(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) **1-0046013**

(51)[2014.01] **H04N 19/14**; H04N 19/162; H04N 19/46; H04N 19/172; H04N 19/174; H04N 19/30; H04N 19/107; H04N 19/169

(13) **B**

(21) 1-2021-06239
(22) 11/03/2020
(86) PCT/US2020/022179 11/03/2020
(87) WO 2020/185956 17/09/2020
(30) 62/816,722 11/03/2019 US; 62/871,020 05/07/2019 US
(45) 26/05/2025 446
(43) 27/12/2021 405A
(73) HUAWEI TECHNOLOGIES CO., LTD. (CN)
Huawei Administration Building Bantian, Longgang District Shenzhen, Guangdong 518129, China
(72) WANG, Ye-Kui (US); HENDRY, Fnu (ID); CHEN, Jianle (CN).
(74) Công ty Luật TNHH Phạm và Liên danh (PHAM & ASSOCIATES)

(54) BỘ MÃ HÓA, PHƯƠNG PHÁP GIẢI MÃ VÀ MÃ HÓA

(21) 1- 2021-06239

(57) Sáng chế đề cập đến phương pháp giải mã dòng bit viđeo được mã hóa được triển khai bằng bộ giải mã viđeo. Phương pháp bao gồm xác định rằng chuỗi viđeo được mã hóa (coded sequence viđeo, CVS) của dòng bit viđeo được mã hóa bao gồm đơn vị lớp trừu tượng mạng (network abstraction layer, NAL) của lớp tạo mã viđeo (video coding layer, VCL) có loại đơn vị NAL làm mới giải mã từng bước (gradual decoding refresh, GDR) (GDR_NUT), đơn VCL NAL có GDR_NUT chứa ảnh GDR; khởi tạo giải mã CVS ở ảnh GDR; và tạo ảnh theo CVS như được giải mã. Sáng chế cũng đề cập đến phương pháp tương ứng mã hóa.

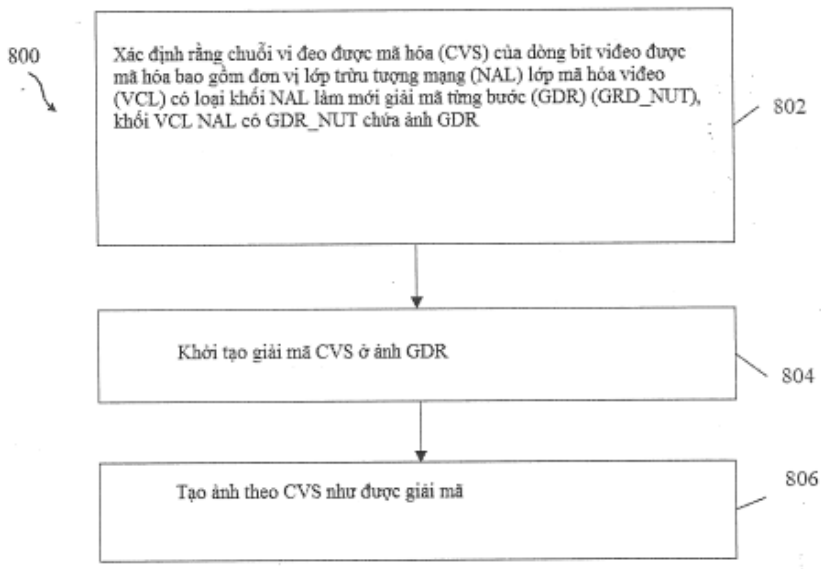

Fig.8

Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập chung đến kỹ thuật hỗ trợ làm mới giải mã từng bước trong kỹ thuật (gradual decoding refresh, GDR) trong kỹ thuật tạo mã viđeo. Cụ thể hơn, sáng chế cho phép làm mới nội bộ từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng các ảnh điểm truy nhập ngẫu nhiên nội ảnh (intra random access point, IRAP).

Tình trạng kỹ thuật của sáng chế

Lượng dữ liệu viđeo cần mô tả viđeo thậm chí rất ngắn có thể là lớn, có thể dẫn đến các khó khăn khi dữ liệu được phát trực tuyến hoặc được truyền qua mạng truyền thông với dung lượng băng thông hữu hạn. Do vậy, dữ liệu viđeo thường được nén trước khi được truyền giữa mạng viễn thông hiện đại. Kích thước của viđeo cũng có thể là vấn đề khi viđeo được lưu trữ trên thiết bị lưu trữ do tài nguyên bộ nhớ có thể bị giới hạn. Các thiết bị nén viđeo thường sử dụng phần mềm và/hoặc phần cứng ở phía nguồn để mã hóa dữ liệu viđeo trước khi truyền hoặc lưu trữ, nhờ đó giảm lượng dữ liệu cần để biểu diễn các ảnh số viđeo. Sau đó dữ liệu được nén được nhận ở phía đích bằng thiết bị giải nén viđeo giải mã dữ liệu viđeo. Với các tài nguyên mạng hữu hạn và các nhu cầu ngày càng tăng của chất lượng viđeo cao hơn, các kỹ thuật nén và giải nén được cải thiện mà cải thiện tỷ lệ nén mà không ảnh hướng đến chất lượng ảnh được mong muốn.

Bản chất kỹ thuật của sáng chế

Khía cạnh thứ nhất đề cập đến phương pháp giải mã dòng bit viđeo được mã hóa được triển khai bằng bộ giải mã viđeo. Phương pháp bao gồm các bước xác định, bằng bộ giải mã viđeo, rằng chuỗi viđeo được mã hóa (coded video sequence, CVS) của dòng bit viđeo được mã hóa bao gồm đơn vị lớp trừu tượng mạng (network abstraction layer, NAL) của lớp tạo mã viđeo (Video Coding

layer, VCL) có loại đơn vị NAL làm mới giải mã từng bước (gradual decoding refresh, GDR) (GDR NAL unit type, GDR_NUT), đơn VCL NAL có GDR_NUT chứa ảnh GDR; khởi tạo, bằng bộ giải mã viđeo, giải mã CVS ở ảnh GDR; và tạo, bằng bộ giải mã viđeo, ảnh theo CVS như được giải mã.

Phương pháp đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được biết là, "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Theo dạng triển khai thứ nhất của phương pháp theo khía cạnh thứ nhất, ảnh GDR là ảnh ban đầu trong CVS.

Theo dạng triển khai thứ hai của phương pháp theo khía cạnh thứ nhất hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ nhất, ảnh GDR là ảnh ban đầu trong chu kỳ GDR.

Theo dạng triển khai thứ ba của phương pháp theo khía cạnh thứ nhất hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ nhất, ảnh GDR có bộ nhận dạng (identifier, ID) thời gian bằng 0.

Theo dạng triển khai thứ tư của phương pháp theo khía cạnh thứ nhất hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ nhất, khối truy nhập chứa đơn VCL NAL có GDR_NUT được chọn là khối truy nhập GDR.

Theo dạng triển khai thứ năm của phương pháp theo khía cạnh thứ nhất hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ nhất, trong đó GDR_NUT chỉ báo cho bộ giải mã viđeo rằng đơn VCL NAL có GDR_NUT chứa ảnh GDR

Khía cạnh thứ hai đề cập đến phương pháp mã hóa dòng bit viđeo được triển khai bằng bộ mã hóa viđeo. Phương pháp bao gồm xác định, bằng bộ mã hóa viđeo, điểm truy nhập ngẫu nhiên cho chuỗi viđeo; mã hóa, bằng bộ mã hóa

viđeo, ảnh GDR thành đơn vị VCL NAL có GDR_NUT ở điểm truy nhập ngẫu nhiên cho chuỗi viđeo; tạo, bằng bộ mã hóa viđeo, dòng bit chứa chuỗi viđeo có ảnh GDR trong đơn VCL NAL có GDR_NUT ở điểm truy nhập ngẫu nhiên; và lưu trữ, bằng bộ mã hóa viđeo, dòng bit để truyền về phía bộ giải mã viđeo.

Phương pháp đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được gọi là "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Theo dạng triển khai thứ nhất của phương pháp theo khía cạnh thứ hai, ảnh GDR là ảnh ban đầu trong CVS.

Theo dạng triển khai thứ hai của phương pháp theo khía cạnh thứ hai hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ hai, ảnh GDR là ảnh ban đầu trong chu kỳ GDR.

Theo dạng triển khai thứ ba của phương pháp theo khía cạnh thứ hai hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ hai, ảnh GDR có ID thời gian bằng 0.

Theo dạng triển khai thứ tư của phương pháp theo khía cạnh thứ hai hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ hai, đơn VCL NAL có GDR_NUT được chọn là khối truy nhập GDR.

Theo dạng triển khai thứ năm của phương pháp theo khía cạnh thứ hai hoặc dạng triển khai đứng trước bất kỳ của khía cạnh thứ hai, trong đó GDR_NUT chỉ báo cho bộ giải mã viđeo rằng đơn VCL NAL có GDR_NUT chứa ảnh GDR

Khía cạnh thứ ba đề cập đến thiết bị giải mã. Thiết bị giải mã bao gồm bộ nhận được tạo cấu hình để nhận dòng bit viđeo được mã hóa; bộ nhớ được ghép nối với bộ nhận, bộ nhớ lưu trữ các lệnh; và bộ xử lý được ghép nối với bộ nhớ,

bộ xử lý được tạo cấu hình để thực thi các lệnh để khiến thiết bị giải mã: xác định rằng CVS của dòng bit viđeo được mã hóa bao gồm đơn vị VCL NAL có GDR_NUT, đơn VCL NAL có GDR_NUT chứa ảnh GDR; khởi tạo giải mã CVS ở ảnh GDR; và tạo ảnh theo CVS như được giải mã.

Thiết bị giải mã đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được biết đến là "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Theo dạng triển khai thứ nhất của thiết bị giải mã theo khía cạnh thứ ba, thiết bị giải mã còn bao gồm bộ phận hiển thị được tạo cấu hình để hiển thị ảnh như được tạo.

Khía cạnh thứ tư đề cập đến thiết bị mã hóa. Thiết bị mã hóa bao gồm bộ nhớ chứa các lệnh; bộ xử lý được ghép nối với bộ nhớ, bộ xử lý được tạo cấu hình để thực hiện các lệnh để khiến thiết bị mã hóa: xác định điểm truy nhập ngẫu nhiên cho chuỗi viđeo; mã hóa ảnh GDR thành đơn vị VCL NAL có GDR_NUT ở điểm truy nhập ngẫu nhiên cho chuỗi viđeo; và tạo dòng bit chứa chuỗi viđeo có ảnh GDR trong đơn VCL NAL có GDR_NUT ở điểm truy nhập ngẫu nhiên; và bộ truyền được ghép nối với bộ xử lý, bộ truyền được tạo cấu hình để truyền dòng bit về phía bộ giải mã viđeo.

Thiết bị mã hóa đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ

mã hóa/bộ giải mã (cũng được biết đến là, "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Theo dạng triển khai thứ nhất của thiết bị mã hóa theo khía cạnh thứ tư, bộ nhớ lưu trữ dòng bit trước bộ truyền truyền dòng bit về phía bộ giải mã viđeo.

Khía cạnh thứ năm đề cập đến thiết bị mã hóa. Thiết bị mã hóa bao gồm bộ nhận được tạo cấu hình để nhận ảnh để mã hóa hoặc để nhận dòng bit để giải mã; bộ truyền được ghép nối với bộ nhận, bộ truyền được tạo cấu hình để truyền dòng bit đến bộ giải mã hoặc truyền ảnh được giải mã đến bộ hiển thị; bộ nhớ được ghép nối với ít nhất một trong bộ nhận hoặc bộ truyền, bộ nhớ được tạo cấu hình để lưu trữ các lệnh; và bộ xử lý được ghép nối với bộ nhớ, bộ xử lý được tạo cấu hình để thực thi các lệnh được lưu trữ trong bộ nhớ để thực hiện phương pháp bất kỳ trong các phương pháp được bộc lộ ở đây.

Thiết bị mã hóa đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được biết đến là, "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Khía cạnh thứ sáu đề cập đến hệ thống. Hệ thống bao gồm bộ mã hóa; và bộ giải mã truyền thông với bộ mã hóa, trong đó bộ mã hóa hoặc bộ giải mã bao gồm thiết bị giải mã, thiết bị mã hóa, hoặc thiết bị mã hóa được bộc lộ ở đây.

Hệ thống đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể

đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được biết đến là "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Khía cạnh thứ bảy đề cập đến phương tiện mã hóa. Phương tiện mã hóa bao gồm phương tiện nhận được tạo cấu hình để nhận ảnh để mã hóa hoặc để nhận dòng bit để giải mã; phương tiện truyền được ghép nối với phương tiện nhận, phương tiện truyền được tạo cấu hình để truyền dòng bit đến phương tiện giải mã hoặc truyền ảnh được giải mã đến phương tiện hiển thị; phương tiện lưu trữ được ghép nối với ít nhất một trong phương tiện nhận hoặc phương tiện truyền, phương tiện lưu trữ được tạo cấu hình để lưu trữ các lệnh; và phương tiện xử lý được ghép nối với phương tiện lưu trữ, phương tiện xử lý được tạo cấu hình để thực thi các lệnh được lưu trữ trong phương tiện lưu trữ để thực hiện phương pháp bất kỳ trong các phương pháp được bộc lộ ở đây.

Phương tiện mã hóa đề cập đến các kỹ thuật cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được biết đến là "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Mô tả vắn tắt các hình vẽ

Để hiểu rõ hơn sáng chế, tham khảo phần mô tả vắn tắt sau, cùng với các hình vẽ đi kèm và phần mô tả chi tiết, trong đó các số chỉ dẫn giống nhau biểu diễn các phần giống nhau.

Fig.1 là sơ đồ khối minh họa hệ thống mã hóa ví dụ có thể sử dụng các kỹ thuật GDR;

Fig.2 là sơ đồ khối minh họa bộ mã hóa viđeo ví dụ có thể triển khai các kỹ thuật GDR;

Fig.3 là sơ đồ khối minh họa ví dụ của bộ giải mã viđeo có thể triển khai các kỹ thuật GDR;

Fig.4 là hình vẽ minh họa mối quan hệ giữa ảnh IRAP so với ảnh dẫn đầu và các ảnh theo sau theo thứ tự giải mã và thứ tự hiển thị;

Fig.5 là hình vẽ minh họa kỹ thuật GDR;

Fig.6 là sơ đồ minh họa tìm kiếm chuyển động không mong muốn;

Fig.7 là hình vẽ minh họa dòng bit mô tả kỹ thuật GDR theo phương án thực hiện sáng chế;

Fig.8 là phương pháp giải mã dòng bit viđeo được mã hóa theo phương án thực hiện;

Fig.9 là phương pháp mã hóa dòng bit viđeo theo phương án thực hiện;

Fig.10 là sơ đồ của thiết bị tạo mã viđeo; và

Fig.11 là sơ đồ của phương tiện mã hóa theo phương án thực hiện.

Mô tả chi tiết sáng chế

Fig.1 là sơ đồ khối minh họa hệ thống mã hóa ví dụ 10 có thể sử dụng các kỹ thuật tạo mã viđeo như được mô tả ở đây. Như được thể hiện trên Fig.1, hệ thống mã hóa 10 bao gồm thiết bị nguồn 12 tạo dữ liệu viđeo được mã hóa cần được giải mã ở thời gian sau này bởi thiết bị đích 14. Cụ thể là, thiết bị nguồn 12 có thể cấp dữ liệu viđeo cho thiết bị đích 14 qua phương tiện máy tính đọc được 16. Thiết bị nguồn 12 và thiết bị đích 14 có thể bao gồm thiết bị bất kỳ trong các loại thiết bị, bao gồm máy tính để bàn, máy tính notebook (chẳng hạn, xách tay), máy tính bảng, set-top box, điện thoại cầm tay, chẳng hạn, điện thoại "thông minh", máy tính bảng "thông minh", TV, camera, thiết bị hiển thị, máy phát media số, bảng điều khiển trò chơi viđeo, thiết bị viđeo streaming, hoặc tương tự. Trong một số trường hợp, thiết bị nguồn 12 và thiết bị đích 14 có thể được trang bị để truyền thông không dây.

Thiết bị đích 14 có thể nhận dữ liệu viđeo được mã hóa cần được giải mã qua phương tiện máy tính đọc được 16. Phương tiện máy tính đọc được 16 có thể bao gồm loại phương tiện hoặc thiết bị có thể di chuyển dữ liệu viđeo được mã hóa từ thiết bị nguồn 12 đến thiết bị đích 14. Trong một ví dụ, phương tiện máy tính đọc được 16 có thể bao gồm phương tiện truyền thông để thiết bị nguồn 12 có thể truyền dữ liệu viđeo được mã hóa trực tiếp đến thiết bị đích 14 theo thời gian thực. Dữ liệu viđeo được mã hóa có thể được điều biến theo chuẩn truyền thông, chẳng hạn giao thức truyền thông không dây, và được truyền đến thiết bị đích 14. Phương tiện truyền thông có thể bao gồm phương tiện truyền thông không dây hoặc hữu tuyến, chẳng hạn, phổ tần số vô tuyến (radio frequency, RF) hoặc một hoặc nhiều đường truyền vật lý. Phương tiện truyền thông có thể tạo thành một phần của mạng gói, chẳng hạn, mạng cục bộ (local area network, LAN), mạng diện rộng (wide-area network, WAN), hoặc mạng toàn cầu, chẳng hạn, mạng Internet. Phương tiện truyền thông có thể bao gồm bộ định tuyến, bộ chuyển mạch, trạm cơ sở, hoặc thiết bị khác bất kỳ có thể hữu ích để tiện truyền thông từ thiết bị nguồn 12 đến thiết bị đích 14.

Trong một số ví dụ, dữ liệu được mã hóa có thể được xuất ra từ giao diện đầu ra 22 đến bộ phận lưu trữ. Tương tự, dữ liệu được mã hóa có thể được truy nhập từ bộ phận lưu trữ bằng giao diện đầu vào. Bộ phận lưu trữ có thể bao gồm một trong các phương tiện lưu trữ dữ liệu được truy nhập cục bộ hoặc phân tán, chẳng hạn, ổ cứng, đĩa Blu-ray, đĩa viđeo số (digital video disk, DVD), đĩa CD chỉ đọc (Compact Disc Read-Only Memory, CD-ROM), bộ nhớ nhanh, bộ nhớ khả biến hoặc bất biến, hoặc phương tiện lưu trữ số bảng thích hợp khác để lưu trữ dữ liệu viđeo được mã hóa. Trong ví dụ khác, bộ phận lưu trữ có thể tương ứng với mảy chủ tệp tin hoặc bộ phận lưu trữ trung gian khác mà có thể lưu trữ viđeo được mã hóa được tạo bằng thiết bị nguồn 12. Thiết bị đích 14 có thể truy nhập dữ liệu viđeo được lưu trữ từ bộ phận lưu trữ qua streaming (phát trực tuyến) hoặc tải xuống. Máy chủ tệp tin có thể là loại máy chủ bất kỳ có thể lưu trữ dữ liệu viđeo được mã hóa và truyền dữ liệu viđeo được mã hóa đó đến thiết bị đích 14. Các máy chủ tệp tin ví dụ bao gồm máy chủ web (chẳng hạn, đối với website), máy chủ giao thức truyền tệp tin (file transfer protocol, FTP), các thiết bị lưu trữ gắn

vào mạng (network attached storage, NAS), hoặc ổ đĩa cục bộ. Thiết bị đích 14 có thể truy nhập dữ liệu viđeo được mã hóa qua kết nối dữ liệu chuẩn bất kỳ, bao gồm kết nối Internet. Kết nối có thể bao gồm kênh không dây (chẳng hạn, kết nối Wi-Fi), kết nối hữu tuyến (chẳng hạn, đường thuê bao số (digital subscriber line, DSL), modem cáp, v.v.), hoặc kết hợp của hai kết nối thích hợp để truy nhập dữ liệu viđeo được mã hóa được lưu trữ trên mảy chủ tệp tin. Phiên truyền dữ liệu viđeo được mã hóa từ bộ phận lưu trữ có thể là phiên truyền streaming, phiên truyền tải xuống, hoặc kết hợp của nó.

Các kỹ thuật của sáng chế không nhất thiết bị giới hạn ở các ứng dụng không dây hoặc các thiết lập. Các kỹ thuật có thể được áp dụng cho tạo mã viđeo với sự hỗ trợ của ứng dụng bất kỳ trong các ứng dụng đa phương tiện, chẳng hạn, phát sóng truyền hình trên không trung, truyền hình cáp, truyền hình vệ tinh, truyền viđeo streaming qua Internet, chẳng hạn, truyền thích ứng động trên HTTP (dynamic adaptive streaming over HTTP, DASH), viđeo số được mã hóa trên phương tiện lưu trữ dữ liệu, giải mã viđeo số được lưu trữ trên phương tiện lưu trữ dữ liệu, hoặc các ứng dụng khác. Trong một số ví dụ, hệ thống mã hóa 10 có thể được tạo cấu hình để hỗ trợ truyền viđeo một chiều hoặc hai chiều để hỗ trợ các ứng dụng, chẳng hạn, viđeo streaming, phát lại viđeo, phát sóng viđeo, và/hoặc điện thoại viđeo.

Ở ví dụ trên Fig.1, thiết bị nguồn 12 bao gồm nguồn viđeo 18, bộ mã hóa viđeo 20, và giao diện đầu ra 22. Thiết bị đích 14 bao gồm giao diện đầu vào 28, bộ giải mã viđeo 30, và bộ phận hiển thị 32. Theo sáng chế, bộ mã hóa viđeo 20 của thiết bị nguồn 12 và/hoặc bộ giải mã viđeo 30 của thiết bị đích 14 có thể được tạo cấu hình để áp dụng các kỹ thuật cho tạo mã viđeo. Trong các ví dụ khác, thiết bị nguồn và thiết bị đích có thể bao gồm các thành phần hoặc các thiết bị khác. Chẳng hạn, thiết bị nguồn 12 có thể nhận dữ liệu viđeo từ nguồn viđeo bên ngoài, chẳng hạn, camera bên ngoài. Tương tự, thiết bị đích 14 có thể gaio tiếp với bộ phận hiển thị ngoài, thay vì bao gồm bộ phận hiển thị được tích hợp.

Hệ thống mã hóa 10 được minh họa trên Fig.1 chỉ là một ví dụ. Các kỹ thuật tạo mã viđeo có thể được thực hiện bằng thiết bị mã hóa viđeo và/hoặc giải mã viđeo số bất kỳ. Mặc dù các kỹ thuật của sáng chế nói chung được thực hiện bằng

thiết bị tạo mã viđeo, các kỹ thuật có thể cũng được thực hiện bằng bộ mã hóa /bộ giải mã viđeo, thường được gọi là "CODEC." Ngoài ra, các kỹ thuật của sáng chế cũng có thể được thực hiện bằng bộ tiền xử lý viđeo. Bộ mã hóa viđeo và/hoặc bộ giải mã có thể là khối xử lý đồ họa (graphics processing unit, GPU) hoặc thiết bị tương tự.

Thiết bị nguồn 12 và thiết bị đích 14 chỉ là các ví dụ của các thiết bị mã hóa này trong đó thiết bị nguồn 12 tạo dữ liệu viđeo được tạo mã để truyền đến thiết bị đích 14. Trong một số ví dụ, thiết bị nguồn 12 và thiết bị đích 14 có thể hoạt động theo cách gần như đối xứng sao cho mỗi thiết bị trong các thiết bị nguồn và thiết bị đích 12, 14 bao gồm các thành phần mã hóa và giải mã viđeo. Do vậy, hệ thống mã hóa 10 có thể hỗ trợ truyền viđeo một chiều hoặc hai chiều giữa các thiết bị viđeo 12, 14, chẳng hạn, cho viđeo streaming, phát lại viđeo, phát sóng viđeo, hoặc điện thoại viđeo.

Nguồn viđeo 18 của thiết bị nguồn 12 có thể bao gồm thiết bị chụp ảnh viđeo, chẳng hạn, viđeo camera, bộ lưu trữ viđeo chứa viđeo đã được ghi lại trước đó, và/hoặc giao diện cấp viđeo để nhận viđeo từ nhà cung cấp nội dung viđeo. Theo tùy chọn khác, nguồn viđeo 18 có thể tạo dữ liệu dựa trên đồ họa máy tính như là viđeo nguồn, hoặc kết hợp của viđeo trực tiếp, viđeo được lưu trữ, và viđeo được tạo bằng máy tính.

Trong một số trường hợp, khi nguồn viđeo 18 là viđeo camera, thiết bị nguồn 12 và thiết bị đích 14 có thể tạo các điện thoại camera hoặc điện thoại viđeo. Như nêu trên, tuy nhiên, các kỹ thuật được mô tả theo sáng chế có thể được áp dụng cho tạo mã viđeo nói chung, và có thể được áp dụng cho các ứng dụng không dây và/hoặc hữu tuyến. Trong mỗi trường hợp, viđeo được ghi nhận, được ghi nhận trước đó, hoặc được tạo bằng máy tính có thể được mã hóa bằng bộ mã hóa viđeo 20. Sau đó, thông tin viđeo được mã hóa có thể được xuất ra bởi giao diện đầu ra 22 trên phương tiện máy tính đọc được 16.

