

	(19) 대한민국특허청(KR) (12) 공개특허공보(A)	(11) 공개번호 10-2017-0116879 (43) 공개일자 2017년10월20일
(51) 국제특허분류(Int. Cl.) H04N 19/513 (2014.01) H04N 19/139 (2014.01) H04N 19/573 (2014.01)	(71) 출원인 인하대학교 산학협력단 인천광역시 남구 인화로 100, 인하대학교 (용현동)	
(52) CPC특허분류 H04N 19/513 (2015.01) H04N 19/139 (2015.01)	(72) 발명자 송병철 서울특별시 양천구 목동서로 38, 125-307(목동, 목동1단지아파트)	
(21) 출원번호 10-2016-0045108	(74) 대리인 성병기, 최윤서	
(22) 출원일자 2016년04월12일 심사청구일자 없음		
전체 청구항 수 : 총 1 항		
(54) 발명의 명칭 패턴 정합 기반 움직임 벡터 생성 기법을 이용한 동영상 부호화 및 복호화 방법 및 장치		

(57) 요약

패턴 정합 기반 움직임 벡터 생성 기법을 이용하여 동영상을 복호화하는 방법 및 장치를 제공한다.

(52) CPC특허분류

H04N 19/573 (2015.01)

명세서

청구범위

청구항 1

패턴 정합 기반 움직임 벡터 생성 기법을 이용하여 동영상상을 복호화하는 방법.

발명의 설명

기술 분야

[0001] 본 발명은 동영상 부호화 및 복호화에 관한 것이다.

배경 기술

[0002] 비디오 영상은 시공간적 중복성 및 시점 간 중복성을 제거하여 압축 부호화되며, 이는 통신 회선을 통해 전송되거나 저장 매체에 적합한 형태로 저장될 수 있다.

발명의 내용

해결하려는 과제

[0003] 본 발명은 비디오 신호의 코딩 효율을 향상시키고자 함에 있다.

과제의 해결 수단

[0004] 상기 과제를 해결하기 위하여 본 발명은 패턴 정합 기반의 움직임 벡터 생성 기법을 이용하여 동영상상을 부호화 또는 복호화하는 방법 및 장치를 제공한다.

발명의 효과

[0005] 본 발명에 따르면, 패턴 정합 기반의 움직임 벡터 생성 기법을 이용하여 비디오 신호 코딩 효율을 향상시킬 수 있다.

발명을 실시하기 위한 구체적인 내용

[0006] 동영상 압축 기술 표준인 MPEG은 동영상의 해상도 향상과 함께 압축률 향상이라는 목적 하에 지속적으로 발전하여 왔다. 2000년대 초반 H.264/MPEG4 AVC 표준이 MPEG2 대비 2배의 압축률을 보이며 제정되었고, 2010년대 초반 HEVC는 다시 H.264 대비 2배의 압축률을 보이며 제정이 되었다. 최근 Full HD를 넘어 UHD 즉, 4K/8K급 콘텐츠에 대한 수요가 증가하고 이런 trend에 선대응하기 위한 노력으로서 MPEG은 2015년부터 차세대 CODEC에 대한 연구를 착수하였다. ISO와 ITU 전문가들이 모여 JVET (Joint Video Exploration Team)을 조직하였고, 현재 HEVC 대비 압축률을 향상시키기 위한 다양한 tool들이 내장된 Joint Exploration Model (JEM) CODEC 알고리즘을 만들고 있는 중이다.

[0007] 본 발명은 JEM의 압축률을 향상시키기 위한 새로운 알고리즘을 제안하고자 한다. HEVC와 달리 JEM와 같은 차세대 CODEC에서는 프레임율 변환 기술을 사용한 블록 단위 움직임 벡터 생성 기법이 존재한다. Bilateral matching과 template matching 등 두 가지 방법이 frame rate up conversion (FRUC) 모드로 이용될 수 있다. 또한, 본 발명은 두 방법을 합쳐서 성능을 올릴 수 있는 방법을 더 제시한다.

