



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2019-0090867
(43) 공개일자 2019년08월02일

(51) 국제특허분류(Int. Cl.) H04N 19/11 (2014.01) H04N 19/105 (2014.01) H04N 19/132 (2014.01) H04N 19/44 (2014.01) H04N 19/96 (2014.01)	(71) 출원인 엘지전자 주식회사 서울특별시 영등포구 여의대로 128 (여의도동) 이화여자대학교 산학협력단 서울특별시 서대문구 이화여대길 52 (대현동, 이화여자대학교)
(52) CPC특허분류 H04N 19/11 (2015.01) H04N 19/105 (2015.01)	(72) 발명자 강제원 서울특별시 서대문구 이화여대길 52, 전자전기공학전공 아산공학관 432호 이재호 서울특별시 서초구 양재대로11길 19, LG전자 특허센터 (뒷면에 계속)
(21) 출원번호 10-2019-7020500	(74) 대리인 특허법인로알
(22) 출원일자(국제) 2016년12월28일 심사청구일자 2019년07월12일	
(85) 번역문제출일자 2019년07월12일	
(86) 국제출원번호 PCT/KR2016/015431	
(87) 국제공개번호 WO 2018/124332 국제공개일자 2018년07월05일	

전체 청구항 수 : 총 17 항

(54) 발명의 명칭 **인트라 예측 모드 기반 영상 처리 방법 및 이를 위한 장치**

(57) 요약

본 발명에서는 인트라 예측 모드 기반 영상 처리 방법 및 이를 위한 장치가 개시된다. 구체적으로, 인트라 예측(intra prediction) 모드 기반으로 영상을 복호화하는 방법에 있어서, 이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 단계; 상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 단계; 상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출하는 단계; 및 상기 인트라 예측 모드에 기반하여 상기 현재 블록의 예측 샘플을 생성하는 단계를 포함하고, 상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력할 수 있다.

(52) CPC특허분류

H04N 19/132 (2015.01)

H04N 19/44 (2015.01)

H04N 19/96 (2015.01)

(72) 발명자

류수경

서울특별시 송파구 송파대로32길 15, 105동 201호

임재현

서울특별시 서초구 양재대로11길 19, LG전자 특허
센터

강민주

서울특별시 서대문구 이화여대2다길 15

명세서

청구범위

청구항 1

인트라 예측(intra prediction) 모드 기반으로 영상을 복호화하는 방법에 있어서,

이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 단계;

상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 단계;

상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출하는 단계; 및

상기 인트라 예측 모드에 기반하여 상기 현재 블록의 예측 샘플을 생성하는 단계를 포함하고,

상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력하는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 2

제 1항에 있어서,

상기 분할 파라미터는 상기 결정 트리의 각 노드에서 좌측 자식 노드 또는 우측 자식 노드로 분할하였을 때 계산되는 불확실성의 감소를 최대화하는 방향으로 학습되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 3

제 1항에 있어서,

복수 개의 결정 트리로 구성된 랜덤 포레스트(random forest)에서 각 결정 트리의 리프 노드를 탐색하여 복수 개의 클래스가 선택되는 경우, 상기 복수 개의 클래스 중에서 가장 많이 선택되는 클래스를 상기 랜덤 포레스트의 클래스로 결정하는 단계를 더 포함하고,

상기 현재 블록의 인트라 예측 모드는 상기 랜덤 포레스트의 클래스를 이용하여 도출되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 4

제 1항에 있어서,

상기 분할 파라미터는 상기 현재 블록에 이웃하는 참조 샘플 중 2개의 참조 샘플을 이용하여 결정되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 5

제 1항에 있어서,

상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우,

상기 분할 파라미터는 상기 다중 참조 샘플 라인 중에서 복수 개의 참조 샘플을 이용하여 결정되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 6

제 1항에 있어서,

상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우,

상기 분할 파라미터는 상기 다중 참조 샘플 라인 중 상기 현재 블록에 인접한 2개의 참조 샘플 라인의 복수 개

의 참조 샘플을 이용하여 결정되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 7

제 1항에 있어서,

상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우,

상기 분할 파라미터는 상기 다중 참조 샘플 라인의 특정 참조 샘플의 샘플값과 수평 방향으로 상기 특정 참조 샘플의 우측에 인접한 참조 샘플의 샘플값과의 차이값과, 상기 특정 참조 샘플의 샘플값과 수직 방향으로 상기 특정 참조 샘플의 하단에 인접한 참조 샘플의 샘플값과의 차이값을 이용하여 결정되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 8

제 1항에 있어서,

상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우,

상기 분할 파라미터는 상기 다중 참조 샘플 라인의 각 참조 샘플의 샘플값과 특정 각도 방향으로 인접한 참조 샘플의 샘플값과의 차이값의 합을 이용하여 결정되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 9

제 1항에 있어서,

상기 클래스에 복수 개의 인트라 예측 모드가 매핑되는 경우, 상기 클래스 내에서 상기 현재 블록의 인트라 예측 모드를 지시하는 인덱스(index) 정보를 복호화하는 단계를 더 포함하고,

상기 현재 블록의 인트라 예측 모드는 상기 클래스 및 상기 인덱스 정보를 이용하여 도출되는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 10

제 1항에 있어서,

상기 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택하는 단계는,

제 1 결정 트리에서 리프 노드를 탐색하여 상기 현재 블록에 적용될 수 있는 복수 개의 인트라 예측 모드로 구성된 인트라 예측 모드 그룹을 결정하는 단계를 더 포함하고,

상기 인트라 예측 모드 그룹에 따라 정해지는 제 2 결정 트리에서 리프 노드를 탐색하여 상기 인트라 예측 모드 그룹 내에서 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택하는 인트라 예측 모드 기반 영상 복호화 방법.

청구항 11

인트라 예측(intra prediction) 모드 기반으로 영상을 복호화하는 장치에 있어서,

이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 리프 노드 탐색부;

상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 클래스 선택부;

상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출하는 예측 모드 도출부; 및

상기 인트라 예측 모드에 기반하여 상기 현재 블록의 예측 샘플을 생성하는 예측 샘플 생성부를 포함하고,

상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력하는 인트라 예측 모드 기반 영상 복호화 장치.

청구항 12

인트라 예측(intra prediction) 모드 기반으로 영상을 부호화하는 방법에 있어서,

이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 단계;

상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 단계; 및

상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 부호화하는 단계를 포함하고,

상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력하는 인트라 예측 모드 기반 영상 부호화 방법.

청구항 13

제 12항에 있어서,

상기 분할 파라미터는 상기 결정 트리의 각 노드에서 좌측 자식 노드 또는 우측 자식 노드로 분할하였을 때 계산되는 불확실성의 감소를 최대화하는 방향으로 학습되는 인트라 예측 모드 기반 영상 부호화 방법.

청구항 14

제 12항에 있어서,

상기 분할 파라미터는 상기 현재 블록에 이웃하는 참조 샘플 중에서 1개의 참조 샘플과 상기 현재 블록의 예측 샘플 중에서 1개의 예측 샘플을 이용하여 결정되는 인트라 예측 모드 기반 영상 부호화 방법.

청구항 15

제 12항에 있어서,

상기 분할 파라미터는 상기 현재 블록에 이웃하는 참조 샘플 중에서 하나 이상의 참조 샘플과 상기 현재 블록의 예측 샘플 중에서 하나 이상의 예측 샘플을 이용하여 결정되는 인트라 예측 모드 기반 영상 부호화 방법.

청구항 16

제 12항에 있어서,

상기 분할 파라미터는 상기 현재 블록의 예측 샘플 중, 특정 예측 샘플의 샘플값과 수평 방향으로 상기 특정 예측 샘플의 우측에 인접한 예측 샘플의 샘플값간의 차이값과, 상기 특정 예측 샘플의 샘플값과 수직 방향으로 상기 특정 예측 샘플의 하단에 인접한 예측 샘플의 샘플값간의 차이값을 이용하여 결정되는 인트라 예측 모드 기반 영상 부호화 방법.

청구항 17

제 12항에 있어서,

상기 분할 파라미터는 상기 현재 블록의 각 예측 샘플의 샘플값과 특정 각도 방향으로 인접한 예측 샘플의 샘플값과의 차이값의 합을 이용하여 결정되는 인트라 예측 모드 기반 영상 부호화 방법.

발명의 설명**기술 분야**

본 발명은 정지 영상 또는 동영상 처리 방법에 관한 것으로서, 보다 상세하게 인트라 예측 모드(intra prediction mode) 기반으로 정지 영상 또는 동영상을 인코딩/디코딩하는 방법 및 이를 지원하는 장치에 관한 것이다.

[0001]

배경 기술

- [0002] 압축 부호화란 디지털화한 정보를 통신 회선을 통해 전송하거나, 저장 매체에 적합한 형태로 저장하기 위한 일련의 신호 처리 기술을 의미한다. 영상, 이미지, 음성 등의 미디어가 압축 부호화의 대상이 될 수 있으며, 특히 영상을 대상으로 압축 부호화를 수행하는 기술을 비디오 영상 압축이라고 일컫는다.
- [0003] 차세대 비디오 콘텐츠는 고해상도(high spatial resolution), 고프레임율(high frame rate) 및 영상 표현의 고차원화(high dimensionality of scene representation)라는 특징을 갖게 될 것이다. 그러한 콘텐츠를 처리하기 위해서는 메모리 저장(memory storage), 메모리 액세스율(memory access rate) 및 처리 전력(processing power) 측면에서 엄청난 증가를 가져올 것이다.
- [0004] 따라서, 차세대 비디오 콘텐츠를 보다 효율적으로 처리하기 위한 코딩 톨을 디자인할 필요가 있다.

발명의 내용

해결하려는 과제

- [0005] 본 발명의 목적은 머신 러닝(machine learning) 기술 중 하나인 랜덤 포레스트(Random Forest) 방법을 사용하여 화면 내 예측 모드의 시그널링을 최적화하여 부호화/복호화하는 방법을 제안한다.
- [0006] 또한, 본 발명의 목적은 화면 내 예측 모드의 시그널링 없이 랜덤 포레스트를 이용하여 화면 내 예측 모드를 도출하는 방법을 제안한다.
- [0007] 또한, 본 발명의 목적은 복수 개의 화면 내 예측 모드를 하나의 클래스로 구성하고, 랜덤 포레스트를 이용하여 클래스를 추정하는 방법을 제안한다.
- [0008] 본 발명에서 이루고자 하는 기술적 과제들은 이상에서 언급한 기술적 과제들로 제한되지 않으며, 언급하지 않은 또 다른 기술적 과제들은 아래의 기재로부터 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 명확하게 이해될 수 있을 것이다.

