(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) **1-0051132**

(51)[2020.01] **H04N 19/56**; H04N 19/70; H04N 19/176; H04N 19/527

(13) **B**

(21) 1-2021-07408
(22) 24/04/2020
(86) PCT/US2020/029913 24/04/2020
(87) WO2020/219945 29/10/2020
(30) 62/838,615 25/04/2019 US
(45) 25/09/2025 450
(43) 25/01/2022 406A
(73) OP SOLUTIONS, LLC (US)
368 Middle Street, Amherst, Massachusetts 01002, United States of America
(72) KALVA, Hari (IN); FURHT, Borivoje (US); ADZIC, Velibor (US).
(74) Công ty Luật TNHH T&G (TGVN)

(54) BỘ GIẢI MÃ VIĐEO VÀ BỘ MÃ HÓA VIĐEO

(21) 1-2021-07408

(57) Sáng chế đề cập đến bộ giải mã bao gồm hệ mạch được tạo cấu hình để nhận dòng bit; xác định, cho khối hiện tại và nhờ sử dụng dòng bit này, ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề; xây dựng danh sách ứng viên vectơ chuyển động bao gồm việc thêm ứng viên vectơ chuyển động toàn cục được xác định vào danh sách ứng viên vectơ chuyển động này; và xây dựng lại dữ liệu điểm ảnh của khối hiện tại và sử dụng danh sách ứng viên vectơ chuyển động. Sáng chế cũng đề cập đến phương pháp giải mã.

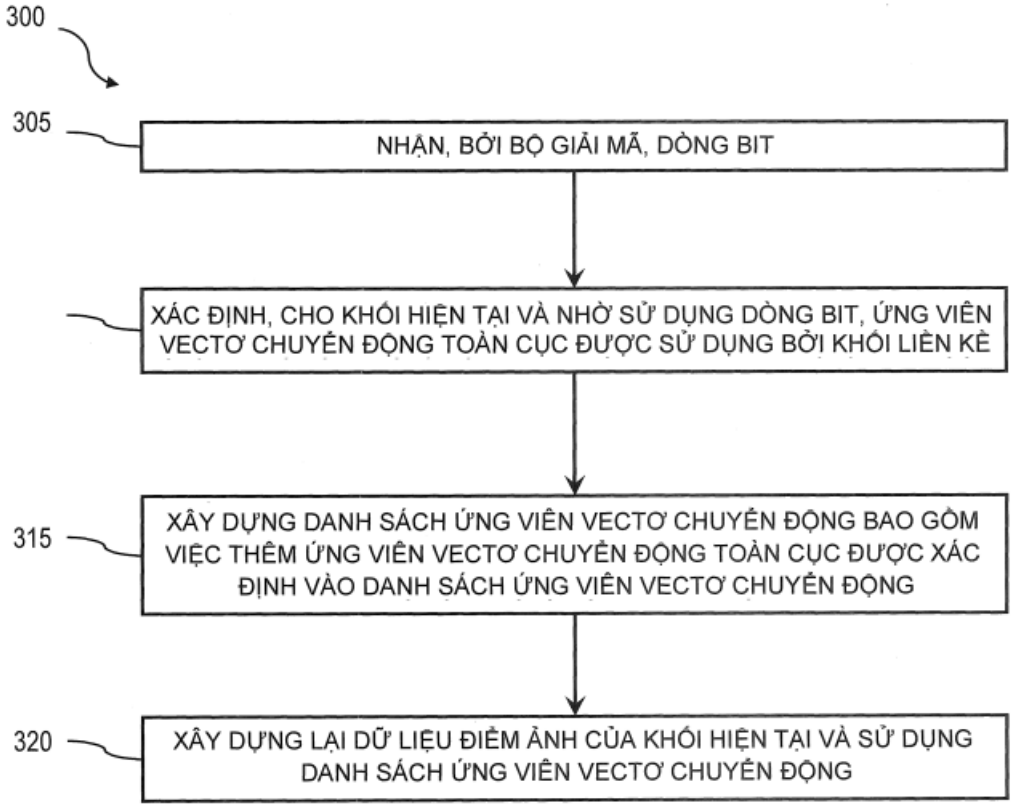


FIG. 3

LĨNH VỰC KỸ THUẬT ĐƯỢC ĐỀ CẬP

Nhìn chung, sáng chế đề cập đến lĩnh vực nén viđeo. Cụ thể, sáng chế hướng tới các ứng viên dự đoán vectơ chuyển động thích ứng trong các khung với chuyển động toàn cục.

Tình trạng kỹ thuật của sáng chế

Codec viđeo có thể bao gồm mạch điện tử hoặc phần mềm mà nó nén hoặc giải nén viđeo kỹ thuật số. Nó có thể chuyển đổi viđeo không được nén thành định dạng được nén hoặc ngược lại. Trong ngữ cảnh của việc nén viđeo, thiết bị mà nó nén viđeo (và/hoặc thực hiện một số chức năng của việc này) thường có thể được gọi là bộ mã hóa, và thiết bị mà nó giải nén viđeo (và/hoặc thực hiện một số chức năng của việc này) có thể được gọi là bộ giải mã.

Định dạng của dữ liệu được nén có thể tuân thủ theo thông số kỹ thuật nén viđeo tiêu chuẩn. Việc nén có thể là có mất mát ở chỗ viđeo được nén thiếu một số thông tin có mặt trong viđeo ban đầu. Hệ quả của điều này có thể bao gồm việc viđeo được giải nén có thể có chất lượng thấp hơn viđeo không được nén ban đầu do có không đủ thông tin để xây dựng lại chính xác viđeo ban đầu.

Có thể có những mối quan hệ phức tạp giữa chất lượng viđeo, lượng dữ liệu được sử dụng để thể hiện viđeo (ví dụ như được xác định bởi tốc độ bit), độ phức tạp của các thuật toán mã hóa và giải mã, độ nhạy với các mất mát và sai số dữ liệu, độ dễ biên tập, việc truy cập ngẫu nhiên, độ trì hoãn đầu mút-đầu mút (ví dụ, độ trễ), và dạng tương tự.

Việc bù chuyển động có thể bao gồm cách tiếp cận để dự đoán khung viđeo hoặc một phần của nó khi đã cho khung tham chiếu, chẳng hạn như các khung trước đó và/hoặc các khung tương lai, bằng cách xem xét tới chuyển động của camera và/hoặc các vật thể trong viđeo. Nó có thể được sử dụng trong việc mã hóa và giải mã dữ liệu viđeo để nén viđeo, ví dụ như trong việc mã hóa và giải mã sử dụng tiêu chuẩn nhóm chuyên gia hình ảnh động (Motion Picture Experts Group, MPEG)-2 (cũng được gọi là tạo mã viđeo cải tiến

(advanced video coding, AVC) và H.264). Việc bù chuyển động có thể mô tả hình ảnh dưới dạng việc biến đổi của hình ảnh tham chiếu thành hình ảnh hiện tại. Hình ảnh tham chiếu có thể là ở trước đó theo thời gian khi so sánh với hình ảnh hiện tại, từ tương lai khi so sánh với hình ảnh hiện tại. Khi các ảnh có thể được tổng hợp một cách chính xác từ các ảnh được truyền và/hoặc được lưu trữ trước đó, hiệu quả nén có thể được cải thiện.

BẢN CHẤT KỸ THUẬT CỦA SÁNG CHẾ

Theo một khía cạnh, bộ giải mã bao gồm hệ mạch được tạo cấu hình để nhận dòng bit, xác định, cho khối hiện tại và nhờ sử dụng dòng bit này, ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề; xây dựng danh sách ứng viên vectơ chuyển động bao gồm việc thêm ứng viên vectơ chuyển động toàn cục được xác định vào danh sách ứng viên vectơ chuyển động, và xây dựng lại dữ liệu điểm ảnh của khối hiện tại và sử dụng danh sách ứng viên vectơ chuyển động này.

Theo một khía cạnh khác, phương pháp bao gồm bước nhận, bởi bộ giải mã, dòng bit, xác định, cho khối hiện tại và nhờ sử dụng dòng bit này, ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề, xây dựng danh sách ứng viên vectơ chuyển động, trong đó việc xây dựng danh sách ứng viên vectơ chuyển động còn bao gồm việc thêm ứng viên vectơ chuyển động toàn cục được xác định vào danh sách ứng viên vectơ chuyển động, và xây dựng lại dữ liệu điểm ảnh của khối hiện tại và sử dụng danh sách ứng viên vectơ chuyển động này.

Các chi tiết của một hoặc nhiều biến thể của đối tượng của sáng chế được mô tả ở đây được nêu trên các hình vẽ kèm theo và phần mô tả dưới đây. Các dấu hiệu và các ưu điểm khác của sáng chế được mô tả ở đây sẽ trở nên rõ ràng từ phần mô tả và các hình vẽ, và từ các điểm yêu cầu bảo hộ.

Mô tả vắn tắt các hình vẽ

Nhằm mục đích minh họa sáng chế, các hình vẽ thể hiện các khía cạnh của một hoặc nhiều phương án của sáng chế. Tuy nhiên, cần hiểu rằng sáng chế không bị giới hạn ở các cách sắp xếp và phương tiện chính xác được thể hiện trên các hình vẽ này, trong đó:

Fig.1 là sơ đồ minh họa các vectơ chuyển động của khung ví dụ với chuyển động toàn cục và cục bộ;

Fig.2 minh họa ba mô hình chuyển động ví dụ mà chúng có thể được sử dụng cho chuyển động toàn cục bao gồm các giá trị chỉ số của chúng (0, 1, hoặc 2);

Fig.3 là sơ đồ luồng xử lý theo một số cách thi hành ví dụ của sáng chế này;

Fig.4 là sơ đồ khối hệ thống của bộ giải mã ví dụ theo một số cách thi hành ví dụ của sáng chế này;

Fig.5 là sơ đồ luồng xử lý theo một số cách thi hành ví dụ của sáng chế này;

Fig.6 là sơ đồ khối hệ thống của bộ mã hóa ví dụ theo một số cách thi hành ví dụ của sáng chế này; và

Fig.7 là sơ đồ khối của hệ thống tính toán mà nó có thể được sử dụng để thi hành một hoặc nhiều phương pháp bất kỳ trong số các phương pháp được bộc lộ ở đây và một hoặc nhiều phần bất kỳ của chúng.

Các hình vẽ này không nhất thiết là đúng tỷ lệ và có thể được minh họa bởi các đường ảo, các cách biểu diễn dạng biểu đồ và các hình rời rạc. Trong các ví dụ nhất định, các chi tiết không cần thiết cho việc hiểu các phương án hoặc tạo ra các chi tiết khác khó lĩnh hội có thể đã được lược bỏ. Các ký hiệu tham chiếu giống nhau trên các hình vẽ khác nhau biểu thị các phần tử giống nhau.

Mô tả chi tiết sáng chế

Chuyển động toàn cục trong viđeo đề cập tới chuyển động mà nó xuất hiện trong toàn bộ khung. Chuyển động toàn cục có thể được gây ra bởi chuyển động của camera, ví dụ, việc lia và phóng to thu nhỏ camera tạo ra chuyển động trong khung mà nó có thể thường ảnh hưởng tới toàn bộ khung. Chuyển động có mặt trong các phần của viđeo có thể được gọi là chuyển động cục bộ. Chuyển động cục bộ có thể được gây ra bởi các vật thể chuyển động trong cảnh, chẳng hạn như, nhưng không giới hạn ở, vật thể chuyển động từ bên trái sang bên phải trong cảnh này. Các viđeo có thể chứa tổ hợp của chuyển động cục bộ và chuyển động toàn cục. Một số cách thi hành của sáng chế này có thể đề xuất các cách tiếp cận hiệu quả để truyền thông chuyển động toàn cục tới bộ giải mã và việc sử dụng các vectơ

chuyển động toàn cục trong việc cải thiện hiệu quả nén.

Fig.1 là sơ đồ minh họa các vectơ chuyển động của khung ví dụ 100 với chuyển động toàn cục và cục bộ. Khung 100 có thể bao gồm một số các khối của các điểm ảnh được minh họa dưới dạng các hình vuông, và các vectơ chuyển động được liên kết của chúng được minh họa dưới dạng các mũi tên. Các hình vuông (ví dụ, các khối của các điểm ảnh) với các mũi tên chỉ lên và sang bên trái có thể chỉ ra các khối với chuyển động mà nó có thể được xem như là chuyển động toàn cục và các hình vuông với các mũi tên chỉ theo các hướng khác (được chỉ ra bởi 104) chỉ ra các khối với chuyển động cục bộ. Trong ví dụ được minh họa trên Fig.1, nhiều khối có chuyển động toàn cục giống hệt nhau. Việc báo hiệu chuyển động toàn cục trong tiêu đề, chẳng hạn như tập tham số hình ảnh (picture parameter set, PPS) và/hoặc tập tham số chuỗi (sequence parameter set, SPS) và sử dụng chuyển động toàn cục được báo hiệu có thể làm giảm thông tin vectơ chuyển động cần bởi các khối và có thể dẫn đến việc dự đoán được cải thiện. Mặc dù để nhằm mục đích minh họa các ví dụ được mô tả dưới đây đề cập tới việc xác định và/hoặc áp dụng các vectơ chuyển động toàn cục hoặc cục bộ tại mức khối, các vectơ chuyển động toàn cục có thể được xác định và/hoặc áp dụng cho vùng bất kỳ của khung và/hoặc hình ảnh, bao gồm các vùng được tạo nên từ nhiều khối, các vùng được giới hạn bởi dạng hình học bất kỳ chẳng hạn như, mà không giới hạn ở, các vùng được định ra bởi việc tạo mã hình học và/hoặc dạng mũ trong đó một hoặc nhiều đường thẳng và/hoặc đường cong giới hạn hình dạng này có thể được tạo góc và/hoặc được làm cong, và/hoặc toàn bộ của khung và/hoặc hình ảnh. Mặc dù việc báo hiệu được mô tả ở đây như là được thực hiện tại mức khung và/hoặc trong tiêu đề và/hoặc tập tham số của khung, việc báo hiệu theo cách khác hoặc thêm vào đó có thể được thực hiện tại mức hình ảnh con, trong đó hình ảnh con có thể bao gồm vùng bất kỳ của khung và/hoặc hình ảnh như được mô tả trên đây.

