mã hóa ảnh theo điểm 9, trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra là một mô hình trong số nhiều mô hình ngữ cảnh dựa trên các cờ dấu của nhiều hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

13. Phương pháp mã hóa ảnh theo điểm 9,

trong đó khi khối con hiện tại trong khối hiện tại không là khối con được mã hóa theo thứ tự thứ nhất và hệ số phần dư hiện tại là hệ số phần dư được mã hóa đầu tiên trong khối con hiện tại, hệ số phần dư được mã hóa trước hệ số phần dư hiện tại được dẫn ra là hệ số phần dư được mã hóa cuối cùng trong khối con được mã hóa trước khối con hiện tại.

14. Phương pháp mã hóa ảnh theo điểm 9, còn bao gồm bước xác định xem liệu cờ dấu có được mã hóa dựa trên mô hình ngữ cảnh hay không, dựa trên liệu việc biến đổi khối hiện tại có được áp dụng hay không.

15. Phương tiện lưu trữ phi chuyển tiếp đọc được bằng máy tính lưu trữ luồng bit được tạo ra bởi phương pháp, phương pháp này bao gồm các bước:

dẫn ra mẫu phần dư của khối hiện tại;

dẫn ra hệ số phần dư hiện tại của khối hiện tại dựa trên mẫu phần dư;

dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại;

mã hóa cờ dấu dựa trên mô hình ngữ cảnh; và

tạo ra luồng bit này bao gồm cờ dấu,

trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

16. Phương pháp truyền dữ liệu cho ảnh, phương pháp này bao gồm các bước:

thu luồng bit của thông tin ảnh bao gồm cờ dấu của hệ số phần dư hiện tại của khối hiện tại; và

truyền dữ liệu bao gồm luồng bit của thông tin ảnh bao gồm cờ dấu này,

trong đó hệ số phần dư hiện tại được dẫn ra dựa trên mẫu phần dư của khối hiện tại,

trong đó cờ dấu được mã hóa dựa trên mô hình ngữ cảnh cho cờ dấu, và

trong đó mô hình ngữ cảnh cho cờ dấu được dẫn ra dựa trên cờ dấu của hệ số phần dư được mã hóa trước hệ số phần dư hiện tại trong khối hiện tại.

FIG. 1

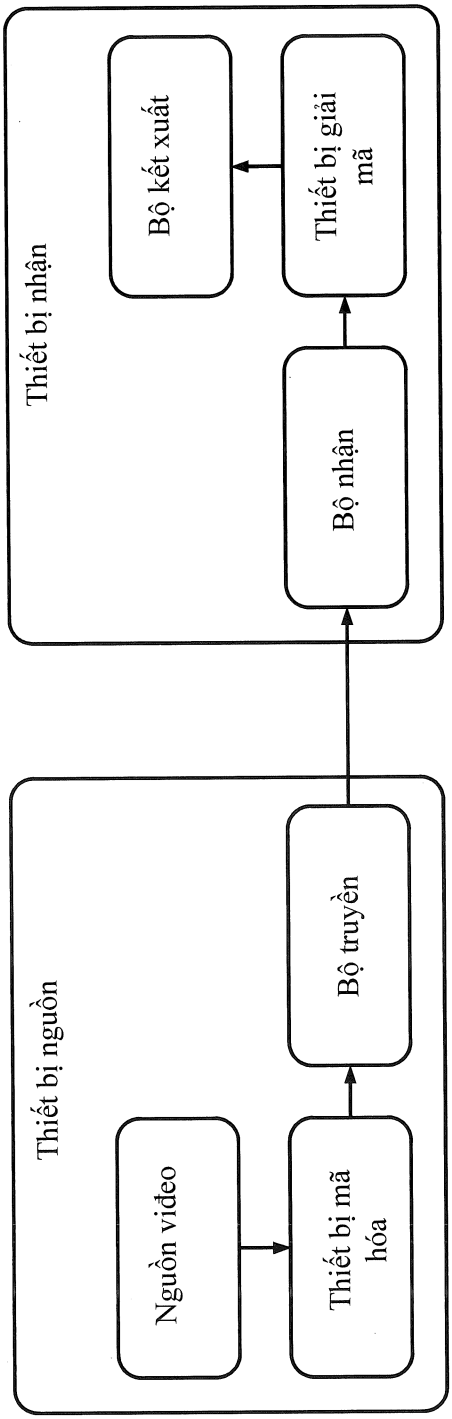

Thiết bị nguồn
Nguồn video
Thiết bị mã hóa
Bộ truyền
Thiết bị nhận
Bộ nhận
Thiết bị giải mã
Bộ kết xuất