과제의 해결 수단

- [0009] 본 발명의 일 양상은, 인트라 예측(intra prediction) 모드 기반으로 영상을 복호화하는 방법에 있어서, 이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 단계; 상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 단계; 상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출하는 단계; 및 상기 인트라 예측 모드에 기반하여 상기 현재 블록의 예측 샘플을 생성하는 단계를 포함하고, 상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력할 수 있다.
- [0010] 본 발명의 일 양상은, 인트라 예측(intraprediction) 모드 기반으로 영상을 복호화하는 장치에 있어서, 이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 리프 노드 탐색부; 상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 클래스 선택부; 상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출하는 예측 모드 도출부; 및 상기 인트라 예측 모드에 기반하여 상기 현재 블록의 예측 샘플을 생성하는 예측 샘플 생성부를 포함하고, 상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력할 수 있다.
- [0011] 바람직하게, 상기 분할 파라미터는 상기 결정 트리의 각 노드에서 좌측 자식 노드 또는 우측 자식 노드로 분할하였을 때 계산되는 불확실성의 감소를 최대화하는 방향으로 학습될 수 있다.
- [0012] 바람직하게, 복수 개의 결정 트리로 구성된 랜덤 포레스트(random forest)에서 각 결정 트리의 리프 노드를 탐색하여 복수 개의 클래스가 선택되는 경우, 상기 복수 개의 클래스 중에서 가장 많이 선택되는 클래스를 상기 랜덤 포레스트의 클래스로 결정하는 단계를 더 포함하고, 상기 현재 블록의 인트라 예측 모드는 상기 랜덤 포레

스트의 클래스를 이용하여 도출될 수 있다.

- [0013] 바람직하게, 상기 분할 파라미터는 상기 현재 블록에 이웃하는 참조 샘플 중 2개의 참조 샘플을 이용하여 결정될 수 있다.
- [0014] 바람직하게, 상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우, 상기 분할 파라미터는 상기 다중 참조 샘플 라인 중에서 복수 개의 참조 샘플을 이용하여 결정될 수 있다.
- [0015] 바람직하게, 상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우, 상기 분할 파라미터는 상기 다중 참조 샘플 라인 중 상기 현재 블록에 인접한 2개의 참조 샘플 라인의 복수 개의 참조 샘플을 이용하여 결정될 수 있다.
- [0016] 바람직하게, 상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우, 상기 분할 파라미터는 상기 다중 참조 샘플 라인의 특정 참조 샘플의 샘플값과 수평 방향으로 상기 특정 참조 샘플의 우측에 인접한 참조 샘플의 샘플값과의 차이값과, 상기 특정 참조 샘플의 샘플값과 수직 방향으로 상기 특정 참조 샘플의 하단에 인접한 참조 샘플의 샘플값과의 차이값을 이용하여 결정될 수 있다.
- [0017] 바람직하게, 상기 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우, 상기 분할 파라미터는 상기 다중 참조 샘플 라인의 각 참조 샘플의 샘플값과 특정 각도 방향으로 인접한 참조 샘플의 샘플값과의 차이값의 합을 이용하여 결정될 수 있다.
- [0018] 바람직하게, 상기 클래스에 복수 개의 인트라 예측 모드가 매핑되는 경우, 상기 클래스 내에서 상기 현재 블록의 인트라 예측 모드를 지시하는 인덱스(index) 정보를 복호화하는 단계를 더 포함하고, 상기 현재 블록의 인트라 예측 모드는 상기 클래스 및 상기 인덱스 정보를 이용하여 도출될 수 있다.
- [0019] 바람직하게, 상기 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택하는 단계는, 제 1 결정 트리에서 리프 노드를 탐색하여 상기 현재 블록에 적용될 수 있는 복수 개의 인트라 예측 모드로 구성된 인트라 예측 모드 그룹을 결정하는 단계를 더 포함하고, 상기 인트라 예측 모드 그룹에 따라 정해지는 제 2 결정 트리에서 리프 노드를 탐색하여 상기 인트라 예측 모드 그룹 내에서 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택할 수 있다.
- [0020] 본 발명의 일 양상은, 인트라 예측(intra prediction) 모드 기반으로 영상을 부호화하는 방법에 있어서, 이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색하는 단계; 상기 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택하는 단계; 및 상기 클래스를 이용하여 현재 블록의 인트라 예측 모드를 부호화하는 단계를 포함하고, 상기 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력할 수 있다.
- [0021] 바람직하게, 상기 분할 파라미터는 상기 결정 트리의 각 노드에서 좌측 자식 노드 또는 우측 자식 노드로 분할하였을 때 계산되는 불확실성의 감소를 최대화하는 방향으로 학습될 수 있다.
- [0022] 바람직하게, 상기 분할 파라미터는 상기 현재 블록에 이웃하는 참조 샘플 중에서 1개의 참조 샘플과 상기 현재 블록의 예측 샘플 중에서 1개의 예측 샘플을 이용하여 결정될 수 있다.
- [0023] 바람직하게, 상기 분할 파라미터는 상기 현재 블록에 이웃하는 참조 샘플 중에서 하나 이상의 참조 샘플과 상기 현재 블록의 예측 샘플 중에서 하나 이상의 예측 샘플을 이용하여 결정될 수 있다.
- [0024] 바람직하게, 상기 분할 파라미터는 상기 현재 블록의 예측 샘플 중, 특정 예측 샘플의 샘플값과 수평 방향으로 상기 특정 예측 샘플의 우측에 인접한 예측 샘플의 샘플값간의 차이값과, 상기 특정 예측 샘플의 샘플값과 수직 방향으로 상기 특정 예측 샘플의 하단에 인접한 예측 샘플의 샘플값간의 차이값을 이용하여 결정될 수 있다.
- [0025] 바람직하게, 상기 분할 파라미터는 상기 현재 블록의 각 예측 샘플의 샘플값과 특정 각도 방향으로 인접한 예측 샘플의 샘플값과의 차이값의 합을 이용하여 결정될 수 있다.

발명의 효과

- [0026] 본 발명의 실시예에 따르면, 랜덤 포레스트 방법을 사용함으로써 화면 내 예측 모드를 표현하기 위해 사용되는

비트 수를 줄일 수 있고, 이를 통해 부호화 효율을 향상시킬 수 있다.

[0027] 또한, 본 발명의 실시예에 따르면, 랜덤 포레스트를 통해 결정되는 클래스에 하나의 화면 내 예측 모드를 대응 시킴으로써 화면 내 예측 모드를 시그널링하지 않고도 화면 내 예측 모드를 복호화할 수 있고, 이를 통해 화면 내 예측 모드를 표현하기 위해 사용되는 비트를 대폭 절약할 수 있다.

[0028] 또한, 본 발명의 실시예에 따르면, 랜덤 포레스트를 통해 결정되는 클래스에 복수 개의 예측 모드들의 집합을 대응시킴으로써, 화면 내 예측 모드 추정의 정확도를 높일 수 있고, 이를 통해 부호화 효율을 향상시킬 수 있다.

[0029] 본 발명에서 얻을 수 있는 효과는 이상에서 언급한 효과로 제한되지 않으며, 언급하지 않은 또 다른 효과들은 아래의 기재로부터 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 명확하게 이해될 수 있을 것이다.

도면의 간단한 설명

[0030] 본 발명에 관한 이해를 돕기 위해 상세한 설명의 일부로 포함되는, 첨부 도면은 본 발명에 대한 실시예를 제공하고, 상세한 설명과 함께 본 발명의 기술적 특징을 설명한다.

도 1은 본 발명이 적용되는 실시예로서, 정지 영상 또는 동영상 신호의 인코딩이 수행되는 인코더의 개략적인 블록도를 나타낸다.

도 2는 본 발명이 적용되는 실시예로서, 정지 영상 또는 동영상 신호의 인코딩이 수행되는 디코더의 개략적인 블록도를 나타낸다.

도 3은 본 발명에 적용될 수 있는 코딩 유닛의 분할 구조를 설명하기 위한 도면이다.

도 4는 본 발명에 적용될 수 있는 예측 유닛을 설명하기 위한 도면이다.

도 5는 본 발명이 적용되는 실시예로서, 인트라 예측 방법을 예시하는 도면이다.

도 6은 인트라 예측 모드에 따른 예측 방향을 예시한다.

도 7은 본 발명이 적용되는 실시예로서, 화면 내 예측 모드를 복호화하는 방법을 나타내는 흐름도이다.

도 8은 본 발명이 적용되는 실시예로서, MPM 모드를 결정하는 방법을 설명하기 위한 도면이다.

도 9는 본 발명이 적용되는 실시예로서, 랜덤 포레스트 및 결정 트리를 설명하기 위한 도면이다.

도 10 내지 도 22는 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

도 23은 본 발명이 적용될 수 있는 실시예로서, 클래스에 복수 개의 화면 내 예측 모드를 매핑하는 방법을 예시하는 도면이다.

도 24는 본 발명의 일 실시예에 따른 랜덤 포레스트에 기반한 계층 구조의 화면 내 예측 모드의 결정 방법을 예시하는 도면이다.

도 25는 본 발명이 적용될 수 있는 실시예로서, 화면 내 예측 모드 별 확률값을 결정하는 방법을 예시하는 도면이다.

도 26은 본 발명의 일 실시예에 따른 인트라 예측 방법을 예시하는 도면이다.

도 27은 본 발명의 일 실시예에 따른 인트라 예측부를 예시하는 도면이다.

발명을 실시하기 위한 구체적인 내용

[0031] 이하, 본 발명에 따른 바람직한 실시 형태를 첨부된 도면을 참조하여 상세하게 설명한다. 첨부된 도면과 함께 이하에 개시될 상세한 설명은 본 발명의 예시적인 실시형태를 설명하고자 하는 것이며, 본 발명이 실시될 수 있는 유일한 실시형태를 나타내고자 하는 것이 아니다. 이하의 상세한 설명은 본 발명의 완전한 이해를 제공하기 위해서 구체적 세부사항을 포함한다. 그러나, 당업자는 본 발명이 이러한 구체적 세부사항 없이도 실시될 수 있음을 안다.

[0032] 몇몇 경우, 본 발명의 개념이 모호해지는 것을 피하기 위하여 공지의 구조 및 장치는 생략되거나, 각 구조 및

장치의 핵심기능을 중심으로 한 블록도 형식으로 도시될 수 있다.