Để làm ví dụ, và vẫn tham chiếu đến Fig.1, chuyển động tịnh tiến đơn giản có thể được mô tả nhờ sử dụng vectơ chuyển động (motion vector, MV) với hai thành phần MVx, MVy mà chúng mô tả sự dịch chuyển của các khối và/hoặc các điểm ảnh trong khung hiện tại. Chuyển động phức tạp hơn chẳng hạn như việc quay, phóng to/thu nhỏ, và lượn có thể được mô tả nhờ sử dụng các vectơ chuyển động afin, trong đó "vectơ chuyển động afin,"

như được sử dụng trong bản mô tả này, là vectơ mô tả sự dịch chuyển đồng nhất của tập hợp các điểm ảnh hoặc các điểm được biểu diễn trong hình ảnh viđeo và/hoặc hình ảnh, chẳng hạn như tập hợp các điểm ảnh minh họa vật thể chuyển động qua hình trong viđeo mà không thay đổi hình dạng biểu kiến trong suốt chuyển động. Một số cách tiếp cận với việc mã hóa và/hoặc giải mã viđeo có thể sử dụng các mô hình afin 4 tham số hoặc 6 tham số cho việc bù chuyển động trong việc tạo mã hình ảnh liên.

Ví dụ, và tiếp tục tham chiếu đến Fig.1, chuyển động afin sáu tham số có thể được mô tả dưới dạng:

$x' = ax + by + c$

$y' = dx + ey + f$

Để ví dụ thêm, chuyển động afin bốn tham số có thể được mô tả dưới dạng:

$x' = ax + by + c$

$y' = -bx + ay + f$

trong đó (x,y) và (x',y') là các vị trí điểm ảnh trong các hình ảnh hiện tại và tham chiếu, theo cách tương ứng; a, b, c, d, e, và f là các tham số của mô hình chuyển động afin.

Vẫn tham chiếu đến Fig.1, các tham số được sử dụng để mô tả chuyển động afin có thể được báo hiệu cho bộ giải mã để áp dụng việc bù chuyển động afin tại bộ giải mã này. Theo một số cách tiếp cận, các tham số chuyển động có thể được báo hiệu một cách tường minh hoặc bằng cách báo hiệu các vectơ chuyển động của điểm điều khiển (control point motion vector, CPMV) tịnh tiến và sau đó dẫn ra các tham số chuyển động afin từ các vectơ chuyển động tịnh tiến này. Hai vectơ chuyển động của điểm điều khiển (CPMV) có thể được sử dụng để dẫn ra các tham số chuyển động afin cho mô hình chuyển động afin bốn tham số và ba vectơ chuyển động của điểm điều khiển (CPMV) tịnh tiến có thể được sử dụng để thu các tham số cho mô hình chuyển động sáu tham số. Việc báo hiệu các tham số chuyển động afin sử dụng các vectơ chuyển động của điểm điều khiển có thể cho phép việc sử dụng các phương pháp tạo mã vectơ chuyển động hiệu quả để báo hiệu các tham số chuyển động afin.

Trong một phương án, và tiếp tục tham chiếu đến Fig.1, **sps_affine_enabled_flag** trong PPS và/hoặc SPS có thể chỉ rõ xem liệu việc bù chuyển động dựa trên mô hình afin

có thể được sử dụng cho việc dự đoán liên hay không. Nếu sps_affine_enabled_flag bằng 0 thì cú pháp này có thể được ràng buộc sao cho không có việc bù chuyển động dựa trên mô hình afin nào được sử dụng trong chuỗi viđeo muộn hơn mã (code later video sequence, CLVS), và inter_affine_flag và cu_affine_type_flag có thể không có mặt trong cú pháp đơn vị tạo mã của CLVS này. Theo cách khác (sps_affine_enabled_flag bằng 1), việc bù chuyển động dựa trên mô hình afin có thể được sử dụng trong CLVS này.

Tham chiếu thêm đến Fig.1, **sps_affine_type_flag** trong PPS và/hoặc SPS có thể chỉ rõ xem liệu việc bù chuyển động dựa trên mô hình afin 6 tham số có thể được sử dụng cho việc dự đoán liên hay không. Nếu sps_affine_type_flag bằng 0 thì cú pháp có thể được ràng buộc sao cho không có việc bù chuyển động dựa trên mô hình afin 6 tham số nào được sử dụng trong CLVS, và cu_affine_type_flag có thể không có mặt trong cú pháp đơn vị tạo mã trong CLVS. Nếu không (sps_affine_type_flag bằng 1) thì việc bù chuyển động dựa trên mô hình afin 6 tham số có thể được sử dụng trong CLVS. Khi không có mặt, giá trị của sps_affine_type_flag có thể được suy ra là bằng 0.

Tiếp tục tham chiếu đến Fig.1, việc xử lý danh sách của các ứng viên dự đoán MV có thể là bước được thực hiện trong một số cách tiếp cận nén mà chúng sử dụng việc bù chuyển động tại bộ giải mã. Một số cách tiếp cận có trước có thể định ra việc sử dụng các ứng viên vectơ chuyển động theo không gian và theo thời gian. Chuyển động toàn cục được báo hiệu trong tiêu đề, chẳng hạn như SPS hoặc PPS, có thể chỉ ra sự có mặt của chuyển động toàn cục trong viđeo. Chuyển động toàn cục có thể được dự liệu là chung cho hầu hết các khối trong khung. Việc tạo mã vectơ chuyển động có thể được cải thiện và tốc độ bit được giảm bằng cách sử dụng chuyển động toàn cục làm ứng viên dự đoán. MV ứng viên được thêm vào danh sách dự đoán MV có thể được chọn phụ thuộc vào mô hình chuyển động được sử dụng để biểu diễn chuyển động toàn cục và/hoặc mô hình chuyển động được sử dụng trong việc tạo mã liên.

Vẫn tham chiếu đến Fig.1, một số cách thi hành của chuyển động toàn cục có thể được mô tả với một hoặc nhiều vectơ chuyển động của điểm điều khiển (CPMV) phụ thuộc vào mô hình chuyển động được sử dụng. Do đó, phụ thuộc vào mô hình chuyển động được sử dụng, một đến ba vectơ chuyển động của điểm điều khiển có thể là khả dụng và có thể được

sử dụng làm các ứng viên cho việc dự đoán. Trong một số cách thi hành, tất cả các CPMV khả dụng có thể được thêm dưới dạng các ứng viên dự đoán vào danh sách. Trường hợp chung của việc thêm tất cả các CPMV khả dụng có thể làm tăng khả năng có thể có của việc tìm ra việc dự đoán vectơ chuyển động tốt và cải thiện hiệu quả nén. Bảng 1:

Mô hình chuyển động toàn cục được sử dụng (MM, các MV khả dụng)	MV của PU được tạo mã	Ứng viên dự đoán GM được thêm vào
MM=0, MV0	MV0 (MM=0)	MV0
MM=0, MV0	MV1 (MM=1 hoặc 2)	MV0
MM=0, MV0	MV2 (MM=2)	MV0
MM=1, MV0, MV1	MV0 (MM=0)	MV0, MV1
MM=1, MV0, MV1	MV1 (MM=1 hoặc 2)	MV0, MV1
MM=1, MV0, MV1	MV2 (MM=2)	MV0, MV1
MM=2, MV0, MV1, MV2	MV0 (MM=0)	MV0, MV1, MV2
MM=2, MV0, MV1, MV2	MV1 (MM=1 hoặc 2)	MV0, MV1, MV2
MM=2, MV0, MV1, MV2	MV2 (MM=2)	MV0, MV1, MV2

Tiếp tục tham chiếu đến Fig.1, việc xử lý danh sách của các ứng viên dự đoán vectơ chuyển động (motion vector, MV) có thể là một bước trong việc thực hiện việc bù chuyển động tại bộ giải mã.

Vẫn tham chiếu đến Fig.1, chuyển động toàn cục được báo hiệu trong tiêu đề, chẳng hạn như SPS, có thể chỉ ra sự có mặt của chuyển động toàn cục trong viđeo. Chuyển động toàn cục này có thể có khả năng có mặt trong nhiều khối trong khung. Do đó, khối đã cho có thể có khả năng có chuyển động tương tự như chuyển động toàn cục. Việc tạo mã vectơ chuyển động có thể được cải thiện và tốc độ bit được giảm bằng cách sử dụng chuyển động toàn cục làm ứng viên dự đoán. MV ứng viên có thể được thêm vào danh sách dự đoán MV và MV ứng viên có thể được chọn phụ thuộc vào mô hình chuyển động được sử dụng để biểu diễn chuyển động toàn cục và mô hình chuyển động được sử dụng trong việc tạo mã liên.

Tiếp tục tham chiếu đến Fig.1, chuyển động toàn cục có thể được mô tả với một hoặc nhiều vectơ chuyển động của điểm điều khiển (CPMV) phụ thuộc vào mô hình chuyển

động được sử dụng. Do đó, phụ thuộc vào mô hình chuyển động được sử dụng, một đến ba vectơ chuyển động của điểm điều khiển có thể là khả dụng và có thể được sử dụng làm các ứng viên cho việc dự đoán. Danh sách của các ứng viên dự đoán MV có thể được giảm xuống qua việc thêm có chọn lọc một CPMV vào danh sách ứng viên dự đoán. Việc làm giảm kích thước danh sách có thể làm giảm độ phức tạp tính toán và cải thiện hiệu quả nén.

Trong một số cách thi hành, và vẫn tham chiếu đến Fig.1, việc chọn CPMV làm ứng viên có thể là tùy theo việc ánh xạ định trước, chẳng hạn như được minh họa trong bảng 3 sau đây:

Mô hình chuyển động toàn cục được sử dụng (M, các MV khả dụng)	MV của PU được tạo mã	Ứng viên dự đoán GM được thêm vào
M=0, MV0	MV0 (M=0, hoặc 1, hoặc 2)	MV0
M=0, MV0	MV1 (M=1 hoặc 2)	MV0
M=0, MV0	MV2 (M=2)	MV0
M=1, MV0, MV1	MV0 (M=0, hoặc 1, hoặc 2)	MV0
M=1, MV0, MV1	MV1 (M=1 hoặc 2)	MV1
M=1, MV0, MV1	MV2 (M=2)	MV0
M=2, MV0, MV1, MV2	MV0 (M=0, hoặc 1, hoặc 2)	MV0
M=2, MV0, MV1, MV2	MV1 (M=1 hoặc 2)	MV1
M=2, MV0, MV1, MV2	MV2 (M=2)	MV2

Vẫn tham chiếu đến Fig.1, việc sử dụng ứng viên dự đoán có chọn lọc từ chuyển động toàn cục có thể được báo hiệu trong tập tham số hình ảnh của tập tham số chuỗi, qua đó làm giảm độ phức tạp mã hóa và giải mã.

Tham chiếu thêm đến Fig.1, do khối có khả năng có chuyển động tương tự như chuyển động toàn cục, việc thêm vectơ chuyển động toàn cục dưới dạng ứng viên thứ nhất vào trong danh sách dự đoán có thể làm giảm các bit cần thiết để báo hiệu các ứng viên dự đoán được sử dụng và để mã hóa các chênh lệch vectơ chuyển động.

Tiếp tục tham chiếu đến Fig.1, việc xử lý danh sách của các ứng viên dự đoán vectơ chuyển động (MV) có thể là một bước trong việc thực hiện việc bù chuyển động tại bộ giải mã.

Vẫn tham chiếu đến Fig.1, chuyển động toàn cục được báo hiệu trong tiêu đề, chẳng hạn như SPS, có thể chỉ ra sự có mặt của chuyển động toàn cục trong viđeo. Chuyển động

toàn cục này có khả năng có mặt trong nhiều khối trong khung. Do đó, khối đã cho có khả năng có chuyển động tương tự như chuyển động toàn cục. Việc tạo mã vectơ chuyển động có thể được cải thiện và tốc độ bit được giảm bằng cách sử dụng chuyển động toàn cục làm ứng viên dự đoán.

Tham chiếu thêm đến Fig.1, ví dụ, MV ứng viên được thêm vào danh sách dự đoán MV có thể được chọn theo cách thích ứng phụ thuộc vào việc véctơ chuyển động của điểm điều khiển (CPMV) nào được chọn làm ứng viên dự đoán trong các khối lân cận (ví dụ, các đơn vị dự đoán (prediction unit, PU)).

Để ví dụ thêm, và vẫn tham chiếu đến Fig.1, nếu các PU lân cận sử dụng CPMV cụ thể làm MV dự đoán cho điểm điều khiển cụ thể, thì CPMV đó có thể được thêm vào danh sách ứng viên MV. Nếu các PU lân cận sử dụng nhiều hơn một CPMV làm MV dự đoán (ví dụ, các PU bên trái sử dụng CPMV0 và PU trên đỉnh sử dụng CPMV1), thì tất cả các CPMV được sử dụng làm các ứng viên dự đoán có thể được thêm vào danh sách. Nếu các PU lân cận không sử dụng CPMV, thì không có các CPMV nào có thể được thêm vào danh sách dự đoán. Trong một số cách thi hành, thông tin chuyển động toàn cục có thể được thêm vào như là ứng viên thứ nhất trong danh sách dự đoán.

Tiếp tục tham chiếu đến Fig.1, việc sử dụng ứng viên dự đoán thích ứng từ chuyển động toàn cục có thể được báo hiệu trong tiêu đề, chẳng hạn như PPS và/hoặc SPS, qua đó làm giảm độ phức tạp mã hóa và giải mã.

Vẫn tham chiếu đến Fig.1, do khối có khả năng có chuyển động tương tự như chuyển động toàn cục, việc thêm vectơ chuyển động toàn cục dưới dạng ứng viên thứ nhất trong danh sách dự đoán có thể làm giảm các bit cần thiết để báo hiệu các ứng viên dự đoán được sử dụng và để mã hóa các chênh lệch vectơ chuyển động. Fig.2 minh họa ba mô hình chuyển động ví dụ 200 mà chúng có thể được sử dụng cho chuyển động toàn cục bao gồm các giá trị chỉ số của chúng (0, 1, hoặc 2).