Phương tiện máy tính đọc được 16 có thể bao gồm phương tiện khả biến, chẳng hạn truyền mạng hữu tuyến hoặc phát không dây, hoặc phương tiện lưu trữ (tức là, phương tiện lưu trữ bất biến), chẳng hạn đĩa cứng, ổ nhớ nhanh, CD, đĩa viđeo số, đĩa Blu-ray, hoặc các phương tiện máy tính đọc được khác. Trong

một số ví dụ, máy chủ mạng (không được thể hiện trên hình vẽ) có thể nhận dữ liệu viđeo được mã hóa từ thiết bị nguồn 12 và cấp dữ liệu viđeo được mã hóa cho thiết bị đích 14, chẳng hạn, qua truyền mạng. Tương tự, thiết bị tính toán của bộ phận sản xuất đa phương tiện, chẳng hạn, bộ phận dán nhãn đĩa, có thể nhận dữ liệu viđeo được mã hóa từ thiết bị nguồn 12 và tạo đĩa chứa dữ liệu viđeo được mã hóa. Do vậy, phương tiện máy tính đọc được 16 có thể được hiểu là bao gồm một hoặc nhiều các phương tiện máy tính đọc được ở các dạng khác nhau, trong các ví dụ khác nhau.

Giao diện đầu vào 28 của thiết bị đích 14 nhận thông tin từ phương tiện máy tính đọc được 16. Thông tin của phương tiện máy tính đọc được 16 có thể bao gồm thông tin cú pháp được định nghĩa bằng bộ mã hóa viđeo 20, cũng được sử dụng bằng bộ giải mã viđeo 30, mà bao gồm các phần tử cú pháp mà mô tả các đặc tính và/hoặc xử lý của các khối và các đơn vị được mã hóa khác, chẳng hạn, nhóm ảnh (group of picture, GOP). Bộ phận hiển thị 32 hiển thị dữ liệu viđeo được giải mã đến người dùng, và có thể bao gồm thiết bị bất kỳ trong các bộ phận hiển thị, chẳng hạn, ống tia catôt (cathode ray tube, CRT), màn hình tinh thể lỏng (liquid crystal display, LCD), màn hình plasma, màn hình điôt phát quang hữu cơ (organic light emitting diode, OLED), hoặc loại khác của bộ phận hiển thị.

Bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 có thể hoạt động theo chuẩn tạo mã viđeo, chẳng hạn chuẩn tạo mã viđeo hiệu suất cao (High Efficiency Video Coding, HEVC) hiện đang được phát triển, và có thể tuân theo mô hình kiểm tra HEVC (HEVC Test Model, HM). Theo cách khác, bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 có thể hoạt động theo chuẩn công nghiệp hoặc độc quyền khác, chẳng hạn chuẩn H.264 ngành chuẩn hóa viễn thông của liên đoàn viễn thông quốc tế (International Telecommunications Union Telecommunication, ITU-T) H.264, được gọi theo cách khác là chuẩn của nhóm chuyên gia ảnh động (Moving Picture Expert Group, MPEG)-4, Part 10, chuẩn tạo mã viđeo cải tiến (Advanced Video Coding, AVC), H.265/HEVC, hoặc các mở rộng của các chuẩn này. Tuy nhiên, các kỹ thuật của sáng chế không bị giới hạn ở chuẩn mã hóa cụ thể bất kỳ. Các ví dụ khác của các chuẩn tạo mã viđeo bao gồm MPEG-2 và ITU-T H.263.

Mặc dù không được thể hiện trên Fig.1, theo một số khía cạnh, mỗi bộ trong bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 có thể được tích hợp với bộ mã hóa audio và bộ giải mã audio, và có thể bao gồm các khối ghép kênh – phân kênh (multiplexer-demultiplexer, MUX-DEMUX) thích hợp, hoặc phần cứng và phần mềm khác, để xử lý mã hóa cả audio lẫn video trong dòng dữ liệu chung hoặc các dòng dữ liệu riêng rẽ. Nếu có thể áp dụng, các khối MUX-DEMUX có thể tuân theo giao thức ghép kênh ITU H.223, hoặc các giao thức khác, chẳng hạn, giao thức lượng dữ liệu người dùng (user datagram protocol, UDP).

Mỗi bộ phận trong bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 có thể được triển khai dưới dạng thiết bị bất kỳ trong hệ mạch mã hóa thích hợp, chẳng hạn một hoặc nhiều bộ vi xử lý, bộ xử lý tín hiệu số (digital signal processor, DSP), mạch tích hợp ứng dụng cụ thể (application specific integrated circuit, ASIC), mảng cổng dạng trường lập trình được (field programmable gate array, FPGA), lôgic rời rạc, phần mềm, phần cứng, firmware hoặc các tổ hợp bất kỳ của nó. Khi các kỹ thuật được triển khai một phần trong phần mềm, thiết bị có thể lưu trữ các lệnh cho phần mềm trong phương tiện máy tính đọc được bất biến, thích hợp và thực thi các lệnh trong phần cứng nhờ sử dụng một hoặc nhiều bộ xử lý để thực hiện các kỹ thuật của sáng chế. Mỗi bộ phận trong bộ mã hóa viđeo 20 và bộ giải mã viđeo 30 có thể được bao gồm trong một hoặc nhiều bộ mã hóa hoặc bộ giải mã, mỗi bộ phận có thể được tích hợp dưới dạng một phần của bộ mã hóa/bộ giải mã (CODEC) được kết hợp trong thiết bị tương ứng. Thiết bị bao gồm bộ mã hóa viđeo 20 và/hoặc bộ giải mã viđeo 30 có thể bao gồm mạch tích hợp, bộ vi xử lý, và/hoặc bộ phận truyền thông không dây, chẳng hạn, điện thoại tế bào.

Fig.2 là sơ đồ khối minh họa ví dụ của bộ mã hóa viđeo 20 có thể triển khai các kỹ thuật tạo mã viđeo. Bộ mã hóa viđeo 20 có thể thực hiện tạo mã trong và ngoài các khối viđeo trong các lát viđeo. Kỹ thuật tạo mã trong dựa vào dự báo không gian để giảm hoặc loại bỏ dư thừa không gian trong viđeo trong khung viđeo hoặc ảnh cụ thể. Tạo mã ngoài dựa vào dự báo thời gian để giảm hoặc loại bỏ dư thừa thời gian trong viđeo trong các khung hoặc ảnh lân cận của chuỗi viđeo. Chế độ nội vùng (chế độ I) có thể đề cập đến chế độ bất kỳ trong vài chế độ mã hóa dựa trên không gian. Các chế độ liên vùng, chẳng hạn, dự báo đơn

hướng (hay còn gọi là, dự báo đơn) (chế độ P) hoặc dự báo hai hướng (hay còn gọi là dự báo kép) (chế độ B), có thể đề cập đến chế độ bất kỳ trong vài chế độ mã hóa dựa trên thời gian.

Như được thể hiện trên Fig.2, bộ mã hóa viđeo 20 nhận khối viđeo hiện tại trong khung viđeo cần được mã hóa. Ở ví dụ trên Fig.2, bộ mã hóa viđeo 20 bao gồm khối chọn chế độ 40, bộ nhớ khung tham chiếu 64, bộ cộng 50, khối xử lý biến đổi 52, khối lượng tử hóa 54, và khối mã hóa entropy 56. Sau đó, khối chọn chế độ 40 bao gồm khối bù chuyển động 44, khối ước tính chế độ 42, khối dự báo trong (cũng được biết là dự báo trong) 46, và khối phân vùng 48. Để tái tạo khối viđeo, bộ mã hóa viđeo 20 cũng bao gồm khối lượng tử hóa ngược 58, khối biến đổi ngược 60, và bộ cộng 62. Bộ lọc tách khối (deblocking filter, DBF) (không được thể hiện trên Fig.2) có thể cũng được bao gồm để lọc các đường biên khối để loại bỏ các giả tượng khối từ viđeo được tái tạo. Nếu muốn, DBF sẽ thường lọc ra đầu ra của bộ cộng 62. Các bộ lọc bổ sung (trong vòng hoặc sau vòng) cũng có thể được sử dụng bên cạnh DBF. Các bộ lọc này không được thể hiện trên hình vẽ để ngắn gọn, nhưng nếu cần, có thể lọc đầu ra của bộ cộng 50 (dưới dạng bộ lọc trong vòng).

Trong quá trình mã hóa, bộ mã hóa viđeo 20 nhận khung hoặc lát viđeo cần được tạo mã. Khung hoặc lát có thể được phân chia thành nhiều khối viđeo. Khối ước tính chế độ 42 và khối bù chuyển động 44 thực hiện tạo mã dự báo ngoài khối viđeo được nhận so với một hoặc nhiều khối trong một hoặc nhiều khung tham chiếu để dự báo thời gian. Theo cách khác, khối dự báo trong 46 có thể thực hiện tạo mã dự báo trong khối viđeo được nhận so với một hoặc nhiều các khối lân cận trong khung hoặc lát giống như khối cần được tạo mã để dự báo không gian. Bộ mã hóa viđeo 20 có thể thực hiện nhiều bước tạo mã, chẳng hạn, để chọn chế độ tạo mã thích hợp cho mỗi khối của dữ liệu viđeo.

Ngoài ra, khối phân vùng 48 có thể phân vùng các khối dữ liệu viđeo thành các khối phụ, dựa trên đánh giá các phương án phân vùng trước đó trong các bước tạo mã trước đó. Chẳng hạn, khối phân vùng 48 có thể ban đầu phân vùng khung hoặc lát thành các đơn vị mã lớn nhất (largest coding unit, LCU), và phân vùng mỗi LCU trong các LCU thành các đơn vị mã phụ (sub-CUs) dựa trên phân

tích méo dạng tốc độ (chẳng hạn, tối ưu hóa méo dạng tốc độ). Khối chọn chế độ 40 có thể còn tạo cấu trúc dữ liệu cây tứ phân chỉ báo phân vùng LCU thành các CU phụ. Các CU nút lá của cây tứ phân có thể bao gồm một hoặc nhiều đơn vị dự báo (prediction unit, PU) và một hoặc nhiều đơn vị biến đổi (transform unit, TU).

Sáng chế sử dụng cụm từ "khối" để đề cập đến một trong CU, PU, hoặc TU, trong ngữ cảnh của HEVC, hoặc các cấu trúc dữ liệu tương tự trong ngữ cảnh của các chuẩn khác (chẳng hạn, các khối lớn và các khối phụ của nó trong H.264/AVC). CU bao gồm nút mã, PU, và TU được liên kết với nút mã. Kích thước của CU tương ứng với kích thước của nút mã và có hình vuông. Kích thước của CU có thể nằm trong khoảng từ 8×8 pixel đến kích thước của khối cây có tối đa 64×64 pixel hoặc lớn hơn. Mỗi CU có thể chứa một hoặc nhiều PU và một hoặc nhiều TU. Dữ liệu cú pháp được liên kết với CU có thể mô tả, chẳng hạn, phân vùng CU thành một hoặc nhiều PU. Các chế độ phân vùng có thể khác nhau giữa liệu CU là chế độ bỏ qua hoặc trực tiếp được mã hóa, chế độ dự báo trong được mã hóa, hoặc chế độ dự báo ngoài (cũng được biết đến là dự báo ngoài) được mã hóa. Các PU có thể được phân vùng là hình không vuông. Dữ liệu cú pháp được liên kết với CU cũng có thể mô tả, chẳng hạn, phân vùng CU thành một hoặc nhiều TU theo cây tứ phân. TU có thể là hình vuông hoặc không phải hình vuông (chẳng hạn, hình chữ nhật).

Khối chọn chế độ 40 có thể lựa chọn một trong các chế độ tạo mã, trong hoặc ngoài, chẳng hạn, dựa trên các kết quả sai số, và cấp khối được mã hóa trong hoặc ngoài cho bộ cộng 50 để tạo dữ liệu khối dư và cho bộ cộng 62 để tạo khối được mã hóa để sử dụng làm khung tham chiếu. Khối chọn chế độ 40 cũng cấp các phần tử cú pháp, chẳng hạn các vectơ chuyển động (motion vector, MV), các bộ chỉ báo chế độ nội vùng, thông tin phân vùng, và thông tin cú pháp khác, cho khối mã hóa entropy 56.

Khối ước tính chế độ 42 và khối bù chuyển động 44 có thể được tích hợp chặt chẽ, mà được minh họa một cách riêng rẽ về mặt khái niệm. Ước tính chuyển động, được thực hiện bằng khối ước tính chế độ 42, là quá trình tạo các MV, mà ước tính chuyển động cho các khối viđeo. MV, chẳng hạn, có thể chỉ báo dịch

chuyển của PU của khối viđeo trong khung viđeo hoặc ảnh hiện tại so với khối dự báo trong khung tham chiếu (hoặc khối được tạo mã khác) so với khối hiện tại đang được tạo mã trong khung hiện tại (hoặc khối được tạo mã khác). Khối dự báo là khối được tìm ra để so khớp chặt chẽ với khối cần được tạo mã, liên quan đến hiệu số pixel, mà có thể được xác định bằng tổng hiệu số tuyệt đối (sum of absolute difference, SAD), tổng hiệu số bình phương (sum of square difference, SSD), hoặc các phép tính hiệu số khác. Trong một số ví dụ, bộ mã hóa viđeo 20 có thể tính toán các giá trị đối với các vị trí pixel phụ số nguyên của các ảnh tham chiếu được lưu trữ trong bộ nhớ khung tham chiếu 64. Chẳng hạn, bộ mã hóa viđeo 20 có thể nội suy các giá trị của các vị trí pixel 1/4, các vị trí pixel 1/8, hoặc các vị trí pixel phân số khác của ảnh tham chiếu. Do vậy, khối ước tính chế độ 42 có thể thực hiện tìm kiếm chuyển động so với các vị trí pixel hoàn chỉnh và các vị trí pixel phân số và MV đầu ra có độ chính xác pixel phân số.

Khối ước tính chế độ 42 tính toán MV cho PU của khối viđeo trong lát được tạo mã ngoài bằng cách so sánh vị trí của PU với vị trí của khối dự báo của ảnh tham chiếu. Ảnh tham chiếu có thể được chọn từ danh sách ảnh tham chiếu thứ nhất (Danh sách 0) hoặc danh sách ảnh tham chiếu thứ hai (Danh sách 1), mỗi danh sách trong đó nhận diện một hoặc nhiều các ảnh tham chiếu được lưu trữ trong bộ nhớ khung tham chiếu 64. Khối ước tính chế độ 42 gửi MV được tính toán đến khối mã hóa entropy 56 và khối bù chuyển động 44.

Kỹ thuật bù chuyển động, được thực hiện bằng khối bù chuyển động 44, có thể bao gồm nạp hoặc tạo khối dự báo dựa trên MV được xác định bằng khối ước tính chế độ 42. Khối ước tính chế độ 42 và khối bù chuyển động 44 lại có thể được tích hợp về mặt chức năng, trong một số ví dụ. Khi nhận MV đối với PU của khối viđeo hiện tại, khối bù chuyển động 44 có thể định vị khối dự báo mà MV trỏ đến ở một trong các danh sách ảnh tham chiếu. Bộ cộng 50 tạo khối viđeo dư bằng cách lấy các giá trị pixel của khối viđeo hiện tại đang được tạo mã trừ đi các giá trị pixel của khối dự báo, tạo các giá trị hiệu số pixel, như được đề cập dưới đây. Nói chung, khối ước tính chế độ 42 thực hiện ước tính chuyển động so với các thành phần độ sáng, và khối bù chuyển động 44 sử dụng các MV được

tính toán dựa trên các thành phần độ sáng cho cả thành phần sắc độ lẫn thành phần độ sáng. Khối chọn chế độ 40 có thể cũng tạo các phần tử cú pháp được liên kết với các khối viđeo và lát viđeo để sử dụng bằng bộ giải mã viđeo 30 khi giải mã các khối viđeo của lát viđeo.

Khối dự báo trong 46 có thể dự báo trong khối hiện tại, như là tùy chọn với dự báo ngoài được thực hiện bằng khối ước tính chế độ 42 và khối bù chuyển động 44, như được mô tả trên đây. Cụ thể là, khối dự báo trong 46 có thể xác định chế độ dự báo trong để sử dụng để mã hóa khối hiện tại. Trong một số ví dụ, khối dự báo trong 46 có thể mã hóa khối hiện tại nhờ sử dụng các chế độ dự báo trong khác nhau, chẳng hạn, trong các bước mã hóa riêng rẽ, và khối dự báo trong 46 (hoặc khối chọn chế độ 40, trong một số ví dụ) có thể lựa chọn chế độ dự báo trong thích hợp để sử dụng từ các chế độ được kiểm tra.

Chẳng hạn, khối dự báo trong 46 có thể tính toán các giá trị méo dạng tốc độ nhờ sử dụng phân tích méo dạng tốc độ đối với các chế độ dự báo trong được kiểm tra khác nhau, và lựa chọn chế độ dự báo trong có các đặc tính méo dạng tốc độ tốt nhất trong số các chế độ được kiểm tra. Phân tích méo dạng tốc độ nói chung xác định lượng méo dạng (hoặc sai số) giữa khối được mã hóa và khối ban đầu, khối không được mã hóa đã được mã hóa để tạo khối được mã hóa, cũng như tốc độ bit (tức là, số lượng bit) được sử dụng để tạo khối được mã hóa. Khối dự báo trong 46 có thể tính toán các tỷ lệ từ các méo dạng và các tốc độ cho các khối được mã hóa khác nhau để xác định chế độ dự báo trong nào biểu diễn giá trị méo dạng tốc độ tốt nhất cho khối.

Ngoài ra, khối dự báo trong 46 có thể được tạo cấu hình để mã hóa các khối chiều sâu của bản đồ chiều sâu sử dụng chế độ tạo mô hình chiều sâu (depth modeling mode, DMM). Khối chọn chế độ 40 có thể xác định liệu chế độ DMM khả dụng tạo các kết quả mã hóa tốt hơn chế độ dự báo trong và các chế độ DMM khác, chẳng hạn, sử dụng tối ưu hóa méo dạng tốc độ (rate distortion optimization, RDO). Dữ liệu đối với ảnh kết cấu tương ứng với bản đồ chiều sâu có thể được lưu trữ trong bộ nhớ khung tham chiếu 64. Khối ước tính chế độ 42 và khối bù chuyển động 44 có thể cũng được tạo cấu hình để dự báo ngoài các khối chiều sâu của bản đồ chiều sâu.

Sau khi lựa chọn chế độ dự báo trong cho khối (chẳng hạn, chế độ dự báo trong đã biết hoặc một trong các chế độ DMM), khối dự báo trong 46 có thể cấp thông tin chỉ báo chế độ dự báo trong được chọn cho khối cho khối mã hóa entropy 56. Khối mã hóa entropy 56 có thể mã hóa thông tin chỉ báo chế độ dự báo trong được chọn. Bộ mã hóa viđeo 20 có thể bao gồm trong dữ liệu cấu hình dòng bit được truyền, mà có thể bao gồm các bảng chỉ số chế độ dự báo trong và các bảng chỉ số chế độ dự báo trong được chỉnh sửa (cũng được gọi là các bảng ánh xạ từ mã), các định nghĩa mã hóa các ngữ cảnh cho các khối khác nhau, và các chỉ báo về chế độ dự báo trong có xác suất lớn nhất, bảng chỉ số chế độ dự báo trong, và bảng chỉ số chế độ dự báo trong được chỉnh sửa để sử dụng cho mỗi ngữ cảnh trong các ngữ cảnh.

Bộ mã hóa viđeo 20 tạo khối viđeo dư bằng cách lấy khối viđeo ban đầu đang được tạo mã trừ đi dữ liệu dự báo từ khối chọn chế độ 40. Bộ cộng 50 là thành phần hoặc các thành phần thực hiện phép trừ này.

Khối xử lý biến đổi 52 áp dụng phép biến đổi, chẳng hạn, biến đổi côsin rời rạc (discrete cosine transform, DCT) hoặc biến đổi tương tự về mặt khái niệm, cho khối dư, tạo khối viđeo bao gồm các giá trị hệ số biến đổi dư. Khối xử lý biến đổi 52 có thể thực hiện các phép biến đổi khác vốn tương tự về mặt khái niệm với DCT. Các phép biến đổi Wavelet, các phép biến đổi số nguyên, các phép biến đổi băng phụ hoặc các loại biến đổi khác cũng có thể được sử dụng.

Khối xử lý biến đổi 52 áp dụng phép biến đổi cho khối dư, tạo khối của các hệ số biến đổi dư. Phép biến đổi có thể biến đổi thông tin dư từ miền giá trị pixel sang miền biến đổi, chẳng hạn miền tần số. Khối xử lý biến đổi 52 có thể gửi các hệ số biến đổi thu được đến khối lượng tử hóa 54. Khối lượng tử hóa 54 lượng tử hóa các hệ số biến đổi để giảm tiếp tốc độ bit. Quá trình lượng tử hóa có thể giảm chiều sâu bit được liên kết với một số hoặc tất cả các hệ số. Mức độ lượng tử hóa có thể được thay đổi bằng cách điều chỉnh tham số lượng tử (quantization parameter, QP). Sau đó, trong một số ví dụ, khối lượng tử hóa 54 có thể thực hiện quét ma trận bao gồm các hệ số biến đổi được lượng tử hóa. Theo cách khác, khối mã hóa entropy 56 có thể thực hiện quét.

Tiếp theo lượng tử hóa, khối mã hóa entropy 56 mã hóa entropy các hệ số

biến đổi được lượng tử hóa. Chẳng hạn, khối mã hóa entropy 56 có thể thực hiện tạo mã có chiều biến thiên thích ứng ngữ cảnh (context adaptive variable length coding, CAVLC), tạo mã số học nhị phân thích ứng ngữ cảnh (context adaptive binary arithmetic coding, CABAC), CABAC dựa trên cú pháp (syntax-based context-adaptive binary arithmetic coding, SBAC), mã hóa entropy phân vùng khoảng xác suất (probability interval partitioning entropy, PIPE) hoặc kỹ thuật mã hóa entropy khác. Trong trường hợp mã hóa entropy dựa trên ngữ cảnh, ngữ cảnh có thể dựa trên các khối lân cận. Tiếp theo mã hóa entropy bởi khối mã hóa entropy 56, dòng bit được mã hóa có thể được truyền đến thiết bị khác (chẳng hạn, bộ giải mã viđeo 30) hoặc được lưu trữ để truyền hoặc truy xuất sau này.

Khối lượng tử hóa ngược 58 và khối biến đổi ngược 60 lần lượt áp dụng phép lượng tử hóa ngược và biến đổi ngược, để tạo khối dư trong miền pixel, chẳng hạn, để sử dụng sau này làm khối tham chiếu. Khối bù chuyển động 44 có thể tính toán khối tham chiếu bằng cách bổ sung khối dư vào khối dự báo của một trong các khung của bộ nhớ khung tham chiếu 64. Khối bù chuyển động 44 có thể cũng áp dụng một hoặc nhiều bộ lọc nội suy cho khối dư được tái tạo để tính toán các giá trị pixel con số nguyên để sử dụng trong ước tính chuyển động. Bộ cộng 62 bổ sung khối dư được tái tạo vào khối dự báo bù chuyển động được tạo bằng khối bù chuyển động 44 để tạo khối viđeo được tái tạo để lưu trữ trong bộ nhớ khung tham chiếu 64. Khối viđeo được tái tạo có thể được sử dụng làm khối ước tính chế độ 42 và khối bù chuyển động 44 làm khối tham chiếu để mã hóa ngoài khối trong khung viđeo tiếp theo.

Fig.3 là sơ đồ khối minh họa ví dụ của bộ giải mã viđeo 30 có thể thực hiện các kỹ thuật tạo mã viđeo. Ở ví dụ trên Fig.3, bộ giải mã viđeo 30 bao gồm khối giải mã entropy 70, khối bù chuyển động 72, khối dự báo trong 74, khối lượng tử hóa ngược 76, khối biến đổi ngược 78, bộ nhớ khung tham chiếu 82, và bộ cộng 80. Bộ giải mã viđeo 30 có thể, trong một số ví dụ, thực hiện bước giải mã nói chung ngược với bước mã hóa được mô tả dựa vào bộ mã hóa viđeo 20 (Fig.2). Khối bù chuyển động 72 có thể tạo dữ liệu dự báo dựa trên các MV được nhận từ khối giải mã entropy 70, trong khi khối dự báo trong 74 có thể tạo dữ liệu dự báo dựa trên các bộ chỉ báo chế độ dự báo trong được nhận từ khối giải

mã entropy 70.

Trong quá trình giải mã, bộ giải mã viđeo 30 nhận được mã hóa video dòng bit vốn là các khối viđeo của lát viđeo được mã hóa và các phần tử cú pháp được liên kết từ bộ mã hóa viđeo 20. Khối giải mã entropy 70 của bộ giải mã viđeo 30 giải mã entropy dòng bit để tạo các hệ số được lượng tử hóa, các MV hoặc các bộ chỉ báo chế độ dự báo trong, và các phần tử cú pháp khác. Khối giải mã entropy 70 chuyển tiếp các MV và các phần tử cú pháp khác đến khối bù chuyển động 72. Bộ giải mã viđeo 30 có thể nhận các phần tử cú pháp ở mức lát viđeo và/hoặc mức khối viđeo.