- [0033] 아울러, 본 발명에서 사용되는 용어는 가능한 한 현재 널리 사용되는 일반적인 용어를 선택하였으나, 특정한 경우에는 출원인이 임의로 선정한 용어를 사용하여 설명한다. 그러한 경우에는 해당 부분의 상세 설명에서 그 의미를 명확히 기재하므로, 본 발명의 설명에서 사용된 용어의 명칭만으로 단순 해석되어서는 안 될 것이며 그 해당 용어의 의미까지 파악하여 해석되어야 함을 밝혀두고자 한다.
- [0034] 이하의 설명에서 사용되는 특정 용어들은 본 발명의 이해를 돕기 위해서 제공된 것이며, 이러한 특정 용어의 사용은 본 발명의 기술적 사상을 벗어나지 않는 범위에서 다른 형태로 변경될 수 있다. 예를 들어, 신호, 데이터, 샘플, 픽처, 프레임, 블록 등의 경우 각 코딩 과정에서 적절하게 대체되어 해석될 수 있을 것이다.
- [0035] 이하 본 명세서에서 '처리 유닛'은 예측, 변환 및/또는 양자화 등과 같은 인코딩/디코딩의 처리 과정이 수행되는 단위를 의미한다. 이하, 설명의 편의를 위해 처리 유닛은 '처리 블록' 또는 '블록'으로 지칭될 수도 있다.
- [0036] 처리 유닛은 휘도(luma) 성분에 대한 단위와 색차(chroma) 성분에 대한 단위를 포함하는 의미로 해석될 수 있다. 예를 들어, 처리 유닛은 코딩 트리 유닛(CTU: Coding Tree Unit), 코딩 유닛(CU: Coding Unit), 예측 유닛(PU: Prediction Unit) 또는 변환 유닛(TU: Transform Unit)에 해당될 수 있다.
- [0037] 또한, 처리 유닛은 휘도(luma) 성분에 대한 단위 또는 색차(chroma) 성분에 대한 단위로 해석될 수 있다. 예를 들어, 처리 유닛은 휘도(luma) 성분에 대한 코딩 트리 블록(CTB: Coding Tree Block), 코딩 블록(CB: Coding Block), 예측 블록(PU: Prediction Block) 또는 변환 블록(TB: Transform Block)에 해당될 수 있다. 또는, 색차(chroma) 성분에 대한 코딩 트리 블록(CTB), 코딩 블록(CB), 예측 블록(PU) 또는 변환 블록(TB)에 해당될 수 있다. 또한, 이에 한정되는 것은 아니며 처리 유닛은 휘도(luma) 성분에 대한 단위와 색차(chroma) 성분에 대한 단위를 포함하는 의미로 해석될 수도 있다.
- [0038] 또한, 처리 유닛은 반드시 정사각형의 블록으로 한정되는 것은 아니며, 3개 이상의 꼭지점을 가지는 다각형 형태로 구성될 수도 있다.
- [0039] 또한, 이하 본 명세서에서 픽셀 또는 화소 등을 샘플로 통칭한다. 그리고, 샘플을 이용한다는 것은 픽셀 값 또는 화소 값 등을 이용한다는 것을 의미할 수 있다.
- [0040] 도 1은 본 발명이 적용되는 실시예로서, 정지 영상 또는 동영상 신호의 인코딩이 수행되는 인코더의 개략적인 블록도를 나타낸다.
- [0041] 도 1을 참조하면, 인코더(100)는 영상 분할부(110), 감산기(115), 변환부(120), 양자화부(130), 역양자화부(140), 역변환부(150), 필터링부(160), 복호 픽처 버퍼(DPB: Decoded Picture Buffer)(170), 예측부(180) 및 엔트로피 인코딩부(190)를 포함하여 구성될 수 있다. 그리고, 예측부(180)는 인터 예측부(181), 인트라 예측부(182)를 포함하여 구성될 수 있다.
- [0042] 영상 분할부(110)는 인코더(100)에 입력된 입력 영상 신호(Input video signal)(또는, 픽처, 프레임)를 하나 이상의 처리 유닛으로 분할한다.
- [0043] 감산기(115)는 입력 영상 신호에서 예측부(180)로부터(즉, 인터 예측부(181) 또는 인트라 예측부(182))로부터 출력된 예측 신호(prediction signal)(또는 예측 블록)를 감산하여 차분 신호(residual signal)(또는 차분 블록)를 생성한다. 생성된 차분 신호(또는 차분 블록)는 변환부(120)로 전송된다.
- [0044] 변환부(120)는 차분 신호(또는 차분 블록)에 변환 기법(예를 들어, DCT(Discrete Cosine Transform), DST(Discrete Sine Transform), GBT(Graph-Based Transform), KLT(Karhunen-Loeve transform) 등)을 적용하여 변환 계수(transform coefficient)를 생성한다. 이때, 변환부(120)는 차분 블록에 적용된 예측 모드와 차분 블록의 크기에 따라서 결정된 변환 기법을 이용하여 변환을 수행함으로써 변환 계수들을 생성할 수 있다.
- [0045] 양자화부(130)는 변환 계수를 양자화하여 엔트로피 인코딩부(190)로 전송하고, 엔트로피 인코딩부(190)는 양자화된 신호(quantized signal)를 엔트로피 코딩하여 비트 스트림으로 출력한다.
- [0046] 한편, 양자화부(130)로부터 출력된 양자화된 신호(quantized signal)는 예측 신호를 생성하기 위해 이용될 수 있다. 예를 들어, 양자화된 신호(quantized signal)는 루프 내의 역양자화부(140) 및 역변환부(150)를 통해 역양자화 및 역변환을 적용함으로써 차분 신호를 복원할 수 있다. 복원된 차분 신호를 인터 예측부(181) 또는 인트라 예측부(182)로부터 출력된 예측 신호(prediction signal)에 더함으로써 복원 신호(reconstructed signal)가 생성될 수 있다.

- [0047] 한편, 위와 같은 압축 과정에서 인접한 블록들이 서로 다른 양자화 파라미터에 의해 양자화됨으로써 블록 경계가 보이는 열화가 발생할 수 있다. 이러한 현상을 블록킹 열화(blocking artifacts)라고 하며, 이는 화질을 평가하는 중요한 요소 중의 하나이다. 이러한 열화를 줄이기 위해 필터링 과정을 수행할 수 있다. 이러한 필터링 과정을 통해 블록킹 열화를 제거함과 동시에 현재 픽처에 대한 오차를 줄임으로써 화질을 향상시킬 수 있게 된다.
- [0048] 필터링부(160)는 복원 신호에 필터링을 적용하여 이를 재생 장치로 출력하거나 복호 픽처 버퍼(170)에 전송한다. 복호 픽처 버퍼(170)에 전송된 필터링된 신호는 인터 예측부(181)에서 참조 픽처로 사용될 수 있다. 이처럼, 필터링된 픽처를 화면간 예측 모드에서 참조 픽처로 이용함으로써 화질 뿐만 아니라 부호화 효율도 향상시킬 수 있다.
- [0049] 복호 픽처 버퍼(170)는 필터링된 픽처를 인터 예측부(181)에서의 참조 픽처로 사용하기 위해 저장할 수 있다.
- [0050] 인터 예측부(181)는 복원 픽처(reconstructed picture)를 참조하여 시간적 중복성 및/또는 공간적 중복성을 제거하기 위해 시간적 예측 및/또는 공간적 예측을 수행한다. 여기서, 예측을 수행하기 위해 이용되는 참조 픽처는 이전 시간에 부호화/복호화 시 블록 단위로 양자화와 역양자화를 거친 변환된 신호이기 때문에, 블록킹 아티팩트(blocking artifact)나 링잉 아티팩트(ringing artifact)가 존재할 수 있다.
- [0051] 따라서, 인터 예측부(181)는 이러한 신호의 불연속이나 양자화로 인한 성능 저하를 해결하기 위해, 로우패스 필터(lowpass filter)를 적용함으로써 픽셀들 사이의 신호를 서브 픽셀 단위로 보간할 수 있다. 여기서, 서브 픽셀은 보간 필터를 적용하여 생성된 가상의 화소를 의미하고, 정수 픽셀은 복원된 픽처에 존재하는 실제 화소를 의미한다. 보간 방법으로는 선형 보간, 양선형 보간(bi-linear interpolation), 위너 필터(wiener filter) 등이 적용될 수 있다.
- [0052] 보간 필터는 복원 픽처(reconstructed picture)에 적용되어 예측의 정밀도를 향상시킬 수 있다. 예를 들어, 인터 예측부(181)는 정수 픽셀에 보간 필터를 적용하여 보간 픽셀을 생성하고, 보간 픽셀들(interpolated pixels)로 구성된 보간 블록(interpolated block)을 예측 블록(prediction block)으로 사용하여 예측을 수행할 수 있다.
- [0053] 인트라 예측부(182)는 현재 부호화를 진행하려고 하는 블록의 주변에 있는 샘플들을 참조하여 현재 블록을 예측한다. 인트라 예측부(182)는, 인트라 예측을 수행하기 위해 다음과 같은 과정을 수행할 수 있다. 먼저, 예측 신호를 생성하기 위해 필요한 참조 샘플을 준비할 수 있다. 그리고, 준비된 참조 샘플을 이용하여 예측 신호를 생성할 수 있다. 또한, 예측 모드를 부호화하게 된다. 이때, 참조 샘플은 참조 샘플 패딩 및/또는 참조 샘플 필터링을 통해 준비될 수 있다. 참조 샘플은 예측 및 복원 과정을 거쳤기 때문에 양자화 에러가 존재할 수 있다. 따라서, 이러한 에러를 줄이기 위해 인트라 예측에 이용되는 각 예측 모드에 대해 참조 샘플 필터링 과정이 수행될 수 있다.
- [0054] 인터 예측부(181) 또는 상기 인트라 예측부(182)를 통해 생성된 예측 신호(prediction signal)(또는 예측 블록)는 복원 신호(또는 복원 블록)를 생성하기 위해 이용되거나 차분 신호(또는 차분 블록)를 생성하기 위해 이용될 수 있다.
- [0055] 도 2는 본 발명이 적용되는 실시예로서, 정지 영상 또는 동영상 신호의 인코딩이 수행되는 디코더의 개략적인 블록도를 나타낸다.
- [0056] 도 2를 참조하면, 디코더(200)는 엔트로피 디코딩부(210), 역양자화부(220), 역변환부(230), 가산기(235), 필터링부(240), 복호 픽처 버퍼(DPB: Decoded Picture Buffer Unit)(250), 예측부(260)를 포함하여 구성될 수 있다. 그리고, 예측부(260)는 인터 예측부(261) 및 인트라 예측부(262)를 포함하여 구성될 수 있다.
- [0057] 그리고, 디코더(200)를 통해 출력된 복원 영상 신호(reconstructed video signal)는 재생 장치를 통해 재생될 수 있다.
- [0058] 디코더(200)는 도 1의 인코더(100)로부터 출력된 신호(즉, 비트 스트림)를 수신하고, 수신된 신호는 엔트로피 디코딩부(210)를 통해 엔트로피 디코딩된다.
- [0059] 역양자화부(220)에서는 양자화 스텝 사이즈 정보를 이용하여 엔트로피 디코딩된 신호로부터 변환 계수(transform coefficient)를 획득한다.
- [0060] 역변환부(230)에서는 역변환 기법을 적용하여 변환 계수를 역변환하여 차분 신호(residual signal)(또는 차분

블록)를 획득하게 된다.