Vẫn tham chiếu đến Fig.2, các PPS có thể được sử dụng để báo hiệu các tham số mà chúng có thể thay đổi giữa các ảnh của chuỗi. Các tham số mà chúng vẫn giữ không đổi cho chuỗi của các hình ảnh có thể được báo hiệu trong tập tham số chuỗi để làm giảm kích thước của PPS và làm giảm tốc độ bit của viđeo. Tập tham số hình ảnh (picture parameter

set, PPS) ví dụ được thể hiện trên bảng 2:

pic_parameter_set_rbsp( ) {	**Descriptor**
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	u(4)
mixed_nalu_types_in_pic_flag	u(1)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
pps_conformance_window_flag	u(1)
if(pps_conformance_window_flag) {	
pps_conf_win_left_offset	ue(v)
pps_conf_win_right_offset	ue(v)
pps_conf_win_top_offset	ue(v)
pps_conf_win_bottom_offset	ue(v)
}	
scaling_window_explicit_signalling_flag	u(1)
if(scaling_window_explicit_signalling_flag) {	
scaling_win_left_offset	ue(v)
scaling_win_right_offset	ue(v)
scaling_win_top_offset	ue(v)
scaling_win_bottom_offset	ue(v)
}	
output_flag_present_flag	u(1)
subpic_id_mapping_in_pps_flag	u(1)
if(subpic_id_mapping_in_pps_flag) {	
pps_num_subpics_minus1	ue(v)
pps_subpic_id_len_minus1	ue(v)
for(i = 0; i <= pps_num_subpic_minus1; i++)	
pps_subpic_id[i]	u(v)
}	
no_pic_partition_flag	u(1)
if(!no_pic_partition_flag) {	
pps_log2_ctu_size_minus5	u(2)
num_exp_tile_columns_minus1	ue(v)
num_exp_tile_rows_minus1	ue(v)
for(i = 0; i <= num_exp_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i <= num_exp_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
if(NumTilesInPic > 1)	
rect_slice_flag	u(1)

if(rect_slice_flag)	
single_slice_per_subpic_flag	u(1)
if(rect_slice_flag && !single_slice_per_subpic_flag) {	
num_slices_in_pic_minus1	ue(v)
if(num_slices_in_pic_minus1 > 0)	
tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
if(NumTileColumns > 1)	
slice_width_in_tiles_minus1[i]	ue(v)
if(NumTileRows > 1 && (tile_idx_delta_present_flag || SliceTopLeftTileIdx[i] % NumTileColumns == 0))	
slice_height_in_tiles_minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0 && RowHeight[SliceTopLeftTileIdx[i] / NumTileColumns] > 1) {	
num_exp_slices_in_tile[i]	ue(v)
for(j = 0; j < num_exp_slices_in_tile[i]; j++)	
exp_slice_height_in_ctus_minus1[i][j]	ue(v)
i += NumSlicesInTile[i] − 1	
}	
if(tile_idx_delta_present_flag && i < num_slices_in_pic_minus1)	
tile_idx_delta[i]	se(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
loop_filter_across_slices_enabled_flag	u(1)
}	
cabac_init_present_flag	u(1)
for(i = 0; i < 2; i++)	
num_ref_idx_default_active_minus1[i]	ue(v)
rpl1_idx_present_flag	u(1)
init_qp_minus26	se(v)
cu_qp_delta_enabled_flag	u(1)
pps_chroma_tool_offsets_present_flag	u(1)
if(pps_chroma_tool_offsets_present_flag) {	
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbcr_qp_offset_present_flag	u(1)
if(pps_joint_cbcr_qp_offset_present_flag)	
pps_joint_cbcr_qp_offset_value	se(v)

pps_slice_chroma_qp_offsets_present_flag	u(1)
pps_cu_chroma_qp_offset_list_enabled_flag	u(1)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
chroma_qp_offset_list_len_minus1	ue(v)
for(i = 0; i <= chroma_qp_offset_list_len_minus1; i++) {	
cb_qp_offset_list[i]	se(v)
cr_qp_offset_list[i]	se(v)
if(pps_joint_cbcr_qp_offset_present_flag)	
joint_cbcr_qp_offset_list[i]	se(v)
}	
}	
pps_weighted_pred_flag	u(1)
pps_weighted_bipred_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
deblocking_filter_override_enabled_flag	u(1)
pps_deblocking_filter_disabled_flag	u(1)
if(!pps_deblocking_filter_disabled_flag) {	
pps_beta_offset_div2	se(v)
pps_tc_offset_div2	se(v)
pps_cb_beta_offset_div2	se(v)
pps_cb_tc_offset_div2	se(v)
pps_cr_beta_offset_div2	se(v)
pps_cr_tc_offset_div2	se(v)
}	
}	
rpl_info_in_ph_flag	u(1)
if(deblocking_filter_override_enabled_flag)	
dbf_info_in_ph_flag	u(1)
sao_info_in_ph_flag	u(1)
alf_info_in_ph_flag	u(1)
if((pps_weighted_pred_flag || pps_weighted_bipred_flag) && rpl_info_in_ph_flag)	
wp_info_in_ph_flag	u(1)
qp_delta_info_in_ph_flag	u(1)
pps_ref_wraparound_enabled_flag	u(1)
if(pps_ref_wraparound_enabled_flag)	
pps_ref_wraparound_offset	ue(v)
picture_header_extension_present_flag	u(1)
slice_header_extension_present_flag	u(1)
pps_extension_flag	u(1)

if(pps_extension_flag)	
while(more_rbsp_data( ))	
pps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

Các trường bổ sung có thể được thêm vào PPS để báo hiệu chuyển động toàn cục. Trong trường hợp của chuyển động toàn cục, sự có mặt của các tham số chuyển động toàn cục trong chuỗi của các hình ảnh có thể được báo hiệu trong SPS và PPS có thể tham chiếu SPS này qua ID của SPS. SPS trong một số cách tiếp cận với việc giải mã có thể được cải biến để thêm trường để báo hiệu sự có mặt của các tham số chuyển động toàn cục trong SPS. Ví dụ, trường một bit có thể được thêm vào SPS này. Nếu bit global_motion_present là 1 thì các tham số liên quan tới chuyển động toàn cục có thể được dự kiến trong PPS. Nếu bit global_motion_present là 0 thì không có các trường liên quan tới tham số chuyển động toàn cục nào có thể có mặt trong PPS. Ví dụ, PPS trong bảng 2 có thể được mở rộng để bao gồm trường global_motion_present, ví dụ, như được thể hiện trên bảng 3:

sequence_parameter_set_rbsp( ) {	**Descriptor**
sps_sequence_parameter_set_id	ue(v)
. . .	
global_motion_present	u(1)
rbsp_trailing_bits( )	
}	

Tương tự, PPS có thể bao gồm trường pps_global_motion_parameters cho khung, ví dụ như được thể hiện trên bảng 4:

pic_parameter_set_rbsp( ) {	**Descriptor**
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
. . .	

pps_global_motion_parameters ( )	
rbsp_trailing_bits( )	
}	

Chi tiết hơn, PPS có thể bao gồm các trường để đặc trưng cho các tham số chuyển động toàn cục sử dụng các vectơ chuyển động của điểm điều khiển, ví dụ như được thể hiện trên bảng 5:

pps_global_motion_parameters ( ) {	**Descriptor**
motion_model_used	**u(2)**
mv0_x	**se(v)**
mv1_y	**se(v)**
if(motion_model_used == 1){	
mv1_x	**se(v)**
mv1_y	**se(v)**
}	
if(motion_model_used == 2){	
mv2_x	**se(v)**
mv2_y	**se(v)**
}	
}	

Như một ví dụ thêm không nhằm mục đích giới hạn, bảng 6 dưới đây có thể biểu diễn SPS làm ví dụ:

seq_parameter_set_rbsp( ) {	**Descriptor**
sps_seq_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sublayers_minus1	u(3)
sps_reserved_zero_4bits	u(4)
sps_ptl_dpb_hrd_params_present_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
profile_tier_level(1, sps_max_sublayers_minus1)	
gdr_enabled_flag	u(1)
chroma_format_idc	u(2)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
res_change_in_clvs_allowed_flag	u(1)
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
sps_conformance_window_flag	u(1)

if(sps_conformance_window_flag) {	
sps_conf_win_left_offset	ue(v)
sps_conf_win_right_offset	ue(v)
sps_conf_win_top_offset	ue(v)
sps_conf_win_bottom_offset	ue(v)
}	
sps_log2_ctu_size_minus5	u(2)
subpic_info_present_flag	u(1)
if(subpic_info_present_flag) {	
sps_num_subpics_minus1	ue(v)
sps_independent_subpics_flag	u(1)
for(i = 0; sps_num_subpics_minus1 > 0 && i <= sps_num_subpics_minus1; i++) {	
if(i > 0 && pic_width_max_in_luma_samples > CtbSizeY)	
subpic_ctu_top_left_x[i]	u(v)
if(i > 0 && pic_height_max_in_luma_samples > CtbSizeY) {	
subpic_ctu_top_left_y[i]	u(v)
if(i < sps_num_subpics_minus1 && pic_width_max_in_luma_samples > CtbSizeY)	
subpic_width_minus1[i]	u(v)
if(i < sps_num_subpics_minus1 && pic_height_max_in_luma_samples > CtbSizeY)	
subpic_height_minus1[i]	u(v)
if(!sps_independent_subpics_flag) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	
sps_subpic_id_len_minus1	ue(v)
subpic_id_mapping_explicitly_signalled_flag	u(1)
if(subpic_id_mapping_explicitly_signalled_flag) {	
subpic_id_mapping_in_sps_flag	u(1)
if(subpic_id_mapping_in_sps_flag)	
for(i = 0; i <= sps_num_subpics_minus1; i++)	
sps_subpic_id[i]	u(v)
}	
}	
bit_depth_minus8	ue(v)
sps_entropy_coding_sync_enabled_flag	u(1)
if(sps_entropy_coding_sync_enabled_flag)	
sps_wpp_entry_point_offsets_present_flag	u(1)
sps_weighted_pred_flag	u(1)

sps_weighted_bipred_flag	u(1)
log2_max_pic_order_cnt_lsb_minus4	u(4)
sps_poc_msb_flag	u(1)
if(sps_poc_msb_flag)	
poc_msb_len_minus1	ue(v)
num_extra_ph_bits_bytes	u(2)
extra_ph_bits_struct(num_extra_ph_bits_bytes)	
num_extra_sh_bits_bytes	u(2)
extra_sh_bits_struct(num_extra_sh_bits_bytes)	
if(sps_max_sublayers_minus1 > 0)	
sps_sublayer_dpb_params_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
dpb_parameters(sps_max_sublayers_minus1, sps_sublayer_dpb_params_flag)	
long_term_ref_pics_flag	u(1)
inter_layer_ref_pics_present_flag	u(1)
sps_idr_rpl_present_flag	u(1)
rpl1_same_as_rpl0_flag	u(1)
for(i = 0; i < rpl1_same_as_rpl0_flag ? 1 : 2; i++) {	
num_ref_pic_lists_in_sps[i]	ue(v)
for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)	
ref_pic_list_struct(i, j)	
}	
if(ChromaArrayType != 0)	
qtbtt_dual_tree_intra_flag	u(1)
log2_min_luma_coding_block_size_minus2	ue(v)
partition_constraints_override_enabled_flag	u(1)
sps_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
sps_log2_diff_min_qt_min_cb_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_inter_slice	ue(v)
if(sps_max_mtt_hierarchy_depth_inter_slice != 0) {	
sps_log2_diff_max_bt_min_qt_inter_slice	ue(v)
sps_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)

if(sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
sps_max_luma_transform_size_64_flag	u(1)
if(ChromaArrayType != 0) {	
sps_joint_cbcr_enabled_flag	u(1)
same_qp_table_for_chroma	u(1)
numQpTables = same_qp_table_for_chroma ? 1 : (sps_joint_cbcr_enabled_flag ? 3 : 2)	
for(i = 0; i < numQpTables; i++) {	
qp_table_start_minus26[i]	se(v)
num_points_in_qp_table_minus1[i]	ue(v)
for(j = 0; j <= num_points_in_qp_table_minus1[i]; j++) {	
delta_qp_in_val_minus1[i][j]	ue(v)
delta_qp_diff_val[i][j]	ue(v)
}	
}	
}	
sps_sao_enabled_flag	u(1)
sps_alf_enabled_flag	u(1)
if(sps_alf_enabled_flag && ChromaArrayType != 0)	
sps_ccalf_enabled_flag	u(1)
sps_transform_skip_enabled_flag	u(1)
if(sps_transform_skip_enabled_flag) {	
log2_transform_skip_max_size_minus2	ue(v)
sps_bdpcm_enabled_flag	u(1)
}	
sps_ref_wraparound_enabled_flag	u(1)
sps_temporal_mvp_enabled_flag	u(1)
if(sps_temporal_mvp_enabled_flag)	
sps_sbtmvp_enabled_flag	u(1)
sps_amvr_enabled_flag	u(1)
sps_bdof_enabled_flag	u(1)
if(sps_bdof_enabled_flag)	
sps_bdof_pic_present_flag	u(1)
sps_smvd_enabled_flag	u(1)
sps_dmvr_enabled_flag	u(1)
if(sps_dmvr_enabled_flag)	
sps_dmvr_pic_present_flag	u(1)
sps_mmvd_enabled_flag	u(1)

sps_isp_enabled_flag	u(1)
sps_mrl_enabled_flag	u(1)
sps_mip_enabled_flag	u(1)
if(ChromaArrayType != 0)	
sps_cclm_enabled_flag	u(1)
if(chroma_format_idc == 1) {	
sps_chroma_horizontal_collocated_flag	u(1)
sps_chroma_vertical_collocated_flag	u(1)
}	
sps_mts_enabled_flag	u(1)
if(sps_mts_enabled_flag) {	
sps_explicit_mts_intra_enabled_flag	u(1)
sps_explicit_mts_inter_enabled_flag	u(1)
}	
six_minus_max_num_merge_cand	ue(v)
sps_sbt_enabled_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag) {	
five_minus_max_num_subblock_merge_cand	ue(v)
sps_affine_type_flag	u(1)
if(sps_amvr_enabled_flag)	
sps_affine_amvr_enabled_flag	u(1)
sps_affine_prof_enabled_flag	u(1)
if(sps_affine_prof_enabled_flag)	
sps_prof_pic_present_flag	u(1)
}	
sps_palette_enabled_flag	u(1)
if(ChromaArrayType == 3 && !sps_max_luma_transform_size_64_flag)	
sps_act_enabled_flag	u(1)
if(sps_transform_skip_enabled_flag || sps_palette_enabled_flag)	
min_qp_prime_ts_minus4	ue(v)
sps_bcw_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
if(sps_ibc_enabled_flag)	
six_minus_max_num_ibc_merge_cand	ue(v)
sps_ciip_enabled_flag	u(1)
if(sps_mmvd_enabled_flag)	
sps_fpel_mmvd_enabled_flag	u(1)
if(MaxNumMergeCand >= 2) {	
sps_gpm_enabled_flag	u(1)
if(sps_gpm_enabled_flag && MaxNumMergeCand >= 3)	

max_num_merge_cand_minus_max_num_gpm_cand	ue(v)
}	
sps_lmcs_enabled_flag	u(1)
sps_lfnst_enabled_flag	u(1)
sps_ladf_enabled_flag	u(1)
if(sps_ladf_enabled_flag) {	
sps_num_ladf_intervals_minus2	u(2)
sps_ladf_lowest_interval_qp_offset	se(v)
for(i = 0; i < sps_num_ladf_intervals_minus2 + 1; i++) {	
sps_ladf_qp_offset[i]	se(v)
sps_ladf_delta_threshold_minus1[i]	ue(v)
}	
}	
log2_parallel_merge_level_minus2	ue(v)
sps_explicit_scaling_list_enabled_flag	u(1)
sps_dep_quant_enabled_flag	u(1)
if(!sps_dep_quant_enabled_flag)	
sps_sign_data_hiding_enabled_flag	u(1)
sps_virtual_boundaries_enabled_flag	u(1)
if(sps_virtual_boundaries_enabled_flag) {	
sps_virtual_boundaries_present_flag	u(1)
if(sps_virtual_boundaries_present_flag) {	
sps_num_ver_virtual_boundaries	u(2)
for(i = 0; i < sps_num_ver_virtual_boundaries; i++)	
sps_virtual_boundaries_pos_x[i]	u(13)
sps_num_hor_virtual_boundaries	u(2)
for(i = 0; i < sps_num_hor_virtual_boundaries; i++)	
sps_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(sps_ptl_dpb_hrd_params_present_flag) {	
sps_general_hrd_params_present_flag	u(1)
if(sps_general_hrd_params_present_flag) {	
general_hrd_parameters( )	
if(sps_max_sublayers_minus1 > 0)	
sps_sublayer_cpb_params_present_flag	u(1)
firstSubLayer = sps_sublayer_cpb_params_present_flag ? 0 : sps_max_sublayers_minus1	
ols_hrd_parameters(firstSubLayer, sps_max_sublayers_minus1)	
}	
}	
field_seq_flag	u(1)

vui_parameters_present_flag	u(1)
if(vui_parameters_present_flag)	
vui_parameters( ) /* Được đặc tả trong ITU-T H.SEI | ISO/IEC 23002-7 */	
sps_extension_flag	u(1)
if(sps_extension_flag)	
while(more_rbsp_data( ))	
sps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