FIG. 2

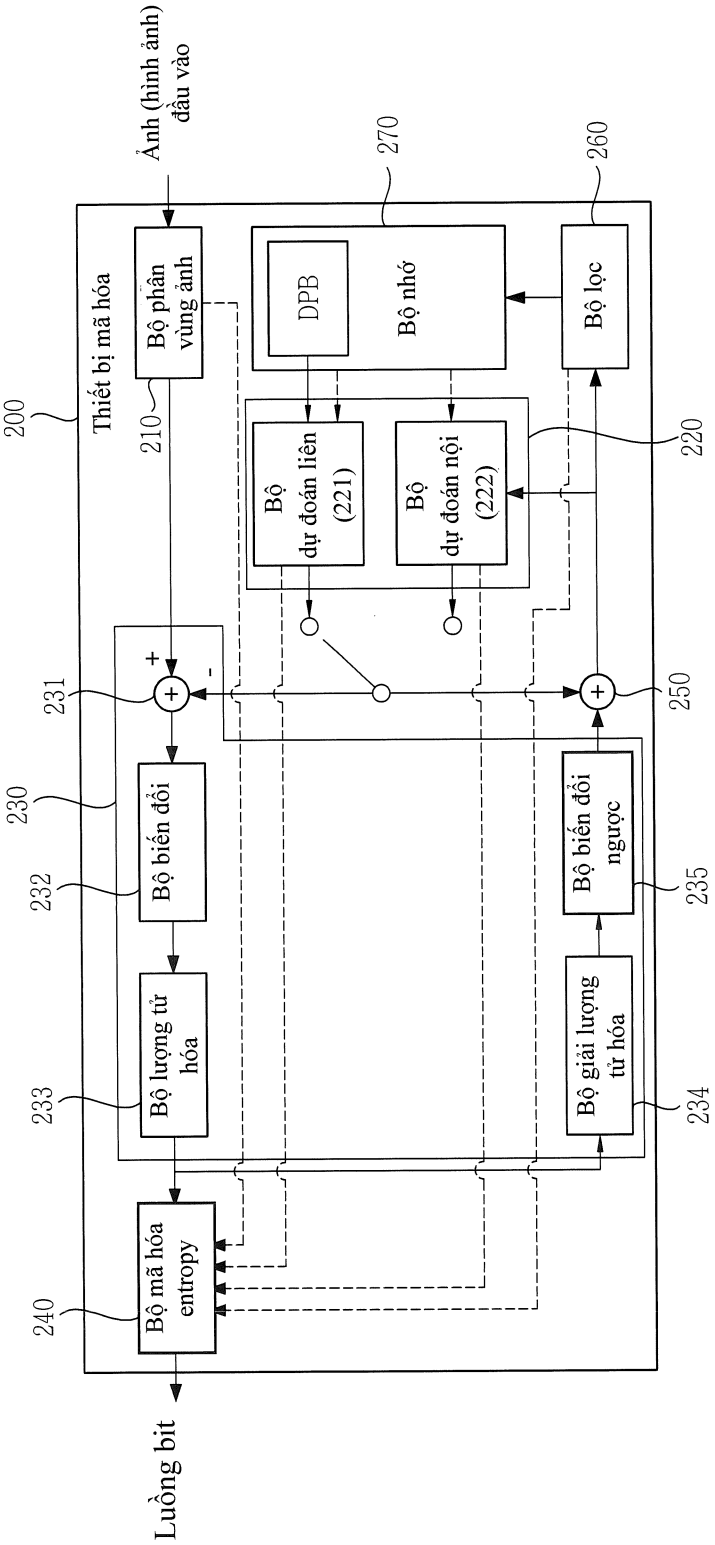

Ảnh (hình ảnh) đầu vào
Thiết bị mã hóa
200
Bộ phân vùng ảnh
210
270
DPB
Bộ nhớ
260
Bộ lọc
Bộ dự đoán liên (221)
Bộ dự đoán nội (222)
220
+
-
231
250
230
Bộ biến đổi
232
Bộ lượng tử hóa
233
Bộ biến đổi ngược
235
Bộ giải lượng tử hóa
234
Bộ mã hóa entropy
240
Luồng bit