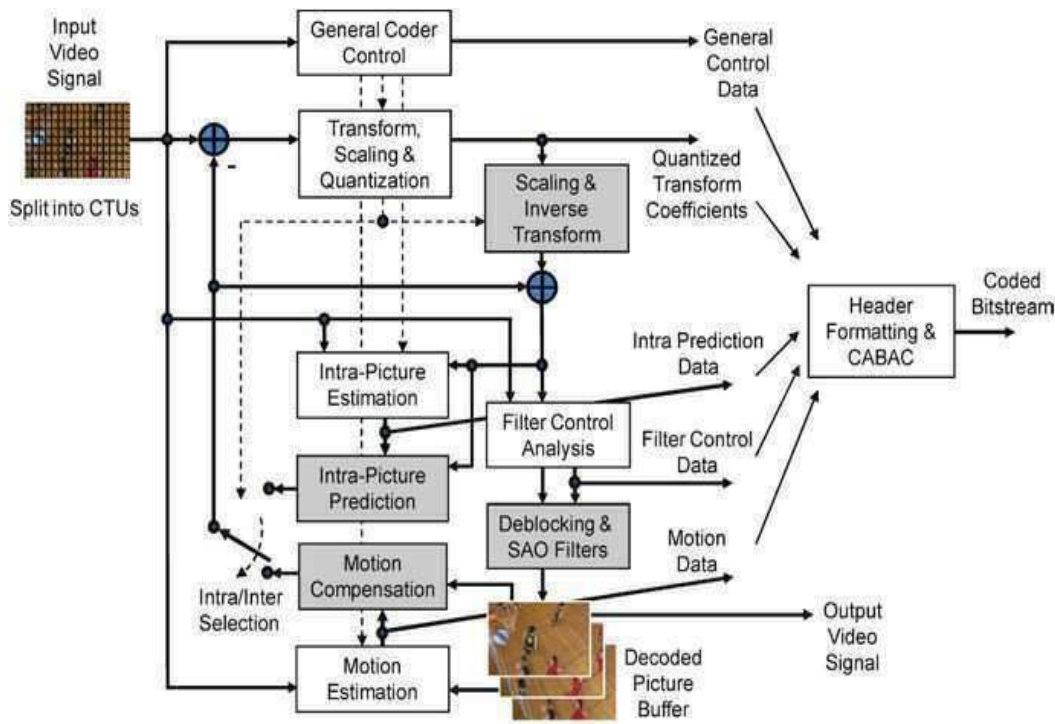


그림 1. HEVC 인코더 블록도

HEVC는 이전의 동영상 압축 표준과 마찬가지로 블록 단위의 하이브리드 영상 압축 기술을 기반으로 영상 압축을 수행한다. 즉, 입력 영상을 일정 블록 단위로 분할한 후, 각 블록에 대해서 예측, 변환, 양자화, 엔트로피 부호화 과정을 수행한다. 그림 1은 전형적인 HEVC의 인코더를 보이는데, 샘플 적응적 오프셋(SAO) 기술이 추가된 것을 제외하면 큰 그림 측면에서 H.264와 구조는 동일하다.

입력 동영상 각 프레임은 그림 1과 같이 CTU(coding tree unit)이라는 단위로 먼저 분할하고, 분할된 각 CTU에 대해서 순차적으로 encoding을 수행한다. 예를 들면, CTU는 64x64 휘도성분 (Y) 블록과 이에 대응하는 32x32크기의 두 개의 색차성분 (Cb, Cr)의 블록들이라고 생각하자. HEVC 인코더는 각 CTU를 효과적으로 인코딩하기 위해서, CU (coding unit)라는 블록으로 분할하여 인코딩할 수 있다. 예를 들어, CTU가 64x64일 때, 이를 네 개의 32x32 블록으로 분할하여 부호화할 수 있는데, 이 때 각 32x32블록이 CU가 된다. 보다 일반적으로 CTU는 쿼드트리 (quad-tree)형태로 다계층 분할 가능하다. 즉, CU는 CTU보다 작은 블록이고 계층적 측면에서 하위 블록이다. CU는 HEVC에서 화면 간 예측 및 화면 내 예측의 모드가 결정되는 단위이다. 각 CU 내에서 보다 정확한 예측을 위해 CU를 다시 PU (prediction unit)으로 나눌 수 있다. PU는 예측 블록을 생성하는 기본 단위이다. 일반적으로 PU는 CU로부터 분할되며, 단일 계층을 가지면서 여러 개의 part로 분할이 가능하다.