Bảng SPS như trên có thể được mở rộng như được mô tả trên đây để tích hợp chỉ dấu có mặt chuyển động toàn cục như được thể hiện trên bảng 7:

sequence_parameter_set_rbsp( ) {	**Descriptor**
sps_sequence_parameter_set_id	ue(v)
. . .	
global_motion_present	u(1)
rbsp_trailing_bits( )	
}	

Các trường bổ sung có thể được tích hợp trong SPS để phản ánh các chỉ dấu khác như được mô tả trong sáng chế này.

Trong một phương án, và vẫn tham chiếu đến Fig.2, **sps_affine_enabled_flag** trong PPS và/hoặc SPS có thể chỉ rõ xem liệu việc bù chuyển động dựa trên mô hình afin có thể được sử dụng cho việc dự đoán liên hay không. Nếu sps_affine_enabled_flag bằng 0 thì cú pháp này có thể được ràng buộc sao cho không có việc bù chuyển động dựa trên mô hình afin nào được sử dụng trong chuỗi viđeo muộn hơn mã (code later video sequence, CLVS), và inter_affine_flag và cu_affine_type_flag có thể không có mặt trong cú pháp đơn vị tạo mã của CLVS này. Theo cách khác (sps_affine_enabled_flag bằng 1), việc bù chuyển động dựa trên mô hình afin có thể được sử dụng trong CLVS này.

Tiếp tục tham chiếu đến Fig.2, **sps_affine_type_flag** trong PPS và/hoặc SPS có thể chỉ rõ xem liệu việc bù chuyển động dựa trên mô hình afin 6 tham số có thể được sử dụng cho việc dự đoán liên hay không. Nếu sps_affine_type_flag bằng 0 thì cú pháp có thể được

ràng buộc sao cho không có việc bù chuyển động dựa trên mô hình afin 6 tham số nào được sử dụng trong CLVS, và cu_affine_type_flag có thể không có mặt trong cú pháp đơn vị tạo mã trong CLVS. Nếu không (sps_affine_type_flag bằng 1) thì việc bù chuyển động dựa trên mô hình afin 6 tham số có thể được sử dụng trong CLVS. Khi không có mặt, giá trị của sps_affine_type_flag có thể được suy ra là bằng 0.

Vẫn tham chiếu đến Fig.2, các CPMV tịnh tiến có thể được báo hiệu trong PPS. Các điểm điều khiển có thể được định trước. Ví dụ, điểm điều khiển MV 0 có thể là so với góc trên cùng bên trái của hình ảnh, MV 1 có thể là so với góc trên cùng bên phải, và MV 3 có thể là so với góc dưới cùng bên trái của hình ảnh. Bảng 5 minh họa cách tiếp cận ví dụ cho việc báo hiệu dữ liệu CPMV phụ thuộc vào mô hình chuyển động được sử dụng.

Trong phương án làm ví dụ, và vẫn tham chiếu đến Fig.2, mảng **amvr_precision_idx**, mà nó có thể được báo hiệu theo đơn vị tạo mã, cây tạo mã hoặc dạng tương tự, có thể chỉ rõ độ phân giải AmvrShift của chênh lệch vectơ chuyển động, mà nó có thể được định ra như là ví dụ không nhằm mục đích giới hạn như được thể hiện trên bảng 8 như được thể hiện dưới đây. Các chỉ số mảng x0, y0 có thể chỉ rõ vị trí (x0, y0) của mẫu độ chói trên cùng bên trái của khối tạo mã được xem xét so với mẫu độ chói trên cùng bên trái của hình ảnh; khi amvr_precision_idx[x0][y0] không có mặt, nó có thể được suy ra là bằng 0. Khi inter_affine_flag[x0][y0] bằng 0, các biến số MvdL0[x0][y0][0], MvdL0[x0][y0][1], MvdL1[x0][y0][0], MvdL1[x0][y0][1] thể hiện các giá trị chênh lệch vectơ chuyển động tương ứng với các khối được xem xét, có thể được cải biến bằng cách dịch chuyển các giá trị này một lượng bằng AmvrShift, ví dụ như nhờ sử dụng MvdL0[x0][y0][0] = MvdL0[x0][y0][0] << AmvrShift; MvdL0[x0][y0][1] = MvdL0[x0][y0][1] << AmvrShift; MvdL1[x0][y0][0] = MvdL1[x0][y0][0] << AmvrShift; và MvdL1[x0][y0][1] = MvdL1[x0][y0][1] << AmvrShift. Khi inter_affine_flag[x0][y0] bằng 1, các biến MvdCpL0[x0][y0][0][0], MvdCpL0[x0][y0][0][1], MvdCpL0[x0][y0][1][0], MvdCpL0[x0][y0][1][1], MvdCpL0[x0][y0][2][0] và MvdCpL0[x0][y0][2][1] có thể được cải biến thông qua việc dịch chuyển, ví dụ như sau: MvdCpL0[x0][y0][0][0] =

MvdCpL0[x0][y0][0][0] << AmvrShift; MvdCpL1[x0][y0] [0][1] =
MvdCpL1[x0][y0][0][1] << AmvrShift; MvdCpL0[x0][y0][1][0] =
MvdCpL0[x0][y0][1][0] << AmvrShift; MvdCpL1[x0][y0] [1][1] =
MvdCpL1[x0][y0][1][1] << AmvrShift; MvdCpL0[x0][y0][2][0] =
MvdCpL0[x0][y0][2][0] << AmvrShift; và MvdCpL1[x0][y0] [2][1] =
MvdCpL1[x0][y0][2][1] << AmvrShift

amvr_flag	amvr_precision_idx	AmvrShift		
		inter_affine_flag ==1	CuPredMode[chType][x 0][y0] == MODE_IBC)	inter_affine_flag ==0 && CuPredMode[chType][x 0][y0] != MODE_IBC
0	-	2 (1/4 mẫu độ chói)	-	2 (1/4 mẫu độ chói)
1	0	0 (1/16 mẫu độ chói)	4 (1 mẫu độ chói)	3 (1/2 mẫu độ chói)
1	1	4 (1 mẫu độ chói)	6 (4 mẫu độ chói)	4 (1 mẫu độ chói)
1	2	-	-	6 (4 mẫu độ chói)

Fig.3 là sơ đồ luồng xử lý minh họa quy trình làm ví dụ 300 của việc xây dựng danh sách ứng viên vectơ chuyển động nhờ sử dụng ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề.

Tại bước 305, và vẫn tham chiếu đến Fig.3, khối hiện tại được nhận bởi bộ giải mã. Khối hiện tại có thể được chứa trong dòng bit mà bộ giải mã nhận. Dòng bit có thể bao gồm, ví dụ, dữ liệu được tìm thấy trong dòng của các bit mà nó là đầu vào cho bộ giải mã khi sử dụng việc nén dữ liệu. Dòng bit có thể bao gồm thông tin cần thiết để giải mã viđeo. Việc nhận có thể bao gồm việc trích xuất và/hoặc phân tích cú pháp khối và thông tin báo hiệu được liên kết với nó từ dòng bit. Trong một số cách thi hành, khối hiện tại có thể bao gồm đơn vị cây tạo mã (coding tree unit, CTU), đơn vị tạo mã (coding unit, CU), hoặc đơn vị dự đoán (prediction unit, PU).

Tại bước 310, và tham chiếu thêm đến Fig.3, ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề có thể được xác định cho khối hiện tại và nhờ sử dụng dòng

bit. Tại bước 330, danh sách ứng viên vectơ chuyển động có thể được xây dựng bao gồm việc thêm ứng viên vectơ chuyển động toàn cục được xác định vào danh sách ứng viên vectơ chuyển động. Tại bước 140, dữ liệu điểm ảnh của khối hiện tại có thể được xây dựng lại nhờ sử dụng danh sách ứng viên vectơ chuyển động.

Fig.4 là sơ đồ khối hệ thống minh họa bộ giải mã ví dụ 400 có khả năng giải mã dòng bit bao gồm, qua việc xây dựng danh sách ứng viên vectơ chuyển động nhờ sử dụng ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề. Bộ giải mã 400 có thể bao gồm bộ xử lý của bộ giải mã entropy 404, bộ xử lý lượng tử hóa ngược và biến đổi ngược 408, bộ lọc khử khối 412, bộ đệm khung 416, bộ xử lý bù chuyển động 420 và/hoặc bộ xử lý dự đoán nội 424.

Khi vận hành, và vẫn tham chiếu đến Fig.4, dòng bit 428 có thể được nhận bởi bộ giải mã 400 và được nhập vào bộ xử lý của bộ giải mã entropy 404, mà nó có thể giải mã entropy các phần của dòng bit thành các hệ số được lượng tử hóa. Các hệ số được lượng tử hóa có thể được cung cấp cho bộ xử lý lượng tử hóa ngược và biến đổi ngược 408, mà nó có thể thực hiện việc lượng tử hóa ngược và biến đổi ngược để tạo ra tín hiệu phần dư, mà nó có thể được cộng vào đầu ra của bộ xử lý bù chuyển động 420 hoặc bộ xử lý dự đoán nội 424 tùy theo chế độ xử lý. Đầu ra của bộ xử lý bù chuyển động 420 và bộ xử lý dự đoán nội 424 có thể bao gồm việc dự đoán khối dựa trên khối được giải mã trước đó. Tổng của việc dự đoán và phần dư có thể được xử lý bởi bộ lọc khử khối 412 và được lưu trữ trong bộ đệm khung 416.

Fig.5 là sơ đồ luồng xử lý minh họa phương án làm ví dụ của quy trình 500 là việc mã hóa viđeo bao gồm việc xây dựng danh sách ứng viên vectơ chuyển động nhờ sử dụng ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề theo một số khía cạnh của sáng chế mà nó có thể làm giảm độ phức tạp mã hóa trong khi làm tăng hiệu quả nén. Tại bước 505, khung viđeo có thể trải qua việc phân đoạn khối ban đầu, ví dụ và không giới hạn, qua việc sử dụng sơ đồ phân vùng khối macro được tạo cấu trúc cây mà nó có thể bao gồm việc phân vùng khung hình ảnh thành các CTU và các CU. Tại bước 510, chuyển động toàn cục có thể được xác định cho khối. Tại bước 515, khối có thể được mã hóa và được đưa vào trong dòng bit. Việc mã hóa có thể bao gồm việc xây dựng danh sách ứng viên

vectơ chuyển động nhờ sử dụng ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề. Việc mã hóa, ví dụ, có thể bao gồm việc sử dụng các chế độ dự đoán nội và dự đoán liên.

Fig.6 là sơ đồ khối hệ thống minh họa bộ mã hóa viđeo ví dụ 00 có khả năng xây dựng danh sách ứng viên vectơ chuyển động nhờ sử dụng ứng viên vectơ chuyển động toàn cục được sử dụng bởi khối liền kề. Bộ mã hóa viđeo làm ví dụ 600 có thể nhận viđeo đầu vào 604, mà nó có thể đã được phân đoạn ban đầu hoặc chia theo sơ đồ xử lý, chẳng hạn như sơ đồ phân vùng khối macro được tạo cấu trúc cây (ví dụ, cây tứ phân cộng cây nhị phân). Ví dụ về sơ đồ phân vùng khối macro được tạo cấu trúc cây có thể bao gồm việc phân vùng khung hình ảnh thành các phần tử khối lớn được gọi là các đơn vị cây tạo mã (CTU). Trong một số cách thi hành, mỗi CTU có thể còn được phân vùng một hoặc nhiều lần thành một số khối con được gọi là các đơn vị tạo mã (CU). Kết quả cuối cùng của việc phân vùng này có thể bao gồm nhóm các khối con mà chúng có thể được gọi là các đơn vị dự đoán (predictive unit, PU). Các đơn vị biến đổi (Transform unit, TU) cũng có thể được sử dụng.

Vẫn tham chiếu đến Fig.6, bộ mã hóa viđeo ví dụ 600 có thể bao gồm bộ xử lý dự đoán nội 608, bộ xử lý ước lượng/bù chuyển động 612, mà nó cũng có thể được gọi là bộ xử lý dự đoán liên, có khả năng xây dựng danh sách ứng viên vectơ chuyển động bao gồm việc thêm ứng viên vectơ chuyển động toàn cục vào danh sách ứng viên vectơ chuyển động này, bộ xử lý biến đổi/lượng tử hóa 616, bộ xử lý lượng tử hóa ngược/biến đổi ngược 620, bộ lọc trong vòng 624, bộ đệm hình ảnh được giải mã 628, và/hoặc bộ xử lý tạo mã entropy 632. Các tham số dòng bit có thể là đầu vào cho bộ xử lý tạo mã entropy 632 để được đưa vào trong dòng bit đầu ra 636.