FIG. 3

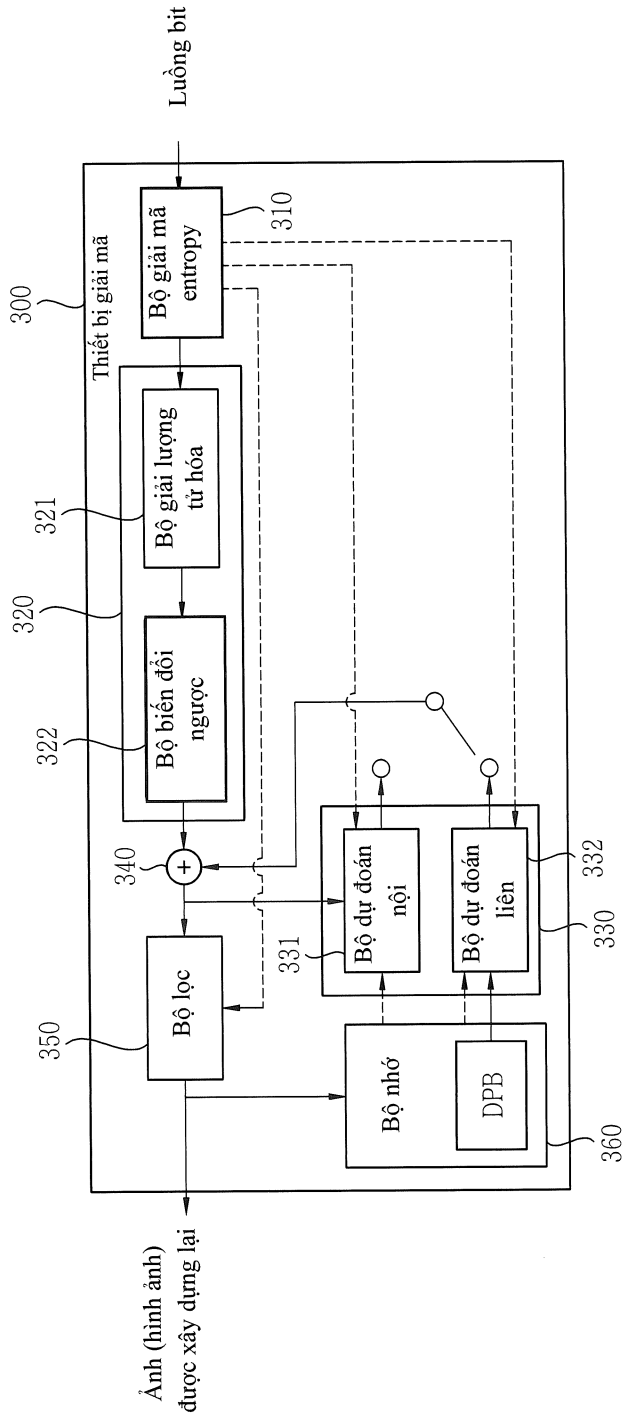

Luồng bit
Thiết bị giải mã
300
Bộ giải mã entropy
310
Bộ giải lượng tử hóa
321
Bộ biến đổi ngược
322
320
340
Bộ dự đoán nội
Bộ dự đoán liên
331
332
330
Bộ lọc
350
Bộ nhớ
DPB
360
Ảnh (hình ảnh) được xây dựng lại