- [0061] 가산기(235)는 획득된 차분 신호(또는 차분 블록)를 예측부(260)(즉, 인터 예측부(261) 또는 인트라 예측부(262))로부터 출력된 예측 신호(prediction signal)(또는 예측 블록)에 더함으로써 복원 신호(reconstructed signal)(또는 복원 블록)가 생성된다.
- [0062] 필터링부(240)는 복원 신호(reconstructed signal)(또는 복원 블록)에 필터링을 적용하여 이를 재생 장치로 출력하거나 복호 픽처 버퍼부(250)에 전송한다. 복호 픽처 버퍼부(250)에 전송된 필터링된 신호는 인터 예측부(261)에서 참조 픽처로 사용될 수 있다.
- [0063] 본 명세서에서, 인코더(100)의 필터링부(160), 인터 예측부(181) 및 인트라 예측부(182)에서 설명된 실시예들은 각각 디코더의 필터링부(240), 인터 예측부(261) 및 인트라 예측부(262)에도 동일하게 적용될 수 있다.
- [0064] **처리 단위 분할 구조**
- [0065] 일반적으로 정지 영상 또는 동영상 압축 기술(예를 들어, HEVC)에서는 블록 기반의 영상 압축 방법을 이용한다. 블록 기반의 영상 압축 방법은 영상을 특정 블록 단위로 나누어서 처리하는 방법으로서, 메모리 사용과 연산량을 감소시킬 수 있다.
- [0066] 도 3은 본 발명에 적용될 수 있는 코딩 유닛의 분할 구조를 설명하기 위한 도면이다.
- [0067] 인코더는 하나의 영상(또는 픽처)을 사각형 형태의 코딩 트리 유닛(CTU: Coding Tree Unit) 단위로 분할한다. 그리고, 래스터 스캔 순서(raster scan order)에 따라 하나의 CTU 씩 순차적으로 인코딩한다.
- [0068] HEVC에서 CTU의 크기는 64×64 , 32×32 , 16×16 중 어느 하나로 정해질 수 있다. 인코더는 입력된 영상의 해상도 또는 입력된 영상의 특성 등에 따라 CTU의 크기를 선택하여 사용할 수 있다. CTU은 휘도(luma) 성분에 대한 코딩 트리 블록(CTB: Coding Tree Block)과 이에 대응하는 두 개의 색차(chroma) 성분에 대한 CTB를 포함한다.
- [0069] 하나의 CTU은 쿼드-트리(Quad-tree) 구조로 분할될 수 있다. 즉, 하나의 CTU은 정사각형 형태를 가지면서 절반의 수평 크기(half horizontal size) 및 절반의 수직 크기(half vertical size)를 가지는 4개의 유닛으로 분할되어 코딩 유닛(CU: Coding Unit)이 생성될 수 있다. 이러한 쿼드-트리 구조의 분할은 재귀적으로 수행될 수 있다. 즉, CU은 하나의 CTU로부터 쿼드-트리 구조로 계층적으로 분할된다.
- [0070] CU은 입력 영상의 처리 과정, 예컨대 인트라(intra)/인터(inter) 예측이 수행되는 코딩의 기본 단위를 의미한다. CU은 휘도(luma) 성분에 대한 코딩 블록(CB: Coding Block)과 이에 대응하는 두 개의 색차(chroma) 성분에 대한 CB를 포함한다. EVC에서 CU의 크기는 64×64 , 32×32 , 16×16 , 8×8 중 어느 하나로 정해질 수 있다.
- [0071] 도 3을 참조하면, 쿼드-트리의 루트 노드(root node)는 CTU와 관련된다. 쿼드-트리는 리프 노드(leaf node)에 도달할 때까지 분할되고, 리프 노드는 CU에 해당한다.
- [0072] 보다 구체적으로 살펴보면, CTU는 루트 노드(root node)에 해당되고, 가장 작은 깊이(depth)(즉, depth=0) 값을 가진다. 입력 영상의 특성에 따라 CTU가 분할되지 않을 수도 있으며, 이 경우 CTU은 CU에 해당한다.
- [0073] CTU은 쿼드 트리 형태로 분할될 수 있으며, 그 결과 깊이 1(depth=1)인 하위 노드들이 생성된다. 그리고, 1의 깊이를 가지는 하위 노드에서 더 이상 분할되지 않은 노드(즉, 리프 노드)는 CU에 해당한다. 예를 들어, 도 3(b)에서 노드 a, b 및 j에 대응하는 CU(a), CU(b), CU(j)는 CTU에서 한 번 분할되었으며, 1의 깊이를 가진다.
- [0074] 1의 깊이를 가지는 노드 중 적어도 어느 하나는 다시 쿼드 트리 형태로 분할될 수 있으며, 그 결과 깊이 1(즉, depth=2)인 하위 노드들이 생성된다. 그리고, 2의 깊이를 가지는 하위 노드에서 더 이상 분할되지 않은 노드(즉, 리프 노드)는 CU에 해당한다. 예를 들어, 도 3(b)에서 노드 c, h 및 i에 대응하는 CU(c), CU(h), CU(i)는 CTU에서 두 번 분할되었으며, 2의 깊이를 가진다.
- [0075] 또한, 2의 깊이를 가지는 노드 중 적어도 어느 하나는 다시 쿼드 트리 형태로 분할될 수 있으며, 그 결과 깊이 3(즉, depth=3)인 하위 노드들이 생성된다. 그리고, 3의 깊이를 가지는 하위 노드에서 더 이상 분할되지 않은 노드(즉, 리프 노드)는 CU에 해당한다. 예를 들어, 도 3(b)에서 노드 d, e, f, g에 대응하는 CU(d), CU(e), CU(f), CU(g)는 CTU에서 3번 분할되었으며, 3의 깊이를 가진다.
- [0076] 인코더에서는 비디오 영상의 특성(예를 들어, 해상도)에 따라서 혹은 부호화의 효율을 고려하여 CU의 최대 크기 또는 최소 크기를 결정할 수 있다. 그리고, 이에 대한 정보 또는 이를 유도할 수 있는 정보가 비트스트림에 포

함될 수 있다. 최대 크기를 가지는 CU를 최대 코딩 유닛(LCU: Largest Coding Unit)이라고 지칭하며, 최소 크기를 가지는 CU를 최소 코딩 유닛(SCU: Smallest Coding Unit)이라고 지칭할 수 있다.

- [0077] 또한, 트리 구조를 갖는 CU은 미리 정해진 최대 깊이 정보(또는, 최대 레벨 정보)를 가지고 계층적으로 분할될 수 있다. 그리고, 각각의 분할된 CU은 깊이 정보를 가질 수 있다. 깊이 정보는 CU의 분할된 횟수 및/또는 정도를 나타내므로, CU의 크기에 관한 정보를 포함할 수도 있다.
- [0078] LCU가 쿼드 트리 형태로 분할되므로, LCU의 크기 및 최대 깊이 정보를 이용하면 SCU의 크기를 구할 수 있다. 또는 역으로, SCU의 크기 및 트리의 최대 깊이 정보를 이용하면, LCU의 크기를 구할 수 있다.
- [0079] 하나의 CU에 대하여, 해당 CU이 분할 되는지 여부를 나타내는 정보(예를 들어, 분할 CU 플래그(split_cu_flag))가 디코더에 전달될 수 있다. 이 분할 정보는 SCU을 제외한 모든 CU에 포함되어 있다. 예를 들어, 분할 여부를 나타내는 플래그의 값이 '1'이면 해당 CU은 다시 4개의 CU으로 나누어지고, 분할 여부를 나타내는 플래그의 값이 '0'이면 해당 CU은 더 이상 나누어지지 않고 해당 CU에 대한 처리 과정이 수행될 수 있다.
- [0080] 상술한 바와 같이, CU는 인트라 예측 또는 인터 예측이 수행되는 코딩의 기본 단위이다. HEVC는 입력 영상을 보다 효과적으로 코딩하기 위하여 CU를 예측 유닛(PU: Prediction Unit) 단위로 분할한다.
- [0081] PU는 예측 블록을 생성하는 기본 단위로서, 하나의 CU 내에서도 PU 단위로 서로 다르게 예측 블록을 생성할 수 있다. 다만, 하나의 CU 내에 속한 PU들은 인트라 예측과 인터 예측이 혼합되어 사용되지 않으며, 하나의 CU 내에 속한 PU들은 동일한 예측 방법(즉, 인트라 예측 혹은 인터 예측)으로 코딩된다.
- [0082] PU는 쿼드-트리 구조로 분할되지 않으며, 하나의 CU에서 미리 정해진 형태로 한번 분할된다. 이에 대하여 아래 도면을 참조하여 설명한다.
- [0083] 도 4는 본 발명에 적용될 수 있는 예측 유닛을 설명하기 위한 도면이다.
- [0084] PU는 PU가 속하는 CU의 코딩 모드로 인트라 예측 모드가 사용되는지 인터 예측 모드가 사용되는지에 따라 상이하게 분할된다.
- [0085] 도 4(a)는 인트라 예측 모드가 사용되는 경우의 PU를 예시하고, 도 4(b)는 인터 예측 모드가 사용되는 경우의 PU를 예시한다.
- [0086] 도 4(a)를 참조하면, 하나의 CU의 크기가 $2N \times 2N$ ($N=4, 8, 16, 32$)인 경우를 가정하면, 하나의 CU는 2가지 타입(즉, $2N \times 2N$ 또는 $N \times N$)으로 분할될 수 있다.
- [0087] 여기서, $2N \times 2N$ 형태의 PU로 분할되는 경우, 하나의 CU 내에 하나의 PU만이 존재하는 것을 의미한다.
- [0088] 반면, $N \times N$ 형태의 PU로 분할되는 경우, 하나의 CU는 4개의 PU로 분할되고, 각 PU 단위 별로 서로 다른 예측 블록이 생성된다. 다만, 이러한 PU의 분할은 CU의 휘도 성분에 대한 CB의 크기가 최소 크기인 경우(즉, CU가 SCU인 경우)에만 수행될 수 있다.
- [0089] 도 4(b)를 참조하면, 하나의 CU의 크기가 $2N \times 2N$ ($N=4, 8, 16, 32$)인 경우를 가정하면, 하나의 CU는 8가지의 PU 타입(즉, $2N \times 2N$, $N \times N$, $2N \times N$, $N \times 2N$, $nL \times 2N$, $nR \times 2N$, $2N \times nU$, $2N \times nD$)으로 분할될 수 있다.
- [0090] 인트라 예측과 유사하게, $N \times N$ 형태의 PU 분할은 CU의 휘도 성분에 대한 CB의 크기가 최소 크기인 경우(즉, CU가 SCU인 경우)에만 수행될 수 있다.
- [0091] 인터 예측에서는 가로 방향으로 분할되는 $2N \times N$ 형태 및 세로 방향으로 분할되는 $N \times 2N$ 형태의 PU 분할을 지원한다.
- [0092] 또한, 비대칭 움직임 분할(AMP: Asymmetric Motion Partition) 형태인 $nL \times 2N$, $nR \times 2N$, $2N \times nU$, $2N \times nD$ 형태의 PU 분할을 지원한다. 여기서, 'n'은 2N의 1/4 값을 의미한다. 다만, AMP는 PU가 속한 CU가 최소 크기의 CU인 경우 사용될 수 없다.
- [0093] 하나의 CTU 내의 입력 영상을 효율적으로 부호화하기 위해 코딩 유닛(CU), 예측 유닛(PU), 변환 유닛(TU)의 최적의 분할 구조는 아래와 같은 수행 과정을 거쳐 최소 율-왜곡(Rate-Distortion) 값을 기반으로 결정될 수 있다. 예를 들어, 64×64 CTU 내 최적의 CU 분할 과정을 살펴보면, 64×64 크기의 CU에서 8×8 크기의 CU까지의 분할 과정을 거치면서 율-왜곡 비용을 계산할 수 있다. 구체적인 과정은 다음과 같다.
- [0094] 1) 64×64 크기의 CU에 대해 인터/인트라 예측, 변환/양자화, 역양자화/역변환 및 엔트로피 인코딩 수행을 통해