Khi lát viđeo được mã hóa dưới dạng lát được tạo mã trong (I), khối dự báo trong 74 có thể tạo dữ liệu dự báo cho khối viđeo của lát viđeo hiện tại dựa trên chế độ dự báo trong và dữ liệu được báo hiệu từ các khối được giải mã trước đó của khung hoặc ảnh hiện tại. Khi khung viđeo được mã hóa dưới dạng lát được mã hóa ngoài (chẳng hạn, B, P, hoặc GPB), khối bù chuyển động 72 tạo các khối dự báo cho khối viđeo của lát viđeo hiện tại dựa trên các MV và các phần tử cú pháp khác được nhận từ khối giải mã entropy 70. Khối dự báos có thể được tạo từ một trong các ảnh tham chiếu trong một trong các danh sách ảnh tham chiếu. Bộ giải mã viđeo 30 có thể tạo các danh sách khung tham chiếu, Danh sách 0 và Danh sách 1, sử dụng các kỹ thuật tạo dựng mặc định dựa trên các ảnh tham chiếu được lưu trữ trong bộ nhớ khung tham chiếu 82.

Khối bù chuyển động 72 xác định thông tin dự báo cho khối viđeo của lát viđeo hiện tại bằng cách phân tách các MV và các phần tử cú pháp khác, và sử dụng thông tin dự báo để tạo các khối dự báo cho khối viđeo hiện tại được giải mã. Chẳng hạn, khối bù chuyển động 72 sử dụng một số phần tử cú pháp được nhận để xác định chế độ dự báo (chẳng hạn, dự báo trong hoặc ngoài) được sử dụng để mã hóa các khối viđeo của lát viđeo, loại lát dự báo ngoài (chẳng hạn, lát B, lát P, hoặc lát GPB), thông tin tạo cho một hoặc nhiều danh sách ảnh tham chiếu cho lát, các MV cho mỗi khối viđeo được mã hóa ngoài của lát, trạng thái dự báo ngoài cho mỗi khối viđeo được mã hóa ngoài của lát, và thông tin khác để giải mã các khối viđeo trong lát viđeo hiện tại.

Khối bù chuyển động 72 có thể cũng thực hiện nội suy dựa trên các bộ lọc

nội suy. Khối bù chuyển động 72 có thể sử dụng các bộ lọc nội suy như được sử dụng bằng bộ mã hóa viđeo 20 trong khi mã hóa các khối viđeo để tính toán các giá trị được nội suy cho các pixel phụ số nguyên của các khối tham chiếu. Trong trường hợp này, khối bù chuyển động 72 có thể xác định các bộ lọc tuyến tính được sử dụng bằng bộ mã hóa viđeo 20 từ các phần tử cú pháp được nhận và sử dụng các bộ lọc tuyến tính để tạo các khối dự báo.

Dữ liệu đối với ảnh kết cấu tương ứng với bản đồ chiều sâu có thể được lưu trữ trong bộ nhớ khung tham chiếu 82. Khối bù chuyển động 72 có thể cũng được tạo cấu hình để dự báo ngoài các khối chiều sâu của bản đồ chiều sâu.

Lưu ý rằng, các kỹ thuật nén viđeo thực hiện dự báo không gian (trong ảnh) và/hoặc dự báo thời gian (ngoài ảnh) để giảm hoặc loại bỏ dư thừa còn lại trong các chuỗi viđeo. Đối với tạo mã viđeo dựa trên khối, lát viđeo (tức là, ảnh viđeo hoặc một phần của ảnh viđeo) có thể được phân vùng thành các khối viđeo, có thể cũng được gọi là các khối cây, các khối cây mã (coding tree block, CTB), các đơn vị cây mã (coding tree unit, CTU), các đơn vị mã (coding unit, CU) và/hoặc các nút mã. Các khối viđeo trong lát I của ảnh được mã hóa sử dụng dự báo không gian dựa vào các mẫu tham chiếu trong các khối lân cận trong cùng ảnh. Các khối viđeo trong lát được mã hóa ngoài (P hoặc B) của ảnh có thể sử dụng dự báo không gian dựa vào các mẫu tham chiếu trong các khối lân cận trong cùng ảnh hoặc dự báo thời gian dựa vào các mẫu tham chiếu trong các ảnh tham chiếu khác. Các ảnh có thể được gọi là các khung, và các ảnh tham chiếu có thể được gọi là các khung tham chiếu.

Dự báo thời gian hoặc không gian dẫn đến khối dự báo cho khối cần được tạo mã. Dữ liệu dư là các hiệu số pixel giữa khối ban đầu cần được tạo mã và khối dự báo. Khối được mã hóa ngoài được mã hóa theo MV mà trỏ đến khối của các mẫu tham chiếu tạo khối dự báo, và dữ liệu dư chỉ báo hiệu số giữa khối được mã hóa và khối dự báo. Khối được mã hóa trong được mã hóa theo chế độ mã hóa trong và dữ liệu dư. Để nén tiếp, dữ liệu dư có thể được biến đổi từ miền pixel sang miền biến đổi, dẫn đến các hệ số biến đổi dư, mà sau đó có thể được lượng tử hóa. Các hệ số biến đổi được lượng tử hóa, được bố trí ban đầu trong mảng hai chiều, có thể được quét để tạo MV một chiều của các hệ số biến đổi,

và mã hóa entropy có thể được áp dụng để nén được nhiều hơn.

Kỹ thuật nén ảnh và viđeo đã phát triển nhanh chóng, dẫn đến các chuẩn mã hóa khác nhau. Các chuẩn tạo mã viđeo này bao gồm ITU-T H.261, ISO/IEC MPEG-1 Part 2, ITU-T H.262 hoặc ISO/IEC MPEG-2 Part 2, ITU-T H.263, ISO/IEC MPEG-4 Part 2, AVC, cũng được biết là ITU-T H.264 hoặc ISO/IEC MPEG-4 Part 10, và HEVC, cũng được biết là ITU-T H.265 hoặc MPEG-H Part 2. AVC bao gồm các phần mở rộng, chẳng hạn, tạo mã viđeo khả mở (Scalable video coding, SVC), tạo mã viđeo đa khung nhìn (Multiview video coding, MVC) và MVC có chiều sâu (MVC plus Depth, MVC+D), và 3D AVC (3D-AVC). HEVC bao gồm các phần mở rộng chẳng hạn HEVC khả mở (Scalable HEVC, SHVC), HEVC đa khung nhìn (Multiview HEVC, MV-HEVC), và 3D HEVC (3D-HEVC).

Cũng có chuẩn tạo mã viđeo mới, tên là VVC, được phát triển bởi nhóm chuyên gia viđeo chung (joint video experts team, JVET) của ITU-T và ISO/IEC. Trong khi chuẩn VVC có vài phác thảo công việc, một phác thảo (Working Draft, WD) của VVC cụ thể là, tức là B. Bross, J. Chen, và S. Liu, "Versatile Video Coding (Draft 4)," JVET-M1001-v5, 13th JVET Meeting, Jan. 2019 (VVC Draft 4) được tham khảo ở đây.

Phần mô tả của các kỹ thuật được bộc lộ ở đây dựa trên chuẩn tạo mã viđeo đang được phát triển VVC bởi JVET của ITU-T và ISO/IEC. Tuy nhiên, các kỹ thuật also áp dụng cho các đặc tả codec viđeo khác.

Fig.4 là biểu diễn 400 của mối quan hệ giữa ảnh IRAP 402 so với ảnh dẫn đầu 404 và các ảnh theo sau 406 theo thứ tự giải mã 408 và thứ tự hiển thị 410. Theo phương án thực hiện, ảnh IRAP 402 được gọi là ảnh truy nhập ngẫu nhiên (clean random access, CRA) hoặc ảnh IDR có ảnh đầu có thể giải mã truy nhập ngẫu nhiên (random access decodable leading, RADL). Trong chuẩn HEVC, các ảnh IDR, các ảnh CRA, và các ảnh truy nhập liên kết đứt gãy (Broken Link Access, BLA) đều được xem là ảnh IRAP 402. Đối với chuẩn VVC, trong cuộc gọi thứ 12 vào tháng 10, 2018, đồng ý xem các ảnh IDR và CRA là các ảnh IRAP.

Như được thể hiện trên Fig.4, ảnh dẫn đầu 404 (chẳng hạn, các ảnh 2 và 3) tiếp sau ảnh IRAP 402 theo thứ tự giải mã 408, nhưng đứng trước ảnh IRAP 402

theo thứ tự hiển thị 410. Ảnh sau 406 đi sau ảnh IRAP 402 cả theo thứ tự giải mã 408 và theo thứ tự hiển thị 410. Trong khi hai ảnh dẫn đầu 404 và một ảnh sau 406 được mô tả trên Fig.4, người có hiểu biết trung bình trong lĩnh vực sẽ hiểu rằng vài ảnh dẫn đầu 404 và/hoặc các ảnh theo sau 406 có thể xuất hiện theo thứ tự giải mã 408 và thứ tự hiển thị 410 trong các ứng dụng thực.

Ảnh dẫn đầu 404 trên Fig.4 đã được phân chia thành hai loại, tức là ảnh dẫn đần bỏ qua truy nhập ngẫu nhiên (random access skipped leading, RASL) và ảnh RADL. Khi giải mã bắt đầu với ảnh IRAP 402 (chẳng hạn, ảnh 1), ảnh RADL (chẳng hạn, ảnh 3) có thể được giải mã đúng; Tuy nhiên, ảnh RASL (chẳng hạn, ảnh 2) không thể được giải mã đúng. Do vậy, ảnh RASL bị loại bỏ. Do khác biệt giữa các ảnh RADL và RASL, loại ảnh dẫn đầu được liên kết với ảnh IRAP nên được nhận diện như là RADL học RASL để mã hóa đúng và hiệu quả. Trong chuẩn HEVC, khi các ảnh RASL và RADL xuất hiện, nó bị giới hạn rằng đối với các ảnh RASL và RADL được liên kết với cùng ảnh IRAP, các ảnh RASL sẽ đứng trước các ảnh RADL theo thứ tự hiển thị 410.

Ảnh IRAP 402 tạo hai ưu điểm/chức năng quan trọng sau. Trước hết, sự xuất hiện của ảnh IRAP 402 chỉ báo rằng quá trình giải mã có thể bắt đầu từ ảnh đó. Hàm này cho phép đặc tính truy nhập ngẫu nhiên trong đó quá trình giải mã bắt đầu ở vị trí đó trong dòng bit, không nhất thiết là bắt đầu của dòng bit, miễn là ảnh IRAP 402 xuất hiện ở vị trí đó. Thứ hai, sự xuất hiện của ảnh IRAP 402 làm mới quá trình giải mã sao cho ảnh được tạo mã bắt đầu ở ảnh IRAP 402, không bao gồm các ảnh RASL, được mã hóa không có tham chiếu bất kỳ đến các ảnh đứng trước. Kết quả là, việc có ảnh IRAP 402 xuất hiện trong dòng bit sẽ dừng lỗi bất kỳ mà có thể xuất hiện trong khi giải mã các ảnh được tạo mã trước ảnh IRAP 402 để làm truyền ảnh IRAP 402 và các ảnh đó theo sau ảnh IRAP 402 trong thứ tự giải mã 408.

Trong khi các ảnh IRAP 402 tạo các chức năng quan trọng, chúng gây ảnh hưởng đến hiệu quả nén. Sự xuất hiện của ảnh IRAP 402 làm tăng vọt tốc độ bit. Việc ảnh hưởng đến hiệu quả nén là do hai nguyên nhân. Trước hết, do ảnh IRAP 402 là ảnh được dự báo trong, chính ảnh sẽ yêu cầu tương đối nhiều bit hơn để biểu diễn khi so với các ảnh khác (chẳng hạn, ảnh dẫn đầu 404, các ảnh theo sau

406) vốn là các ảnh được dự báo ngoài. Thứ hai, do sự xuất hiện của ảnh IRAP 402 làm hỏng dự báo thời gian (đây là do bộ giải mã sẽ làm mới quá trình giải mã, trong đó một hoạt động của quá trình giải mã cho nó là loại bỏ các ảnh tham chiếu đứng trước trong DPB), ảnh IRAP 402 khiến mã hóa ảnh tiếp theo ảnh IRAP 402 theo thứ tự giải mã 408 trở nên ít hiệu quả hơn (tức là, cần nhiều bit hơn để biểu diễn) do chúng có ít ảnh tham chiếu hơn để mã hóa dự báo ngoài.

Trong số các loại ảnh được xem xét là các ảnh IRAP 402, ảnh IDR trong chuẩn HEVC có báo hiệu và dẫn xuất khác khi so với các loại ảnh khác. Một số hiệu số như sau.

Đối với báo hiệu và dẫn xuất của giá trị số đếm thứ tự ảnh (picture order count, POC) của ảnh IDR, phần bit có trọng số lớn nhất (most significant bit, MSB) của POC không được dẫn xuất từ ảnh chính trước đó mà đơn giản được gán bằng 0.

Đối với thông tin báo hiệu cần để quản lý ảnh tham chiếu, tiêu đề lát của ảnh IDR không chứa thông tin cần để báo hiệu để hỗ trợ quản lý ảnh tham chiếu. Đối với các loại ảnh khác (tức là, CRA, tiếp sau, truy nhập thời gian lớp con (temporal sub-layer access, TSA), v.v.), thông tin, chẳng hạn, tập hợp ảnh tham chiếu (reference picture set, RPS) được mô tả dưới đây hoặc các dạng khác của thông tin tương tự (chẳng hạn, các danh sách ảnh tham chiếu) cần cho quá trình đánh dấu các ảnh tham chiếu (tức là, quá trình xác định trạng thái của các ảnh tham chiếu trong DPB), được sử dụng để tham chiếu và không được sử dụng để tham chiếu). Tuy nhiên, đối với ảnh IDR, thông tin này không cần được báo hiệu do sự xuất hiện của IDR chỉ báo rằng quá trình giải mã sẽ đơn giản đánh dấu tất cả các ảnh tham chiếu trong DPB như là không được sử dụng để tham chiếu.

Trong chuẩn HEVC và VVC, mỗi ảnh trong các ảnh IRAP 402 và ảnh dẫn đầu 404 có thể được chứa trong một đơn vị NAL. Tập hợp các đơn vị NAL có thể được gọi là khối truy nhập. Các ảnh IRAP 402 và ảnh dẫn đầu 404 là các loại đơn vị NAL khác nhau được cấp sao cho chúng có thể dễ dàng được nhận diện bởi các ứng dụng mức hệ thống. Chẳng hạn, bộ nối viđeo cần hiểu các loại ảnh được tạo mã mà không phải hiểu quá nhiều chi tiết về phần tử cú pháp trong dòng bit được mã hóa, cụ thể là để nhận diện các ảnh IRAP 402 từ các ảnh phi

IRAP và để nhận diện ảnh dẫn đầu 404, bao gồm xác định các ảnh RASL và RADL, từ các ảnh theo sau 406. Các ảnh theo sau 406 là các ảnh được liên kết với ảnh IRAP 402 và tiếp theo ảnh IRAP 402 theo thứ tự hiển thị 410. Ảnh có thể tiếp theo ảnh IRAP 402 cụ thể theo thứ tự giải mã 408 và đứng trước ảnh IRAP 402 khác bất kỳ theo thứ tự giải mã 408. Về vấn đề này, việc cấp cho các ảnh IRAP 402 và ảnh dẫn đầu 404 loại đơn vị NAL của riêng nó sẽ giúp cho các ứng dụng này.

Đối với chuẩn HEVC, các loại đơn vị NAL đối với các ảnh IRAP bao gồm các tham số sau:

BLA có ảnh dẫn đầu (BLA_W_LP): đơn vị NAL của ảnh BLA có thể tiếp sau bởi một hoặc nhiều ảnh dẫn đầu theo thứ tự giải mã.

BLA với RADL (BLA_W_RADL): đơn vị NAL của ảnh BLA có thể tiếp sau bởi một hoặc nhiều ảnh RADL mà không có ảnh RASL theo thứ tự giải mã.

BLA không có ảnh dẫn đầu (BLA_N_LP): đơn vị NAL của ảnh BLA không được tiếp sau bởi ảnh dẫn đầu theo thứ tự giải mã.

IDR với RADL (IDR_W_RADL): đơn vị NAL của ảnh IDR có thể tiếp sau bởi một hoặc nhiều ảnh RADL mà không có ảnh RASL nào theo thứ tự giải mã.

IDR không có ảnh dẫn đầu (IDR_N_LP): đơn vị NAL của ảnh IDR không được tiếp theo bởi ảnh dẫn đầu theo thứ tự giải mã.

CRA: đơn vị NAL của ảnh CRA có thể tiếp sau bởi ảnh dẫn đầu (tức là, các ảnh RASL hoặc các ảnh RADL hoặc cả hai).

RADL: đơn vị NAL của ảnh RADL.

RASL: đơn vị NAL của ảnh RASL.

Đối với chuẩn VVC, loại đơn vị NAL đối với các ảnh IRAP 402 và ảnh dẫn đầu 404 như sau:

IDR có RADL (IDR_W_RADL): đơn vị NAL của ảnh IDR có thể tiếp sau bởi một hoặc nhiều các ảnh RADL nhưng không ảnh RASL nào theo thứ tự giải mã.

IDR không có ảnh dẫn đầu (IDR_N_LP): đơn vị NAL của ảnh IDR không được tiếp theo bởi ảnh dẫn đầu theo thứ tự giải mã.

CRA: đơn vị NAL của ảnh CRA mà có thể tiếp sau bởi ảnh dẫn đầu (tức là,

các ảnh RASL hoặc các ảnh RADL hoặc cả hai).

RADL: đơn vị NAL của ảnh RADL.

RASL: đơn vị NAL của ảnh RASL.

Làm mới nội ảnh từng bước/làm mới giải mã từng bước được nêu dưới đây.

Đối với các ứng dụng độ trễ thấp, cần tránh tạo mã ảnh dưới dạng ảnh IRAP (chẳng hạn, ảnh IRAP 402) do yêu cầu tốc độ bit tương đối lớn so với các ảnh phi IRAP (tức là, P-/B-), kết quả là gây ra nhiều độ trễ hơn. Tuy nhiên, việc hoàn toàn tránh sử dụng IRAP có thể là không thể trong tất cả các ứng dụng độ trễ thấp. Chẳng hạn, đối với các ứng dụng hội thoại, chẳng hạn, hội nghị từ xa đa biên, thì cần thiết cung cấp các điểm chính quy trong đó người dùng mới có thể gia nhập hội nghị từ xa.

Để tạo truy nhập vào dòng bit cho phép người dùng mới gia nhập ứng dụng hội nghị từ xa đa biên, một chiến lược khả thi là sử dụng kỹ thuật làm mới nội ảnh từng bước (Progressive Intra Refresh, PIR) thay cho việc sử dụng các ảnh IRAP để tránh có đỉnh trong tốc độ bit. PIR có thể cũng được gọi là GDR. Cụm từ PIR và GDR có thể được sử dụng qua lại theo sáng chế.

Fig.5 minh họa kỹ thuật GDR 500. Như được thể hiện, kỹ thuật GDR 500 được mô tả nhờ sử dụng ảnh GDR 502, một hoặc nhiều ảnh tiếp sau 504, và ảnh điểm khôi phục 506 trong CVS 508 của dòng bit. Theo phương án thực hiện, ảnh GDR 502, các ảnh tiếp sau 504, và ảnh điểm khôi phục 506 có thể định nghĩa chu kỳ GDR trong CVS 508. CVS 508 là chuỗi ảnh (hoặc các phần của ảnh) bắt đầu với ảnh GDR 502 và bao gồm tất cả các ảnh (hoặc các phần của nó) lên tới, mà không bao gồm, ảnh GDR tiếp theo hoặc cho đến khi kết thúc dòng bit. Chu kỳ GDR là chuỗi ảnh bắt đầu với ảnh GDR 502 và bao gồm tất cả các ảnh tới và bao gồm ảnh điểm khôi phục 506.

Như được thể hiện trên Fig.5, kỹ thuật GDR 500 hoặc nguyên lý làm việc trên chuỗi ảnh bắt đầu với ảnh GDR 502 và kết thúc với ảnh điểm khôi phục 506. Ảnh GDR 502 chứa vùng sạch/được làm mới 510 chứa các khối đều đã được mã hóa sử dụng dự báo trong (tức là, các khối được dự báo trong) vùng bẩn/không được làm mới 512 chứa các khối đều đã được mã hóa sử dụng dự báo ngoài (tức là, các khối được dự báo ngoài).

Ảnh sau 504 liền kề ngay trước ảnh GDR 502 chứa vùng được làm mới/sạch 510 có phần thứ nhất 510A được tạo mã nhờ sử dụng dự báo trong và phần thứ hai 510B được tạo mã nhờ sử dụng dự báo ngoài. Phần thứ hai 510B được tạo mã nhờ tạo tham chiếu vùng được làm mới/sạch 510 của, chẳng hạn, ảnh đứng trước trong chu kỳ GDR của CVS 508. Như được thể hiện, vùng được làm mới/sạch 510 của các ảnh tiếp sau 504 mở rộng khi quá trình tạo mã di chuyển hoặc tiến triển theo hướng nhất quán (chẳng hạn, từ trái sang phải), một cách tương ứng co lại vùng bẩn/không được làm mới 512. Dần dần, ảnh điểm khôi phục 506, chỉ chứa vùng được làm mới/sạch 510, thu được từ quá trình tạo mã. Đáng lưu ý, và như sẽ được nêu thêm dưới đây, phần thứ hai 510B của vùng được làm mới/sạch 510, được mã hóa dưới dạng các khối được dự báo ngoài, có thể chỉ tham chiếu đến vùng được làm mới/vùng sạch 510 trong ảnh tham chiếu.

Trong chuẩn HEVC, kỹ thuật GDR 500 trên Fig.5 được hỗ trợ không chuẩn tắc sử dụng thông điệp thông tin tăng cường bổ trợ (Supplemental Enhancement Information, SEI) điểm khôi phục và thông điệp SEI thông tin làm mới vùng. Hai thông điệp SEI này không định nghĩa cách thức GDR được thực hiện. Thay vào đó, hai thông điệp SEI chỉ nêu cơ chế chỉ báo thứ nhất và các ảnh cuối cùng trong chu kỳ GDR (tức là, được tạo bởi thông điệp SEI điểm khôi phục) và vùng được làm mới (tức là, được tạo bởi thông điệp SEI thông tin làm mới vùng).

Trên thực tế, kỹ thuật GDR 500 được thực hiện bằng cách sử dụng hai kỹ thuật đồng thời. Hai kỹ thuật đó là dự báo trong ràng buộc (Constraint Intra Prediction, CIP) và các ràng buộc của bộ mã hóa đối với các MV. CIP có thể được sử dụng cho GDR, cụ thể là để tạo mã vùng được tạo mã chỉ dưới dạng các khối được dự báo trong (chẳng hạn, phần thứ nhất 510A của vùng được làm mới/sạch 510) do CIP cho phép vùng không sử dụng các mẫu từ vùng được làm mới (chẳng hạn, vùng bẩn/không được làm mới 512) được sử dụng để tham chiếu. Tuy nhiên, việc sử dụng CIP gây ra suy giảm hiệu năng tạo mã nghiêm trọng do ràng buộc với các khối trong phải được áp dụng không chỉ cho các khối trong trong vùng được làm mới, mà còn cả với tất cả các khối trong trong ảnh. Các ràng buộc bộ mã hóa cho các MV giới hạn bộ mã hóa khỏi việc sử dụng các mẫu bất kỳ trong các ảnh tham chiếu mà được đặt ngoài vùng được làm mới.

Ràng buộc này gây ra tìm kiếm chuyển động không tối ưu.

Fig.6 là sơ đồ minh họa tìm kiếm chuyển động không mong muốn 600 khi sử dụng giới hạn bộ mã hóa để hỗ trợ GDR. Như được thể hiện, tìm kiếm chuyển động 600 mô tả ảnh hiện tại 602 và ảnh tham chiếu 604. Mỗi ảnh trong ảnh hiện tại 602 và ảnh tham chiếu 604 bao gồm vùng được làm mới 606 được mã hóa với dự báo trong, vùng được làm mới 608 được mã hóa với dự báo ngoài, và vùng không được làm mới 608. Vùng được làm mới 604, vùng được làm mới 606, và the vùng không được làm mới 608 giống như phần thứ nhất 510A của vùng được làm mới/sạch 510, phần thứ hai 510B của vùng được làm mới/sạch 510, và vùng bẩn/không được làm mới 512 trên Fig.5.

Trong quá trình tìm kiếm chuyển động, bộ mã hóa bị giới hạn hoặc bị ngăn không cho lựa chọn MV 610 bất kỳ dẫn đến một số mẫu của khối tham chiếu 612 được đặt ngoài vùng được làm mới 606. Điều này xảy ra thậm chí khi khối tham chiếu 612 tạo các tiêu chí méo dạng tốc độ tốt nhất khi dự báo khối hiện tại 614 trong ảnh hiện tại 602. Do vậy, Fig.6 minh họa nguyên nhân đối với việc không tối ưu trong tìm kiếm chuyển động 600 khi sử dụng giới hạn bộ mã hóa để hỗ trợ GDR.