FIG. 4

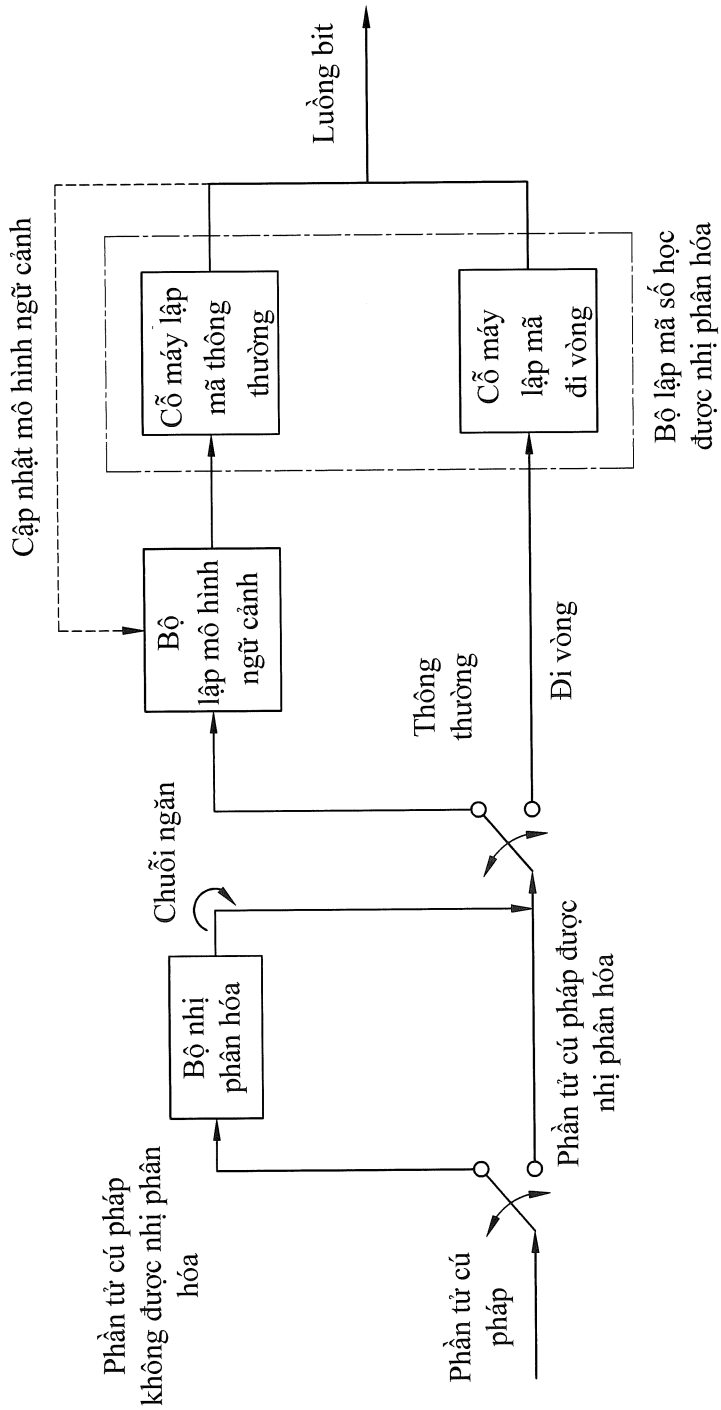

Luồng bit
Cập nhật mô hình ngữ cảnh
Cỗ máy lập mã thông thường
Cỗ máy lập mã đi vòng
Bộ lập mã số học được nhị phân hóa
Bộ lập mô hình ngữ cảnh
Thông thường
Đi vòng
Chuỗi ngăn
Bộ nhị phân hóa
Phần tử cú pháp được nhị phân hóa
Phần tử cú pháp không được nhị phân hóa
Phần tử cú pháp