최소의 율-왜곡 값을 발생시키는 최적의 PU와 TU의 분할 구조를 결정한다.

- [0095] 2) 64×64 CU를 32×32 크기의 CU 4개로 분할하고 각 32×32 CU에 대해 최소의 율-왜곡 값을 발생시키는 최적의 PU와 TU의 분할 구조를 결정한다.
- [0096] 3) 32×32 CU를 16×16 크기의 CU 4개로 다시 분할하고, 각 16×16 CU에 대해 최소의 율-왜곡 값을 발생시키는 최적의 PU와 TU의 분할 구조를 결정한다.
- [0097] 4) 16×16 CU를 8×8 크기의 CU 4개로 다시 분할하고, 각 8×8 CU에 대해 최소의 율-왜곡 값을 발생시키는 최적의 PU와 TU의 분할 구조를 결정한다.
- [0098] 5) 위의 3)의 과정에서 산출한 16×16 CU의 율-왜곡 값과 위의 4)의 과정에서 산출한 4개 8×8 CU의 율-왜곡 값의 합을 비교하여 16×16 블록 내에서 최적의 CU의 분할 구조를 결정한다. 이 과정을 나머지 3개의 16×16 CU들에 대해서도 동일하게 수행한다.
- [0099] 6) 위의 2)의 과정에서 계산된 32×32 CU의 율-왜곡 값과 위의 5)의 과정에서 획득한 4개 16×16 CU의 율-왜곡 값의 합을 비교하여 32×32 블록 내에서 최적의 CU의 분할 구조를 결정한다. 이 과정을 나머지 3개의 32×32 CU들에 대해서도 동일하게 수행한다.
- [0100] 7) 마지막으로, 위의 1)의 과정에서 계산된 64×64 CU의 율-왜곡 값과 위의 6)의 과정에서 획득한 4개 32×32 CU의 율-왜곡 값의 합을 비교하여 64×64 블록 내에서 최적의 CU의 분할 구조를 결정한다.
- [0101] 인트라 예측 모드에서, PU 단위로 예측 모드가 선택되고, 선택된 예측 모드에 대해 실제 TU 단위로 예측과 재구성이 수행된다.
- [0102] TU는 실제 예측과 재구성이 수행되는 기본 단위를 의미한다. TU는 휘도(luma) 성분에 대한 변환 블록(TB: Transform Block)과 이에 대응하는 두 개의 색차(chroma) 성분에 대한 TB를 포함한다.
- [0103] 앞서 도 3의 예시에서 하나의 CTU가 쿼드-트리 구조로 분할되어 CU가 생성되는 것과 같이, TU는 코딩하려는 하나의 CU로부터 쿼드-트리 구조로 계층적으로 분할된다.
- [0104] TU는 쿼드-트리 구조로 분할되므로 CU로부터 분할된 TU는 다시 더 작은 하위 TU로 분할될 수 있다. HEVC에서는 TU의 크기는 32×32 , 16×16 , 8×8 , 4×4 중 어느 하나로 정해질 수 있다.
- [0105] 다시 도 3을 참조하면, 쿼드-트리의 루트 노드(root node)는 CU와 관련된다고 가정한다. 쿼드-트리는 리프 노드(leaf node)에 도달할 때까지 분할되고, 리프 노드는 TU에 해당한다.
- [0106] 보다 구체적으로 살펴보면, CU는 루트 노드(root node)에 해당되고, 가장 작은 깊이(depth)(즉, depth=0) 값을 가진다. 입력 영상의 특성에 따라 CU가 분할되지 않을 수도 있으며, 이 경우 CU는 TU에 해당한다.
- [0107] CU는 쿼드 트리 형태로 분할될 수 있으며, 그 결과 깊이 1(depth=1)인 하위 노드들이 생성된다. 그리고, 1의 깊이를 가지는 하위 노드에서 더 이상 분할되지 않은 노드(즉, 리프 노드)는 TU에 해당한다. 예를 들어, 도 3(b)에서 노드 a, b 및 j에 대응하는 TU(a), TU(b), TU(j)는 CU에서 한 번 분할되었으며, 1의 깊이를 가진다.
- [0108] 1의 깊이를 가지는 노드 중 적어도 어느 하나는 다시 쿼드 트리 형태로 분할될 수 있으며, 그 결과 깊이 1(즉, depth=2)인 하위 노드들이 생성된다. 그리고, 2의 깊이를 가지는 하위 노드에서 더 이상 분할되지 않은 노드(즉, 리프 노드)는 TU에 해당한다. 예를 들어, 도 3(b)에서 노드 c, h 및 i에 대응하는 TU(c), TU(h), TU(i)는 CU에서 두 번 분할되었으며, 2의 깊이를 가진다.
- [0109] 또한, 2의 깊이를 가지는 노드 중 적어도 어느 하나는 다시 쿼드 트리 형태로 분할될 수 있으며, 그 결과 깊이 3(즉, depth=3)인 하위 노드들이 생성된다. 그리고, 3의 깊이를 가지는 하위 노드에서 더 이상 분할되지 않은 노드(즉, 리프 노드)는 CU에 해당한다. 예를 들어, 도 3(b)에서 노드 d, e, f, g에 대응하는 TU(d), TU(e), TU(f), TU(g)는 CU에서 3번 분할되었으며, 3의 깊이를 가진다.
- [0110] 트리 구조를 갖는 TU는 미리 정해진 최대 깊이 정보(또는, 최대 레벨 정보)를 가지고 계층적으로 분할될 수 있다. 그리고, 각각의 분할된 TU는 깊이 정보를 가질 수 있다. 깊이 정보는 TU의 분할된 횟수 및/또는 정도를 나타내므로, TU의 크기에 관한 정보를 포함할 수도 있다.
- [0111] 하나의 TU에 대하여, 해당 TU이 분할 되는지 여부를 나타내는 정보(예를 들어, 분할 TU 플래그(split_transform_flag))가 디코더에 전달될 수 있다. 이 분할 정보는 최소 크기의 TU를 제외한 모든 TU에 포함되어 있다. 예를 들어, 분할 여부를 나타내는 플래그의 값이 '1'이면 해당 TU는 다시 4개의 TU로 나누어지고,

분할 여부를 나타내는 플래그의 값이 '0'이면 해당 TU은 더 이상 나누어지지 않는다.

[0112] **예측(prediction)**

[0113] 디코딩이 수행되는 현재 처리 유닛을 복원하기 위해서 현재 처리 유닛이 포함된 현재 픽처 또는 다른 픽처들의 디코딩된 부분을 이용할 수 있다.

[0114] 복원에 현재 픽처만을 이용하는, 즉 화면내 예측만을 수행하는 픽처(슬라이스)를 인트라 픽처 또는 I 픽처(슬라이스), 각 유닛을 예측하기 위하여 최대 하나의 움직임 벡터 및 레퍼런스 인덱스를 이용하는 픽처(슬라이스)를 예측 픽처(predictive picture) 또는 P 픽처(슬라이스), 최대 두 개의 움직임 벡터 및 레퍼런스 인덱스를 이용하는 픽처(슬라이스)를 쌍예측 픽처(Bi-predictive picture) 또는 B 픽처(슬라이스)라고 지칭할 수 있다.

[0115] 인트라 예측은 동일한 디코딩된 픽처(또는 슬라이스)의 데이터 요소(예를 들어, 샘플 값 등)으로부터 현재 처리 블록을 도출하는 예측 방법을 의미한다. 즉, 현재 픽처 내의 복원된 영역들을 참조하여 현재 처리 블록의 픽셀 값을 예측하는 방법을 의미한다.

[0116] 인터 예측은 현재 픽처 이외의 픽처의 데이터 요소(예를 들어, 샘플 값 또는 움직임 벡터 등)의 기반하여 현재 처리 블록을 도출하는 예측 방법을 의미한다. 즉, 현재 픽처 이외의 복원된 다른 픽처 내의 복원된 영역들을 참조하여 현재 처리 블록의 픽셀값을 예측하는 방법을 의미한다.

[0117] 이하, 인트라 예측에 대하여 보다 상세히 살펴본다.

[0118] **인트라 예측(Intra prediction)(또는 화면 내 예측)**

[0119] 도 5는 본 발명이 적용되는 실시예로서, 인트라 예측 방법을 예시하는 도면이다.

[0120] 도 5를 참조하면, 디코더는 현재 처리 블록의 인트라 예측 모드를 도출(derivation)한다(S501).