Các thành phần JVET JVET-K0212 và JVET-L0160 mô tả triển khai GDR dựa trên việc sử dụng CIP và cách thức ràng buộc bộ mã hóa. Triển khai có thể được tóm tắt như sau: chế độ dự báo trong được tập trung vào CU trên cơ sở cột, dự báo trong bị ràng buộc được kích hoạt để đảm bảo tái tạo CU nội vùng, các MV bị ràng buộc để trỏ đến trong khu vực được làm mới trong khi xem xét biên bổ sung để tránh lan rộng sai số cho các bộ lọc (6 pixel, chẳng hạn), và loại bỏ các ảnh tham chiếu trước đó khi lặp lại cột nội vùng.

Thành phần JVET JVET-M0529 đề cập đến phương pháp chỉ báo rằng ảnh là thứ nhất và cuối cùng trong chu kỳ GDR một cách chuẩn tắc. Ý tưởng được đề xuất như sau.

Định nghĩa đơn vị NAL mới có chỉ báo điểm khôi phục lọi đơn vị NAL dưới dạng đơn vị VCL NAL. Tải tin của đơn vị NAL chứa phần tử cú pháp để xác định thông tin có thể được sử dụng để dẫn xuất giá trị POC của ảnh cuối trong chu kỳ GDR. Khối truy nhập chứa đơn vị phi VCL NAL với chỉ báo điểm khôi

phục loại được gọi là khối truy nhập (access unit, AU) bắt đầu điểm khôi phụ (recovery point begin, RBP) và ảnh trong RBP AU được gọi là ảnh RBP. Quá trình giải mã có thể bắt đầu từ RBP AU. Khi giải mã bắt đầu từ RBP AU, tất cả các ảnh trong chu kỳ GDR, ngoại trừ ảnh cuối cùng, không được xuất ra.

Một số vấn đề với thiết kế GDR hiện tại được thảo luận.

Thiết kế/cách thức hiện tại để hỗ trợ GDR ít nhất có các vấn đề sau.

Phương pháp định nghĩa GDR chuẩn tắc trong JVET-M0529 có các vấn dề sau. Phương pháp được đề xuất không mô tả cách thức GDR được thực hiện. Thay vào đó, phương pháp được đề xuất chỉ nêu một vài báo hiệu cho chỉ báo thứ nhất và các ảnh cuối cùng trong chu kỳ GDR. Đối với chỉ báo thứ nhất và các ảnh cuối cùng trong chu kỳ GDR, cần đơn vị phi VCL NAL mới. Đây là dư thừa do thông tin được chứa trong đơn vị NAL chỉ báo điểm khôi phục (recovery point indication, RPI) có thể chỉ được bao gồm trong tiêu đề nhóm phiến của ảnh thứ nhất trong chu kỳ GDR. Phương pháp được đề xuất cũng không thể mô tả vùng nào trong các ảnh trong chu kỳ GDR là vùng được làm mới và vùng không được làm mới.

Cách thức GDR được mô tả theo JVET-K0212 và JVET-L0160 có các vấn dề sau. Trước hết, sử dụng CIP. Cần thiết tạo mã vùng được làm mới có dự báo trong với một số ràng buộc để ngăn không cho các mẫu bất kỳ với vùng không được làm mới được sử dụng làm tham chiếu không gian. Khi CIP được sử dụng, tạo mã được dựa trên ảnh, nghĩa là tất cả các khối nội vùng trong ảnh cùng phải được tạo mã dưới dạng các khối nội vùng CIP. Kết quả là, gây ra suy giảm hiệu năng. Ngoài ra, việc sử dụng ràng buộc bộ mã hóa để giới hạn tìm kiếm chuyển động ngăn không cho bộ mã hóa chọn MV tốt nhất khi các mẫu của khối tham chiếu được liên kết với MV không hoàn toàn nằm trong vùng được làm mới trong ảnh tham chiếu. Vùng được làm mới được tạo mã chỉ với dự báo trong cũng không phải kích thước CTU. Thay vào đó, vùng được làm mới có thể trở nên nhỏ hơn kích thước CTU, giảm xuống tới kích thước CU nhỏ nhất. Điều này khiến việc triển khai trở nên phức tạp không cần thiết do nó có thể cần chỉ báo ở mức khối.

Các kỹ thuật hỗ trợ GDR trong tạo mã viđeo được bộc lộ. Các kỹ thuật được

bộc lộ cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Như sẽ được giải thích đầy đủ hơn dưới đây, đơn vị VCL NAL có GDR_NUT để chỉ báo rằng đơn VCL NAL chứa ảnh GDR. Tức là, GDR_NUT ngay lập tức chỉ báo đến bộ giải mã viđeo rằng đơn VCL NAL có GDR_NUT chứa ảnh GDR. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, bộ mã hóa/bộ giải mã (cũng được biết là "codec") trong kỹ thuật tạo mã viđeo được cải thiện so với các codec hiện tại. Trên thực tế, quá trình tạo mã viđeo được cải thiện mang lại cho người dùng trải nghiệm tốt hơn khi các viđeo được gửi, được nhận, và/hoặc được xem.

Để giải quyết một hoặc nhiều vấn đề nêu trên, sáng chế bộc lộ các khía cạnh sau. Mỗi khía cạnh trong các khía cạnh có thể được áp dụng riêng rẽ, và một số có thể được áp dụng kết hợp.

1) Đơn vị VCL NAL có loại GDR_NUT được định nghĩa.

a. Các ảnh có loại đơn vị NAL GDR_NUT được gọi là ảnh GDR, tức là, ảnh thứ nhất trong chu kỳ GDR.

b. Ảnh GDR có temporalID bằng 0.

c. Khối truy nhập chứa ảnh GDR được gọi là khối truy nhập GDR. Như lưu ý trên đây, khối truy nhập là tập hợp đơn vị NAL. Mỗi đơn vị NAL có thể chứa một ảnh.

2) CVS có thể bắt đầu với khối truy nhập GDR.

3) Khối truy nhập GDR là khối truy nhập thứ nhất trong CVS khi một trong các điều kiện sau là đúng:

a. Khối truy nhập GDR là khối truy nhập thứ nhất trong dòng bit.

b. Khối truy nhập GDR theo sau ngay khối truy nhập kết thúc chuỗi (end-of-sequence, EOS).

c. Khối truy nhập GDR theo sau ngay khối truy nhập kết thúc dòng bit (end-of-bitstream, EOB).

d. Cờ bộ giải mã NoIncorrectPicOutputFlag được liên kết với ảnh GDR và giá trị của cờ được gán bằng bằng 1 (tức là, true) bởi thực thể nằm ngoài bộ giải

mã.

4) Khi ảnh GDR là khối truy nhập thứ nhất trong CVS, điều kiện sau áp dụng:

a. Tất cả các ảnh tham chiếu trong DPB được đánh dấu là "không được sử dụng để tham chiếu."

b. POC MSB của ảnh được gán bằng bằng 0.

c. Ảnh GDR và tất cả các ảnh đi sau ảnh GDR theo thứ tự đầu ra cho đến khi ảnh cuối cùng trong chu kỳ GDR, không bao gồm ảnh cuối cùng trong chu kỳ GDR, không được xuất ra (tức là, được đánh dấu là "không cần cho đầu ra").

5) Cờ để xác định liệu GDR được kích hoạt được báo hiệu trong tập hợp tham số mức chuỗi (chẳng hạn, trong SPS).

a. Cờ có thể được chỉ định gdr_enabled_flag.

b. Khi cờ bằng 1, ảnh GDR có thể hiện có trong CVS. Ngược lại, khi cờ bằng 0, GDR không được kích hoạt sao cho ảnh GDR không hiện có trong CVS.

6) Thông tin mà có thể được sử dụng để dẫn xuất giá trị POC của ảnh cuối cùng trong chu kỳ GDR được báo hiệu trong tiêu đề nhóm ô vuông của ảnh GDR.

a. Thông tin được báo hiệu dưới dạng POC delta giữa ảnh cuối cùng trong chu kỳ GDR và ảnh GDR. Thông tin có thể được báo hiệu sử dụng phần tử cú pháp được chỉ định recovery_point_cnt.

b. Sự xuất hiện của phần tử cú pháp recovery_point_cnt trong tiêu đề nhóm ô vuông có thể có điều kiện theo giá trị của gdr_enabled flag và loại đơn vị NAL của ảnh, tức là, cờ hiện có chỉ khi GDR_enabled_flag bằng 1 và nal_unit_type của đơn vị NAL chứa nhóm ô vuông là GDR_NUT.

7) Cờ để xác định liệu nhóm ô vuông có phải là một phần của vùng được làm mới hay không được báo hiệu trong tiêu đề nhóm ô vuông.

a. Cờ có thể được chỉ định refreshed_region_flag.

b. Sự xuất hiện cờ có thể tùy thuộc theo giá trị của GDR_enabled_flag và liệu ảnh chứa nhóm ô vuông có nằm trong chu kỳ GDR. Do vậy, cờ hiện có chỉ khi tất cả các điều kiện sau là đúng:

i. Giá trị của gdr_enabled_flag bằng 1.

ii. Giá trị POC của ảnh hiện tại bằng hoặc lớn hơn giá trị POC của ảnh GDR

cuối cùng (khi ảnh hiện tại là ảnh GDR, ảnh GDR cuối cùng là ảnh hiện tại) và nhỏ hơn giá trị POC của ảnh cuối cùng trong chu kỳ GDR.

c. Khi cờ không có trong tiêu đề nhóm phiến, giá trị của cờ được suy ra bằng 1.

8) Tất cả các nhóm ô có refreshed_region_flag bằng 1 bao phủ vùng được kết nối. Tương tự, tất cả các nhóm ô có refreshed_region_flag bằng 0 cũng bao phủ vùng được kết nối.

9) Nhóm ô vuông có refreshed_region_flag có thể có loại I (tức là, nhóm ô vuông trong) hoặc B- hoặc P- (tức là, nhóm ô vuông ngoài).

10)Mỗi ảnh bắt đầu từ ảnh GDR đến ảnh cuối cùng trong chu kỳ GDR chứa ít nhất một nhóm ô vuông có refreshed_region_flag bằng 1.

11)ảnh GDR chứa ít nhất một nhóm ô vuông có refreshed_region_flag bằng 1 và tile_group_type bằng I (tức là, nhóm ô vuông trong).

12)Khi GDR_enabled_flag bằng 1, thông tin về nhóm ô hình chữ nhật, tức là, số lượng nhóm ô và các địa chỉ của nó, được phép được báo hiệu trong PPS hoặc trong tiêu đề nhóm ô vuông. Để làm vậy, cờ được báo hiệu trong PPS để xác định liệu thông tin nhóm ô hình chữ nhật hiện có trong PPS hoặc không. Cờ này có thể được gọi là rect_tile_group_info_in_pps_flag. Cờ này có thể bị giới hạn bằng 1 khi GDR_enabled_flag bằng 1.

a. Theo một tùy chọn, thay cho báo hiệu liệu thông tin nhóm ô hình chữ nhật hiện có trong PPS hay không, cờ tổng quát hơn có thể được báo hiệu trong PPS để xác định liệu thông tin nhóm ô vuông (tức là, loại bất kỳ của nhóm ô – chẳng hạn nhóm ô hình chữ nhật, nhóm ô quét mành) hiện có trong PPS.

13)Khi thông tin nhóm ô vuông không có trong PPS, có thể còn bị giới hạn rằng báo hiệu về thông tin ID nhóm ô tường minh không có. Thông tin ID nhóm ô tường minh bao gồm: signaled_tile_group_id_flag, signaled_tile_group_id_length_minus1, và tile_group_id[i].

14)Cờ được báo hiệu để xác định liệu hoạt động lọc vòng giữa các đường biên giữa vùng được làm mới và vùng không được làm mới trong ảnh được phép.

a. Cờ này có thể được báo hiệu trong PPS và được gọi là loop_filter_across_refreshed_region_enabled_flag.

b. Sự xuất hiện của loop_filter_across_refreshed_region_enabled_flag có thể có điều kiện theo giá trị của loop_filter_across_tile_enabled_flag. Khi loop_filter_across_tile_enabled_flag bằng 0, loop_filter_across_refreshed_region_enabled_flag có thể not hiện có và giá trị của nó được suy ra bằng 0.

c. Theo một tùy chọn, cờ có thể được báo hiệu in tiêu đề nhóm phiến và sự xuất hiện của nó có thể tùy thuộc vào giá trị của refreshed_region_flag, tức là, cờ hiện có chỉ khi giá trị của refreshed_region_flag bằng 1.

15) Khi được chỉ báo rằng nhóm ô vuông là vùng được làm mới và được chỉ báo rằng hoạt động lọc vòng giữa vùng được làm mới không được phép, điều kiện sau áp dụng:

a. Bước tách khối đường mép ở biên của nhóm ô vuông không được thực hiện khi nhóm ô lân cận chia sẻ đường mép là nhóm ô không được làm mới.

b. Quá trình SAO cho các khối ở đường biên của nhóm ô vuông không sử dụng các mẫu bất kỳ từ ngoài đường biên vùng được làm mới.

c. Quá trình lọc vòng thích ứng (Adaptive loop filtering, ALF) cho các khối ở đường biên của nhóm ô vuông không sử dụng các mẫu bất kỳ từ ngoài đường biên vùng được làm mới.

16)Khi GDR_enabled_flag bằng 1, mỗi ảnh được liên kết với các biến để xác định các đường biên của vùng được làm mới trong ảnh. Các biến này có thể được gọi là:

a. PicRefreshedLeftBoundaryPos cho vị trí đường biên trái của vùng được làm mới trong ảnh.

b. PicRefreshedRightBoundaryPos cho vị trí đường biên phải của vùng được làm mới trong ảnh.

c. PicRefreshedTopBoundaryPos cho vị trí đường biên trên của vùng được làm mới trong ảnh.

d. PicRefreshedBotBoundaryPos cho vị trí đường biên dưới của vùng được làm mới trong ảnh.

17)Các đường biên của vùng được làm mới trong ảnh có thể được dẫn xuất. Các đường biên của vùng được làm mới của ảnh được cập nhật bằng bộ giải mã

sau khi tiêu đề nhóm ô vuông được phân tách và giá trị của refreshed_region_flag của nhóm ô vuông bằng 1.

18)Theo một tùy chọn với giải pháp 17), các đường biên của vùng được làm mới trong ảnh được báo hiệu tường minh trong mỗi nhóm ô của ảnh.

a. Cờ có thể được báo hiệu để chỉ báo liệu ảnh mà có nhóm ô vuông có chứa vùng không được làm mới hay không. Khi xác định được rằng ảnh không chứa vùng được làm mới, không thông tin đường biên được làm mới nào được báo hiệu và có thể chỉ được suy ra bằng các đường biên ảnh.

19)Đối với ảnh hiện tại, các đường biên của vùng được làm mới được sử dụng trong quá trình lọc trong vòng như sau:

a. Đối với quá trình lọc tách khối, để xác định đường mép của vùng được làm mới để quyết định liệu đường mép có cần được tách khối hay không.

b. Đối với quá trình SAO, để xác định đường biên của vùng làm mới sao cho quá trình cắt xén có thể được áp dụng để tránh sử dụng các mẫu từ vùng được làm mới khi quá trình lọc vòng giữa vùng được làm mới không được phép.

c. Đối với quá trình ALF, để xác định đường biên của vùng làm mới sao cho quá trình cắt xén có thể được áp dụng để tránh sử dụng các mẫu từ vùng không được làm mới khi quá trình lọc vòng giữa vùng được làm mới không được phép.

20)Đối với quá trình bù chuyển động, thông tin về các đường biên vùng được làm mới, cụ thể là các đường biên của vùng được làm mới trong ảnh tham chiếu, được sử dụng như sau: khi khối hiện tại trong ảnh hiện tại nằm trong nhóm ô vuông có refreshed_region_flag bằng 1 và khối tham chiếu nằm trong ảnh tham chiếu chứa vùng không được làm mới, điều kiện sau áp dụng:

a. MV từ khối hiện tại đến ảnh tham chiếu đó được xén tỉa bằng các đường biên của vùng được làm mới trong ảnh tham chiếu đó.

b. Đối với bộ lọc nội suy phân số cho các mẫu trong ảnh tham chiếu đó, nso được xén bởi các đường biên của vùng được làm mới trong ảnh tham chiếu đó.

Các phương án thực hiện sáng chế được mô tả chi tiết. Phần mô tả được so với văn bản cơ bản, vốn là thành phần JVET JVET-M1001-v5. Tức là, chỉ delta được mô tả, trong khi các văn bản trong văn bản cơ bản không được nêu dưới đây áp dụng đúng theo thực tế. Văn bản được chỉnh sửa so với văn bản cơ bản

được in nghiêng.

Các định nghĩa được nêu.

3.1 Ảnh CRA: Ảnh IRAP mà mỗi đơn vị VCL NAL cho nó có nal_unit_type bằng CRA_NUT.

Lưu ý – Ảnh CRA không đề cập đến ảnh bất kỳ ngoài chính nó để dự báo ngoài trong quá trình giải mã của nó, và có thể là ảnh thứ nhất trong dòng bit theo thứ tự giải mã, hoặc có thể xuất hiện sau trong dòng bit. Ảnh CRA có thể có các ảnh RADL hoặc RASL liên kết. Khi ảnh CRA có *NoIncorrectPicOutputFlag* bằng 1, các ảnh RASL liên kết không được xuất ra bằng bộ giải mã, do các ảnh RASL không thể giải mã được, do chúng có thể chứa các tham chiếu đến các ảnh không có trong dòng bit.

3.2 CVS: Chuỗi khối truy nhập bao gồm, theo thứ tự giải mã, của khối truy nhập IRAP với *NoIncorrectPicOutputFlag* bằng 1 *hoặc khối truy nhập GDR với NoIncorrectPicOutputFlag bằng 1*, tiếp theo bởi 0 hoặc nhiều khối truy nhập hơn mà không phải là các khối truy nhập IRAP với *NoIncorrectPicOutputFlag* bằng 1 *hoặc khối truy nhập GDR có NoIncorrectPicOutputFlag bằng 1*, bao gồm tất cả các khối truy nhập tiếp theo mà không bao gồm khối truy nhập tiếp theo vốn là khối truy nhập IRAP với *NoIncorrectPicOutputFlag* bằng 1 *hoặc khối truy nhập GDR có NoIncorrectPicOutputFlag bằng 1*.

Lưu ý 1 – Khối truy nhập IRAP có thể là khối truy nhập IDR hoặc khối truy nhập CRA. Giá trị của *NoIncorrectPicOutputFlag* bằng 1 cho mỗi khối truy nhập IDR và mỗi khối truy nhập CRA vốn là khối truy nhập thứ nhất trong dòng bit theo thứ tự giải mã, là khối truy nhập thứ nhất tiếp theo đơn vị NAL EOS theo thứ tự giải mã, hoặc có HandleCraAsCvsStartFlag bằng 1.

Lưu ý 2 – Giá trị của NoIncorrectPicOutputFlag bằng 1 cho mỗi khối truy nhập GDR vốn là khối truy nhập thứ nhất trong dòng bit theo thứ tự giải mã, là khối truy nhập thứ nhất tiếp theo đơn vị NAL EOS theo thứ tự giải mã, hoặc có HandleGdrAsCvsStartFlag bằng 1.

3.3 Khối truy nhập GDR: Khối truy nhập trong đó ảnh được tạo mã là ảnh GDR.

3.4 Ảnh GDR: ảnh mà mỗi đơn vị VCL NAL cho nó nal_unit_type bằng

GDR_NUT.

3.5 Ảnh RASL: Ảnh được tạo mã mà mỗi đơn vị VCL NAL cho nó có nal_unit_type bằng RASL_NUT.

Lưu ý– Tất cả các ảnh RASL là ảnh dẫn đầu của ảnh CRA liên kết. Khi ảnh CRA liên kết có *NoIncorrectPicOutputFlag* bằng 1, ảnh RASL không được xuất ra và không thể được giải mã đúng, do ảnh RASL có thể chứa các tham chiếu đến các ảnh không có trong dòng bit. Các ảnh RASL không được sử dụng làm các ảnh tham chiếu cho quá trình giải mã của các ảnh phi RASL. Khi hiện có, tất cả các ảnh RASL đứng trước, theo thứ tự giải mã, tất cả các ảnh tiếp sau của cùng ảnh CRA liên kết.

Cú pháp và ngữ nghĩa tải tin chuỗi byte thô (raw byte sequence payload, RBSP) của tập hợp tham số chuỗi.

seq_parameter_set_rbsp() {	**Mô tả**
...	
gdr_enabled_flag	*u(1)*
...	

gdr_enabled_flag bằng 1 xác định các ảnh GDR có thể hiện có trong CVS. gdr_enabled_flag bằng 0 xác định rằng các ảnh GDR không có trong CVS.

Cú pháp và ngữ nghĩa RBSP tập hợp tham số ảnh.

pic_parameter_set_rbsp() {	**Mô tả**
...	
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
...	ue(v)
single_tile_per_tile_group_flag	u(1)
if(!single_tile_per_tile_group_flag)	
rect_tile_group_flag	u(1)
if(rect_tile_group_flag)	
rect_tile_group_info_in_pps_flag	*u(1)*
if(rect_tile_group_flag && !single_tile_per_tile_group_flag && *rect_tile_group_info_in_pps_flag)* {	
num_tile_groups_in_pic_minus1	ue(v)
for(i = 0; i <= num_tile_groups_in_pic_minus1; i++) {	
if(i > 0)	
top_left_tile_idx[i]	u(v)
bottom_right_tile_idx[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
if(loop_filter_across_tiles_enabled_flag)	
loop_filter_across_tile_groups_enabled_flag	u(1)
if(loop_filter_across_tile_groups_enabled_flag)	

loop_filter_across_refreshed_region_enabled_flag	*u(1)*
}	
...	

rect_tile_group_info_in_pps_flag bằng 1 xác định rằng thông tin nhóm ô hình chữ nhật được báo hiệu trong PPS. rect_tile_group_info_in_pps_flag bằng 0 xác định rằng thông tin nhóm ô hình chữ nhật không được báo hiệu trong PPS.

Sự tương thích dòng bit yêu cầu rằng giá trị của rect_tile_group_info_in_pps_flag phải bằng 0 khi giá trị của gdr_enabled_flag trong SPS hoạt động bằng 0.

loop_filter_across_refreshed_region_enabled_flag bằng 1 xác định rằng các hoạt động lọc trong vòng có thể được thực hiện giữa các đường biên của nhóm ô với refreshed_region_flag bằng 1 trong các ảnh đề cập đến PPS. loop_filter_across_refreshed_region_enabled_flag bằng 0 xác định rằng các hoạt động lọc trong vòng không được thực hiện giữa các đường biên của nhóm ô với refreshed_region_flag bằng 1 trong các ảnh đề cập đến PPS. Các hoạt động lọc trong vòng bao gồm DBF, bộ lọc SAO, và các hoạt động ALF. Khi không có, giá trị của loop_filter_across_refreshed_region_enabled_flag được suy ra bằng 0.

signalled_tile_group_id_flag bằng 1 xác định rằng nhóm ô vuông ID cho mỗi nhóm ô được báo hiệu. signalled_tile_group_index_flag bằng 0 xác định rằng các ID nhóm ô không được báo hiệu. Khi *không có*, giá trị của signalled_tile_group_index_flag được suy ra bằng 0.

signalled_tile_group_id_length_minus1 plus 1 xác định số lượng bit được sử dụng để biểu diễn phần tử cú pháp tile_group_id[i] khi hiện có, và phần tử cú pháp tile_group_address trong các tiêu đề nhóm ô vuông. Giá trị của signalled_tile_group_index_length_minus1 sẽ nằm trọn trong khoảng từ 0 đến 15. *Khi không có, giá trị của signalled_tile_group_index_length_minus1 được suy ra như sau:*

If rect_tile_group_info_in_pps_flag bằng 1, Ceil(Log2(num_tile_groups_in_pic_minus1 + 1)) − 1.

Else, Ceil(Log2(NumTilesInPic)) − 1

Cú pháp và ngữ nghĩa tiêu đề nhóm ô vuông tổng quát.

tile_group_header() {	**Mô tả**
...	
if(rect_tile_group_flag || NumTilesInPic > 1)	
tile_group_address	u(v)
if(*!rect_tile_group_info_in_pps_flag* && !single_tile_per_tile_group_flag) {	
if(rect_tile_group_flag)	
bottom_right_tile_id	*u(v)*
Else	
num_tiles_in_tile_group_minus1	ue(v)
}	
tile_group_type	ue(v)
tile_group_pic_order_cnt_lsb	u(v)
if(gdr_enabled_flag) {	
if(nal_unit_type == GDR_NUT)	
recovery_poc_cnt	*se(v)*
if(PicOrderCntVal >= LastGDRPocVal && PicOrderCntVal < RecoveryPointPocVal)	
refreshed_region_flag	*u(1)*
}	
...	)

tile_group_address xác định địa chỉ ô vuông của ô vuông thứ nhất trong nhóm ô vuông. Khi không có, giá trị của tile_group_address được suy ra bằng 0.

Nếu rect_tile_group_flag bằng 0, điều kiện sau áp dụng:

tile_group_address là ID phiến như được xác định bằng Phương trình 6-7.

Chiều dài của tile_group_address là Ceil(Log2 (NumTilesInPic)) bit.

Giá trị của tile_group_address sẽ nằm trọn trong khoảng từ 0 đến NumTilesInPic − 1.