FIG. 5

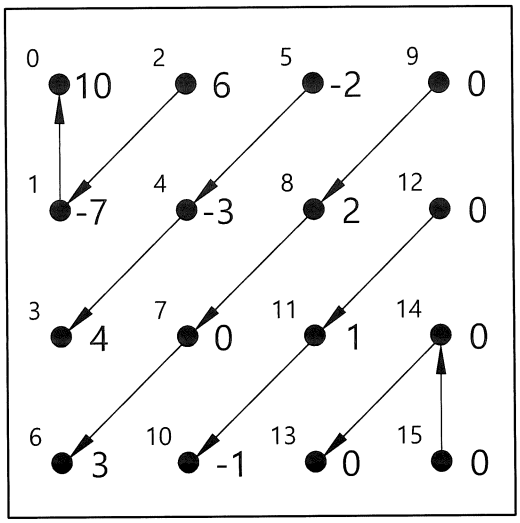


FIG. 6

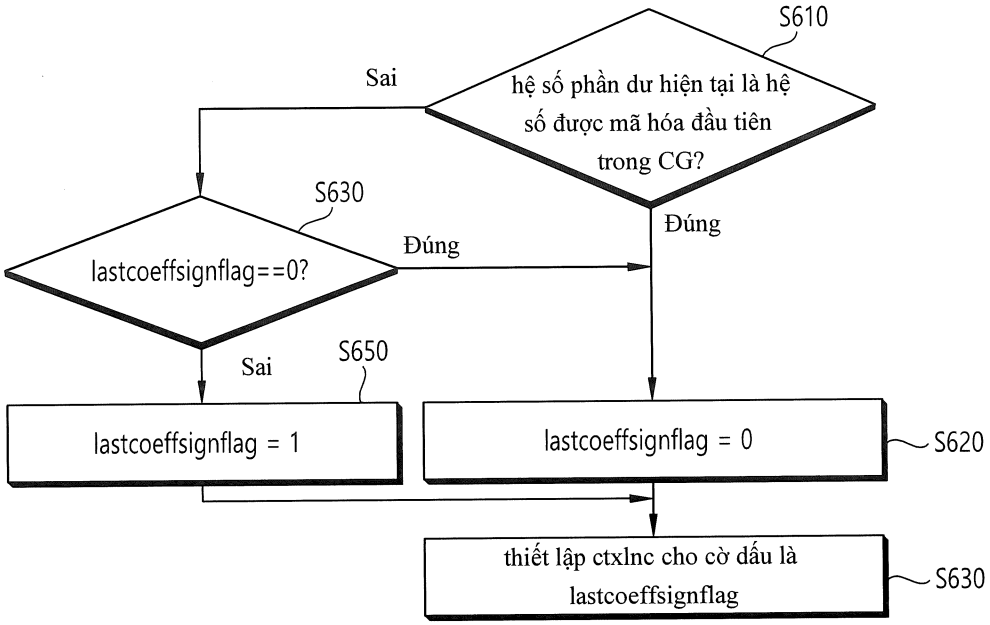
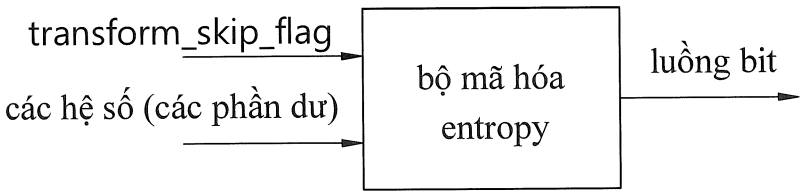
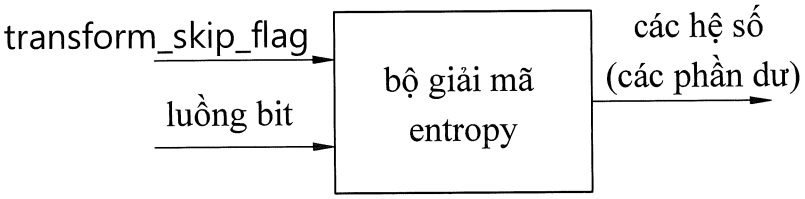

S610
hệ số phần dư hiện tại là hệ số được mã hóa đầu tiên trong CG?
Sai
Đúng
S630
lastcoeffsignflag==0?
Đúng
Sai
S650
lastcoeffsignflag = 1
lastcoeffsignflag = 0
S620
thiết lập ctxlnc cho cờ dấu là lastcoeffsignflag
S630

FIG. 7


transform_skip_flag
các hệ số (các phần dư)
bộ mã hóa entropy
luồng bit

(a)


transform_skip_flag
luồng bit
bộ giải mã entropy
các hệ số (các phần dư)

(b)