[0121] 인트라 예측에서는 예측 모드에 따라 예측에 사용되는 참조 샘플의 위치에 대한 예측 방향을 가질 수 있다. 예측 방향을 가지는 인트라 예측 모드를 인트라 방향성 예측 모드(Intra_Angular prediction mode)라고 지칭한다. 반면, 예측 방향을 가지지 않는 인트라 예측 모드로서, 인트라 플래너(INTRA_PLANAR) 예측 모드, 인트라 DC(INTRA_DC) 예측 모드가 있다.

[0122] 표 1은 인트라 예측 모드와 관련 명칭에 대하여 예시하고, 도 6은 인트라 예측 모드에 따른 예측 방향을 예시한다.

표 1

인트라 예측 모드 (Intra prediction mode)	관련 명칭 (Associated name)
0	인트라 플래너 (INTRA_PLANAR)
1	인트라 DC (INTRA_DC)
2 ... 34	인트라 방향성 2 ... 인트라 방향성 34 (INTRA_ANGULAR2 ... INTRA_ANGULAR34)

[0123]

[0124] 인트라 예측에서는 도출되는 예측 모드에 기반하여 현재 처리 블록에 대한 예측을 수행한다. 예측 모드에 따라 예측에 사용되는 참조 샘플과 구체적인 예측 방법이 달라지므로, 현재 블록이 인트라 예측 모드로 인코딩된 경우, 디코더는 예측을 수행하기 위해 현재 블록의 예측 모드를 도출한다.

[0125] 디코더는 현재 처리 블록의 주변 샘플들(neighboring samples)이 예측에 사용될 수 있는지 확인하고, 예측에 사용할 참조 샘플들을 구성한다(S502).

[0126] 인트라 예측에서 현재 처리 블록의 주변 샘플들은 $nS \times nS$ 크기의 현재 처리 블록의 좌측(left) 경계에 인접한 샘플 및 좌하측(bottom-left)에 이웃하는 총 $2 \times nS$ 개의 샘플들, 현재 처리 블록의 상측(top) 경계에 인접한 샘플 및 우상측(top-right)에 이웃하는 총 $2 \times nS$ 개의 샘플들 및 현재 처리 블록의 좌상측(top-left)에 이웃하는 1개의 샘플을 의미한다.

[0127] 그러나, 현재 처리 블록의 주변 샘플들 중 일부는 아직 디코딩되지 않았거나, 이용 가능하지 않을 수 있다. 이 경우, 디코더는 이용 가능한 샘플들로 이용 가능하지 않은 샘플들을 대체(substitution)하여 예측에 사용할 참

조 샘플들을 구성할 수 있다.

- [0128] 디코더는 인트라 예측 모드에 기반하여 참조 샘플의 필터링을 수행할 수 있다(S503).
- [0129] 참조 샘플의 필터링 수행 여부는 현재 처리 블록의 크기에 기반하여 결정될 수 있다. 또한, 참조 샘플의 필터링 방법은 인코더로부터 전달되는 필터링 플래그에 의해 결정될 수 있다.
- [0130] 디코더는 인트라 예측 모드와 참조 샘플들에 기반하여 현재 처리 블록에 대한 예측 블록을 생성한다(S504). 즉, 디코더는 인트라 예측 모드 도출 단계(S501)에서 도출된 인트라 예측 모드와 참조 샘플 구성 단계(S502)와 참조 샘플 필터링 단계(S503)를 통해 획득한 참조 샘플들에 기반하여 현재 처리 블록에 대한 예측 블록을 생성(즉, 현재 처리 블록 내 예측 샘플 생성)한다.
- [0131] 현재 처리 블록이 INTRA_DC 모드로 인코딩된 경우 처리 블록 간의 경계의 불연속성(discontinuity)을 최소화하기 위해, S504 단계에서 예측 블록의 좌측(left) 경계 샘플(즉, 좌측 경계에 인접한 예측 블록 내 샘플)과 상측(top) 경계 샘플(즉, 상측 경계에 인접한 예측 블록 내 샘플)을 필터링할 수 있다.
- [0132] 또한, S504 단계에서 인트라 방향성 예측 모드들 중 수직 방향 모드(vertical mode) 및 수평 방향 모드(horizontal mode)에 대해서도 INTRA_DC 모드와 유사하게 좌측 경계 샘플 또는 상측 경계 샘플에 필터링을 적용할 수 있다.
- [0133] 보다 구체적으로 살펴보면, 현재 처리 블록이 수직 방향 모드(vertical mode) 또는 수평 방향 모드(horizontal mode)로 인코딩된 경우, 예측 방향에 위치하는 참조 샘플에 기반하여 예측 샘플의 값을 도출할 수 있다. 이때, 예측 블록의 좌측 경계 샘플 또는 상측 경계 샘플 중 예측 방향에 위치하지 않는 경계 샘플이 예측에 사용되지 않는 참조 샘플과 인접할 수 있다. 즉, 예측에 사용되는 참조 샘플과의 거리보다 예측에 사용되지 않는 참조 샘플과의 거리가 훨씬 가까울 수 있다.
- [0134] 따라서, 디코더는 인트라 예측 방향이 수직 방향인지 수평 방향인지에 따라 적응적으로 좌측 경계 샘플들 또는 상측 경계 샘플들에 필터링을 적용할 수 있다. 즉, 인트라 예측 방향이 수직 방향인 경우, 좌측 경계 샘플들에 필터링을 적용하고, 인트라 예측 방향이 수평 방향인 경우, 상측 경계 샘플들에 필터링을 적용할 수 있다.
- [0135] MPM 모드(Most Probable Mode)
- [0136] 앞서 설명한 바와 같이, HEVC에서는 화면 내 예측(또는 인트라 예측)을 위해 33가지의 방향성 예측 방법과 두 가지의 무 방향성 예측 방법, 총 35가지 예측 방법을 사용하여 현재 블록의 예측 블록이 생성된다. 이때, HEVC에서는 이와 같은 35가지 예측 모드를 보다 적은 비트로 표현(또는 시그널링)하기 위해 화면 내 예측 모드의 통계적 특성이 이용된다.
- [0137] 일반적으로 부호화 블록은 주변 블록과 유사한 영상 특징을 가지기 때문에 화면 내 예측 모드 또한 같거나 비슷한 화면 내 예측 모드를 가질 확률이 높다. 이러한 특성을 고려하여 현재 PU의 좌측 PU와 상단 PU의 화면 내 예측 모드를 기반으로 현재 PU의 예측 모드를 부호화한다. 이때, 인코더/디코더는 주변 블록(또는 이웃하는 블록)의 예측 모드와 일반적으로 가장 많이 발생하는 예측 모드를 MPM 모드(Most Probable Mode)로 결정한다.
- [0138] 만약, 현재 PU의 예측 모드가 MPM 모드로 결정된다면 예측 모드를 표현하기 위해 사용되는 비트를 절약할 수 있고(2비트 이내로 표현), MPM 모드가 아닌 다른 예측 모드로 결정된다면 3개의 MPM 모드를 제외한 32개 모드 중 하나의 모드로 부호화하기 때문에 6비트가 아닌 5비트를 사용하여 화면 내 예측 모드를 표현할 수 있다.
- [0139] 도 7은 본 발명이 적용되는 실시예로서, 화면 내 예측 모드를 복호화하는 방법을 나타내는 흐름도이다.
- [0140] 디코더는 현재 블록이 화면 내 예측 모드로 부호화된 경우, MPM 플래그(flag)를 파싱(parsing)한다(S701).
- [0141] 디코더는 MPM 플래그로부터 현재 블록이 MPM 모드로 부호화되었는지 여부를 판단한다(S702).
- [0142] S702 단계에서 판단한 결과, 현재 블록이 MPM 모드로 부호화된 경우, 디코더는 MPM 인덱스(index)를 파싱한다(S703).
- [0143] 디코더는 MPM 인덱스로부터 도출되는 모드를 현재 블록의 화면 내 예측 모드로 설정한다(S704).
- [0144] 즉, 디코더는 주변 블록 중 어떤 블록의 화면 내 예측 모드를 MPM으로 사용할지(또는 비방향성 예측 모드를 사용할지) 결정하고, 결정된 화면 내 예측 모드로 복호화를 수행한다.
- [0145] S702 단계에서 판단한 결과, 현재 블록이 MPM 모드로 부호화되지 않은 경우, 디코더는 화면 내 예측 모드를 파

싱한다(S705).