Khi vận hành, và tiếp tục tham chiếu đến Fig.6, đối với mỗi khối của khung của viđeo đầu vào 604, việc liệu sẽ xử lý khối thông qua việc dự đoán hình ảnh nội hay sử dụng việc ước lượng/bù chuyển động có thể được xác định. Khối có thể được cung cấp cho bộ xử lý dự đoán nội 608 hoặc bộ xử lý ước lượng/bù chuyển động 612. Nếu khối cần được xử lý thông qua việc dự đoán nội thì bộ xử lý dự đoán nội 608 có thể thực hiện việc xử lý để xuất ra bộ dự đoán. Nếu khối cần được xử lý thông qua việc ước lượng/bù chuyển động thì bộ xử lý ước lượng/bù chuyển động 612 có thể thực hiện việc xử lý bao gồm việc xây dựng

danh sách ứng viên vectơ chuyển động bao gồm việc thêm ứng viên vectơ chuyển động toàn cục vào danh sách ứng viên vectơ chuyển động này, nếu có thể áp dụng được.

Tham chiếu thêm đến Fig.6, phần dư có thể được tạo ra bằng việc trừ bộ dự đoán khỏi viđeo đầu vào. Phần dư có thể được nhận bởi bộ xử lý biến đổi/lượng tử hóa 616, mà nó có thể thực hiện việc xử lý biến đổi (ví dụ, biến đổi cosin rời rạc (discrete cosine transform, DCT)) để tạo ra các hệ số, mà chúng có thể được lượng tử hóa. Các hệ số được lượng tử hóa và thông tin báo hiệu bất kỳ được liên kết với chúng có thể được cung cấp cho bộ xử lý tạo mã entropy 632 để mã hóa entropy và đưa vào trong dòng bit đầu ra 636. Bộ xử lý mã hóa entropy 632 có thể hỗ trợ việc mã hóa thông tin báo hiệu liên quan tới việc mã hóa khối hiện tại. Thêm vào đó, các hệ số được lượng tử hóa có thể được cung cấp cho bộ xử lý lượng tử hóa ngược/biến đổi ngược 620, mà nó có thể tái tạo các điểm ảnh, mà chúng có thể được kết hợp với bộ dự đoán và được xử lý bởi bộ lọc trong vòng 624, đầu ra của nó có thể được lưu trữ trong bộ đệm hình ảnh được giải mã 628 để sử dụng bởi bộ xử lý ước lượng/bù chuyển động 612 mà nó có khả năng xây dựng danh sách ứng viên vectơ chuyển động bao gồm việc thêm ứng viên vectơ chuyển động toàn cục vào danh sách ứng viên vectơ chuyển động này.

Tiếp tục tham chiếu đến Fig.6, mặc dù một số biến thể đã được mô tả chi tiết trên đây, các cải biến hoặc bổ sung khác cũng là khả thi. Ví dụ, trong một số cách thi hành, các khối hiện tại có thể bao gồm các khối đối xứng bất kỳ (8x8, 16x16, 32x32, 64x64, 128 x 128, và dạng tương tự) cũng như khối bất đối xứng bất kỳ (8x4, 16x8, và dạng tương tự).

Trong một số cách thi hành, và vẫn tham chiếu đến Fig.6, cây quyết định tứ phân cộng nhị phân (quadtree plus binary decision tree, QTBT) có thể được thi hành. Trong QTBT, tại mức đơn vị cây tạo mã, các tham số phân vùng của QTBT có thể được dẫn ra theo cách động để thích ứng với các đặc điểm cục bộ mà không cần truyền phí tổn bất kỳ. Tiếp đó, tại mức đơn vị tạo mã, cấu trúc cây quyết định bộ phân loại chung (joint-classifier) có thể loại trừ những lần lặp không cần thiết và kiểm soát rủi ro của việc dự đoán sai. Trong một số cách thi hành, chế độ cập nhật khối khung LTR có thể là khả dụng dưới dạng tùy chọn bổ sung khả dụng tại mọi nút lá của QTBT.

Trong một số cách thi hành, và vẫn tham chiếu đến Fig.6, các phần tử cú pháp bổ sung

có thể được báo hiệu tại các mức phân cấp khác nhau của dòng bit. Ví dụ, cờ có thể được cho phép thực hiện đối với toàn bộ chuỗi qua việc đưa vào cờ cho phép thực hiện được tạo mã trong tập tham số chuỗi (Sequence Parameter Set, SPS). Hơn nữa, cờ CTU có thể được tạo mã tại mức đơn vị cây tạo mã (CTU).

Cần lưu ý rằng một hoặc nhiều khía cạnh và phương án bất kỳ trong số các khía cạnh và các phương án được mô tả ở đây có thể được thi hành một cách thuận tiện nhờ sử dụng hệ mạch điện tử kỹ thuật số, hệ mạch tích hợp, các mạch tích hợp chuyên dụng (application specific integrated circuit, ASIC) được thiết kế đặc biệt, các mảng cổng lập trình được dạng trường (field programmable gate array, FPGA), phần cứng, phần sụn, phần mềm máy tính, và/hoặc những sự kết hợp của chúng, như được thực hiện và/hoặc thi hành trong một hoặc nhiều máy (ví dụ, một hoặc nhiều thiết bị tính toán mà chúng được sử dụng làm thiết bị tính toán người dùng cho tài liệu điện tử, một hoặc nhiều thiết bị máy chủ, chẳng hạn như máy chủ tài liệu, v.v.) được lập trình theo các nội dung bộc lộ của bản mô tả này, như sẽ trở nên rõ ràng đối với những người có trình độ trung bình trong lĩnh vực máy tính. Các khía cạnh hoặc các dấu hiệu khác nhau này có thể bao gồm cách thi hành trong một hoặc nhiều chương trình và/hoặc phần mềm máy tính mà chúng là có thể thực thi được và/hoặc có thể diễn dịch được trên hệ thống lập trình được bao gồm ít nhất một bộ xử lý lập trình được, mà nó có thể là chuyên dụng hoặc đa năng, được ghép nối để nhận dữ liệu và các lệnh từ, và để truyền dữ liệu và các lệnh tới, hệ thống lưu trữ, ít nhất một thiết bị đầu vào, và ít nhất một thiết bị đầu ra. Việc tạo mã phần mềm thích hợp hoàn toàn có thể được chuẩn bị bởi các lập trình viên có trình độ dựa trên các nội dung bộc lộ của sáng chế, như sẽ trở nên rõ ràng đối với những người có trình độ trung bình trong lĩnh vực phần mềm. Các khía cạnh và các cách thi hành nêu trên, sử dụng phần mềm và/hoặc các môđun phần mềm, cũng có thể bao gồm phần cứng thích hợp để hỗ trợ trong việc thi hành các lệnh có thể thực thi được bằng máy của phần mềm và/hoặc môđun phần mềm này.

Phần mềm này có thể là sản phẩm chương trình máy tính mà nó sử dụng phương tiện lưu trữ đọc được bằng máy. Phương tiện lưu trữ đọc được bằng máy có thể là phương tiện bất kỳ mà nó có khả năng lưu trữ và/hoặc mã hóa chuỗi các lệnh để thực thi bởi máy (ví

dụ, thiết bị tính toán) và khiến cho máy này thực hiện phương pháp và/hoặc phương án bất kỳ trong số các phương pháp và/hoặc các phương án được mô tả ở đây. Các ví dụ về phương tiện lưu trữ đọc được bằng máy bao gồm, nhưng không bị giới hạn ở, đĩa từ, đĩa quang (ví dụ, CD, CD-R, DVD, DVD-R, v.v.), đĩa từ quang, thiết bị nhớ chỉ đọc (read-only memory, "ROM"), thiết bị nhớ truy cập ngẫu nhiên (random access memory, "RAM"), thẻ từ, thẻ quang, thiết bị nhớ trạng thái rắn, EPROM, EEPROM, các thiết bị lôgic lập trình được (Programmable Logic Device, PLD), và/hoặc những sự kết hợp bất kỳ của chúng. Phương tiện đọc được bằng máy, như được sử dụng ở đây, được dự tính bao gồm một phương tiện đơn cũng như một tập hợp các phương tiện tách biệt về vật lý, ví dụ chẳng hạn như một tập hợp của các đĩa compăc hoặc một hoặc nhiều ổ đĩa cứng kết hợp với bộ nhớ máy tính. Như được sử dụng ở đây, phương tiện lưu trữ đọc được bằng máy không bao gồm các dạng chuyển tiếp của hoạt động truyền tín hiệu.

Phần mềm này cũng có thể bao gồm thông tin (ví dụ, dữ liệu) được mang dưới dạng tín hiệu dữ liệu trên bộ mang dữ liệu, chẳng hạn như sóng mang. Ví dụ, thông tin có thể thực thi bằng máy có thể được đưa vào dưới dạng tín hiệu mang dữ liệu được thực hiện trong bộ mang dữ liệu trong đó tín hiệu này mã hóa chuỗi lệnh, hoặc một phần của nó, để thực thi bởi máy (ví dụ, thiết bị tính toán) và thông tin liên quan bất kỳ (ví dụ, các cấu trúc dữ liệu và dữ liệu) mà nó khiến cho máy này thực hiện phương pháp và/hoặc phương án bất kỳ trong số các phương pháp và/hoặc các phương án được mô tả ở đây.

Các ví dụ về thiết bị tính toán bao gồm, nhưng không bị giới hạn ở, thiết bị đọc sách điện tử, trạm máy tính, máy tính đầu cuối, máy tính chủ, thiết bị cầm tay (ví dụ, máy tính bảng, điện thoại thông minh, v.v.), ứng dụng web, bộ định tuyến mạng, bộ chuyển mạch mạng, cầu nối mạng, máy bất kỳ có khả năng thực thi chuỗi lệnh mà chúng chỉ rõ hoạt động cần được thực hiện bởi máy đó, và những sự kết hợp bất kỳ của chúng. Trong một ví dụ, thiết bị tính toán có thể bao gồm và/hoặc được trang bị trong ki ốt.

Fig.7 thể hiện phép biểu diễn dạng biểu đồ của một phương án về thiết bị tính toán dưới dạng làm ví dụ của hệ thống máy tính 700 mà tập lệnh để khiến cho hệ thống điều khiển thực hiện một hoặc nhiều khía cạnh và/hoặc phương pháp bất kỳ trong số các khía cạnh và/hoặc các phương pháp của sáng chế có thể được thực thi bên trong nó. Việc nhiều

thiết bị tính toán có thể được sử dụng để thi hành tập lệnh được tạo cấu hình đặc biệt để khiến để khiến cho một hoặc nhiều thiết bị trong số các thiết bị này thực hiện một hoặc nhiều khía cạnh và/hoặc phương pháp bất kỳ trong số các khía cạnh và/hoặc các phương pháp của sáng chế cũng đã được dự liệu. Hệ thống máy tính 700 bao gồm bộ xử lý 704 và bộ nhớ 708 mà chúng truyền thông với nhau, và với các thành phần khác, thông qua bus 712. Bus 712 có thể bao gồm loại bất kỳ trong một số loại cấu trúc bus bao gồm, nhưng không bị giới hạn ở, bus bộ nhớ, bộ điều khiển bộ nhớ, bus ngoại vi, bus cục bộ, và những sự kết hợp bất kỳ của chúng, sử dụng kiến trúc bất kỳ trong số nhiều kiến trúc bus khác nhau.

Bộ nhớ 708 có thể bao gồm các thành phần khác nhau (ví dụ, các phương tiện đọc được bằng máy) bao gồm, nhưng không bị giới hạn ở, thành phần nhớ truy nhập ngẫu nhiên, thành phần chỉ đọc, và những sự kết hợp bất kỳ của chúng. Trong một ví dụ, hệ thống vào/ra cơ bản 716 (basic input/output system, BIOS), bao gồm các đoạn chương trình cơ bản mà chúng giúp truyền thông tin giữa các phần tử bên trong hệ thống máy tính 700, chẳng hạn như khi khởi động, có thể được lưu trữ trong bộ nhớ 708. Bộ nhớ 708 cũng có thể bao gồm (ví dụ, được lưu trữ trên một hoặc nhiều phương tiện đọc được bằng máy) các lệnh (ví dụ, phần mềm) 720 thực hiện một hoặc nhiều khía cạnh và/hoặc phương pháp bất kỳ trong số các khía cạnh và/hoặc các phương pháp của sáng chế. Trong một ví dụ khác, bộ nhớ 708 có thể còn bao gồm số lượng bất kỳ của các môđun chương trình bao gồm, nhưng không bị giới hạn ở, hệ điều hành, một hoặc nhiều chương trình ứng dụng, các môđun chương trình khác, dữ liệu chương trình, và những sự kết hợp bất kỳ của chúng.

Hệ thống máy tính 700 cũng có thể bao gồm thiết bị lưu trữ 724. Các ví dụ về thiết bị lưu trữ (ví dụ, thiết bị lưu trữ 724) bao gồm, nhưng không bị giới hạn ở, ổ đĩa cứng, ổ đĩa từ, ổ đĩa quang kết hợp với phương tiện quang, thiết bị nhớ trạng thái rắn, và những sự kết hợp bất kỳ của chúng. Thiết bị lưu trữ 724 có thể được kết nối với bus 712 bởi giao diện thích hợp (không được thể hiện). Các giao diện ví dụ bao gồm, nhưng không bị giới hạn ở, SCSI, gắn kèm công nghệ tiên tiến (advanced technology attachment, ATA), ATA nối tiếp, bus nối tiếp vạn năng (universal serial bus, USB), IEEE 1394 (FIREWIRE), và những sự kết hợp bất kỳ của chúng. Trong một ví dụ, thiết bị lưu trữ 724 (hoặc một hoặc nhiều thành

phần của nó) có thể được giao tiếp theo cách tháo ra được với hệ thống máy tính 700 (ví dụ, thông qua bộ kết nối cổng ngoài (không được thể hiện)). Cụ thể, thiết bị lưu trữ 724 và phương tiện đọc được bằng máy được liên kết 728 có thể cung cấp bộ lưu trữ bất khả biến và/hoặc khả biến của các lệnh đọc được bằng máy, các cấu trúc dữ liệu, các môđun chương trình, và/hoặc dữ liệu khác cho hệ thống máy tính 700. Trong một ví dụ, phần mềm 720 có thể nằm, hoàn toàn hoặc một phần, bên trong phương tiện đọc được bằng máy 728. Trong một ví dụ khác, phần mềm 720 có thể nằm, hoàn toàn hoặc một phần, bên trong bộ xử lý 704.