FIG. 8

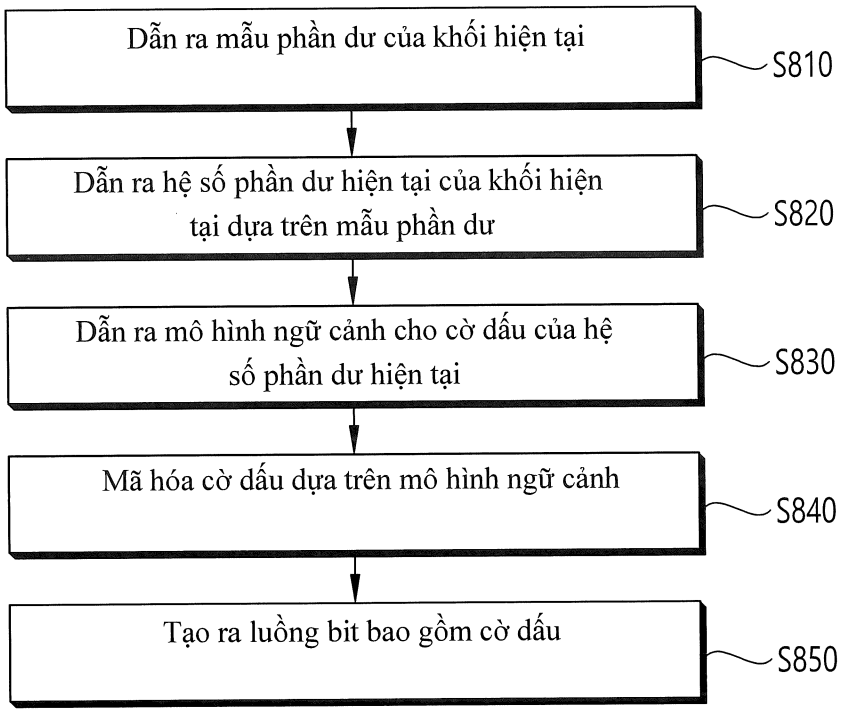

Dẫn ra mẫu phần dư của khối hiện tại
S810
Dẫn ra hệ số phần dư hiện tại của khối hiện tại dựa trên mẫu phần dư
S820
Dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại
S830
Mã hóa cờ dấu dựa trên mô hình ngữ cảnh
S840
Tạo ra luồng bit bao gồm cờ dấu
S850

FIG. 9

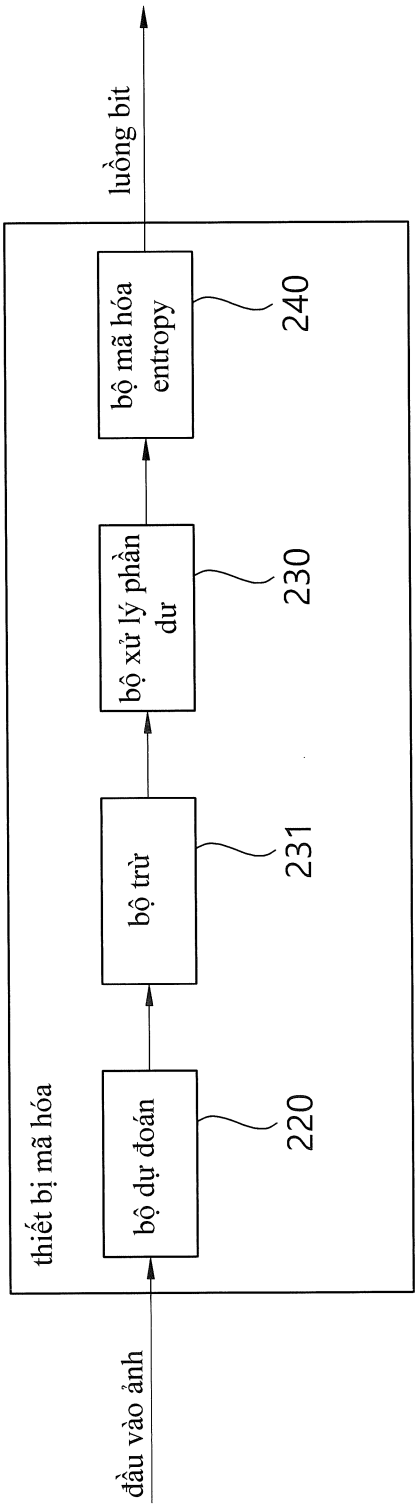

đầu vào ảnh
thiết bị mã hóa
bộ dự đoán
220
bộ trừ
231
bộ xử lý phần dư
230
bộ mã hóa entropy
240
luồng bit