- [0146] 전술한 바와 같이, MPM 모드를 제외한 나머지 32가지 예측 모드 중에서 인코더가 시그널링한 예측 모드를 파싱하고, 파싱된 화면 내 예측 모드로 복호화를 수행한다.
- [0147] 구체적인 MPM 모드의 결정 방법을 아래의 도면을 참조하여 설명한다.
- [0148] 도 8은 본 발명이 적용되는 실시예로서, MPM 모드를 결정하는 방법을 설명하기 위한 도면이다.
- [0149] 디코더는 현재 블록의 좌측에 인접한 블록의 예측 모드(이하, 'L 모드' 라 지칭한다)와 현재 블록의 상단에 인접한 블록의 예측 모드(이하, 'A 모드' 라 지칭한다)가 같은지 여부를 판단한다(S801).
- [0150] S801 단계에서 판단한 결과, L 모드와 A 모드가 다른 경우, 디코더는 첫 번째 MPM 모드(MPM[0])와 두 번째 MPM 모드(MPM[1])는 각각 L 모드, A 모드로 결정하고, 마지막 MPM 모드(MPM[2])는 플래너(planar), DC, 수직 방향(vertical) 모드 중 L 모드와 A 모드가 아닌 모드로 설정한다(S802).
- [0151] S801 단계에서 판단한 결과, L 모드와 A 모드가 같은 경우, 디코더는 L 모드의 예측 모드가 2(앞서 도 6 참조)보다 작은지 여부를 판단한다(S803).
- [0152] S803 단계에서 판단한 결과, L 모드가 2 보다 작지 않은 경우, 디코더는 첫 번째 MPM 모드(MPM[0]), 두 번째 MPM 모드(MPM[1]), 세 번째 MPM 모드(MPM[2])를 각각 L 모드, L 모드 - 1, L 모드 + 1로 설정한다(S804).
- [0153] S803 단계에서 판단한 결과, L 모드가 2 보다 작은 경우, 디코더는 첫 번째 MPM 모드(MPM[0]), 두 번째 MPM 모드(MPM[1]), 세 번째 MPM 모드(MPM[2])를 각각 플래너(planar), DC, 수직 방향(Vertical) 모드로 설정한다(S805).
- [0154] 랜덤 포레스트(RF: Random Forest)
- [0155] 랜덤 포레스트는 결정 트리(Decision tree)(또는 랜덤 트리(Random tree))들의 집합으로 분류나 회귀 분석에 사용되는 기계 학습 기법의 일종이다. 아래의 도면을 참조하여 설명한다.
- [0156] 도 9는 본 발명이 적용되는 실시예로서, 랜덤 포레스트 및 결정 트리를 설명하기 위한 도면이다.
- [0157] 도 9(a)는 결정 트리의 일 예로서, 결정 트리는 리프 노드(903)를 제외한 모든 노드가 2개 이하의 자식 노드(child node)를 가지는 이진 트리(binary tree) 구조를 가질 수 있다. 하나의 결정 트리는 노드(node)와 에지(edge)로 구성되며, 각 노드는 다음 계층의 노드와 에지로 연결되는 계층적 구조를 가진다.
- [0158] 결정 트리 내에서 노드는 다음과 같이, 루트 노드(root node)(901), 인터널 노드(internal node)(또는 분할 노드(split node))(902), 리프 노드(leaf node)(또는 터미널 노드(terminal node))(903)로 나뉘질 수 있다. 그리고, 에지에 의해 부모 노드와 자식 노드가 연결된다.
- [0159] - 루트 노드(901): 결정 트리의 최상위 노드를 말한다. 분할 함수(split function)와 분할 함수를 구성하는 분할 파라미터(split parameter)로 구성된다. 분할 함수는 입력(input)을 좌측 자식 노드(left child node)와 우측 자식 노드(right child node) 중 어느 노드로 보낼지 결정하는 함수에 해당한다. 분할 파라미터는 오프라인 학습(off-line training)을 통해 결정되고 저장될 수 있다.
- [0160] - 인터널 노드(902): 결정 트리를 구성하는 노드 중에서 리프 노드(903)와 루트 노드(901)가 아닌 노드를 말한다. 루트 노드(901)와 마찬가지로 분할 함수와 분할 파라미터로 구성된다.
- [0161] - 리프 노드(903): 결정 트리의 최종 노드를 말한다. 자신에게 도달한 입력의 클래스(class) 정보를 저장하고, 이를 최종 클래스 결정의 투표에 사용한다.
- [0162] 각 노드의 분할 파라미터는 정보 이득(information gain)을 최대화하는 방향으로 학습될 수 있다. 정보 이론에서 정보 이득은 현재 노드에 도달하는 학습 데이터를 자식 노드로 분할하였을 때 계산되는 불확실성의 감소로 정의된다.
- [0163] 즉, 정보 이득은 수학식 1과 같이 정의될 수 있다.

수학식 1

$$\text{Information Gain} = H(S_i) - \sum_{j \in L, R} \frac{|S_i^j|}{|S_i|} H(S_i^j)$$

[0164]

[0165]

여기서 S_i 는 i 번째 노드에서 좌측 자식 노드로 분할될 부분 집합 S_i^L 와 우측 자식 노드로 분할될 부분 집합 S_i^R 을 포함하는 데이터 집합을 나타낸다. 그리고, $H(S)$ 는 확률 변수 S 의 엔트로피를 나타낸다.

[0166]

화면 내 예측 모드 기반 영상 처리 방법

[0167]

본 발명은 머신 러닝(machine learning) 기술 중 하나인 랜덤 포레스트(Random Forest) 방법을 사용하여 화면 내 예측 모드의 시그널링을 최적화하여 부호화/복호화하는 방법을 제안한다. 랜덤 포레스트 방법을 사용함으로써 화면 내 예측 모드를 표현하기 위해 사용되는 비트 수를 줄일 수 있고, 이를 통해 부호화 효율을 향상시킬 수 있다.

[0168]

본 발명에서는 화면 내 예측 모드의 시그널링 없이 랜덤 포레스트를 이용하여 화면 내 예측 모드를 추정하는 방법을 제안한다.

[0169]

또한, 본 발명에서는 복수의 화면 내 예측 모드를 하나의 클래스로 구성하고, 랜덤 포레스트를 이용하여 클래스를 추정하는 방법을 제안한다.

[0170]

또한, 본 발명에서는 하나 이상의 랜덤 포레스트를 계층적으로 적용하여 화면 내 예측 모드를 복호화하는 방법을 제안한다.

[0171]

또한, 본 발명에서는 랜덤 포레스트를 이용하여 MPM 모드(most probable mode)를 재구성 또는 추가하는 방법을 제안한다.

[0172]

실시예 1

[0173]

본 실시예에서는 화면 내 예측 모드(또는 인트라 예측 모드)의 시그널링 없이 랜덤 포레스트를 이용하여 화면 내 예측 모드를 추정하는 방법을 제안한다.

[0174]

구체적으로, 랜덤 포레스트 내 각 결정 트리(또는 랜덤 트리)에서 루트 노드(root node)로부터 리프 노드(leaf node)까지 탐색하여 클래스(class)를 결정하고, 각 결정 트리에서 결정된 클래스를 종합하여 최종 클래스를 결정함으로써 화면 내 예측 모드를 도출하는 방법을 제안한다.

[0175]

여기서, 랜덤 포레스트는 하나 이상의 결정 트리(또는 랜덤 트리)로 구성된다. 또한, 결정 트리의 각 리프 노드에는 클래스의 확률 정보가 저장되어 있을 수 있으며, 각 결정 트리에서 탐색을 통해 출력되는 클래스는 리프 노드에 저장된 클래스 중 가장 확률이 높은 클래스로 결정될 수 있다.

[0176]

그리고, 본 실시예에서는 각 클래스 마다 하나의 화면 내 예측 모드가 매핑되어 저장될 수 있다. 다시 말해, 각각의 리프 노드는 특정 화면 내 예측 모드에 대응될 수 있다.

[0177]

본 실시예에서 제안하는 방법은 인코더와 디코더에서 동일하게 적용될 수 있으며, 설명의 편의상 디코더에서 수행되는 방법을 기준으로 설명한다.

[0178]

랜덤 포레스트를 구성하는 랜덤 트리(또는 결정 트리) 구조는 앞서 도 9(a)에서 살펴본 바와 같다.

[0179]

다시 도 9(a)를 참조하면, 랜덤 트리는 노드와 에지로 구성된다. 루트 노드(901) 또는 인터널 노드(902)는 입력 블록의 화소 또는 특징 정보를 입력 받거나 저장할 수 있다.

[0180]

인코더/디코더는 랜덤 포레스트를 구성하는 랜덤 트리의 루트 노드(901)로부터 리프 노드(903)까지 분할(또는 탐색)하여, 도달된 리프 노드(903)의 클래스 정보(또는 화면 내 예측 모드)를 저장하고, 이를 최종 클래스 결정(또는 현재 블록의 화면 내 예측 모드 결정)의 투표에 사용할 수 있다.

[0181]

랜덤 포레스트가 복수 개의 랜덤 트리(또는 결정 트리)로 구성된 경우, 각각의 랜덤 트리에서 결정된 클래스(또는 화면 내 예측 모드)가 랜덤 포레스트의 클래스 결정 투표(voting)에 사용된다. 그리고, 랜덤 포레스트 내에서 가장 많은 투표를 얻은 클래스가 최적의 클래스(즉, 랜덤 포레스트의 클래스)로 결정될 수 있다.

[0182]

만약, 랜덤 포레스트가 단일의 랜덤 트리(또는 결정 트리)로 구성된 경우, 랜덤 트리에서 결정된 클래스(또는 화면 내 예측

모드)가 최적의 클래스(또는 현재 블록의 화면 내 예측 모드)로 결정될 수 있다.

- [0183] 인코더/디코더는 랜덤 트리를 설계하기 위해, 다음을 고려하여 분할 함수(split function)를 결정할 수 있다.
- [0184] - 분할 함수의 입력(Split function input): 분할 함수의 입력은 최적의 부호화 모드(즉, 현재 블록의 화면 내 예측 모드)를 정확하게 결정할 수 있도록 하는 특징(feature)으로서, 인코더/디코더는 현재 블록(예를 들면, 코딩 블록, CU, PU 등)에 이웃하는 샘플(neighbouring sample)을 분할 함수의 입력으로 사용할 수 있다.
- [0185] - 분할 파라미터(Split parameter): 분할 함수를 구성하는 분할 파라미터는 학습을 통해 결정될 수 있다. 분할 파라미터를 결정하기 위한 학습 방법은 후술한다. 분할 파라미터는 이웃하는 참조 샘플의 위치(또는 해당 위치의 참조 샘플의 값), 이웃하는 블록의 화면 내 예측 모드, 결정 임계값(decision threshold) t , 분할 함수 내 사용되는 참조 샘플의 개수 등, 또는 이들의 조합으로 결정될 수 있다.
- [0186] - 분할 함수의 출력(Split function output): 출력은 0 또는 1의 값을 가지며, 각각의 값은 좌측 자식 노드 또는 우측 자식 노드에 대응될 수 있다. 즉, 분할 함수의 출력 값에 따라 좌측 자식 노드 또는 우측 자식 노드로 분할될 수 있다.
- [0187] 분할 파라미터 셋(split parameter set)은 복수의 분할 파라미터 집합으로서, 이하, 본 발명의 설명에 있어 분할 파라미터 셋을 세타(theta)로 지칭한다.
- [0188] 하나의 랜덤 트리에서 각 노드의 분할 함수의 형태는 동일하지만, 각 노드의 분할 함수에 사용되는 파라미터 또는 파라미터 셋(parameter set)의 값은 다를 수 있다. 또한, 하나의 랜덤 트리에서 파라미터 셋을 공유하고, 각 노드마다 전체 파라미터 셋의 전체 또는 일부를 사용하여 분할 함수를 구성할 수 있다.
- [0189] 랜덤 포레스트는 학습 단계와 테스트 단계를 거쳐 설계될 수 있다. 학습 단계를 통해 랜덤 트리의 각 노드에 저장된 분할 함수의 분할 파라미터가 결정되고, 테스트 단계를 통해 결정된 분할 파라미터를 이용한 데이터의 분류 결과(또는 각 클래스의 확률 분포 결과)가 축적될 수 있다.
- [0190] 이 경우, 학습 단계와 테스트 단계에서 이용되는 샘플 데이터는 서로 다를 수도 있고, 동일할 수도 있다. 이하, 설명의 편의를 위해 학습 단계에서 이용되는 샘플 데이터를 학습 샘플 데이터, 테스트 단계에서 이용되는 샘플 데이터를 테스트 샘플 데이터라 지칭한다.
- [0191] 랜덤 포레스트(또는 랜덤 트리) 내 분할 함수는 다양한 여러 방법에 의해 결정될 수 있다. 분할 함수를 설계하는 방법들을 예로 들어 설명한다. 다만, 본 발명이 이하에서 설명하는 예시에만 한정되는 것이 아님은 물론이다.
- [0192] 1) 첫 번째 방법에 의한 분할 함수 h 는 수학적식 2 와 같이 정의될 수 있다.