각 CU에 대해 motion estimation 및 motion compensation에 근거한 화면 간 예측 또는 화면 내 예측 (intra prediction)을 수행하여 예측블록 P를 생성하고, 이를 원본 CU (0)에서 빼주면 CU크기와 동일한 차분 블록 R이 생성된다. 생성된 차분 블록 R은 변환 (transform), 양자화 (quantization) 및 엔트로피 부호화 과정을 거친다. 이때 효과적으로 변환 부호화하기 위해 TU(transform unit)라는 단위를 사용한다. 즉, 차분 블록 R은 TU단위로 나뉘어서 부호화가 수행된다. 여기서 TU는 CU로부터 다계층 쿼드트리 형태로 분할될 수 있다. 다만, PU에 의존적이어서 TU가 PU보다 큰 블록이 될 수는 없다.

인코더와 디코더에서 미스매치가 발생하지 않도록 하기 위해서는 인코더에서 예측을 수행할 때 디코더에서 사용 가능한 값을 사용해야 한다. 이를 위해 HEVC 인코더에서는 인코딩한 현재 블록을 다시 복원하는 과정을 수행한다. 그림 1과 같이 양자화된 계수값들에 다시 역양자화를 수행하고 이를 다시 역변환하여 복원된 차분 R'이 얻어진다. R'은 예측 블록 P와 더해져서 복원 블록 O'이 얻어진다. 이와 같은 과정을 모든 CU에 대해 적용한 후에 디블록킹 필터와 SAO 필터가 순차적으로 복원 영상에 적용된 후에 결과 영상들은 이후에 이루어지는 예측 과정을 위해 decoded picture buffer에 저장된다.

패턴 정합 움직임 벡터 유도 (pattern matched motion vector derivation; 이하 PMMVD)모드는 소위 프레임율 상승 변환 (frame rate up conversion; 이하 FURC)에 기반한 특별한 모드이다. 이 모드가 설정이 되면, 현재 블록의 움직임 정보는 encoder에서 decoder로 전송하지 않고, decoder에서 직접 유도해서 사용한다. HEVC와 달

리 JEM은 PMMVD 모드를 가지고 있다.

소위 FRUC 플래그가 off이면, 일반적인 머지 (merge) 모드가 사용된다. 반면 FRUC 플래그가 on이면 다음 두 가지 모드 중 하나가 선택되고 추가적인 FRUC 모드 플래그로 결정된다. 선택된 FRUC 모드 플래그에 따라 encoder와 decoder 모두에서 현재 블록의 MV는 해당 모드로 유도가 된다.

- Bilateral matching

- Template matching

먼저 현재 PU의 merge 리스트가 체크되고, 최소 정합 오차를 갖는 후보 MV가 시작점이 된다. 다음으로 그 후보 MV를 중심으로 지향적인 탐색을 수행한다. 이때 bilateral matching이나 temporal matching이 수행된다. 최소 정합 오차를 갖는 MV가 해당 PU의 최종 MV로 선택된다.

그림 2는 bilateral 정합을 설명한다. 두 개의 서로 다른 참조 프레임들에서 현재 블록의 움직임 trajectory를 따라 두 블록 간 정합 오차가 최소가 되는 블록을 찾는 방식이다. 연속적인 움직임 trajectory를 가정하고 MV0과 MV1은 프레임간 거리 TD0과 TD1에 비례해서 결정된다.

한편, 그림 3은 template matching을 설명한다. 현재 블록과 참조 프레임 내 후보 블록과의 정합 시 그림과 같이 정합 대상 상단과 좌측 template 간 정합이 수행된다. 그 오차가 최소가 되는 방향으로 MV가 결정된다.

Bilateral 정합의 경우 현재 블록의 정보를 사용하지 않고 움직임 벡터를 유도할 수 있고, template matching의 경우 한 방향만 고려하여 움직임 벡터를 유도할 수 있다.