FIG. 10

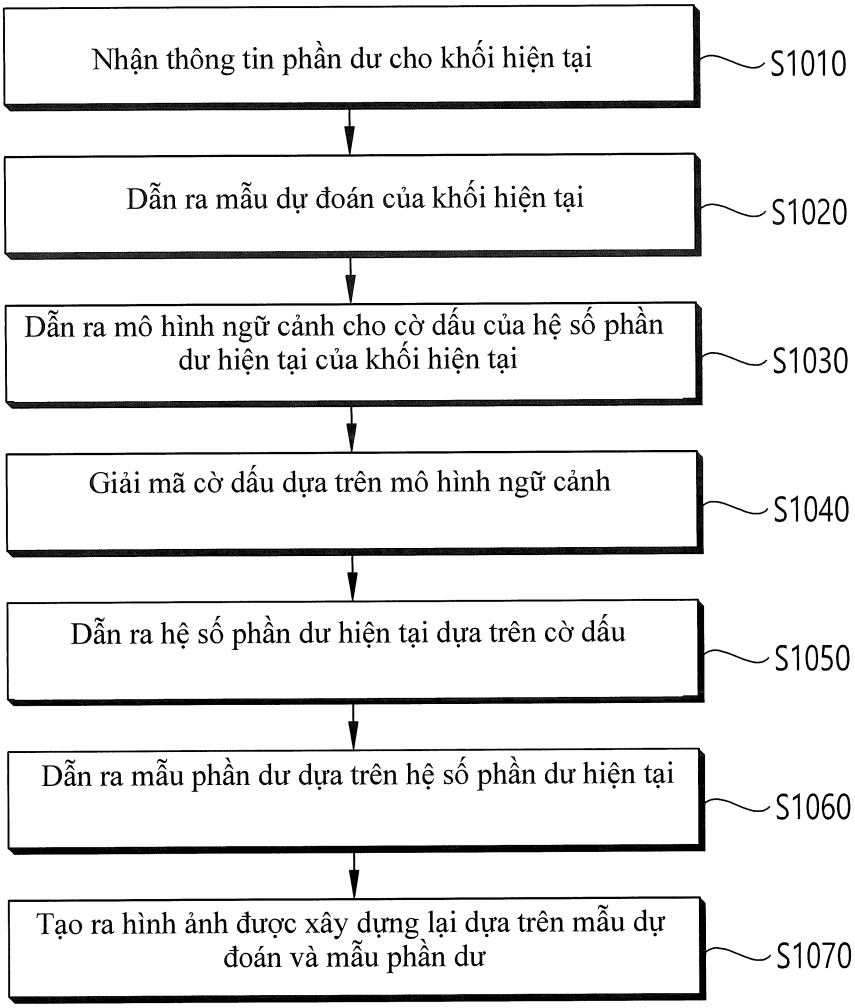

Nhận thông tin phần dư cho khối hiện tại
S1010
Dẫn ra mẫu dự đoán của khối hiện tại
S1020
Dẫn ra mô hình ngữ cảnh cho cờ dấu của hệ số phần dư hiện tại của khối hiện tại
S1030
Giải mã cờ dấu dựa trên mô hình ngữ cảnh
S1040
Dẫn ra hệ số phần dư hiện tại dựa trên cờ dấu
S1050
Dẫn ra mẫu phần dư dựa trên hệ số phần dư hiện tại
S1060
Tạo ra hình ảnh được xây dựng lại dựa trên mẫu dự đoán và mẫu phần dư
S1070