Hệ thống máy tính 700 cũng có thể bao gồm thiết bị đầu vào 732. Trong một ví dụ, người dùng của hệ thống máy tính 700 có thể nhập các mệnh lệnh và/hoặc thông tin khác vào trong hệ thống máy tính 700 thông qua thiết bị đầu vào 732. Các ví dụ về thiết bị đầu vào 732 bao gồm, nhưng không bị giới hạn ở, thiết bị đầu vào chữ và số (ví dụ, bàn phím), thiết bị trỏ, cần điều khiển, tay cầm điều khiển trò chơi, thiết bị đầu vào âm thanh (ví dụ, micrô, hệ thống đáp ứng giọng nói, v.v.), thiết bị điều khiển con trỏ (ví dụ, con chuột), tấm chạm, máy quét quang học, thiết bị chụp viđeo (ví dụ, camera tĩnh, camera viđeo), màn hình cảm ứng chạm, và những sự kết hợp bất kỳ của chúng. Thiết bị đầu vào 732 có thể được giao tiếp với bus 712 thông qua giao diện bất kỳ trong số nhiều giao diện khác nhau (không được thể hiện) bao gồm, nhưng không bị giới hạn ở, giao diện nối tiếp, giao diện song song, cổng trò chơi, giao diện USB, giao diện FIREWIRE, giao diện trực tiếp với bus 712, và những sự kết hợp bất kỳ của chúng. Thiết bị đầu vào 732 có thể bao gồm giao diện màn hình chạm mà nó có thể là một phần của hoặc tách biệt khỏi bộ hiển thị 736, được mô tả thêm bên dưới. Thiết bị đầu vào 732 có thể được sử dụng làm thiết bị chọn của người dùng để chọn một hoặc nhiều biểu diễn đồ họa trong giao diện đồ họa như được mô tả trên đây.

Người dùng cũng có thể nhập các mệnh lệnh và/hoặc thông tin khác vào hệ thống máy tính 700 thông qua thiết bị lưu trữ 724 (ví dụ, ổ đĩa tháo ra được, ổ đĩa tác động nhanh, v.v.) và/hoặc thiết bị giao diện mạng 740. Thiết bị giao diện mạng, chẳng hạn như thiết bị giao diện mạng 740, có thể được sử dụng để kết nối hệ thống máy tính 700 với một hoặc nhiều mạng trong số các loại mạng khác nhau, chẳng hạn như mạng 744, và một hoặc nhiều thiết

bị ở xa 748 được kết nối vào đó. Các ví dụ về thiết bị giao diện mạng bao gồm, nhưng không bị giới hạn ở, cạc giao diện mạng (ví dụ, cạc giao diện mạng di động, cạc LAN), môđem, và sự kết hợp bất kỳ của chúng. Các ví dụ về mạng bao gồm, nhưng không bị giới hạn ở, mạng diện rộng (ví dụ, Internet, mạng doanh nghiệp), mạng cục bộ (ví dụ, mạng được liên kết với văn phòng, tòa nhà, khuôn viên hoặc không gian địa lý tương đối nhỏ khác), mạng điện thoại, mạng dữ liệu được liên kết với nhà cung cấp điện thoại/thoại (ví dụ, mạng dữ liệu và/hoặc thoại của nhà cung cấp truyền thông di động), kết nối trực tiếp giữa hai thiết bị tính toán, và những sự kết hợp bất kỳ của chúng. Mạng, chẳng hạn như mạng 744, có thể sử dụng chế độ truyền thông có dây và/hoặc không dây. Nhìn chung, cấu trúc liên kết mạng bất kỳ có thể được sử dụng. Thông tin (ví dụ, dữ liệu, phần mềm 720, v.v.) có thể được truyền thông tới và/hoặc từ hệ thống máy tính 700 thông qua thiết bị giao diện mạng 740.

Hệ thống máy tính 700 có thể còn bao gồm bộ thích ứng hiển thị viđeo 752 để truyền thông ảnh có thể hiển thị được tới thiết bị hiển thị, chẳng hạn như thiết bị hiển thị 736. Các ví dụ về thiết bị hiển thị bao gồm, nhưng không bị giới hạn ở, bộ hiển thị tinh thể lỏng (liquid crystal display, LCD), đèn tia âm cực (cathode ray tube, CRT), bộ hiển thị plasma, bộ hiển thị điôt phát sáng (light emitting diode, LED), và những sự kết hợp bất kỳ của chúng. Bộ thích ứng hiển thị 752 và thiết bị hiển thị 736 có thể được sử dụng kết hợp với bộ xử lý 704 để cung cấp các biểu diễn đồ họa của các khía cạnh của sáng chế. Ngoài thiết bị hiển thị, hệ thống máy tính 700 có thể bao gồm một hoặc nhiều thiết bị đầu ra ngoại vi khác bao gồm, nhưng không bị giới hạn ở, loa âm thanh, máy in, và những sự kết hợp bất kỳ của chúng. Các thiết bị đầu ra ngoại vi này có thể được kết nối với bus 712 thông qua giao diện ngoại vi 756. Các ví dụ về giao diện ngoại vi bao gồm, nhưng không bị giới hạn ở, cổng nối tiếp, kết nối USB, kết nối FIREWIRE, kết nối song song, và những sự kết hợp bất kỳ của chúng.

Phần nêu trên là phần mô tả chi tiết của các phương án minh họa của sáng chế. Các cải biến và những sự bổ sung khác nhau có thể được thực hiện mà không tách rời khỏi nguyên lý và phạm vi của sáng chế. Các dấu hiệu của mỗi phương án trong số các phương án khác nhau đã được mô tả trên đây có thể được kết hợp với các dấu hiệu của các phương

án khác đã được mô tả theo cách thích hợp để cung cấp nhiều cách kết hợp về dấu hiệu trong các phương án mới có liên quan. Hơn thế nữa, tuy phần nêu trên mô tả một số phương án riêng biệt, những gì đã được mô tả ở đây chỉ đơn thuần mang tính minh họa về việc ứng dụng các nguyên lý của sáng chế. Thêm vào đó, mặc dù các phương pháp cụ thể ở đây có thể được minh họa và/hoặc được mô tả là được thực hiện theo một thứ tự nhất định, việc sắp xếp thứ tự này có khả năng biến thiên cao trong phạm vi trình độ trung bình để đạt được các phương án như được bộc lộ ở đây. Theo đó, phần mô tả này chỉ nhằm để lấy làm ví dụ, và theo cách khác không giới hạn phạm vi của sáng chế.

Trong các phần mô tả trên đây và trong các điểm yêu cầu bảo hộ, các cụm từ như "ít nhất một" hoặc "một hoặc nhiều" có thể xuất hiện theo sau bởi danh sách liên hợp của các phần tử hoặc các dấu hiệu. Thuật ngữ "và/hoặc" cũng có thể xuất hiện trong danh sách của hai hay nhiều phần tử hoặc dấu hiệu. Trừ khi được phủ nhận theo cách khác theo cách không tường minh hoặc tường minh bởi ngữ cảnh mà nó được sử dụng trong đó, cụm từ như vậy nhằm đề cập tới riêng phần tử hoặc dấu hiệu bất kỳ trong số các phần tử hoặc các dấu hiệu được liệt kê, hoặc phần tử hoặc dấu hiệu bất kỳ trong số các phần tử hoặc các dấu hiệu đã nêu kết hợp với phần tử hoặc dấu hiệu bất kỳ trong số các phần tử hoặc các dấu hiệu đã nêu khác. Ví dụ, mỗi cụm từ trong số "ít nhất một thành phần trong số A và B;" "một hoặc nhiều thành phần trong số A và B;" và "A và/hoặc B" nhằm đề cập tới "chỉ riêng A, chỉ riêng B, hoặc A và B cùng với nhau." Cách diễn giải tương tự cũng là dành cho các danh sách bao gồm ba hay hơn ba khoản mục. Ví dụ, mỗi cụm từ trong số "ít nhất một thành phần trong số A, B, và C;" "một hoặc nhiều thành phần trong số A, B, và C;" và "A, B, và/hoặc C" nhằm đề cập tới "chỉ riêng A, chỉ riêng B, chỉ riêng C, A và B cùng với nhau, A và C cùng với nhau, B và C cùng với nhau, hoặc A và B và C cùng với nhau." Thêm vào đó, việc sử dụng thuật ngữ "dựa trên" trên đây và trong các điểm yêu cầu bảo hộ là để có nghĩa là, "dựa ít nhất một phần trên," sao cho dấu hiệu hoặc phần tử không được nêu cũng có thể được cho phép.

Đối tượng được mô tả ở đây có thể được thực hiện trong các hệ thống, các thiết bị, các phương pháp, và/hoặc các vật phẩm phụ thuộc vào cấu hình mong muốn. Các cách thi hành được nêu trong phần mô tả trên đây không thể hiện tất cả các cách thi hành nhất quán

với đối tượng được mô tả ở đây. Thay vào đó, chúng chỉ đơn thuần là một số ví dụ nhất quán với các khía cạnh liên quan đến đối tượng được mô tả. Mặc dù một vài biến thể đã được mô tả một cách chi tiết trên đây, có thể có các cải biến hoặc các bổ sung khác. Cụ thể là, thêm các dấu hiệu và/hoặc các biến thể khác có thể được cung cấp bổ sung cho những gì đã được nêu ra ở đây. Ví dụ, các cách thi hành được mô tả trên đây có thể được hướng tới các tổ hợp và các tổ hợp con khác nhau của các dấu hiệu đã được bộc lộ và/hoặc các tổ hợp và các tổ hợp con của một số dấu hiệu khác đã được bộc lộ trên đây. Thêm vào đó, các luồng logic được thể hiện trên các hình vẽ kèm theo và/hoặc đã được mô tả ở đây không nhất thiết yêu cầu thứ tự cụ thể được thể hiện, hoặc thứ tự tuần tự, để đạt được các kết quả mong muốn. Các cách thi hành khác có thể nằm trong phạm vi của các điểm yêu cầu bảo hộ sau đây.

YÊU CẦU BẢO HỘ

1. Bộ giải mã được tạo cấu hình để:

nhận dòng bit bao gồm hình ảnh được tạo mã, hình ảnh được tạo mã này bao gồm vùng thứ nhất có chuyển động toàn cục và bao gồm nhiều đơn vị tạo mã liền kề thứ nhất và vùng thứ hai có chuyển động cục bộ khác với chuyển động toàn cục này và bao gồm nhiều đơn vị tạo mã liền kề thứ hai, tất cả các đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất này có mô hình chuyển động afin giống nhau;

xây dựng, cho mỗi đơn vị tạo mã trong số các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ nhất này, danh sách ứng viên vectơ chuyển động, mỗi danh sách ứng viên vectơ chuyển động bao gồm một hoặc nhiều ứng viên vectơ chuyển động được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận;

giải mã mỗi đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất này của vùng thứ nhất nhờ sử dụng một ứng viên được chọn trong số các ứng viên vectơ chuyển động từ danh sách ứng viên vectơ chuyển động của đơn vị tạo mã này, ứng viên vectơ chuyển động được chọn này là ứng viên được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận, trong đó mô hình chuyển động afin là mô hình chuyển động afin 6 tham số đặc trưng bởi các tham số a, b, c, d, e và f trong đó:

$x' = ax + by + c$

$y' = dx + ey + f$

(x,y) là vị trí điểm ảnh trong vùng thứ nhất của hình ảnh được tạo mã và (x', y') là vị trí điểm ảnh trong hình ảnh tham chiếu, trong đó mỗi ứng viên vectơ chuyển động được chọn bao gồm ba vectơ chuyển động của điểm điều khiển, và trong đó các ứng viên vectơ chuyển động được chọn thi hành a, b, c, d, e và f với các giá trị về cơ bản là giống nhau trong tất cả các đơn vị tạo mã của vùng thứ nhất, qua đó chuyển động toàn cục trong vùng thứ nhất được xây dựng lại; và

giải mã các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ hai này, qua đó chuyển động cục bộ được xây dựng lại.

2. Bộ giải mã theo điểm 1 được tạo cấu hình để làm rõ, từ dòng bit, các vectơ chuyển động được xác định riêng rẽ cho mỗi đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ hai đã nêu, các đơn vị tạo mã liền kề trong nhiều đơn vị tạo mã thứ hai này có các vectơ chuyển động được xác định riêng rẽ khác nhau.

3. Bộ giải mã theo điểm 1, trong đó vùng thứ nhất và vùng thứ hai bao gồm toàn bộ hình ảnh.

4. Bộ giải mã được tạo cấu hình để:

nhận dòng bit bao gồm hình ảnh được tạo mã, hình ảnh được tạo mã này bao gồm vùng thứ nhất có chuyển động toàn cục và bao gồm nhiều đơn vị tạo mã liền kề thứ nhất và vùng thứ hai có chuyển động cục bộ khác với chuyển động toàn cục này và bao gồm nhiều đơn vị tạo mã liền kề thứ hai, tất cả các đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất này có mô hình chuyển động afin giống nhau;

xây dựng, cho mỗi đơn vị tạo mã trong số các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ nhất này, danh sách ứng viên vectơ chuyển động, mỗi danh sách ứng viên vectơ chuyển động bao gồm một hoặc nhiều ứng viên vectơ chuyển động được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận;

giải mã mỗi đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất của vùng thứ nhất nhờ sử dụng một ứng viên được chọn trong số các ứng viên vectơ chuyển động từ danh sách ứng viên vectơ chuyển động của đơn vị tạo mã này, ứng viên vectơ chuyển động được chọn này là ứng viên được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận, trong đó mô hình chuyển động afin là mô hình chuyển động afin 4 tham số đặc trưng bởi các tham số a, b, c, và f trong đó:

$x' = ax + by + c$

$y' = - bx + ay + f$

(x,y) là vị trí điểm ảnh trong vùng thứ nhất của hình ảnh được tạo mã và (x', y') là vị trí điểm ảnh trong hình ảnh tham chiếu, trong đó mỗi ứng viên vectơ chuyển động được chọn bao gồm hai vectơ chuyển động của điểm điều khiển, và trong đó các ứng viên vectơ chuyển động được chọn thi hành a, b, c, và f với các giá trị về cơ bản là giống nhau trong tất cả các đơn vị tạo mã của vùng thứ nhất, qua đó chuyển động toàn cục trong vùng thứ

nhất được xây dựng lại; và

giải mã các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ hai này, qua đó chuyển động cục bộ được xây dựng lại.