Else if rect_tile_group_flag bằng 1 và rect_tile_group_info_in_pps bằng 0, điều kiện sau áp dụng:

tile_group_address là chỉ số ô của ô được đặt ở góc trên bên trái của nhóm ô thứ i.

Chiều dài của tile_group_address được báo hiệu_tile_group_index_length_minus1 + 1 bit.

if signalled_tile_group_id_flag bằng 0, giá trị của tile_group_address sẽ nằm trọn trong khoảng từ 0 đến NumTilesInPic − 1. Ngược lại, giá trị của tile_group_address sẽ nằm trọn trong khoảng từ 0 đến $2^{(signalled\_tile\_group\_index\_length\_minus1 + 1)} - 1$.

Else (rect_tile_group_flag bằng 1 và rect_tile_group_info_in_pps bằng 1), điều kiện sau áp dụng:

tile_group_address là ID nhóm ô vuông của nhóm ô vuông.

Chiều dài của tile_group_address được báo hiệu_tile_group_index_length_minus1 + 1 bit.

Nếu signalled_tile_group_id_flag bằng 0, giá trị của tile_group_address sẽ nằm trọn trong khoảng từ 0 đến num_tile_groups_in_pic_minus1. Ngược lại, giá trị của tile_group_address sẽ nằm trọn trong khoảng từ 0 đến $2^{(signalled\_tile\_group\_index\_length\_minus1 + 1)} - 1$.

bottom_right_tile_id xác định chỉ số ô của ô được đặt ở góc dưới bên phải của nhóm ô vuông. Khi single_tile_per_tile_group_flag bằng 1 bottom_right_tile_id được suy ra bằng tile_group_address. Chiều dài của phần tử cú pháp bottom_right_tile_id là Ceil(Log2(NumTilesInPic)) bit.

Biến NumTilesInCurrTileGroup, xác định số lượng ô trong nhóm ô hiện tại, TopLeftTileIdx mà xác định chỉ số ô của ô trên bên trái của nhóm ô vuông, BottomRightTileIdx mà xác định chỉ số ô của ô dưới bên phải của nhóm ô vuông, và TgTileIdx[i], mà xác định chỉ số ô của ô thứ i trong nhóm ô hiện tại, được dẫn xuất như sau:

```
if(rect_tile_group_flag) {
    if (tile_group_info_in_pps) {
        tileGroupIdx = 0
        trong khi(tile_group_address != rect_tile_group_id[tileGroupIdx])
            tileGroupIdx++
        tileIdx = top_left_tile_idx[tileGroupIdx]
        BottomRightTileIdx = bottom_right_tile_idx[tileGroupIdx]
    } else {
        tileIdx = tile_group_address
        BottomRightTileIdx = bottom_right_tile_id
    }
    TopLeftTileIdx = tileIdx
```

deltaTileIdx = BottomRightTileIdx – TopLeftTileIdx

NumTileRowsInTileGroupMinus1 = deltaTileIdx/(num_tile_columns_minus1 + 1) (7-35)

NumTileColumnsInTileGroupMinus1 = deltaTileIdx % (num_tile_columns_minus1 + 1)

*NumTilesInCurrTileGroup = (NumTileRowsInTileGroupMinus1 + 1) * (NumTileColumnsInTileGroupMinus1 + 1)*

```
for(j = 0, tIdx = 0; j < NumTileRowsInTileGroupMinus1 + 1;
                                    j++,    tileIdx    += num_tile_columns_minus1 + 1)
    for(i = 0, currTileIdx = tileIdx; i < NumTileColumnsInTileGroupMinus1 + 1;
                                        i++, currTileIdx++, tIdx++)
                TgTileIdx[tIdx] = currTileIdx
} else {
    NumTilesInCurrTileGroup = num_tiles_in_tile_group_minus1 + 1
    TgTileIdx[0] = tile_group_address
    for(i = 1; i < NumTilesInCurrTileGroup; i++)
        TgTileIdx[i] = TgTileIdx[i − 1] + 1
}
```

recovery_poc_cnt *xác định điểm khôi phục của các ảnh được giải mã theo thứ tự đầu ra. Nếu có ảnh picA tiếp sau ảnh hiện tại (tức là, ảnh GDR) theo thứ tự giải mã trong CVS và có PicOrderCntVal bằng PicOrderCntVal của ảnh hiện tại cộng giá trị của recovery_poc_cnt, ảnh picA được gọi là ảnh điểm khôi phục. Ngược lại, ảnh thứ nhất theo thứ tự đầu ra có PicOrderCntVal lớn hơn PicOrderCntVal của ảnh hiện tại plus giá trị của recovery_poc_cnt được gọi là ảnh điểm khôi phục. ảnh điểm khôi phục sẽ không đứng trước ảnh hiện tại theo thứ tự giải mã. Tất cả các ảnh được giải mã theo thứ tự đầu ra được chỉ báo là đúng hoặc gần đúng trong nội dung bắt đầu từ vị trí thứ tự đầu ra của ảnh điểm khôi phục. Giá trị của recovery_poc_cnt sẽ nằm trọn trong khoảng từ −MaxPicOrderCntLsb/2 đến MaxPicOrderCntLsb/2 − 1.*

Giá trị RecoveryPointPocVal được dẫn xuất như sau:

RecoveryPointPocVal = PicOrderCntVal + recovery_poc_cnt

refreshed_region_flag *bằng 1 xác định rằng giải mã nhóm ô vuông tạo các giá trị mẫu được tái tạo đúng bất kể liệu giá trị của NoIncorrectPicOutputFlag của ảnh GDR liên kết. refreshed_region_flag bằng 0 xác định rằng giải mã nhóm ô vuông có thể tạo các giá trị mẫu được tái tạo không đúng khi bắt đầu từ ảnh GDR liên kết với NoIncorrectPicOutputFlag bằng 1. Khi không có, giá trị của refreshed_region_flag được suy ra bằng 1.*

Lưu ý x – chính ảnh hiện tại có thể là ảnh GDR với NoIncorrectPicOutputFlag bằng 1.

Các đường biên có nhóm ô vuông được làm mới được dẫn xuất như sau:

tileColIdx = TopLeftTileIdx % (num_tile_columns_minus1 + 1)

tileRowIdx = TopLeftTileIdx/(num_tile_columns_minus1 + 1)

TGRefreshedLeftBoundary = ColBd[tileColIdx] << CtbLog2SizeY

TGRefreshedTopBoundary = RowBd[tileRowIdx] << CtbLog2SizeY

tileColIdx = BottomRightTileIdx % (num_tile_columns_minus1 + 1)

tileRowIdx = BottomRightTileIdx/(num_tile_columns_minus1 + 1)

TGRefreshedRightBoundary = ((ColBd[tileColIdx] + ColWidth[tileColIdx]) << CtbLog2SizeY) – 1

TGRefreshedRightBoundary = TGRefreshedRightBoundary > pic_width_in_luma_samples ?

pic_width_in_luma_samples : TGRefreshedRightBoundary

TGRefreshedBotBoundary = ((RowBd[tileRowIdx] + RowHeight[tileRowIdx]) << CtbLog2SizeY) – 1

TGRefreshedBotBoundary = TGRefreshedBotBoundary > pic_height_in_luma_samples ?

pic_height_in_luma_samples : TGRefreshedBotBoundary

Ngữ nghĩa tiêu đề đơn vị NAL.

Bảng 7-1 – Mã loại đơn vị NAL và lớp loại đơn vị NAL

nal_unit_type	**Tên của nal_unit_type**	**Nội dung của cấu trúc cú pháp đơn vị NAL và RBSP**	**Lớp loại đơn vị NAL**
0	TRAIL_NUT	Nhóm ô được mã hóa của ảnh tiếp sau phi STSA tile_group_layer_rbsp()	VCL
1	STSA_NUT	Nhóm ô được mã hóa của ảnh STSA tile_group_layer_rbsp()	VCL
2	RASL_NUT	Nhóm ô được mã hóa của ảnh RASL tile_group_layer_rbsp()	VCL
3	RADL_NUT	Nhóm ô được mã hóa của ảnh RADL tile_group_layer_rbsp()	VCL
4..7	RSV_VCL_4.. RSV_VCL_7	Các loại đơn vị VCL NAL phi IRAP dành riêng	VCL
8 9	IDR_W_RADL IDR_N_LP	Nhóm ô được mã hóa của ảnh IDR tile_group_layer_rbsp()	VCL
10	CRA_NUT	Nhóm ô được mã hóa của ảnh CRA tile_group_layer_rbsp()	VCL
11 12	RSV_IRAP_VCL11 RSV_IRAP_VCL12	Các loại đơn vị VCL NAL IRAP dành riêng	VCL
13	*GDR_NUT*	*Nhóm ô được mã hóa của ảnh GDR tile_group_layer_rbsp()*	*VCL*
14..15	RSV_VCL14.. RSV_VCL15	Các loại đơn vị VCL NAL phi IRAP dành riêng	VCL
16	SPS_NUT	SPS seq_parameter_set_rbsp()	phi VCL
17	PPS_NUT	PPS pic_parameter_set_rbsp()	phi VCL
18	APS_NUT	Tập hợp tham số thích ứng adaptation_parameter_set_rbsp()	phi VCL
19	AUD_NUT	Dấu phân cách AU access_unit_delimiter_rbsp()	phi VCL
20	EOS_NUT	Kết thúc của chuỗi end_of_seq_rbsp()	phi VCL
21	EOB_NUT	Kết thúc của dòng bit end_of_dòng bit_rbsp()	phi VCL
22, 23	PREFIX_SEI_NUT SUFFIX_SEI_NUT	SEI sei_rbsp()	phi VCL
24..27	RSV_NVCL24.. RSV_NVCL27	Các loại đơn vị phi VCL NAL dành riêng	phi VCL
28..31	UNSPEC28.. UNSPEC31	Các loại đơn vị phi VCL NAL không xác định	phi VCL

...

Khi nal_unit_type bằng GDR_NUT, nhóm ô được mã hóa thuộc ảnh GDR, TemporalId phải bằng 0.

Thứ tự của UE và liên kết với các CVS được đề cập.

Dòng bit tuân theo chuẩn đặc tả (tức là, thành phần JVET JVET-M1001-v5) bao gồm một hoặc nhiều CVS.

CVS bao gồm một hoặc nhiều AU. Thứ tự của các đơn vị NAL và các ảnh

được tạo mã và liên kết của nó với các AU được mô tả trong mệnh đề 7.4.2.4.4.

Khối truy nhập thứ nhất của CVS là một trong các khối sau:

– *khối truy nhập IRAP có NoBrokenPictureOutputFlag bằng 1.*

– *khối truy nhập GDR có NoIncorrectPicOutputFlag bằng 1.*

Tương thích dòng bit yêu cầu rằng, khi xuất hiện, AU tiếp theo sau khối truy nhập mà chứa đơn vị NAL EOS hoặc đơn vị NAL EOB sẽ là một trong các khối sau:

– *khối truy nhập IRAP, có thể là khối truy nhập IDR hoặc khối truy nhập CRA.*

– *khối truy nhập GDR.*

8.1.1 Quá trình giải mã cho ảnh được tạo mã được đề cập.

...

Khi ảnh hiện tại là ảnh IRAP, điều kiện sau áp dụng:

– Khi ảnh hiện tại là ảnh IDR, ảnh thứ nhất trong dòng bit theo thứ tự giải mã, hoặc ảnh thứ nhất tiếp sau đơn vị NAL EOS theo thứ tự giải mã, biến *NoIncorrectPicOutputFlag* được gán bằng 1.

– Ngược lại, khi một số phương tiện ngoài không được xác định trong đặc tả này (chẳng hạn, đầu vào người dùng) khả dụng để gán biến HandleCraAsCvsStartFlag bằng giá trị cho ảnh hiện tại, biến HandleCraAsCvsStartFlag được gán bằng giá trị được tạo bởi các phương tiện ngoài và biến *NoIncorrectPicOutputFlag* được gán bằng HandleCraAsCvsStartFlag.

– Ngược lại, biến HandleCraAsCvsStartFlag được gán bằng 0 và biến NoIncorrectPicOutputFlag được gán bằng 0.

Khi ảnh hiện tại là ảnh GDR, điều kiện sau áp dụng:

– *Khi ảnh hiện tại là ảnh GDR, ảnh thứ nhất trong dòng bit theo thứ tự giải mã, hoặc ảnh thứ nhất tiếp sau NAL EOS theo thứ tự giải mã, biến NoIncorrectPicOutputFlag được gán bằng 1.*

– *Ngược lại, khi một số phương tiện ngoài không được xác định theo sáng chế khả dụng để gán biến HandleGdrAsCvsStartFlag bằng giá trị cho ảnh hiện tại, biến HandleGdrAsCvsStartFlag được gán bằng giá trị được tạo bởi phương*

tiện ngoàis và biến NoIncorrectPicOutputFlag được gán bằng HandleGdrAsCvsStartFlag.

– *Ngược lại, biến HandleGdrAsCvsStartFlag được gán bằng 0 và biến NoIncorrectPicOutputFlag được gán bằng 0.*

...

Các quá trình giải mã hoạt động như sau cho ảnh hiện tại CurrPic:

1. Các đơn vị NAL giải mã được xác định trong mệnh đề 8.2.

2. Các quá trình trong mệnh đề 8.3 xác định các quá trình giải mã sau sử dụng các phần tử cú pháp trong lớp tiêu đề nhóm phiến và nêu trên:

– Các biến và các hàm liên quan đến POC được dẫn xuất như được xác định trong mệnh đề 8.3.1. Điều này cần được gọi chỉ cho nhóm ô vuông thứ nhất của ảnh.

– Lúc bắt đầu quá trình giải mã cho mỗi nhóm ô của ảnh phi IDR, quá trình giải mã để tạo các danh sách ảnh tham chiếu được xác định trong mệnh đề 8.3.2 được gọi để dẫn xuất ảnh tham chiếu danh sách 0 (RefPicList[0]) và ảnh tham chiếu danh sách 1 (RefPicList[1]).

– Quá trình giải mã để đánh dấu ảnh tham chiếu trong mệnh đề 8.3.3 được gọi, trong đó các ảnh tham chiếu có thể được đánh dấu là "không được sử dụng để tham chiếu" hoặc "được sử dụng cho tham chiếu dài hạn." Tham số được gọi chỉ cho nhóm ô vuông thứ nhất của ảnh.

– *PicOutputFlag được thiết lập như sau:*

– *Khi một trong các điều kiện sau là đúng, ảnhOutputFlag được gán bằng 0:*

– *ảnh hiện tại là ảnh RASL và NoIncorrectPicOutputFlag của ảnh IRAP được liên kết bằng 1.*

– *gdr_enabled_flag bằng 1 và ảnh hiện tại là ảnh GDR có NoIncorrectPicOutputFlag bằng 1.*

– *gdr_enabled_flag bằng 1 và ảnh hiện tại chứa một hoặc nhiều nhóm ô vuông có refreshed_region_flag bằng 0 và NoBrokenPictureOutputFlag của ảnh GDR liên kết bằng 1.*

– Ngược lại, PicOutputFlag được gán bằng 1.

3. Các quá trình giải mã được gọi cho các CTU, phép chia tỷ lệ, biến đổi, lọc trong vòng, v.v..

4. Sau khi tất cả các nhóm ô của ảnh hiện tại đã được giải mã, ảnh được giải mã hiện tại được đánh dấu là "được sử dụng cho tham chiếu ngắn hạn."

Quá trình giải mã cho POC được đề cập.

Đầu ra của quá trình này là PicOrderCntVal, POC của ảnh hiện tại.

Mỗi ảnh được tạo mã được liên kết với biến POC, được ký hiệu là PicOrderCntVal.

Khi ảnh hiện tại không phải là ảnh IRAP với NoIncorrectPicOutputFlag bằng *1 hoặc ảnh GDR có NoIncorrectPicOutputFlag bằng 1*, các biến prevPicOrderCntLsb và prevPicOrderCntMsb được dẫn xuất như sau:

– Giả sử prevTid0Pic là ảnh đứng trước theo thứ tự giải mã có TemporalId bằng 0 và không phải là ảnh RASL hoặc RADL.

– Biến prevPicOrderCntLsb được gán bằng tile_group_pic_order_cnt_lsb của prevTid0Pic.

– Biến prevPicOrderCntMsb được gán bằng PicOrderCntMsb của prevTid0Pic.

Biến PicOrderCntMsb của ảnh hiện tại được dẫn xuất như sau:

– Khi ảnh hiện tại là ảnh IRAP có *NoIncorrectPicOutputFlag* bằng 1 *hoặc ảnh GDR có NoIncorrectPicOutputFlag bằng 1*, PicOrderCntMsb được gán bằng 0.

– Ngược lại, PicOrderCntMsb được dẫn xuất như sau:

if((tile_group_pic_order_cnt_lsb < prevPicOrderCntLsb) &&
((prevPicOrderCntLsb − tile_group_pic_order_cnt_lsb) >=
(MaxPicOrderCntLsb/2)))

PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb (8-1)

else if((tile_group_pic_order_cnt_lsb > prevPicOrderCntLsb) &&
((tile_group_pic_order_cnt_lsb − prevPicOrderCntLsb) >
(MaxPicOrderCntLsb/2)))

PicOrderCntMsb = prevPicOrderCntMsb − MaxPicOrderCntLsb

else

PicOrderCntMsb = prevPicOrderCntMsb

PicOrderCntVal được dẫn xuất như sau:

PicOrderCntVal = PicOrderCntMsb + tile_group_pic_order_cnt_lsb

(8-2)

Lưu ý 1 – Tất cả các ảnh IRAP có *NoIncorrectPicOutputFlag* bằng 1 sẽ có PicOrderCntVal bằng tile_group_pic_order_cnt_lsb do đối với các ảnh IRAP có NoIncorrectPicOutputFlag bằng 1 PicOrderCntMsb được gán bằng 0.

Lưu ý 1 – Tất cả các ảnh GDR có NoIncorrectPicOutputFlag bằng 1 sẽ có PicOrderCntVal bằng tile_group_pic_order_cnt_lsb do đối với các ảnh GDR có NoIncorrectPicOutputFlag bằng 1 PicOrderCntMsb được gán bằng 0.

Giá trị của PicOrderCntVal sẽ nằm trọn trong khoảng từ -2^{31} đến $2^{31} - 1$.

Khi ảnh hiện tại là ảnh GDR, giá trị của LastGDRPocVal được gán bằng bằng PicOrderCntVal.

Quá trình giải mã đối với vị trí biên có ảnh được làm mới được đề cập.

Quá trình này được gọi chỉ khi gdr_enabled_flag bằng 1.

Quá trình này được gọi sau khi hoàn thành phân tách tiêu đề nhóm phiến.

Đầu ra cùa quá trình này là PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, PicRefreshedTopBoundaryPos, và PicRefreshedBotBoundaryPos, vị trí đường biên của vùng được làm mới của ảnh hiện tại.

Mỗi ảnh được tạo mã được liên kết với tập hợp các biến của các vị trí đường biên vùng được làm mới, được ký hiệu là PicOrderCntVal.

PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, PicRefreshedTopBoundaryPos, và PicRefreshedBotBoundaryPos được dẫn xuất như sau:

Nếu nhóm ô vuông là nhóm ô vuôngđược nhận thứ nhất của ảnh hiện tại có refreshed_region_flag bằng 1, điều kiện sau áp dụng:

PicRefreshedLeftBoundaryPos = TGRefreshedLeftBoundary

PicRefreshedRightBoundaryPos = TGRefreshedRightBoundary

PicRefreshedTopBoundaryPos = TGRefreshedTopBoundary

PicRefreshedBotBoundaryPos = TileGroupBotBoundary

Else if refreshed_region_flag bằng 1, the followings apply:

PicRefreshedLeftBoundaryPos = TGRefreshedLeftBoundary <

PicRefreshedLeftBoundaryPos ?

TGRefreshedLeftBoundary : PicRefreshedLeftBoundaryPos

PicRefreshedRightBoundaryPos = TGRefreshedRightBoundary >

PicRefreshedRightBoundaryPos ?

TGRefreshedRightBoundary : PicRefreshedRightBoundaryPos

PicRefreshedTopBoundaryPos = TGRefreshedTopBoundary <

PicRefreshedTopBoundaryPos ?

TGRefreshedTopBoundary : RefreshedRegionTopBoundaryPos

PicRefreshedBotBoundaryPos = TileGroupBotBoundary >

PicRefreshedBotBoundaryPos ?

TileGroupBotBoundary : PicRefreshedBotBoundaryPos

Quá trình giải mã để tạo các danh sách ảnh tham chiếu được đề cập.

...

Tương thích dòng bit yêu cầu rằng cho mỗi ảnh hiện tại mà không phải là ảnh IRAP có *NoIncorrectPicOutputFlag* bằng 1 hoặc *ảnh GDR có NoIncorrectPicOutputFlag bằng 1*, giá trị của maxPicOrderCnt − minPicOrderCnt phải nhỏ hơn MaxPicOrderCntLsb/2.

...

Quá trình giải mã để đánh dấu ảnh tham chiếu

...

Nếu ảnh hiện tại là ảnh IRAP có NoIncorrectPicOutputFlag bằng 1 *hoặc ảnh GDR có NoIncorrectPicOutputFlag bằng 1*, tất cả các ảnh tham chiếu hiện tại trong DPB (nếu có) được đánh dấu là "không được sử dụng để tham chiếu."

...

Quá trình dẫn xuất cho dự báo vectơ chuyển động độ sáng thời gian được đề cập.

...

Biến currCb xác định CB độ sáng hiện tại ở vị trí độ sáng (xCb, yCb).

Các biến mvLXCol và availableFlagLXCol được dẫn xuất như sau:

– Nếu tile_group_temporal_mvp_enabled_flag bằng 0, cả hai thành phần của mvLXCol được gán bằng 0 và availableFlagLXCol được gán bằng 0.

– Ngược lại (tile_group_temporal_mvp_enabled_flag bằng 1), các bước có thứ tự sau áp dụng:

1. MV cùng vị trí bên phải phía dưới được dẫn xuất như sau:

xColBr = xCb + cbWidth (8-414)

yColBr = yCb + cbHeight (8-415)

leftBoundaryPos = gdr_enabled_flag ?

PicRefreshedLeftBoundaryPos của ảnh được gọi bằng RefPicList[X][refIdxLX] :

0 (8-415)

topBoundaryPos = gdr_enabled_flag ?

PicRefreshedTopBoundaryPos của ảnh được gọi bằng RefPicList[X][refIdxLX] :

0 (8-415)

rightBoundaryPos = gdr_enabled_flag ?

PicRefreshedRightBoundaryPos của ảnh được gọi bằng RefPicList[X][refIdxLX] :

pic_width_in_luma_samples (8-415)

botBoundaryPos = gdr_enabled_flag ?

PicRefreshedBotBoundaryPos của ảnh được gọi bằng RefPicList[X][refIdxLX] :

pic_height_in_luma_samples (8-415)

– Nếu yCb >> CtbLog2SizeY bằng yColBr >> CtbLog2SizeY, yColBr nằm trọn trong khoảng từ topBoundaryPos đến botBoundaryPos, và xColBr nằm trọn trong khoảng từ leftBoundaryPos đến rightBoundaryPos, điều kiện sau áp dụng:

– Biến colCb xác định CB độ sáng bao phủ vị trí được chỉnh sửa thu được bằng ((xColBr >> 3) << 3, (yColBr >> 3) << 3) bên trong ảnh cùng vị trí được xác định bởi ColPic.

– Vị trí độ sáng (xColCb, yColCb) được gán bằng mẫu trên bên trái của CB độ sáng cùng vị trí được xác định bởi colCb so với mẫu độ sáng trên bên trái của ảnh cùng vị trí được xác định bởi ColPic.

– Quá trình dẫn xuất cho các vectơ chuyển động cùng vị trí như được xác định trong mệnh đề 8.5.2.12 được gọi với currCb, colCb, (xColCb, yColCb), refIdxLX và sbFlag được gán bằng 0 dưới dạng các đầu vào, và đầu ra được gán bằng mvLXCol và availableFlagLXCol.

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

2. ...

Quá trình nội suy song tuyến tính mẫu độ sáng được đề cập.

Các đầu vào cho quá trình này là:

– vị trí độ sáng theo các đơn vị mẫu hoàn chỉnh ($xInt_L$, $yInt_L$),

– vị trí độ sáng theo các đơn vị mẫu phân số ($xFrac_L$, $yFrac_L$),

– mảng mẫu tham chiếu độ sáng $refPicLX_L$.

– *các đường biên vùng được làm mới của ảnh tham chiếu PicRefreshedLeftBoundaryPos, PicRefreshedTopBoundaryPos, PicRefreshedRightBoundaryPos, và PicRefreshedBotBoundaryPos.*

...

Các vị trí độ sáng theo các đơn vị mẫu hoàn chỉnh ($xInt_i$, $yInt_i$) được dẫn xuất như sau với i = 0..1:

– *nếu gdr_enabled_flag bằng 1, điều kiện sau áp dụng:*

$xInt_i = Clip3(PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, xInt_L + i)$ *(8-458)*

$yInt_i = Clip3(PicRefreshedTopBoundaryPos, PicRefreshedBotBoundaryPos, yInt_L + i)$ *(8-458)*

– Ngược lại (gdr_enabled_flag bằng 0), điều kiện sau áp dụng:

$xInt_i$ = sps_ref_wraparound_enabled_flag ?

ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW, ($xInt_L$ + i)) : (8-459)

Clip3(0, picW − 1, $xInt_L$ + i)

$yInti = Clip3(0, picH - 1, yInt_L + i)$ (8-460)

...

Quá trình lọc nội suy 8 bậc mẫu độ sáng được đề cập.

Các đầu vào cho quá trình này là:

– vị trí độ sáng theo các đơn vị mẫu hoàn chỉnh ($xInt_L$, $yInt_L$),

– vị trí độ sáng theo các đơn vị mẫu phân số ($xFrac_L$, $yFrac_L$),

– mảng mẫu tham chiếu độ sáng $refPicLX_L$,

– danh sách padVal[dir] với dir = 0,1 xác định hướng và lượng đệm mẫu tham chiếu.

– *các đường biên vùng được làm mới của ảnh tham chiếu PicRefreshedLeftBoundaryPos, PicRefreshedTopBoundaryPos, PicRefreshedRightBoundaryPos, và PicRefreshedBotBoundaryPos.*

...

Các vị trí độ sáng theo các đơn vị mẫu hoàn chỉnh ($xInt_i$, $yInt_i$) được dẫn xuất như sau với i = 0..7:

– *Nếu gdr_enabled_flag bằng 1, điều kiện sau áp dụng:*

$xInt_i = Clip3(PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, xInt_L + i - 3)$ *(8-830)*

$yInt_i = Clip3(PicRefreshedTopBoundaryPos, PicRefreshedBotBoundaryPos, yInt_L + i - 3)$ *(8-830)*

– Ngược lại (gdr_enabled_flag bằng 0), điều kiện sau áp dụng:

$xInt_i$ = sps_ref_wraparound_enabled_flag ?

ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW, $xInt_L + i - 3$) : (8-831)

Clip3(0, picW − 1, $xInt_L + i - 3$)

$yInt_i = Clip3(0, picH - 1, yInt_L + i - 3)$ (8-832)

Quá trình nội suy mẫu sắc độ được đề cập.

Các đầu vào cho quá trình này là:

– vị trí sắc độ theo các đơn vị mẫu hoàn chỉnh ($xInt_C$, $yInt_C$),

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

– tmảng mẫu tham chiếu sắc độ $refPicLX_C$.

– *các đường biên vùng được làm mới của ảnh tham chiếu PicRefreshedLeftBoundaryPos, PicRefreshedTopBoundaryPos, PicRefreshedRightBoundaryPos, và PicRefreshedBotBoundaryPos.*

...

Biến xOffset được gán bằng (sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY)/SubWidthC.

Các vị trí sắc độ theo các đơn vị mẫu hoàn chỉnh ($xInt_i$, $yInt_i$) được dẫn xuất như sau với i = 0..3:

– *Nếu gdr_enabled_flag bằng 1, điều kiện sau áp dụng:*

$xInt_i$ = *Clip3(PicRefreshedLeftBoundaryPos/SubWidthC,*

PicRefreshedRightBoundaryPos/SubWidthC, $xInt_L$ + *i) (8-844)*

yInti = Clip3(PicRefreshedTopBoundaryPos/SubHeightC,

PicRefreshedBotBoundaryPos/SubHeightC, $yInt_L$ + *i) (8-844)*

– Ngược lại (gdr_enabled_flag bằng 0), điều kiện sau áp dụng:

$xInt_i$ = sps_ref_wraparound_enabled_flag ? ClipH(xOffset, $picW_C$, $xInt_C$ + i − 1) : (8-845)

Clip3(0, $picW_C$ − 1, $xInt_C$ + i − 1)

$yInt_i$ = Clip3(0, $picH_C$ − 1, $yInt_C$ + i − 1) (8-846)

Quá trình lọc tách khối được đề cập.

Quá trình tổng quát.

...

Quá trình DBF được áp dụng cho tất cả các mép khối phụ mã và các mép khối biến đổi của ảnh, ngoại trừ các loại mép sau:

– Các mép nằm ở đường biên của ảnh,

– *Các đường mép trùng với đường biên trên của nhóm ô vuông tgA khi tất cả các điều kiện sau được thỏa mãn:*

– *gdr_enabled_flag bằng 1*

– *loop_filter_across_refreshed_region_enabled_flag bằng 0*

– *các đường mép trùng với đường biên dưới của nhóm ô vuông tgB và giá trị của refreshed_region_flag của tgB khác với giá trị của refreshed_region_flag của tgA*

– *Các đường mép trùng với đường biên trái của nhóm ô vuông tgA khi tất cả các điều kiện sau được thỏa mãn:*

– *gdr_enabled_flag bằng 1*

– *loop_filter_across_refreshed_region_enabled_flag bằng 0*

– *các đường mép trùng với đường biên phải của nhóm ô vuông tgB và giá trị của refreshed_region_flag của tgB khác với giá trị của refreshed_region_flag của tgA*

– Các đường mép trùng với các đường biên ô khi loop_filter_across_tiles_enabled_flag bằng 0,

– Các đường mép trùng với các đường biên trên hoặc trái của các nhóm ô có tile_group_loop_filter_across_tile_groups_enabled_flag bằng 0 hoặc tile_group_deblocking_filter_disabled_flag bằng 1,

– Các đường mép trong các nhóm ô có tile_group_deblocking_filter_disabled_flag bằng 1,

– Các đường mép không tương ứng với các đường biên lưới mẫu 8x8 của thành phần được xem xét

– Các đường mép trong các thành phần sắc độ mà cả hai phía của đường mép sử dụng dự báo ngoài,

– Các đường mép của các khối biến đổi sắc độ không phải là các đường mép của TU liên kết.

– Các đường mép giữa các khối biến đổi độ sáng của CU mà có giá trị IntraSubPartitionsSplit không bằng ISP_NO_SPLIT.

Quá trình DBF cho một hướng được đề cập.

...

Đối với mỗi CU có chiều rộng CB log2CbW, chiều cao CB log2CbH và vị trí của mẫu trên bên trái của CB (xCb, yCb), khi edgeType bằng EDGE_VER và xCb % 8 bằng 0 hoặc khi edgeType bằng EDGE_HOR và yCb % 8 bằng 0, các đường mép được lọc bằng các bước có thứ tự sau:

1. Chiều rộng CB nCbW được gán bằng 1 << log2CbW và chiều cao CB nCbH được gán bằng 1 << log2CbH

2. Biến filterEdgeFlag được dẫn xuất như sau:

– Nếu edgeType bằng EDGE_VER và một hoặc nhiều điều kiện sau là đúng, filterEdgeFlag được gán bằng 0:

– Đường biên trái của khối mã hiện tại là đường biên trái của ảnh.

– Đường biên trái của khối mã hiện tại là đường biên trái của ô vuông và loop_filter_across_tiles_enabled_flag bằng 0.

– Đường biên trái của khối mã hiện tại là đường biên trái của nhóm ô vuông và tile_group_loop_filter_across_tile_groups_enabled_flag bằng 0.

– *Đường biên trái của khối mã hiện tại là đường biên trái của nhóm ô vuông hiện tại và tất cả các điều kiện sau được thỏa mãn:*

– *gdr_enabled_flag bằng 1*

– *loop_filter_across_refreshed_region_enabled_flag bằng 0.*

– *nhóm ô vuông đã chia sẻ đường biên với đường biên trái của nhóm ô vuông hiện tại hiện có và giá trị của its refreshed_region_flag khác với giá trị của refreshed_region_flag của nhóm ô vuông hiện tại.*

– Ngược lại nếu edgeType bằng EDGE_HOR và một hoặc nhiều điều kiện sau là đúng, biến filterEdgeFlag được gán bằng 0:

– Đường biên trên của CB độ sáng hiện tại là đường biên trên của ảnh.

– Đường biên trên của khối mã hiện tại là đường biên trên của ô vuông và loop_filter_across_tiles_enabled_flag bằng 0.

– Đường biên trên của khối mã hiện tại là đường biên trên của nhóm ô vuông và tile_group_loop_filter_across_tile_groups_enabled_flag bằng 0.

– *Đường biên trên của khối mã hiện tại là đường biên trên của nhóm ô vuông hiện tại và tất cả các điều kiện sau được thỏa mãn:*

– *gdr_enabled_flag bằng 1*

– *loop_filter_across_refreshed_region_enabled_flag bằng 0.*

– *nhóm ô vuông đã chia sẻ đường biên với đường biên trên của nhóm ô vuông hiện tại hiện có và giá trị của its refreshed_region_flag khác với giá trị của refreshed_region_flag của nhóm ô vuông hiện tại.*

– Ngược lại, filterEdgeFlag được gán bằng 1.

Thích ứng cú pháp khi các ô được tích hợp.

3. Tất cả các phần tử của mảng (nCbW)x(nCbH) hai chiều edgeFlags được

khởi tạo bằng 0.

Quá trình chỉnh sửa CTB cho SAO được đề cập.

...

Đối với tất cả các vị trí mẫu (xS_i, yS_j) và (xY_i, yY_j) có i = 0..nCtbSw − 1 và j = 0..nCtbSh − 1, tùy thuộc vào các giá trị của pcm_loop_filter_disabled_flag, pcm_flag[xY_i][yY_j] và *cu_transquant_bypass_flag* của CU bao gồm CB bao phủ recPicture[xSi][ySj], điều kiện sau áp dụng:

– ....

Thay đổi các phần được in đậm tùy thuộc vào bước đi vòng lượng tử hóa/biến đổi quyết định sau này.

– Ngược lại, nếu SaoTypeIdx[cIdx][rx][ry] bằng 2, các bước có thứ tự sau áp dụng:

1. Các giá trị hPos[k] và vPos[k] với k = 0..1 được xác định trong Bảng 8-18 dựa trên SaoEoClass[cIdx][rx][ry].
2. Biến edgeIdx được dẫn xuất như sau:

– Các vị trí mẫu ($xS_{ik'}$, $yS_{jk'}$) và ($xY_{ik'}$, $yY_{jk'}$) được chỉnh sửa được dẫn xuất như sau:

($xS_{ik'}$, $yS_{jk'}$) = (xS_i + hPos[k], yS_j + vPos[k]) (8-1128)

($xY_{ik'}$, $yY_{jk'}$) = (cIdx == 0) ? ($xS_{ik'}$, $yS_{jk'}$) : ($xS_{ik'}$ * SubWidthC, $yS_{jk'}$ * SubHeightC) (8-1129)

– Nếu một hoặc nhiều điều kiện sau với tất cả các vị trí mẫu ($xS_{ik'}$, $yS_{jk'}$) và ($xY_{ik'}$, $yY_{jk'}$) với k = 0..1 là đúng, edgeIdx được gán bằng 0:

– Mẫu ở vị trí ($xS_{ik'}$, $yS_{jk'}$) nằm ngoài các đường biên ảnh.

– *gdr_enabled_flag bằng 0, loop_filter_across_refreshed_region_enabled_flag bằng 0, refreshed_region_flag của nhóm ô vuông hiện tại bằng 1, và refreshed_region_flag của nhóm ô vuông chứa mẫu ở vị trí ($xS_{ik'}$, $yS_{jk'}$) bằng 0.*

– Mẫu ở vị trí ($xS_{ik'}$, $yS_{jk'}$) thuộc nhóm ô vuông khác nhau và một trong hai điều kiện sau là đúng:

– MinTbAddrZs[$xY_{ik'}$ >> MinTbLog2SizeY][$yY_{jk'}$ >> MinTbLog2SizeY] nhỏ hơn MinTbAddrZs[xY_i >> MinTbLog2SizeY][yY_j >>

MinTbLog2SizeY] và tile_group_loop_filter_across_tile_groups_enabled_flag trong nhóm ô vuông mà mẫu recPicture[xS_i][yS_j] thuộc bằng 0.

– MinTbAddrZs[xY_i >> MinTbLog2SizeY][yY_j >> MinTbLog2SizeY] nhỏ hơn MinTbAddrZs[$xY_{ik'}$ >> MinTbLog2SizeY][$yY_{jk'}$ >> MinTbLog2SizeY] và tile_group_loop_filter_across_tile_groups_enabled_flag trong nhóm ô vuông mà mẫu recPicture[$xS_{ik'}$][$yS_{jk'}$] thuộc bằng 0.

– loop_filter_across_tiles_enabled_flag bằng 0 và mẫu ở vị trí ($xS_{ik'}$, $yS_{jk'}$) thuộc ô khác.

Chỉnh sửa các phần được in đậm khi các ô mà không có các nhóm ô được bao gồm

– Ngược lại, edgeIdx được dẫn xuất như sau:

– Điều kiện sau áp dụng:

edgeIdx = 2 + Sign(recPicture[xS_i][yS_j] − recPicture[xS_i + hPos[0]][yS_j + vPos[0]]) +

Sign(recPicture[xS_i][yS_j] − recPicture[xS_i + hPos[1]][yS_j + vPos[1]])(8-1130)

– Khi edgeIdx bằng 0, 1, hoặc 2, edgeIdx được chỉnh sửa như sau:

edgeIdx = (edgeIdx == 2) ? 0 : (edgeIdx + 1) (8-1131)

3. Mảng mẫu ảnh được chỉnh sửa saoPicture[xS_i][yS_j] được dẫn xuất như sau:

saoPicture[xS_i][yS_j] = Clip3(0, (1 << bitDepth) − 1, recPicture[xS_i][yS_j] + SaoOffsetVal[cIdx][rx][ry][edgeIdx]) (8-1132)

Quá trình lọc CTB cho các mẫu độ sáng cho ALF được đề cập.

...

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

– ...

– Các vị trí (h_x, v_y) cho mỗi mẫu của các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng được dẫn xuất như sau:

– *Nếu gdr_enabled_flag bằng 1,*

loop_filter_across_refreshed_region_enabled_flag bằng 0, refreshed_region_flag của nhóm ô vuông tgA chứa mẫu độ sáng ở vị trí (x, y) bằng 1, điều kiện sau áp dụng:

– Nếu vị trí (h_x, v_y) được đặt trong nhóm ô vuông tgB khác và refreshed_region_flag của tgB bằng 0, các biến leftBoundary, rightBoundary, topBoundary, và botBoundary được gán lần lượt bằng TGRefreshedLeftBoundary, TGRefreshedRightBoundary, TGRefreshedTopBoundary, và TGRefreshedBotBoundary.

– Ngược lại, các biến leftBoundary, rightBoundary, topBoundary, và botBoundary được gán lần lượt bằng PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, PicRefreshedTopBoundaryPos, và PicRefreshedBotBoundaryPos.

h_x = Clip3(leftBoundary, rightBoundary, xCtb + x) (8-1140)

v_y = Clip3(topBoundary, botBoundary, yCtb + y) (8-1141)

– Ngược lại, điều kiện sau áp dụng:

h_x = Clip3(0, pic_width_in_luma_samples − 1, xCtb + x) (8-1140)

v_y = Clip3(0, pic_height_in_luma_samples − 1, yCtb + y) (8-1141)

– ...

Quá trình dẫn xuất đối với chỉ số lọc và chuyển vị ALF cho các mẫu độ sáng được đề cập.

...

Các vị trí (h_x, v_y) cho mỗi mẫu của các mẫu độ sáng tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu độ sáng được dẫn xuất như sau:

– Nếu gdr_enabled_flag bằng 1, loop_filter_across_refreshed_region_enabled_flag bằng 0, refreshed_region_flag của nhóm ô vuông tgA chứa the mẫu độ sáng ở vị trí (x, y) bằng 1, điều kiện sau áp dụng:

– Nếu vị trí (h_x, v_y) được đặt trong nhóm ô vuông tgB khác và refreshed_region_flag of tgB bằng 0, các biến leftBoundary, rightBoundary, topBoundary, và botBoundary được gán lần lượt bằng TGRefreshedLeftBoundary, TGRefreshedRightBoundary,

TGRefreshedTopBoundary, và TGRefreshedBotBoundary.

– Ngược lại, các biến leftBoundary, rightBoundary, topBoundary, và botBoundary được gán lần lượt bằng PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, PicRefreshedTopBoundaryPos, và PicRefreshedBotBoundaryPos.

h_x = *Clip3(leftBoundary, rightBoundary, x)* *(8-1140)*

v_y = *Clip3(topBoundary, botBoundary, y)* *(8-1141)*

– Ngược lại, điều kiện sau áp dụng:

h_x = Clip3(0, pic_width_in_luma_samples − 1, x) (8-1145)

v_y = Clip3(0, pic_height_in_luma_samples − 1, y) (8-1146)

Quá trình lọc CTB cho các mẫu sắc độ được đề cập.

...

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

– Các vị trí (h_x, v_y) cho mỗi mẫu trong các mẫu sắc độ tương ứng (x, y) bên trong mảng cụ thể recPicture của các mẫu sắc độ được dẫn xuất như sau:

– Nếu gdr_enabled_flag bằng 1, loop_filter_across_refreshed_region_enabled_flag bằng 0, refreshed_region_flag của nhóm ô vuông tgA chứa mẫu độ sáng ở vị trí (x, y) bằng 1, điều kiện sau áp dụng:

– Nếu vị trí (h_x, v_y) được đặt trong nhóm ô vuông tgB khác và refreshed_region_flag của tgB bằng 0, các biến leftBoundary, rightBoundary, topBoundary, và botBoundary được gán lần lượt bằng TGRefreshedLeftBoundary, TGRefreshedRightBoundary, TGRefreshedTopBoundary, và TGRefreshedBotBoundary.

– Ngược lại, các biến leftBoundary, rightBoundary, topBoundary, và botBoundary được gán lần lượt bằng PicRefreshedLeftBoundaryPos, PicRefreshedRightBoundaryPos, PicRefreshedTopBoundaryPos, và PicRefreshedBotBoundaryPos.

h_x = *Clip3(leftBoundary/SubWidthC , rightBoundary/SubWidthC , xCtbC +*

x) *(8-1140)*

$v_y = Clip3(topBoundary/SubWidthC , botBoundary/SubWidthC , yCtbC + y)$

(8-1141)

– Ngược lại, điều kiện sau áp dụng:

$h_x = Clip3(0, pic\_width\_in\_luma\_samples/SubWidthC - 1, xCtbC + x)$ (8-1177)

$v_y = Clip3(0, pic\_height\_in\_luma\_samples/SubHeightC - 1, yCtbC + y)$ (8-1178)

Fig.7 minh họa dòng bit viđeo 750 được tạo cấu hình để thực hiện kỹ thuật GDR 700 theo phương án thực hiện sáng chế. Kỹ thuật GDR 700 có thể giống như kỹ thuật GDR 500 trên Fig.5. Như được sử dụng ở đây, video dòng bit 750 có thể cũng được gọi là dòng bit viđeo được mã hóa, dòng bit, hoặc các biến thể của nó. Như được thể hiện trên Fig.7, dòng bit 750 bao gồm SPS 752, PPS 754, tiêu đề lát 756, và dữ liệu ảnh 758.

SPS 752 chứa dữ liệu chung cho tất cả các ảnh trong SOP. Ngược lại, PPS 754 chứa dữ liệu chung cho toàn bộ ảnh. Tiêu đề lát 756 chứa thông tin về lát hiện tại chẳng hạn, chẳng hạn, loại lát, ảnh nào trong các ảnh tham chiếu sẽ được sử dụng, và v.v.. SPS 752 và PPS 754 có thể được gọi chung là tập hợp tham số. SPS 752, PPS 754, và tiêu đề lát 756 là các loại đơn vị NAL. Đơn vị NAL là cấu trúc cú pháp chứa chỉ báo loại dữ liệu tiếp theo (chẳng hạn, dữ liệu viđeo được tạo mã). Các đơn vị NAL được phân loại thành các đơn vị NAL VCL và phi VCL. Các đơn vị VCL NAL chứa dữ liệu vốn là các giá trị của các mẫu trong các ảnh viđeo, và các đơn vị phi VCL NAL chứa thông tin bổ sung liên kết bất kỳ, chẳng hạn, các tập hợp tham số (dữ liệu tiêu đề quan trọng có thể áp dụng cho số lượng lớn đơn vị VCL NAL) và SEI (thông tin thời gian và dữ liệu bổ sung khác mà có thể tăng cường khả năng sử dụng của tín hiệu viđeo được giải mã mà không cần để giải mã các giá trị của các mẫu trong các ảnh viđeo). Người có hiểu biết trung bình trong lĩnh vực sẽ hiểu rằng dòng bit 750 có thể chứa các tham số khác và thông tin trong các ứng dụng thực.

Dữ liệu ảnh 758 trên Fig.7 bao gồm dữ liệu được liên kết với các ảnh hoặc viđeo được mã hóa hoặc được giải mã. Dữ liệu ảnh 758 có thể được gọi đơn giản

là tải tin hoặc dữ liệu được mang trong dòng bit 750. Theo phương án thực hiện, dữ liệu ảnh 758 bao gồm CVS 708 chứa ảnh GDR 702, một hoặc nhiều các ảnh tiếp sau 704, và ảnh điểm khôi phục 706. Theo phương án thực hiện, ảnh GDR 702, các ảnh tiếp sau 704, và ảnh điểm khôi phục 706 có thể định nghĩa chu kỳ GDR trong CVS 708.

Như được thể hiện trên Fig.7, kỹ thuật GDR 700 hoặc nguyên lý hoạt động trên chuỗi ảnh bắt đầu với ảnh GDR 702 và kết thúc với ảnh điểm khôi phục 706. Ảnh GDR 702 chứa vùng được làm mới/sạch 710 chứa các khối đều được mã hóa sử dụng dự báo trong (tức là, các khối được dự báo trong) vùng bẩn/không được làm mới 712 chứa các khối đều đã được mã hóa sử dụng dự báo ngoài (tức là, các khối được dự báo ngoài).

Ảnh tiếp sau 704 đứng ngay gần ảnh GDR 702 chứa vùng được làm mới/sạch 710 có phần thứ nhất 710A được tạo mã nhờ sử dụng dự báo trong và phần thứ hai 710B được tạo mã nhờ sử dụng dự báo ngoài. Phần thứ hai 710B được tạo mã nhờ tạo tham chiếu vùng được làm mới/sạch 710 của, chẳng hạn, ảnh đứng trước trong chu kỳ GDR của CVS 708. Như được thể hiện, vùng được làm mới/sạch 710 của các ảnh tiếp sau 704 mở rộng khi quá trình tạo mã di chuyển hoặc tiến triển theo hướng nhất quán (chẳng hạn, từ trái sang phải), một cách tương ứng co lại vùng bẩn/không được làm mới 712. Dần dần, ảnh điểm khôi phục 706, chỉ chứa vùng được làm mới/sạch 710, thu được từ quá trình tạo mã. Phần thứ hai 710B của vùng được làm mới/sạch 710, mà được mã hóa dưới dạng các khối được dự báo ngoài, có thể chỉ tham chiếu đến vùng được làm mới/vùng sạch 710 trong ảnh tham chiếu.

Như được thể hiện trên Fig.7, mỗi ảnh trong ảnh GDR 702, các ảnh tiếp sau 704, và ảnh điểm khôi phục 706 trong CVS 708 được chứa trong đơn vị NAL VCL 730 của nó. Các tập hợp đơn vị VCL NAL 730 trong CVS 708 có thể được gọi là khối truy nhập.

Đơn vị NAL 730 chứa ảnh GDR 702 trong CVS 708 có GDR_NUT. Tức là, theo phương án thực hiện, đơn vị NAL 730 chứa ảnh GDR 702 trong CVS 708 có loại đơn vị NAL duy nhất so với các ảnh tiếp sau 704 và ảnh điểm khôi phục 706. Theo phương án thực hiện, GDR_NUT cho phép dòng bit bắt đầu với ảnh

GDR 702 thay cho dòng bit có bắt đầu với ảnh IRAP. Việc chỉ định đơn VCL NAL 730 của ảnh GDR 702 làm GDR_NUT có thể chỉ báo cho, chẳng hạn, bộ giải mã rằng đơn vị VCL NAL 730 ban đầu trong CVS 708 chứa ảnh GDR 702.

Theo phương án thực hiện, ảnh GDR 702 là ảnh ban đầu trong CVS 708. Theo phương án thực hiện, ảnh GDR 702 là ảnh ban đầu trong chu kỳ GDR. Theo phương án thực hiện, ảnh GDR 702 có ID thời gian bằng 0. ID thời gian là giá trị hoặc số nhận diện vị trí hoặc thứ tự của ảnh so với các ảnh khác. Theo phương án thực hiện, khối truy nhập chứa đơn VCL NAL 730 có GDR_NUT được chọn là khối truy nhập GDR. Theo phương án thực hiện, ảnh GDR 702 là lát được mã hóa của ảnh GDR khác (chẳng hạn, lớn hơn). Tức là, ảnh GDR 702 có thể là phần của ảnh GDR lớn hơn.

Theo phương án thực hiện, Fig.8 là phương pháp 800 giải mã dòng bit viđeo được mã hóa được triển khai bằng bộ giải mã viđeo (chẳng hạn, bộ giải mã viđeo 30). Phương pháp 800 có thể được thực hiện sau khi dòng bit được giải mã đã được nhận gián tiếp hoặc trực tiếp từ bộ mã hóa viđeo (chẳng hạn, bộ mã hóa viđeo 20). Phương pháp 800 cải thiện quá trình giải mã do phương pháp cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, trên thực tế, hiệu năng của codec được cải thiện, làm làm cho trải nghiệm người dùng tốt hơn.