FIG. 11

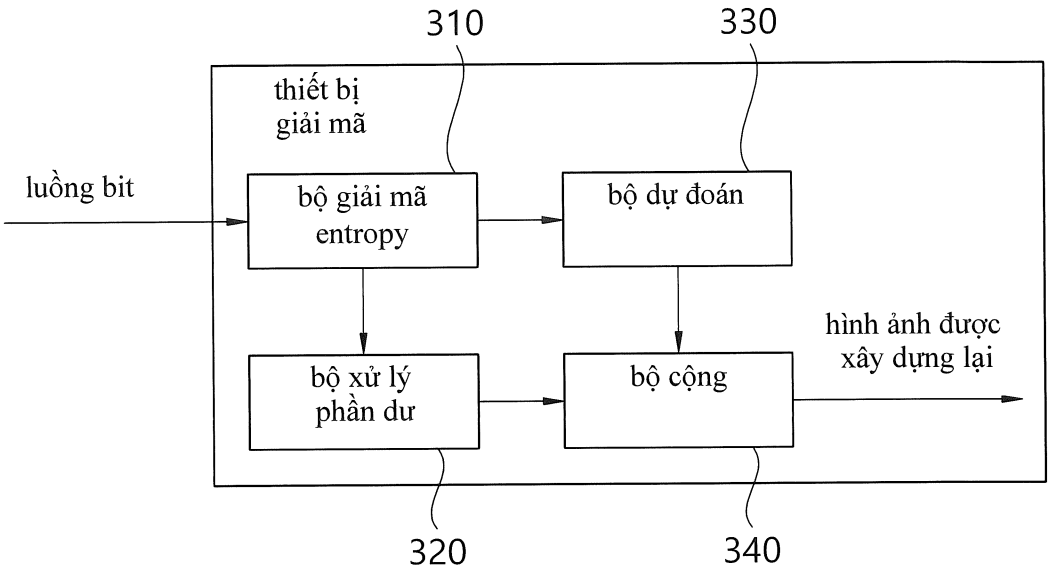
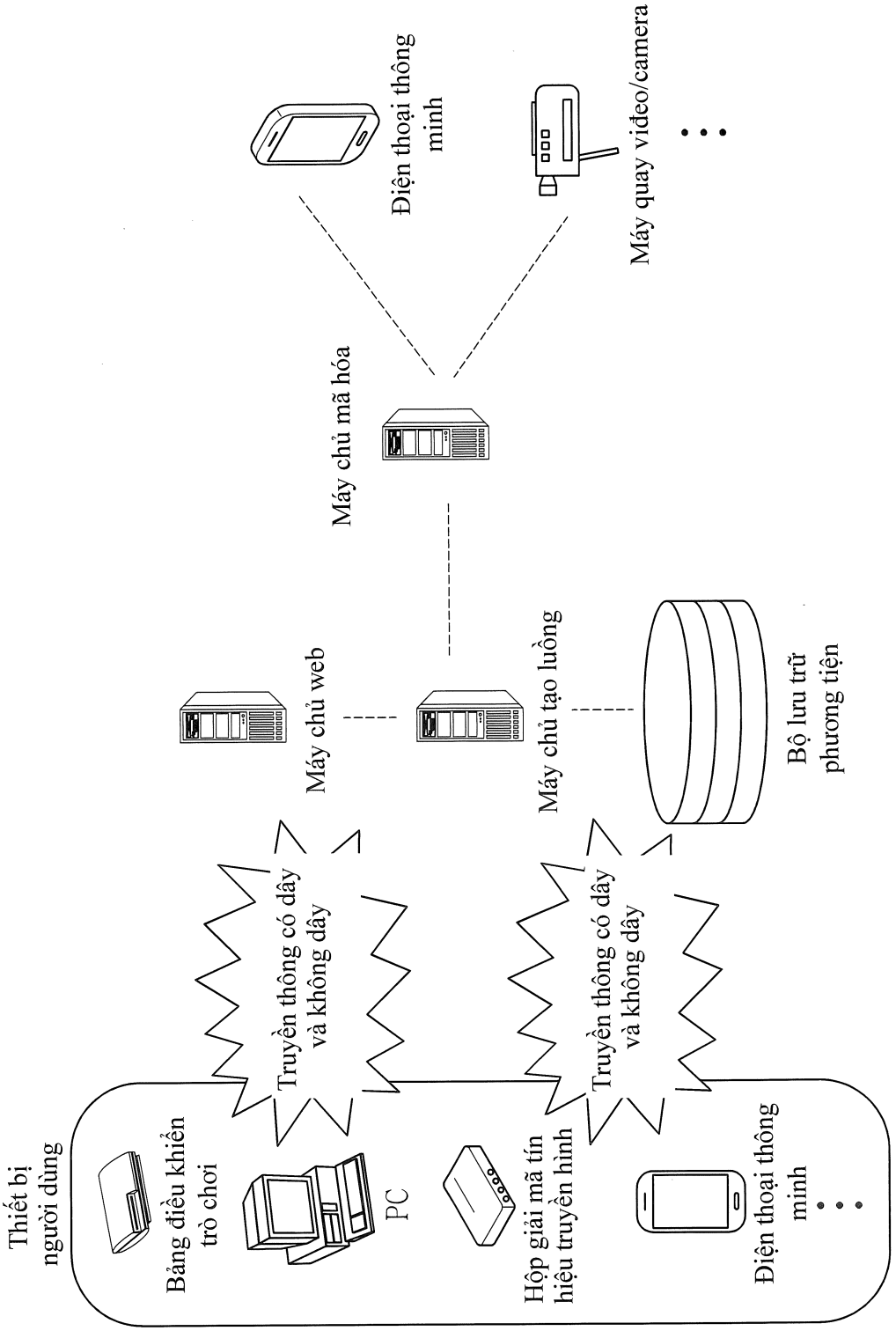

310
330
thiết bị
giải mã
luồng bit
bộ giải mã
entropy
bộ dự đoán
hình ảnh được
xây dựng lại
bộ xử lý
phần dư
bộ cộng
320
340

FIG. 12


Thiết bị người dùng
Bảng điều khiển trò chơi
PC
Hộp giải mã tín hiệu truyền hình
Điện thoại thông minh
Truyền thông có dây và không dây
Truyền thông có dây và không dây
Máy chủ web
Máy chủ tạo luồng
Bộ lưu trữ phương tiện
Máy chủ mã hóa
Điện thoại thông minh
Máy quay video/camera