5. Bộ mã hóa viđeo để mã hóa dòng bit để được giải mã bởi bộ giải mã, bộ giải mã này được tạo cấu hình để:

nhận dòng bit bao gồm hình ảnh được tạo mã, hình ảnh được tạo mã này bao gồm vùng thứ nhất có chuyển động toàn cục và bao gồm nhiều đơn vị tạo mã liền kề thứ nhất và vùng thứ hai có chuyển động cục bộ khác với chuyển động toàn cục này và bao gồm nhiều đơn vị tạo mã liền kề thứ hai, tất cả các đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất này có mô hình chuyển động afin giống nhau;

xây dựng, cho mỗi đơn vị tạo mã trong số các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ nhất này, danh sách ứng viên vectơ chuyển động, mỗi danh sách ứng viên vectơ chuyển động bao gồm một hoặc nhiều ứng viên vectơ chuyển động được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận;

giải mã mỗi đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất này của vùng thứ nhất nhờ sử dụng một ứng viên được chọn trong số các ứng viên vectơ chuyển động từ danh sách ứng viên vectơ chuyển động của đơn vị tạo mã này, ứng viên vectơ chuyển động được chọn này là ứng viên được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận, trong đó mô hình chuyển động afin là mô hình chuyển động afin 6 tham số đặc trưng bởi các tham số a, b, c, d, e và f trong đó:

$x' = ax + by + c$

$y' = dx + ey + f$

(x,y) là vị trí điểm ảnh trong vùng thứ nhất của hình ảnh được tạo mã và (x', y') là vị trí điểm ảnh trong hình ảnh tham chiếu, trong đó mỗi ứng viên vectơ chuyển động được chọn bao gồm ba vectơ chuyển động của điểm điều khiển, và trong đó các ứng viên vectơ chuyển động được chọn thi hành a, b, c, d, e và f với các giá trị về cơ bản là giống nhau trong tất cả các đơn vị tạo mã của vùng thứ nhất, qua đó chuyển động toàn cục trong vùng thứ nhất được xây dựng lại; và

giải mã các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ hai này, qua đó chuyển động

cục bộ được xây dựng lại.

6. Bộ mã hóa viđeo để mã hóa dòng bit để được giải mã bởi bộ giải mã, bộ giải mã này được tạo cấu hình để:

nhận dòng bit bao gồm hình ảnh được tạo mã, hình ảnh được tạo mã này bao gồm vùng thứ nhất có chuyển động toàn cục và bao gồm nhiều đơn vị tạo mã liền kề thứ nhất và vùng thứ hai có chuyển động cục bộ khác với chuyển động toàn cục này và bao gồm nhiều đơn vị tạo mã liền kề thứ hai, tất cả các đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất này có mô hình chuyển động afin giống nhau;

xây dựng, cho mỗi đơn vị tạo mã trong số các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ nhất này, danh sách ứng viên vectơ chuyển động, mỗi danh sách ứng viên vectơ chuyển động bao gồm một hoặc nhiều ứng viên vectơ chuyển động được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận;

giải mã mỗi đơn vị tạo mã trong số nhiều đơn vị tạo mã thứ nhất của vùng thứ nhất nhờ sử dụng một ứng viên được chọn trong số các ứng viên vectơ chuyển động từ danh sách ứng viên vectơ chuyển động của đơn vị tạo mã này, ứng viên vectơ chuyển động được chọn này là ứng viên được tạo ra nhờ sử dụng thông tin vectơ chuyển động của đơn vị tạo mã lân cận, trong đó mô hình chuyển động afin là mô hình chuyển động afin 4 tham số đặc trưng bởi các tham số a, b, c, và f trong đó:

$x' = ax + by + c$

$y' = - bx + ay + f$

(x,y) là vị trí điểm ảnh trong vùng thứ nhất của hình ảnh được tạo mã và (x', y') là vị trí điểm ảnh trong hình ảnh tham chiếu, trong đó mỗi ứng viên vectơ chuyển động được chọn bao gồm hai vectơ chuyển động của điểm điều khiển, và trong đó các ứng viên vectơ chuyển động được chọn thi hành a, b, c, và f với các giá trị về cơ bản là giống nhau trong tất cả các đơn vị tạo mã của vùng thứ nhất, qua đó chuyển động toàn cục trong vùng thứ nhất được xây dựng lại; và

giải mã các đơn vị tạo mã trong nhiều đơn vị tạo mã thứ hai này, qua đó chuyển động cục bộ được xây dựng lại.

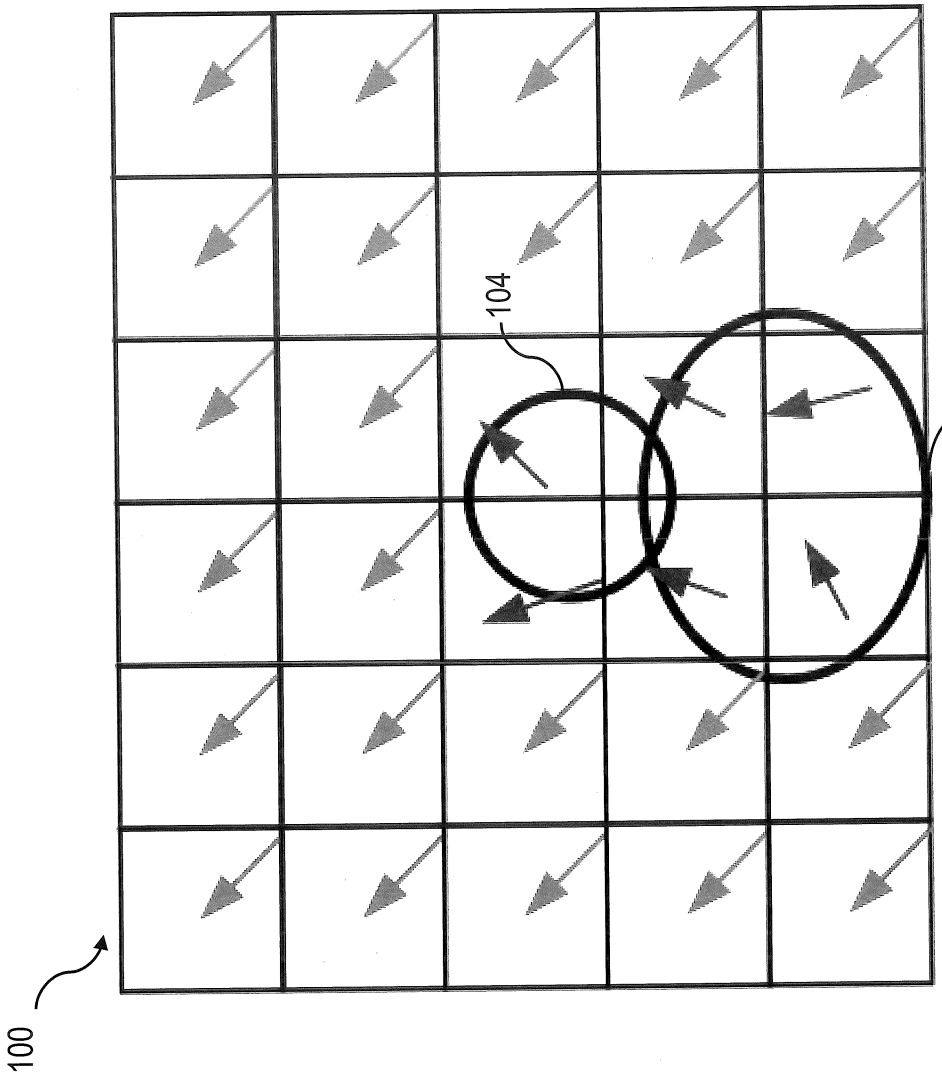
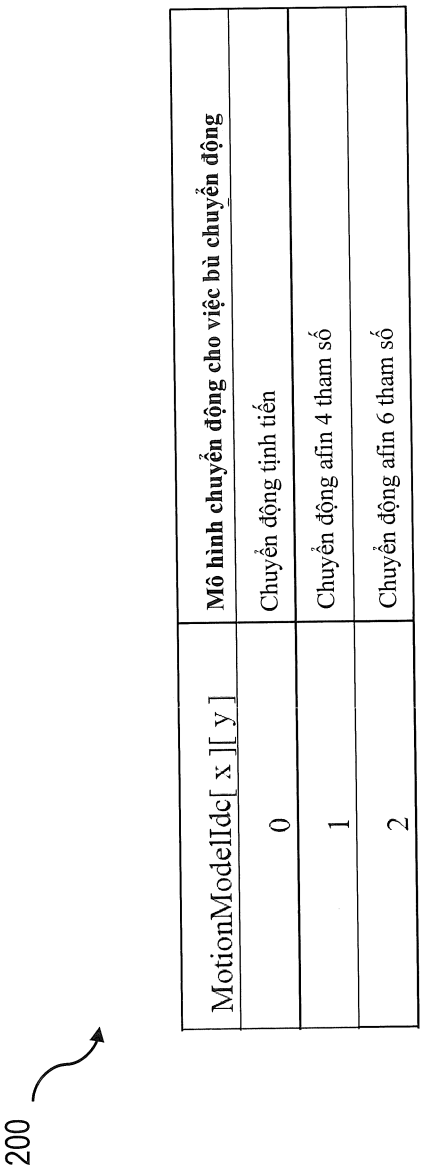


FIG. 1



MotionModelIdc[x][y]	**Mô hình chuyển động cho việc bù chuyển động**
0	Chuyển động tịnh tiến
1	Chuyển động afin 4 tham số
2	Chuyển động afin 6 tham số

FIG. 2

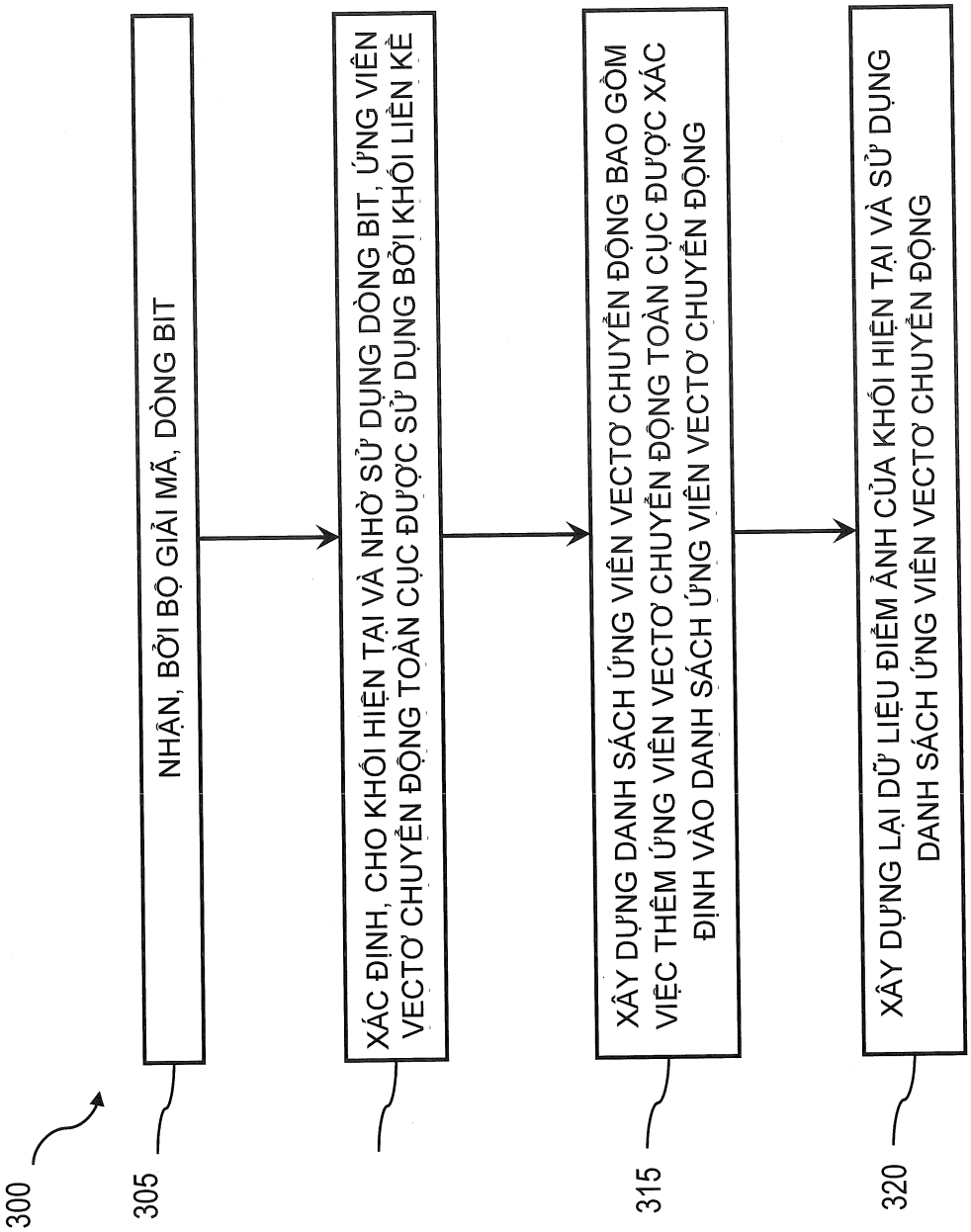

300
305
NHẬN, BỞI BỘ GIẢI MÃ, DÒNG BIT
XÁC ĐỊNH, CHO KHỐI HIỆN TẠI VÀ NHỜ SỬ DỤNG DÒNG BIT, ỨNG VIÊN VECTƠ CHUYỂN ĐỘNG TOÀN CỤC ĐƯỢC SỬ DỤNG BỞI KHỐI LIỀN KỀ
315
XÂY DỰNG DANH SÁCH ỨNG VIÊN VECTƠ CHUYỂN ĐỘNG BAO GỒM VIỆC THÊM ỨNG VIÊN VECTƠ CHUYỂN ĐỘNG TOÀN CỤC ĐƯỢC XÁC ĐỊNH VÀO DANH SÁCH ỨNG VIÊN VECTƠ CHUYỂN ĐỘNG
320
XÂY DỰNG LẠI DỮ LIỆU ĐIỂM ẢNH CỦA KHỐI HIỆN TẠI VÀ SỬ DỤNG DANH SÁCH ỨNG VIÊN VECTƠ CHUYỂN ĐỘNG