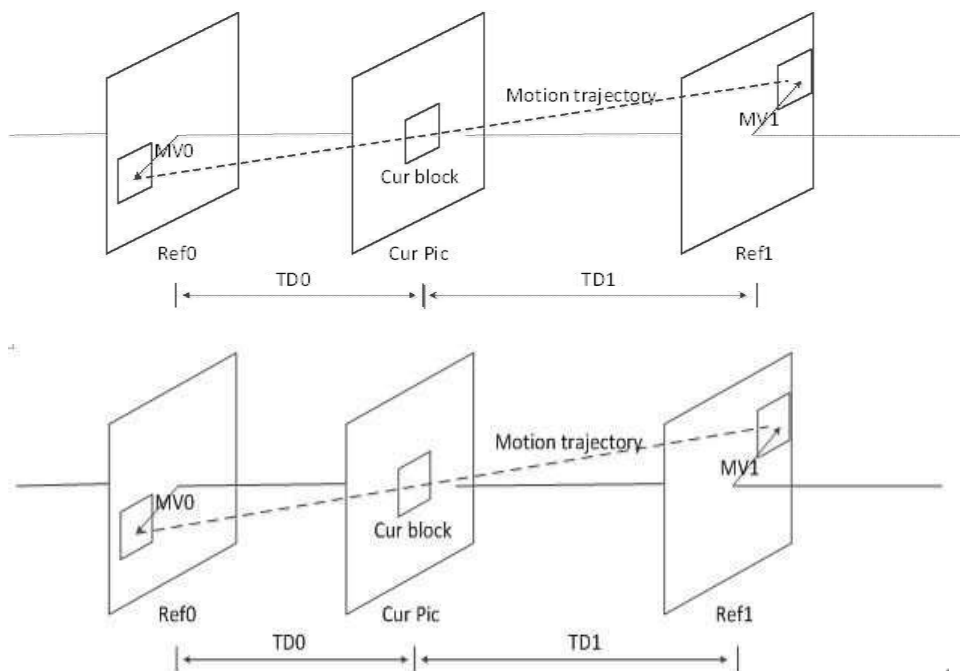


그림 2. Bilateral matching

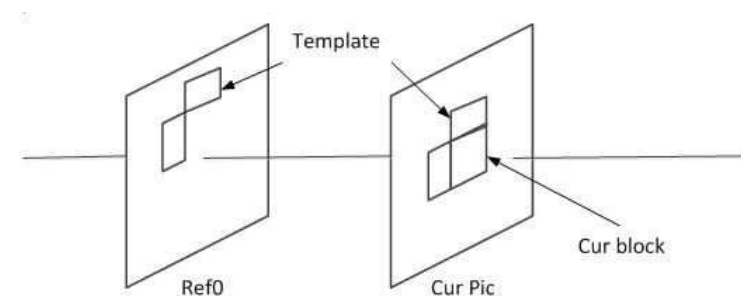


그림 3. Template matching

Bilateral 정합과 template 정합을 merge하여 MV를 유도할 수도 있다. 그림 4는 제안한 기법을 설명한다. 두 개

의 서로 다른 참조 프레임들에서 현재 블록의 움직임 trajectory를 따라 세 블록 간 template 정합 오차가 최소가 되는 블록을 찾는 방식이다. 즉, 현재 블록과 ref0의 후보 블록간 template 정합오차와 현재 블록과 ref1의 후보 블록 간 정합 오차의 합이 최소가 되는 움직임 벡터를 찾는다. 연속적인 움직임 trajectory를 가정하고 MV0와 MV1은 프레임간 거리 TD0와 TD1에 비례해서 결정된다.

또 다른 실시예로는 MV0와 MV1을 분리해서 탐색하는 것도 가능하다. 즉, MV0에 대해서만 지역적 template matching을 수행하고, 독립적으로 MV1에 대해서 별도의 template matching을 수행한 다음 각각의 최적의 MV를 찾을 수도 있다.

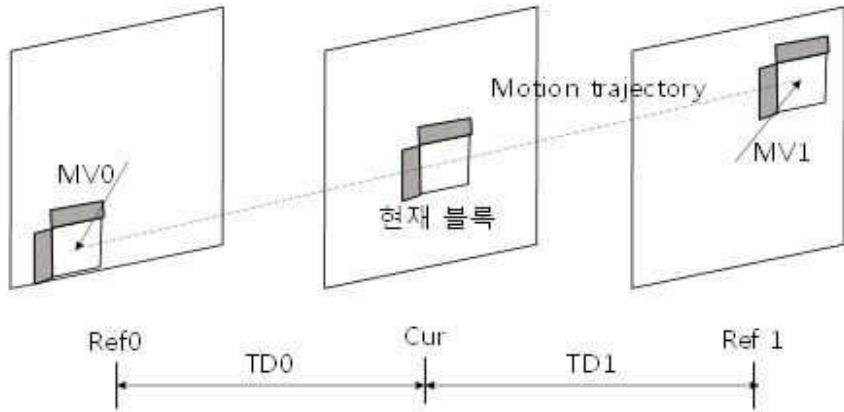


그림 4. 제안 기법에 따른 패턴 정합. 회색 영역이 template임