Trong khối 802, bộ giải mã viđeo xác định rằng CVS của dòng bit viđeo được mã hóa bao gồm đơn vị VCL NAL có GDR_NUT, đơn VCL NAL có GDR_NUT chứa ảnh GDR.

Theo phương án thực hiện, ảnh GDR là ảnh ban đầu trong CVS. Theo phương án thực hiện, ảnh GDR là ảnh ban đầu trong chu kỳ GDR. Theo phương án thực hiện, ảnh GDR có ID thời gian bằng 0. ID thời gian 0 có thể chỉ báo rằng ảnh GDR là ảnh thứ nhất trong, chẳng hạn, CVS hoặc chu kỳ GDR. Theo phương án thực hiện, khối truy nhập chứa đơn VCL NAL có GDR_NUT được chọn là khối truy nhập GDR. Theo phương án thực hiện, ảnh GDR là lát được mã hóa của ảnh

GDR khác.

Trong khối 804, bộ giải mã viđeo khởi tạo giải mã CVS ở ảnh GDR.

Trong khối 806, bộ giải mã viđeo tạo ảnh theo CVS như được giải mã. Sau đó, ảnh có thể được hiển thị cho người dùng của thiết bị điện tử (chẳng hạn, điện thoại thông minh, tablet, máy tính xách tay, máy tính cá nhân, v.v.).

Theo phương án thực hiện, Fig.9 là phương pháp 900 mã hóa dòng bit viđeo được triển khai bằng bộ mã hóa viđeo (chẳng hạn, bộ mã hóa viđeo 20). Phương pháp 900 có thể được thực hiện khi ảnh (chẳng hạn, từ viđeo) cần được mã hóa thành dòng bit viđeo và sau đó được truyền về phía bộ giải mã viđeo (chẳng hạn, bộ giải mã viđeo 30). Phương pháp 900 cải thiện quá trình mã hóa do phương pháp cho phép làm mới nội ảnh từng bước để có thể truy nhập ngẫu nhiên mà không phải sử dụng ảnh IRAP. Bằng cách sử dụng ảnh GDR thay cho ảnh IRAP, tốc độ bit mượt hơn, nhất quán hơn có thể đạt được do, chẳng hạn, kích thước của ảnh GDR so với kích thước của ảnh IRAP, cho phép độ trễ giảm từ đầu này đến đầu kia (tức là, độ trễ). Do vậy, trên thực tế, hiệu năng của codec được cải thiện, làm cho trải nghiệm người dùng tốt hơn.

Trong khối 902, bộ mã hóa viđeo xác định điểm truy nhập ngẫu nhiên cho chuỗi viđeo. Trong khối 904, bộ mã hóa viđeo mã hóa ảnh GDR thành đơn vị VCL NAL có GDR_NUT ở điểm truy nhập ngẫu nhiên cho chuỗi viđeo.

Theo phương án thực hiện, ảnh GDR là ảnh ban đầu trong CVS. Theo phương án thực hiện, ảnh GDR là ảnh ban đầu trong chu kỳ GDR. Theo phương án thực hiện, ảnh GDR có ID thời gian bằng 0. ID thời gian 0 có thể chỉ báo rằng ảnh GDR là ảnh thứ nhất trong, chẳng hạn, CVS hoặc chu kỳ GDR. Theo phương án thực hiện, khối truy nhập chứa đơn VCL NAL có GDR_NUT được chọn là khối truy nhập GDR. Theo phương án thực hiện, ảnh GDR là lát được mã hóa của ảnh GDR khác.

Trong khối 906, bộ mã hóa viđeo tạo dòng bit chứa chuỗi viđeo có ảnh GDR trong đơn VCL NAL có GDR_NUT ở điểm truy nhập ngẫu nhiên.

Trong khối 908, dòng bit được lưu trữ để truyền về phía bộ giải mã viđeo. Dòng bit viđeo có thể cũng được gọi là dòng bit viđeo được mã hóa hoặc dòng bit viđeo được mã hóa. Bộ mã hóa viđeo có thể truyền dòng bit về phía bộ giải

mã viđeo. Khi được nhận bằng bộ giải mã viđeo, viđeo được mã hóa dòng bit có thể được giải mã (chẳng hạn, như được mô tả trên đây) để tạo ảnh để hiển thị đến người dùng trên màn hình của thiết bị điện tử (chẳng hạn, điện thoại thông minh, máy tính bảng, máy tính xách tay, máy tính cá nhân, v.v.).

Fig.10 là sơ đồ của thiết bị tạo mã viđeo 1000 (chẳng hạn, bộ mã hóa viđeo 20 hoặc bộ giải mã viđeo 30) theo phương án thực hiện sáng chế. Thiết bị tạo mã viđeo 1000 thích hợp để thực hiện các phương án thực hiện được bộc lộ như được mô tả ở đây. Thiết bị tạo mã viđeo 1000 bao gồm các cổng vào 1010 và các bộ nhận (Rx) 1020 để nhận dữ liệu; bộ xử lý, đơn vị lôgic, hoặc CPU 1030 để xử lý dữ liệu; các bộ truyền (Tx) 1040 và các cổng ra 1050 để truyền dữ liệu; và bộ nhớ 1060 để lưu trữ dữ liệu. Thiết bị tạo mã viđeo 1000 có thể cũng bao gồm các thành phần quang sang điện (optical-to-electrical, OE) và các thành phần điện sang quang (electrical-to-optical, EO) được ghép nối với các cổng vào 1010, các bộ nhận 1020, các bộ truyền 1040, và các cổng ra 1050 để xuất hoặc nhập tín hiệu quang hoặc điện.

Bộ xử lý 1030 được triển khai bằng phần cứng và phần mềm. Bộ xử lý 1030 có thể được triển khai dưới dạng một hoặc nhiều chip CPU, lõi (chẳng hạn, dưới dạng bộ xử lý nhiều lõi), FPGA, ASIC, và DSP. Bộ xử lý 1030 truyền thông với các cổng vào 1010, các bộ nhận 1020, các bộ truyền 1040, các cổng ra 1050, và bộ nhớ 1060. Bộ xử lý 1030 bao gồm môđun mã hóa 1070. Môđun mã hóa 1070 triển khai các phương án thực hiện được bộc lộ nêu trên. Chẳng hạn, môđun mã hóa 1070 triển khai, xử lý, chuẩn bị, hoặc tiến hành các chức năng codec khác nhau. Do vậy, việc đưa vào môđun mã hóa 1070 cải thiện đáng kể với chức năng của thiết bị tạo mã viđeo 1000 và biến đổi thiết bị tạo mã viđeo 1000 sang trạng thái khác. Theo cách khác, môđun mã hóa 1070 được triển khai dưới dạng các lệnh được lưu trữ trong bộ nhớ 1060 và được thực thi bằng bộ xử lý 1030.

Thiết bị tạo mã viđeo 1000 có thể cũng bao gồm các bộ phận nhập và/hoặc xuất (Input/Output, I/O) 1080 để truyền thông dữ liệu đến và từ người dùng. Các bộ phận I/O 1080 có thể bao gồm các bộ phận đầu ra, chẳng hạn, bộ phận hiển thị để hiển thị dữ liệu viđeo, loa để xuất ra dữ liệu audio, v.v.. Các bộ phận I/O 1080 có thể cũng bao gồm các bộ phận đầu vào, chẳng hạn bàn phím, chuột, bi

xoay, v.v., và/hoặc các giao diện tương ứng để tương tác với các bộ phận đầu ra này.

Bộ nhớ 1060 bao gồm một hoặc nhiều đĩa, ổ băng, và ổ trạng thái rắn (solid-state drive, SSD) và có thể được sử dụng như là bộ phận lưu trữ dữ liệu tràn, để lưu trữ các chương trình khi các chương trình này được chọn để thực thi, và lưu trữ các lệnh và dữ liệu được đọc trong khi thực thi chương trình. Bộ nhớ 1060 có thể là khả biến và/hoặc bất biến và có thể là bộ nhớ chỉ đọc (read-only memory, ROM), bộ nhớ truy nhập ngẫu nhiên (random access memory, RAM), bộ nhớ có nội dung đánh địa chỉ được (ternary content-addressable memory, TCAM), và/hoặc RAM tĩnh (static random-access memory, SRAM).

Theo phương án thực hiện, Fig.11 là sơ đồ của phương tiện mã hóa 1100. Theo phương án thực hiện, phương tiện mã hóa 1100 được triển khai trong thiết bị tạo mã viđeo 1102 (chẳng hạn, bộ mã hóa viđeo 20 hoặc bộ giải mã viđeo 30). Thiết bị tạo mã viđeo 1102 bao gồm phương tiện nhận 1101. Phương tiện nhận 1101 được tạo cấu hình để nhận ảnh để mã hóa hoặc để nhận dòng bit để giải mã. Thiết bị tạo mã viđeo 1102 bao gồm phương tiện truyền 1107 được ghép nối với phương tiện nhận 1101. Phương tiện truyền 1107 được tạo cấu hình để truyền dòng bit đến bộ giải mã hoặc truyền ảnh được giải mã đến phương tiện hiển thị (chẳng hạn, một trong các bộ phận I/O 1080).

Thiết bị tạo mã viđeo 1102 bao gồm phương tiện lưu trữ 1103. Phương tiện lưu trữ 1103 được ghép nối với ít nhất một trong phương tiện nhận 1101 hoặc phương tiện truyền 1107. Phương tiện lưu trữ 1103 được tạo cấu hình để lưu trữ các lệnh. Thiết bị tạo mã viđeo 1102 cũng bao gồm phương tiện xử lý 1105. Phương tiện xử lý 1105 được ghép nối với phương tiện lưu trữ 1103. Phương tiện xử lý 1105 được tạo cấu hình để thực thi các lệnh được lưu trữ trong phương tiện lưu trữ 1103 để thực hiện các phương pháp được bộc lộ ở đây.

Cũng cần hiểu rằng các bước của các phương pháp lấy làm ví dụ được nêu ở đây không nhất thiết được yêu cầu thực hiện theo thứ tự được mô tả, và thứ tự của các bước của các phương pháp này nên được hiểu chỉ là ví dụ. Tương tự, các bước bổ sung có thể được bao gồm trong các phương pháp này, và các bước cụ thể có thể được bỏ qua hoặc kết hợp, ở các phương pháp nhất quán với các

phương án thực hiện khác nhau của sáng chế.

Trong khi vài phương án thực hiện đã được nêu theo sáng chế, cần hiểu rằng các hệ thống và các phương pháp được bộc lộ có thể được triển khai ở nhiều dạng cụ thể khác mà không xa rời nguyên lý của sáng chế. Các ví dụ hiện tại được xem là minh họa và không giới hạn, và ý định không bị giới hạn ở các chi tiết được nêu ở đây. Chẳng hạn, các phần tử hoặc các thành phần khác có thể được kết hợp hoặc tích hợp trong hệ thống khác hoặc các đặc điểm cụ thể có thể bị bỏ qua hoặc không được triển khai.

Ngoài ra, các kỹ thuật, các hệ thống, các hệ thống phụ, và các phương pháp được mô tả và được minh họa trên các phương án thực hiện khác nhau dưới dạng rời rạc hoặc riêng rẽ có thể được kết hợp hoặc tích hợp với các hệ thống, các môđun, các kỹ thuật, hoặc các phương pháp khác mà không xa rời phạm vi của sáng chế. Các mục khác được thể hiện hoặc được đề cập như là được ghép nối hoặc được ghép nối trực tiếp hoặc truyền thông với nhau có thể được ghép nối gián tiếp hoặc truyền thông qua một số giao diện, thiết bị, hoặc thành phần trung gian bất kể dạng điện, cơ, hoặc ngược lại. Các ví dụ khác về các thay đổi, các thay thế, và các chỉnh sửa được chấp nhận bởi người có kiến thức trung bình trong lĩnh vực và có thể được thực hiện mà không xa rời phạm vi và nguyên lý được bộc lộ ở đây.

YÊU CẦU BẢO HỘ

1. Phương pháp được triển khai bằng bộ giải mã, trong đó phương pháp bao gồm các bước:

nhận dòng bit bao gồm chuỗi viđeo được tạo mã (coded video sequence, CVS) bao gồm đơn vị lớp trừu tượng mạng (network abstraction layer, NAL) của lớp tạo mã viđeo (video coding layer, VCL);

xác định rằng đơn vị VCL NAL có loại đơn vị lớp trừu tượng mạng (network abstraction layer, NAL) làm mới giải mã dần dần (gradual decoding refresh, GDR) (GDR_NUT), trong đó đơn vị truy nhập GDR tương ứng với đơn vị VCL NAL là đơn vị truy nhập bắt đầu CVS; và

giải mã ảnh GDR tương ứng với GDR_NUT.

2. Phương pháp theo điểm 1, trong đó ảnh GDR là ảnh thứ nhất trong CVS.

3. Phương pháp theo điểm 1, trong đó ảnh GDR là ảnh ban đầu trong chu kỳ GDR.

4. Phương pháp theo điểm 1, trong đó ảnh GDR có bộ nhận dạng thời gian (identifier, ID) bằng 0.

5. Phương pháp theo điểm 1, trong đó GDR_NUT chỉ báo cho bộ giải mã rằng đơn vị VCL NAL có GDR_NUT chứa lát của ảnh GDR.

6. Phương pháp mã hóa dòng bit viđeo được triển khai bằng bộ mã hóa viđeo, trong đó phương pháp bao gồm các bước:

xác định điểm truy nhập ngẫu nhiên cho chuỗi viđeo;

mã hóa ảnh GDR thành đơn vị VCL NAL ở điểm truy nhập ngẫu nhiên đối với chuỗi viđeo;

tạo dòng bit chứa chuỗi viđeo có ảnh GDR ở điểm truy nhập ngẫu nhiên, trong đó đơn vị VCL NAL có loại đơn vị GDR NAL (GDR-NUT), trong đó đơn vị truy nhập GDR tương ứng với đơn vị VCL NAL là đơn vị truy nhập bắt đầu CVS; và

lưu trữ dòng bit để truyền về phía bộ giải mã viđeo.

7. Phương pháp theo điểm 6, trong đó ảnh GDR là ảnh ban đầu trong chuỗi viđeo.

8. Phương pháp theo điểm 6, trong đó ảnh GDR là ảnh ban đầu trong chu kỳ GDR.

9. Phương pháp theo điểm 6, trong đó ảnh GDR có ID thời gian bằng 0.

10. Phương pháp theo điểm 6, trong đó GDR_NUT chỉ báo cho bộ giải mã viđeo rằng đơn vị VCL NAL có GDR_NUT chứa lát của ảnh GDR.

11. Bộ giải mã viđeo bao gồm:

bộ nhận được tạo cấu hình để nhận dòng bit bao gồm CVS bao gồm đơn vị VCL NAL; và

bộ xử lý được ghép nối với bộ nhận, bộ xử lý được tạo cấu hình để:

xác định rằng đơn vị VCL NAL có loại đơn vị GDR NAL (GDR-NUT), trong đó đơn vị truy nhập GDR tương ứng với đơn vị VCL NAL là đơn vị truy nhập bắt đầu CVS; và

giải mã ảnh GDR tương ứng với GDR-NUT.

12. Bộ giải mã viđeo theo điểm 11, trong đó ảnh GDR là ảnh thứ nhất trong CVS.

13. Bộ giải mã viđeo theo điểm 11, trong đó ảnh GDR là ảnh ban đầu trong chu kỳ GDR.

14. Bộ giải mã viđeo theo điểm 11, trong đó ảnh GDR có ID thời gian bằng 0.

15. Bộ giải mã viđeo theo điểm 11, trong đó GDR_NUT chỉ báo cho bộ giải mã viđeo rằng đơn vị VCL NAL có GDR_NUT chứa lát của ảnh GDR.

1/10

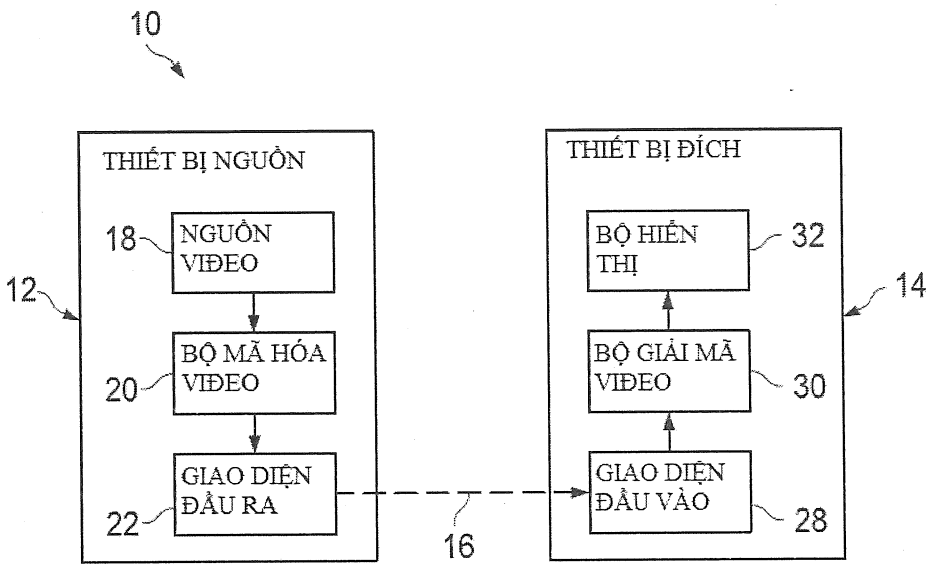

Fig.1

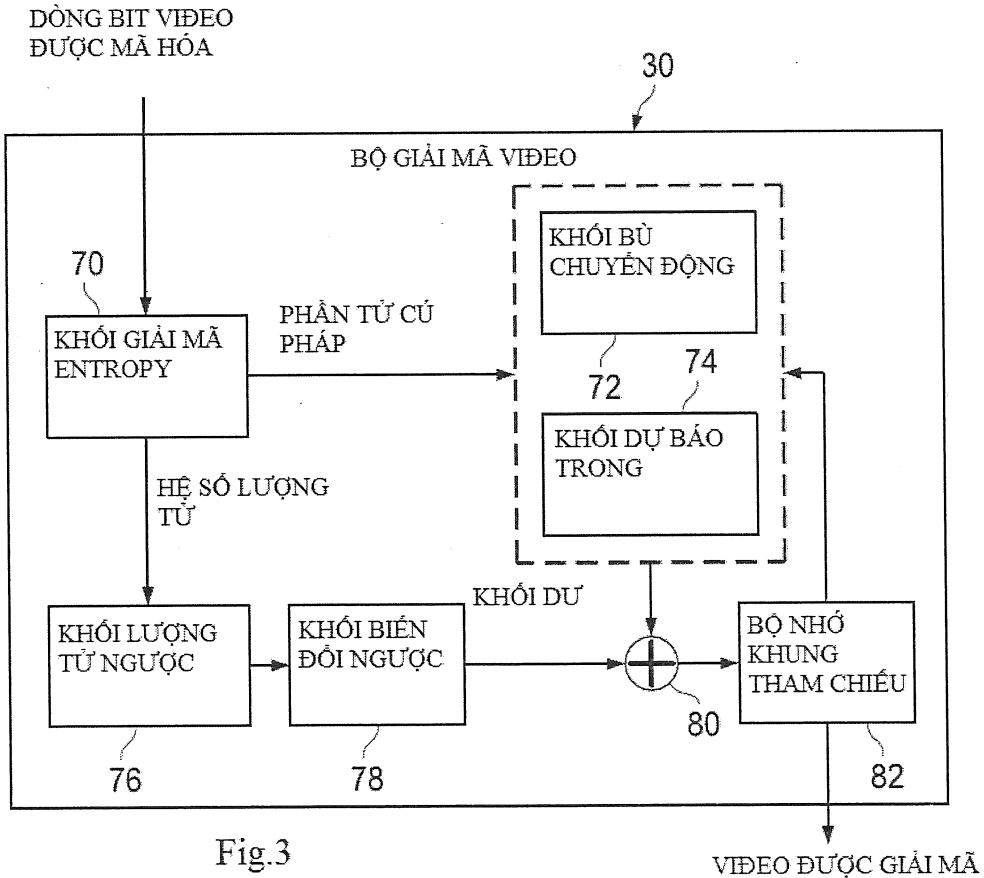

Fig.3

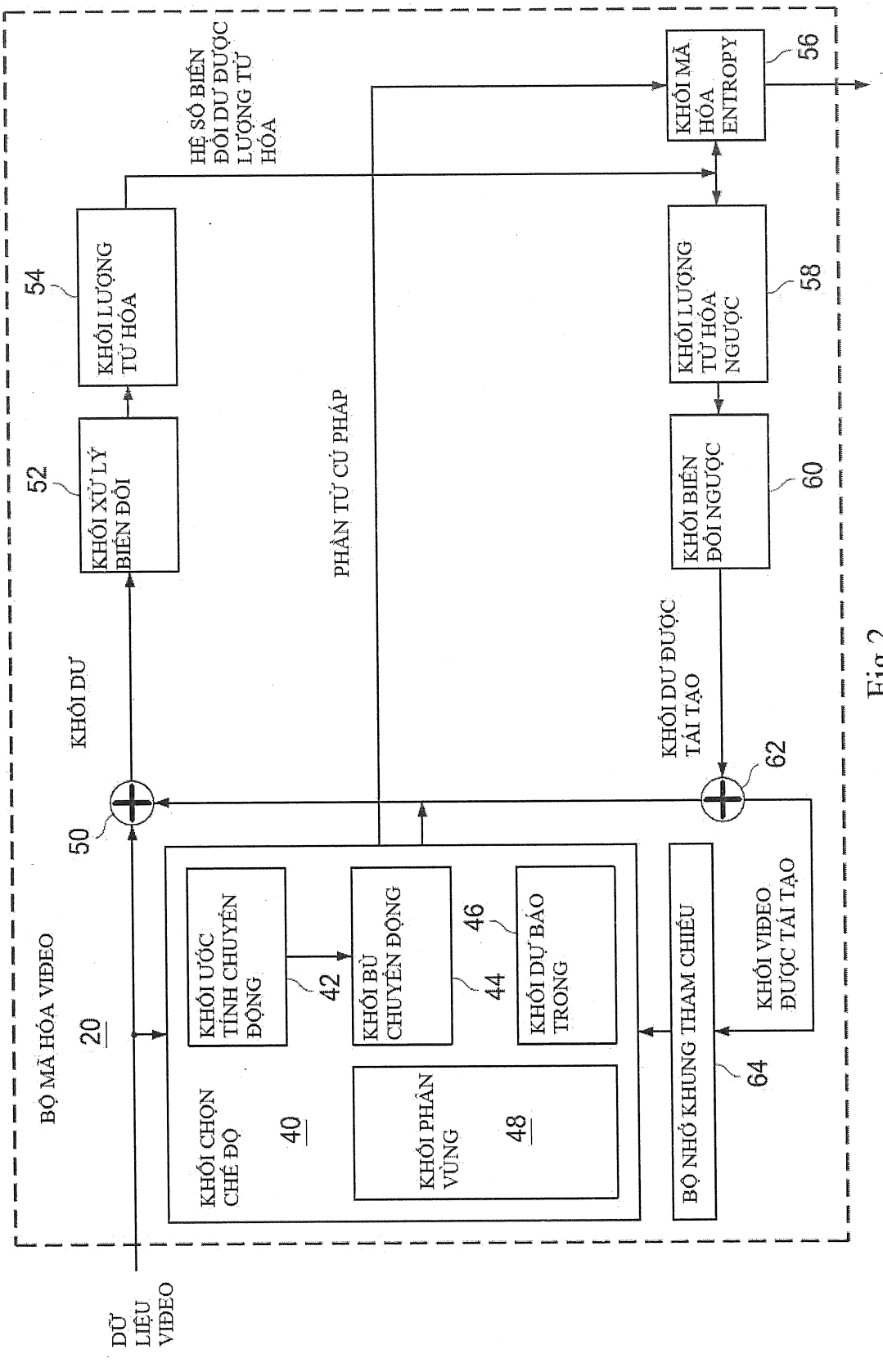
DỮ LIỆU VIĐEO
BỘ MÃ HÓA VIĐEO
20
KHỐI CHỌN CHẾ ĐỘ
40
KHỐI ƯỚC TÍNH CHUYỂN ĐỘNG
42
KHỐI BÙ CHUYỂN ĐỘNG
44
KHỐI DỰ BÁO TRONG
46
KHỐI PHÂN VÙNG
48
KHỐI DƯ
50
KHỐI XỬ LÝ BIẾN ĐỔI
52
KHỐI LƯỢNG TỬ HÓA
54
HỆ SỐ BIẾN ĐỔI ĐƯỢC LƯỢNG TỬ HÓA
PHẦN TỬ CÚ PHÁP
KHỐI MÃ HÓA ENTROPY
56
KHỐI LƯỢNG TỬ HÓA NGƯỢC
58
KHỐI BIẾN ĐỔI NGƯỢC
60
KHỐI DƯ ĐƯỢC TÁI TẠO
62
BỘ NHỚ KHUNG THAM CHIẾU
64
KHỐI VIĐEO ĐƯỢC TÁI TẠO