FIG. 3

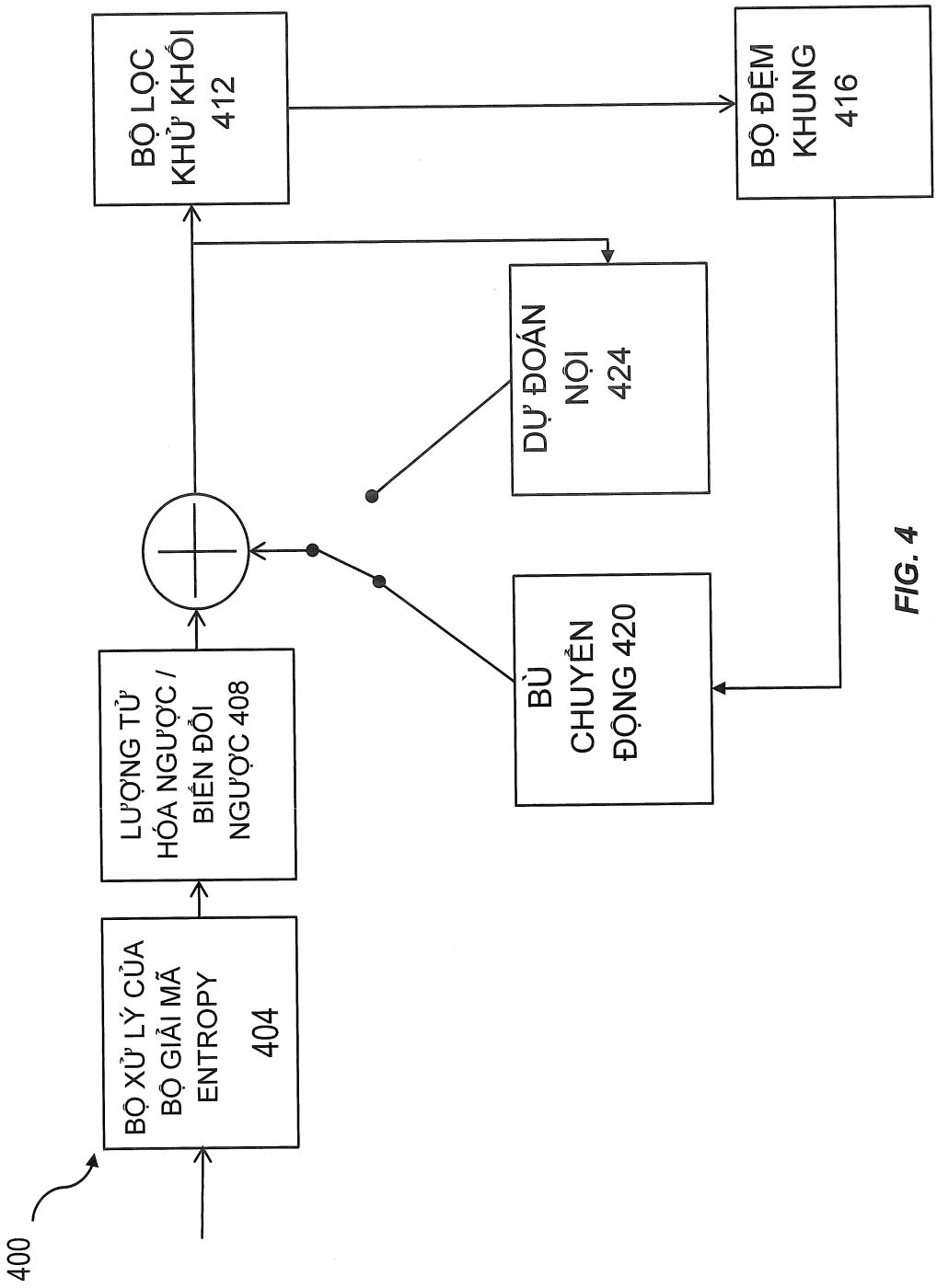

400
BỘ XỬ LÝ CỦA BỘ GIẢI MÃ ENTROPY 404
LƯỢNG TỬ HÓA NGƯỢC / BIẾN ĐỔI NGƯỢC 408
BÙ CHUYỂN ĐỘNG 420
DỰ ĐOÁN NỘI 424
BỘ LỌC KHỬ KHỐI 412
BỘ ĐỆM KHUNG 416

FIG. 4

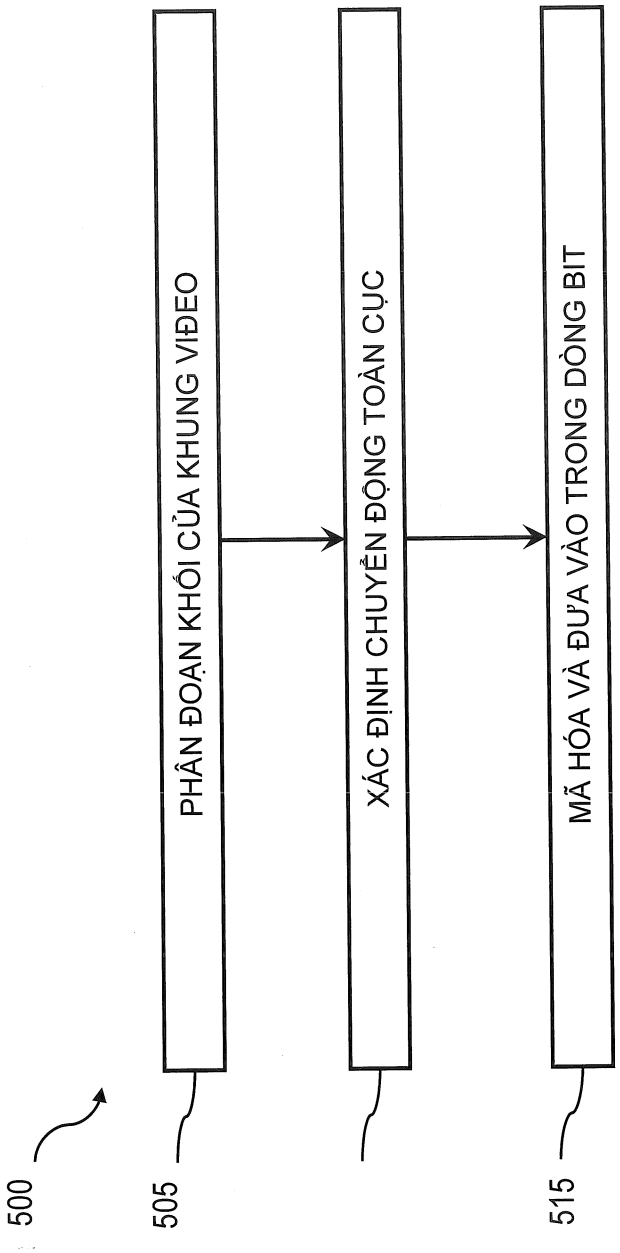

500
505
PHÂN ĐOẠN KHỐI CỦA KHUNG VIĐEO
XÁC ĐỊNH CHUYỂN ĐỘNG TOÀN CỤC
515
MÃ HÓA VÀ ĐƯA VÀO TRONG DÒNG BIT

FIG. 5

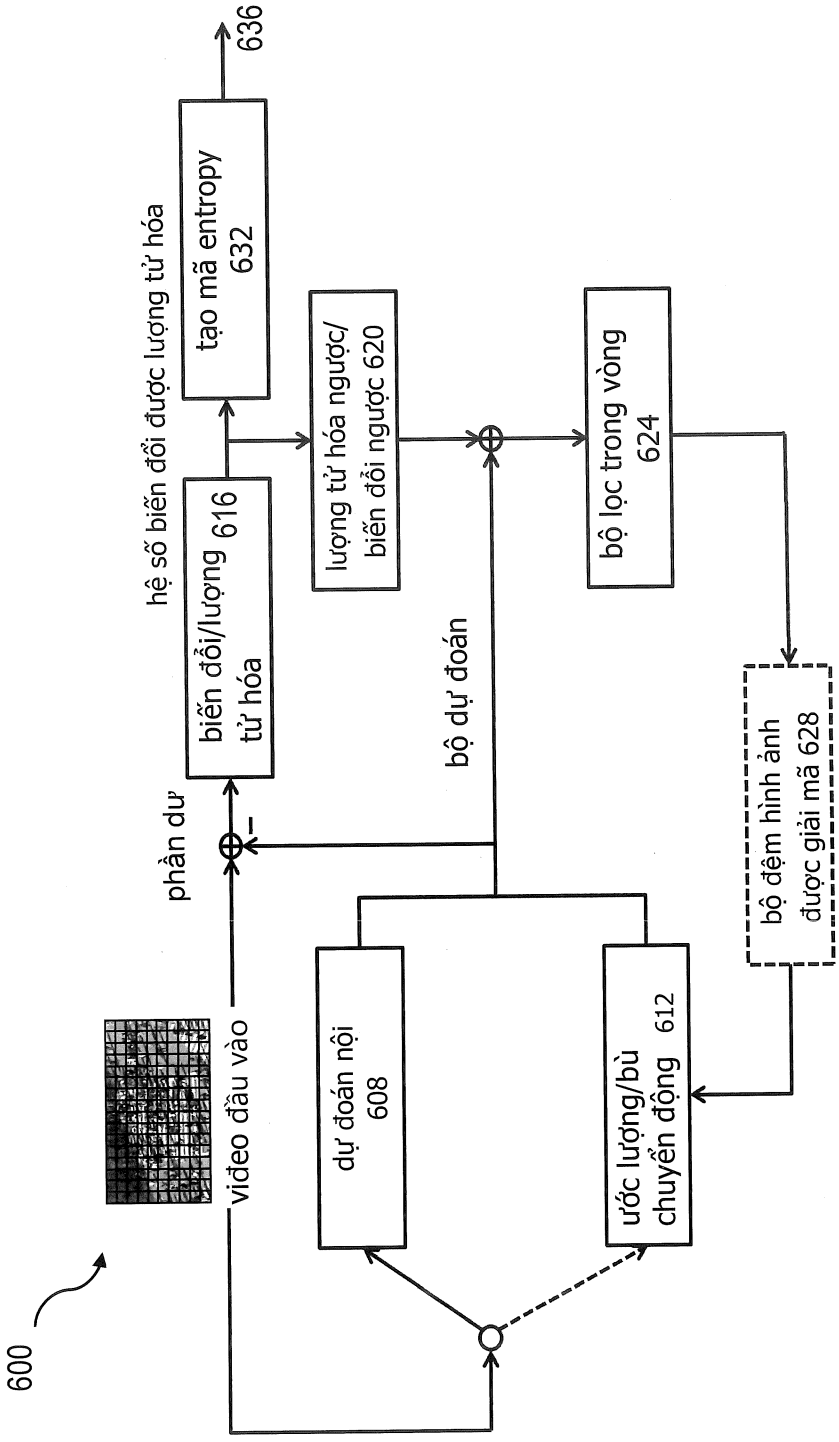

600
video đầu vào
phần dư
biến đổi/lượng tử hóa 616
hệ số biến đổi được lượng tử hóa
tạo mã entropy 632
636
lượng tử hóa ngược/ biến đổi ngược 620
bộ dự đoán
dự đoán nội 608
ước lượng/bù chuyển động 612
bộ lọc trong vòng 624
bộ đệm hình ảnh được giải mã 628

FIG. 6

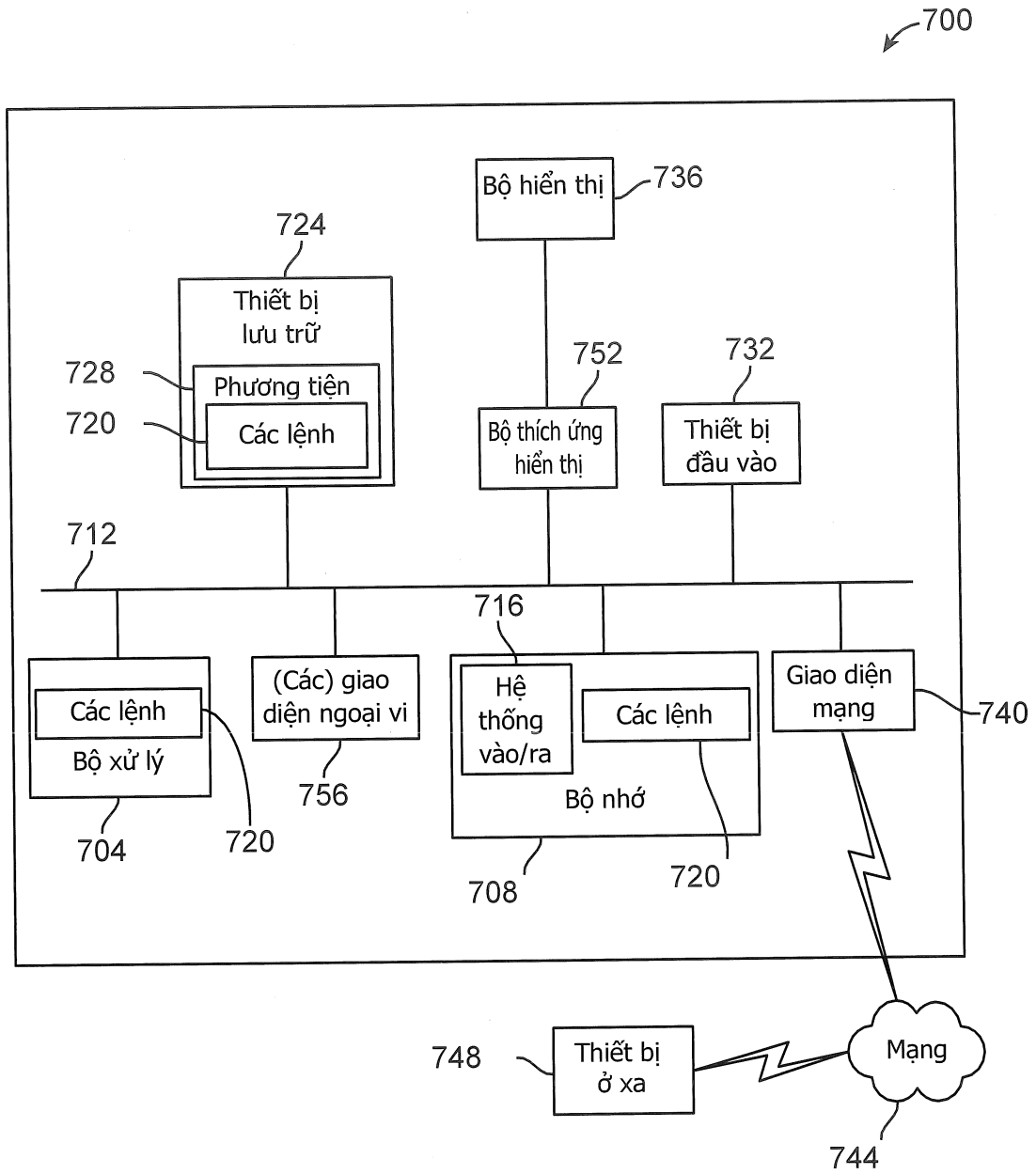

700
Bộ hiển thị
736
724
Thiết bị lưu trữ
728
Phương tiện
720
Các lệnh
752
732
Bộ thích ứng hiển thị
Thiết bị đầu vào
712
716
Các lệnh
Bộ xử lý
704
720
(Các) giao diện ngoại vi
756
Hệ thống vào/ra
Các lệnh
Bộ nhớ
708
720
Giao diện mạng
740
748
Thiết bị ở xa
Mạng
744

FIG. 7