Fig.2

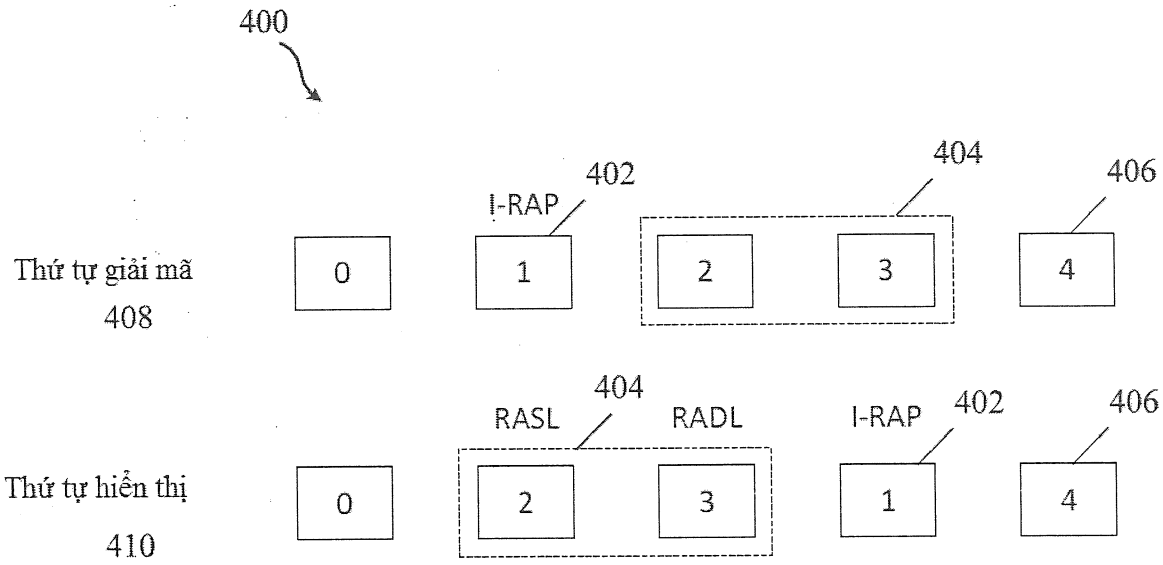
400
402
I-RAP
404
406
Thứ tự giải mã
408
0
1
2
3
4
404
RASL
RADL
402
I-RAP
406
Thứ tự hiển thị
410
0
2
3
1
4

Fig.4

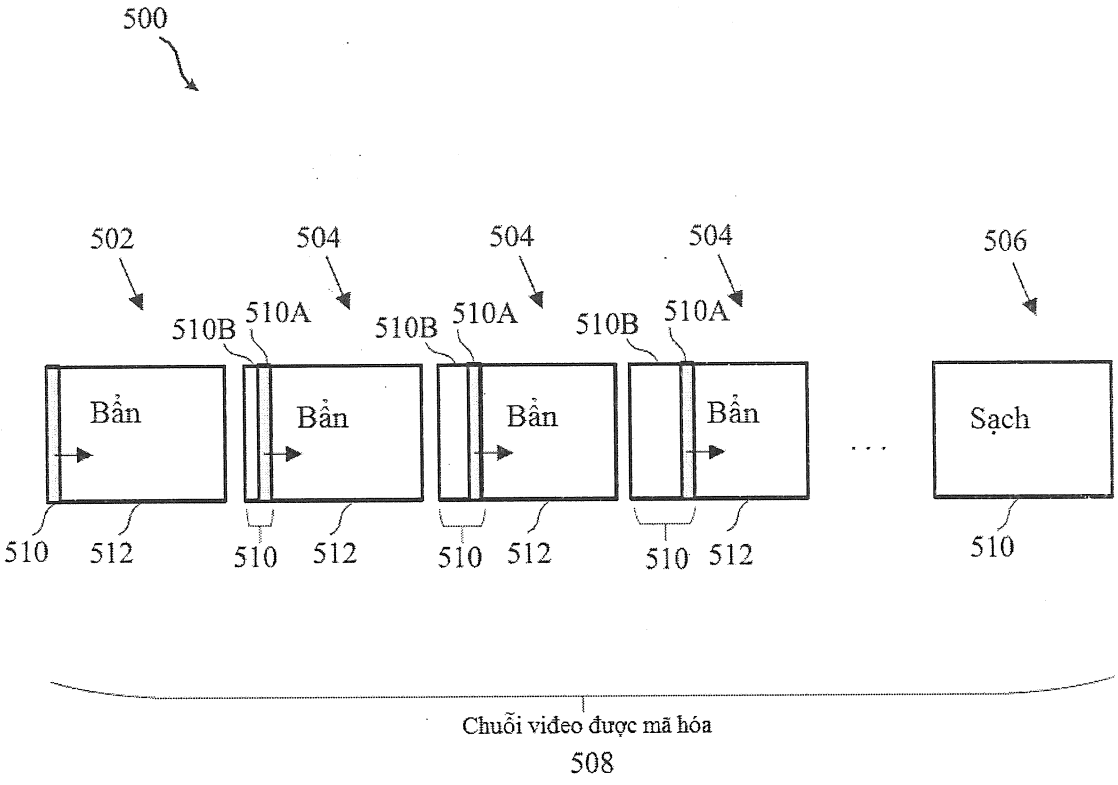
500
502
504
504
504
506
510B
510A
510B
510A
510B
510A
Bẩn
Bẩn
Bẩn
Bẩn
Sạch
510
512
510
512
510
512
510
512
510
Chuỗi viđeo được mã hóa
508

Fig.5

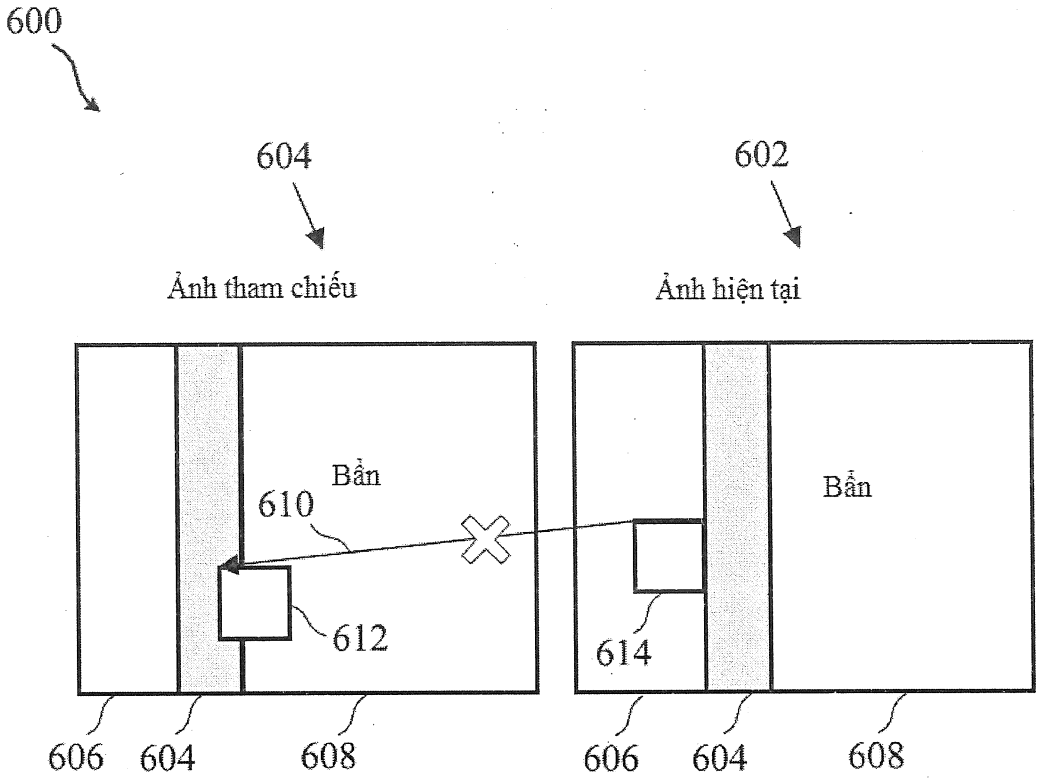
600
604
602
Ảnh tham chiếu
Ảnh hiện tại
Bản
610
612
614
Bản
606
604
608
606
604
608

Fig.6

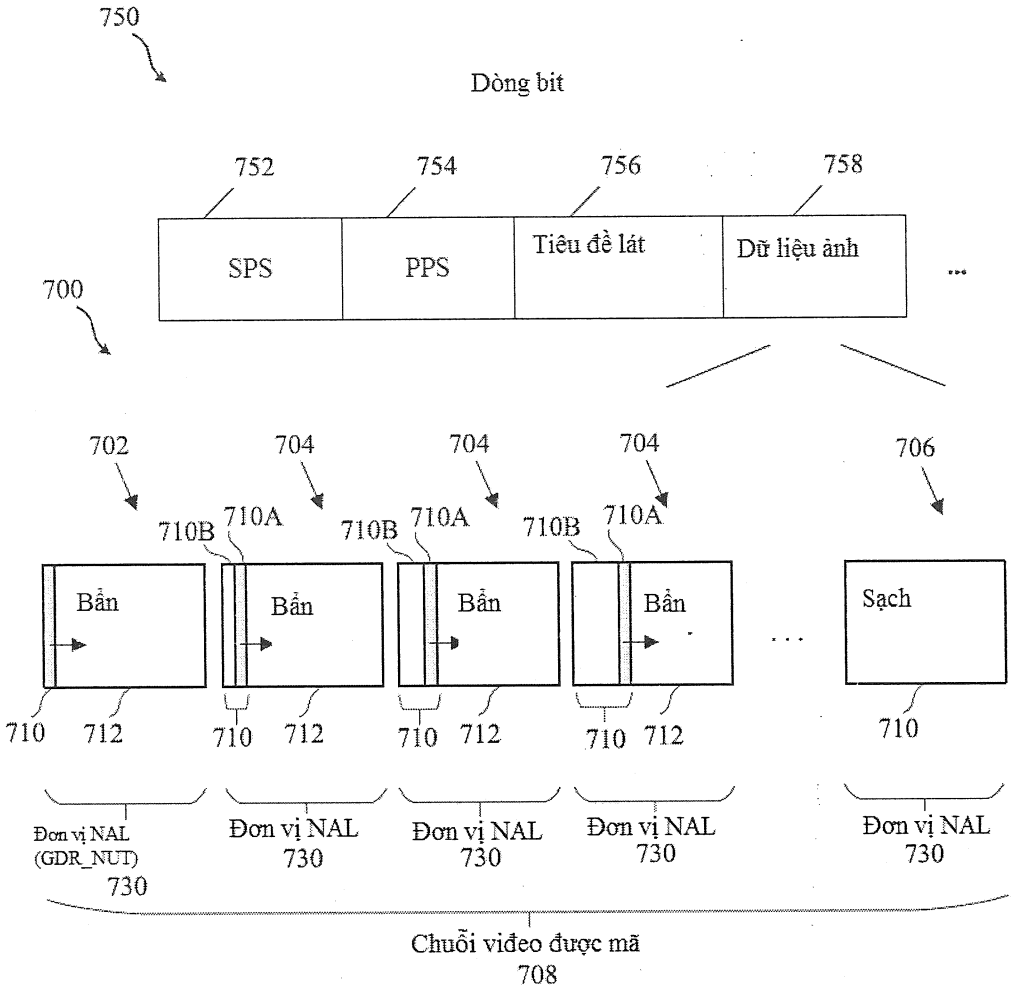
750
Dòng bit
752
754
756
758
SPS
PPS
Tiêu đề lát
Dữ liệu ảnh
...
700
702
704
704
704
706
710B
710A
710B
710A
710B
710A
Bẩn
Bẩn
Bẩn
Bẩn
Sạch
710
712
710
712
710
712
710
712
710
Đơn vị NAL
(GDR_NUT)
730
Đơn vị NAL
730
Đơn vị NAL
730
Đơn vị NAL
730
Đơn vị NAL
730
Chuỗi viđeo được mã
708

Fig.7

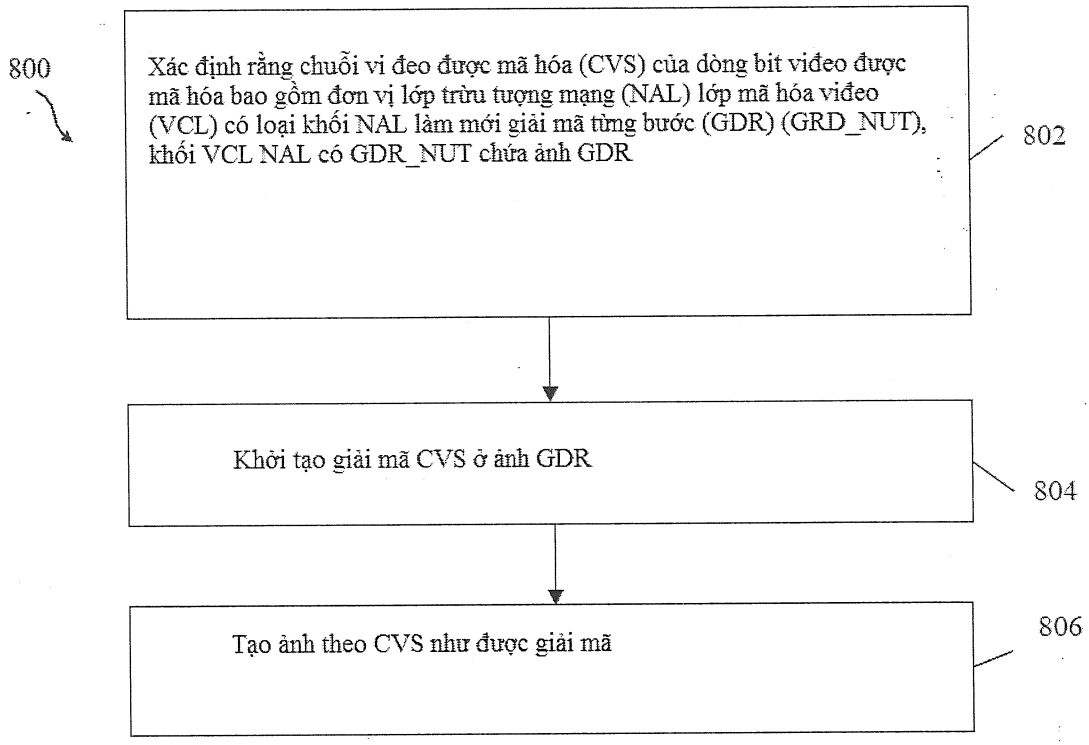
800
Xác định rằng chuỗi vi đeo được mã hóa (CVS) của dòng bit viđeo được mã hóa bao gồm đơn vị lớp trừu tượng mạng (NAL) lớp mã hóa viđeo (VCL) có loại khối NAL làm mới giải mã từng bước (GDR) (GRD_NUT), khối VCL NAL có GDR_NUT chứa ảnh GDR
802
Khởi tạo giải mã CVS ở ảnh GDR
804
Tạo ảnh theo CVS như được giải mã
806

Fig.8

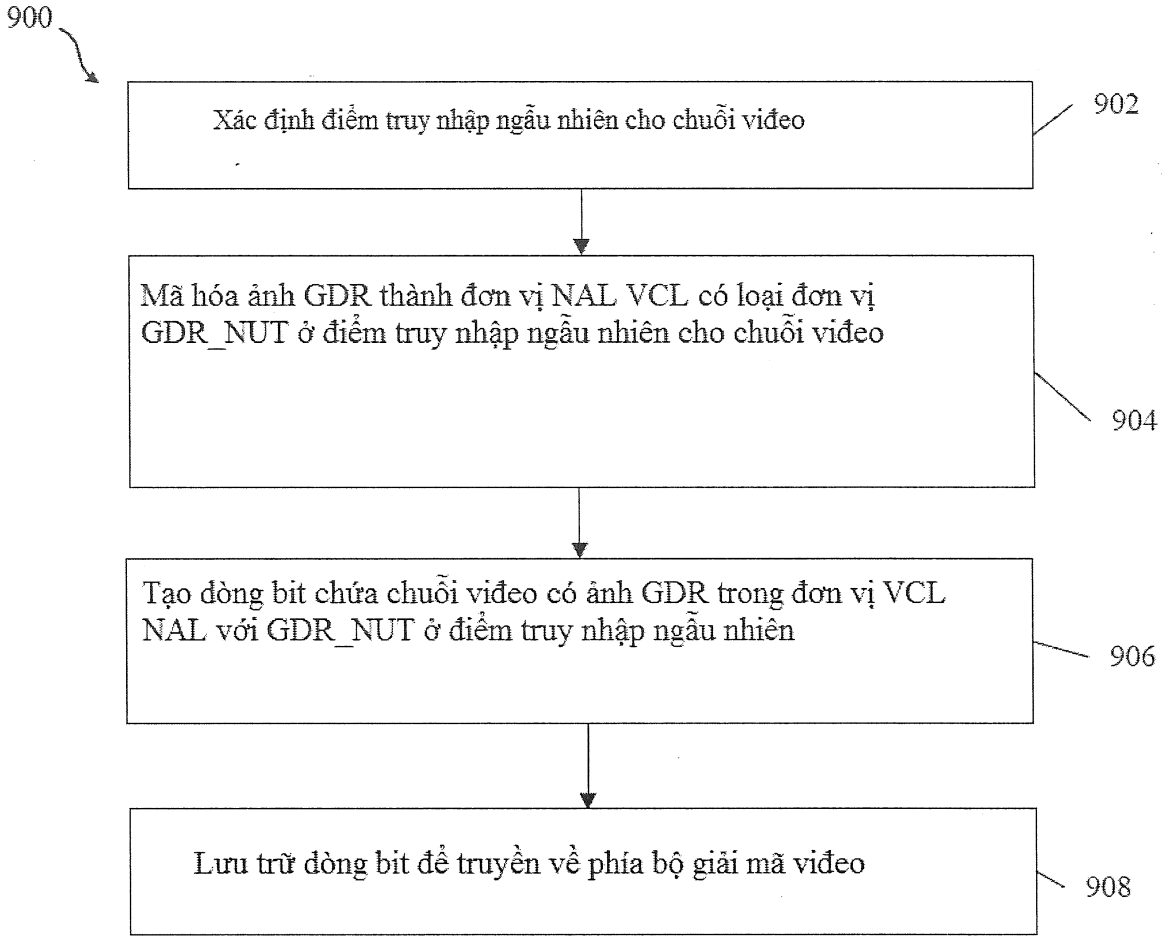
900
Xác định điểm truy nhập ngẫu nhiên cho chuỗi viđeo
902
Mã hóa ảnh GDR thành đơn vị NAL VCL có loại đơn vị GDR_NUT ở điểm truy nhập ngẫu nhiên cho chuỗi viđeo
904
Tạo đòng bit chứa chuỗi viđeo có ảnh GDR trong đơn vị VCL NAL với GDR_NUT ở điểm truy nhập ngẫu nhiên
906
Lưu trữ đòng bit để truyền về phía bộ giải mã viđeo
908

Fig.9

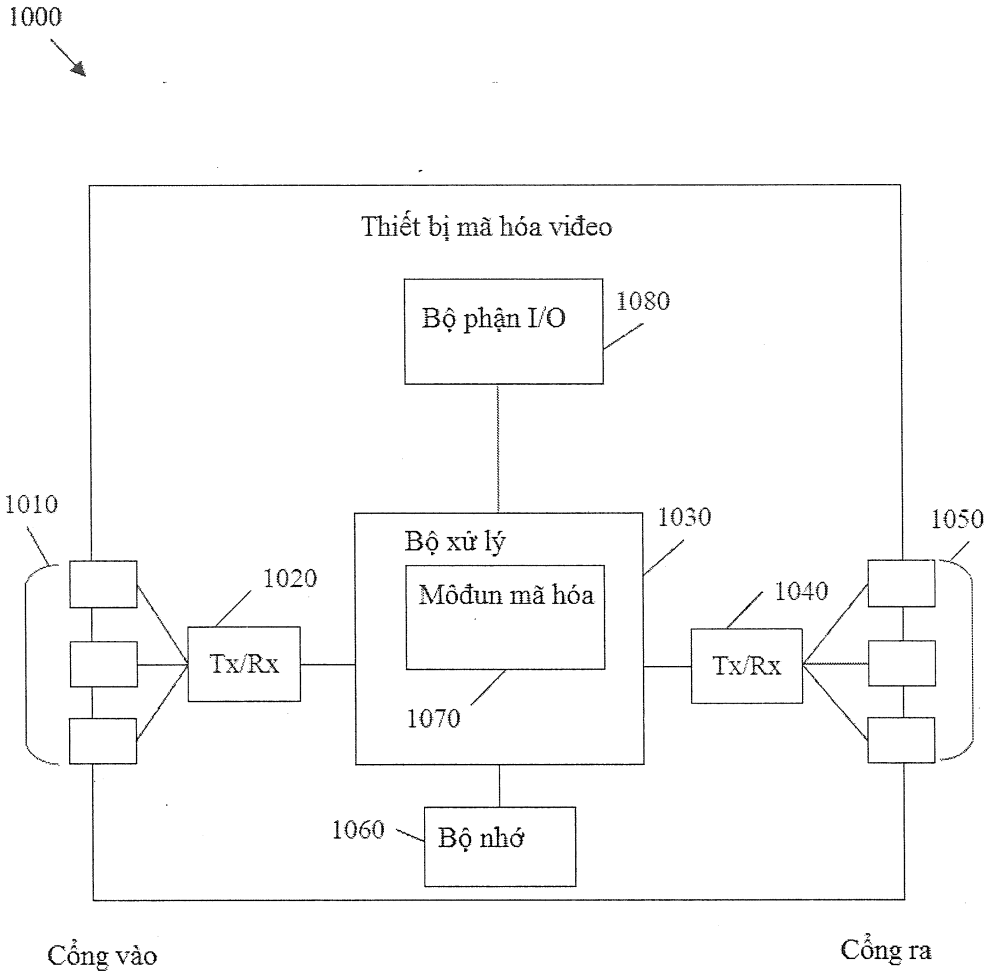
1000
Thiết bị mã hóa viđeo
Bộ phận I/O
1080
1010
1020
Tx/Rx
Bộ xử lý
Môđun mã hóa
1030
1070
1040
Tx/Rx
1050
1060
Bộ nhớ
Cổng vào
Cổng ra

Fig.10

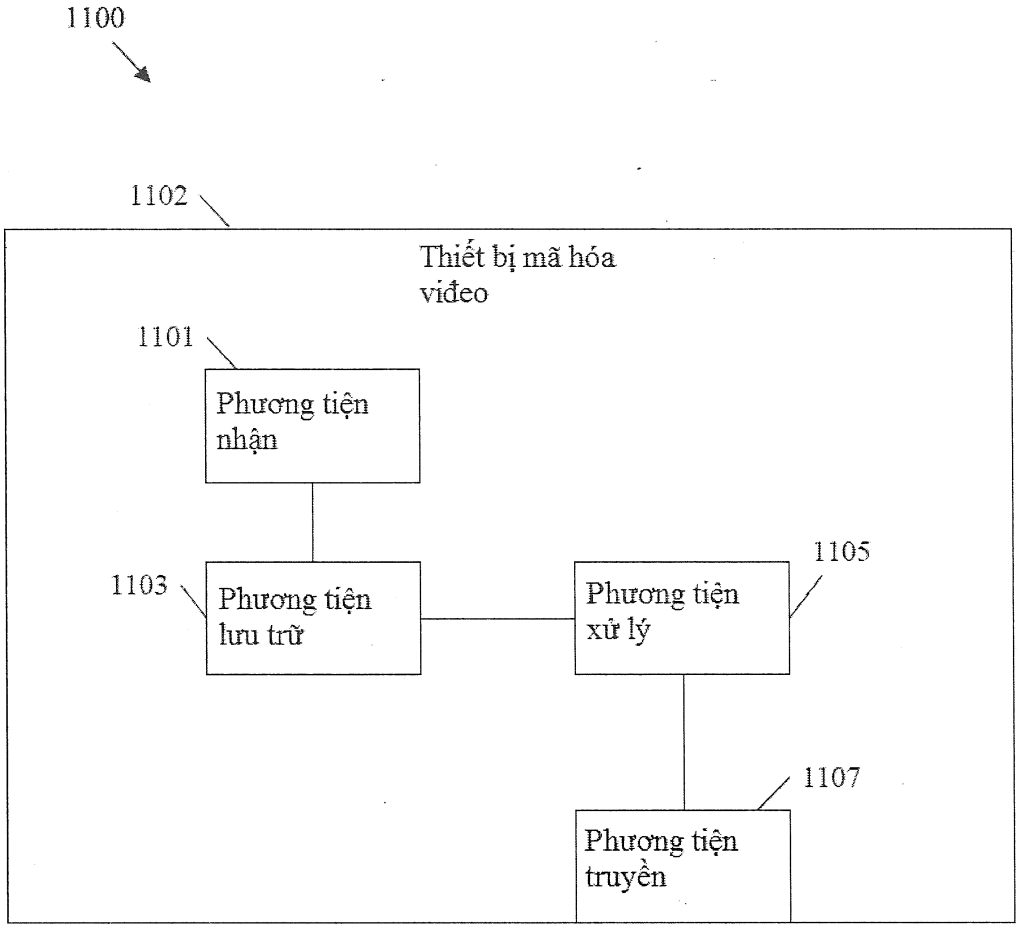
1100
1102
Thiết bị mã hóa video
1101
Phương tiện nhận
1103
Phương tiện lưu trữ
1105
Phương tiện xử lý
1107
Phương tiện truyền

Fig.11