수학적식 2

- [0193] $h(v, \theta) = |R_{x0,y0} - R_{x1,y1}| < t,$

- [0194] 수학적식 2를 참조하면, 현재 블록에 이웃하는 참조 샘플을 분할 함수의 입력으로 사용할 수 있다. 여기서, v 는 랜덤 트리의 각 노드를 나타내며, 전술한 바와 같이 세타(theta)는 분할 함수를 구성하기 위하여 이용되는 분할 파라미터 셋을 의미한다.
- [0195] 구체적으로, 수학적식 2의 분할 함수를 구성하는 세타는 결정 임계값(decision threshold) t 와 현재 블록에 이웃하는 중 좌표 $(x0, y0)$, $(x1, y1)$ 에 위치하는 2개의 참조 샘플 $R_{x0,y0}$, $R_{x1,y1}$ 로 구성된다.
- [0196] 도 10은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.
- [0197] 도 10을 참조하면, $N \times N$ 크기의 현재 블록(HEVC를 예로 들면, PU)(1001)에 이웃하는 참조 샘플은 현재 블록의 좌측(left) 및 좌하측(bottom-left)에 인접한 총 $2 \times N$ 개의 샘플들, 현재 블록의 상측(top) 및 우상측(top-right)에 인접한 총 $2 \times N$ 개의 샘플들 및 현재 블록의 좌상측(top-left)에 인접한 1개의 샘플로 구성될 수 있다.
- [0198] 전술한 바와 같이, 세타는 학습 샘플 데이터를 이용하여 학습의 결과로 결정될 수 있다. 이때, 인코더/디코더가 동일한 학습 샘플 데이터를 이용하여 학습할 수도 있고, 인코더/디코더가 학습 결과 결정된 세타를 공유할 수도

있다.

[0199] 학습 단계에서, 각 노드 별로 $N \times N$ 크기의 현재 블록에 이웃하는 참조 샘플 중 두 화소를 N 번 임의의 선택하여, 그 중에서 정보 이득(information gain)을 최대화하는 2개의 참조 샘플과 임계값 t 가 분할 파라미터로 결정될 수 있다.

[0200] 학습 단계에서 분할 파라미터를 결정한 후 테스트 단계에서, 각 리프 노드에 저장된 클래스의 확률 값(즉, 데이터의 분류 결과)을 축적하기 위한 테스트 절차를 수행할 수 있다. 테스트 단계에서는 학습을 통해 결정된 세타를 이용하여 각 노드에서의 분할(또는 탐색)을 통해 하나의 리프 노드를 탐색할 수 있다. 테스트 절차의 구체적인 방법을 아래의 도면을 참조하여 설명한다.

[0201] 도 11은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

[0202] 도 11을 참조하면, 현재 블록의 화면 내 예측 모드를 추정(또는 예측)하는 랜덤 포레스트 내 랜덤 트리가 도 11(a)와 같이 구성되고, 랜덤 트리 내 각 노드의 분할 파라미터가 아래의 표 2와 같이 학습된 경우를 가정한다. 그리고, 테스트 절차에서 이용되는 테스트 샘플 데이터(즉, 이웃하는 참조 샘플)가 도 11(b)와 같은 경우를 가정한다.

표 2

Split parameters	$R_{(x0,y0)}$	$R_{(x1,y1)}$	t
Node0	$R_{(3,0)}$	$R_{(0,5)}$	100
...	...	...	...
Node2	$R_{(0,4)}$	$R_{(1,0)}$	50
...	...	...	...
Node6	$R_{(0,5)}$	$R_{(3,0)}$	175

[0203]

[0204] 도 11(a)를 참조하면, 학습된 랜덤 트리를 따라 테스트 샘플이 이동하는 경로를 나타낸다. 그리고, 표 2는 랜덤 트리 내 각 노드에 저장된 분할 파라미터(2개의 참조 샘플의 위치($R(x0,y0)$, $R(x1,y1)$)와 임계값 t)를 나타낸다.

[0205] 테스트 샘플 데이터는 노드 0(1101)에서의 이진 테스트(binary test)를 통해 좌측 자식 노드(left child node)로 이동할지 또는 우측 자식 노드(right child node)로 이동할지 결정될 수 있다. 노드 0(1101)에서의 이진 테스트 결과, $|R_{(3,0)} - R_{(0,5)}| = |40 - 210| = 170 < 100$ 로 수학적 2를 만족하지 못하기 때문에, 0의 값이 출력되어 우측 자식 노드인 노드 2(1102)로 이동할 수 있다.

[0206] 노드 2(1102)에서 이진 테스트 결과, $|R_{(0,4)} - R_{(1,0)}| = |200 - 20| = 180 < 50$ 이고 수학적 2를 만족하지 못하기 때문에, 0의 값이 출력되어 우측 자식 노드인 노드 6(1103)으로 이동할 수 있다.

[0207] 노드 6(1103)에서 이진 테스트 결과, $|R_{(0,5)} - R_{(3,0)}| = |210 - 40| = 170 < 175$ 로 수학적 2를 만족하기 때문에, 1의 값이 출력되어 좌측 자식 노드로서 리프 노드(leaf node)인 노드 11(1104)에 도달할 수 있다.

[0208] 이와 같이, 테스트 단계에서는 테스트 샘플 데이터를 이용하여, 학습 단계에서 분할 함수의 분할 파라미터가 결정된 랜덤 트리의 각 리프 노드에 클래스 별 확률 값을 축적할 수 있다.

[0209] 클래스가 9개인 경우를 가정하면, 예를 들어, 테스트 절차를 거쳐 노드 11(1104)에는 {0.1, 0.2, 0.5, 0.0, 0.05, 0.0, 0.15, 0.0}와 같이 각 클래스에 대한 확률 분포가 저장(또는 축적)될 수 있다.

[0210] 부호화/복호화 단계에서는 테스트 절차에서와 동일한 방법으로 랜덤 트리의 루트 노드로부터 리프 노드까지 도달할 수 있다.

- [0211] 즉, 학습 단계에서 결정된 분할 파라미터와 분할 함수를 이용하여 각 노드에서 자식 노드를 탐색하여 리프 노드 까지 도달할 수 있다. 그리고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스가 최종적으로 결정될 수 있다.
- [0212] 만약, 복수 개의 랜덤 트리로 랜덤 포레스트가 구성되는 경우, 인코더/디코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 현재 블록의 화면 내 예측 모드로 결정할 수 있다. 다시 말해, 투표(voting) 단계를 통해, 각 랜덤 트리에서 결정된 클래스가 투표되고 가장 많은 투표수를 받은 클래스가 최종적으로 결정될 수 있다. 그리고 결정된 클래스 정보를 이용하여 현재 블록의 화면 내 예측 모드를 결정될 수 있다. 다시 말해, 결정된 클래스에 매핑된 화면 내 예측 모드를 현재 블록의 화면 내 예측 모드로 결정될 수 있다.
- [0213] 예를 들어, 부호화/복호화 단계에서 도 11(a)와 같은 랜덤 트리의 루트 노드(노드 0)(1101)로부터 노드 11(1104)의 리프 노드까지 분할(또는 탐색)되어 도달한 경우, 노드 11(1104)에 저장된 클래스 중에서 최대 확률을 가지는 클래스인 클래스 3이 예측(또는 결정)될 수 있다.
- [0214] 또한, 예를 들어, 랜덤 포레스트 내에서 10개의 랜덤 트리가 존재한다고 가정하면, 10개의 랜덤 트리 중에서 7개의 랜덤 트리에서 클래스 3, 2개의 랜덤 트리에서 클래스 2, 1개의 랜덤 트리에서 클래스 9가 결정(또는 예측)되었다면, 투표(voting) 절차를 거쳐 가장 많은 투표수를 받은 클래스 3이 최종적으로 랜덤 포레스트의 클래스로 결정될 수 있고, 상기 결정된 클래스로부터 현재 블록의 화면 내 예측 모드가 결정될 수 있다.
- [0215] 2) 두 번째 방법에 의한 분할 함수는 이웃하는 참조 샘플 중 1개의 참조 샘플 라인(또는 참조 샘플 배열)을 사용하는 첫 번째 방법과 다르게, 이웃하는 참조 샘플 중 2개의 참조 샘플 라인을 입력으로 사용할 수 있다.
- [0216] 즉, 현재 블록의 예측 블록을 생성하기 위하여 다중 참조 샘플 라인(또는 참조 샘플 배열)을 참조하는 경우, 분할 함수도 다중 참조 샘플 라인의 참조 샘플을 이용하여 설계될 수 있다.
- [0217] 이하에서, 2개의 이웃하는 참조 샘플 라인을 사용하여 분할 함수를 설계하는 방법을 도 12 내지 도 17를 참조하여 예를 들어 설명한다. 다만, 본 발명이 이하에서 설명하는 예시에만 한정되는 것이 아니다.
- [0218] 도 12는 본 발명의 일 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.
- [0219] 도 12를 참조하면, 예를 들어, 분할 함수 h 는 수학식 3과 같이 정의될 수 있다.

수학식 3

- [0220]
$$h(v, theta) = |R_{x0,y0}^O - R_{x1,y1}^I| < t$$
- [0221] 도 12를 참조하면, 현재 블록에 이웃하는 2개의 참조 샘플 라인 중 현재 블록에 더 인접한(즉, 안쪽에 위치한) 참조 샘플 라인은 $N \times N$ 크기의 현재 블록의 좌측(left) 및 좌하측(bottom-left)에 인접한 총 $2 \times N$ 개의 샘플($R^I_{(0,1)}$, $R^I_{(0,2)}$, ..., $R^I_{(0,2N-1)}$, $R^I_{(0,2N)}$)들, 현재 블록의 상측(top) 및 우상측(top-right)에 인접한 총 $2 \times N$ 개의 샘플($R^I_{(1,0)}$, $R^I_{(2,0)}$, ..., $R^I_{(2N-1,0)}$, $R^I_{(2N,0)}$)들 및 현재 블록의 좌상측(top-left)에 인접한 1개의 샘플($R^I_{(0,0)}$)로 구성될 수 있다.
- [0222] 그리고, 현재 블록에 이웃하는 2개의 참조 샘플 라인 중 현재 블록에 더 인접하지 않은(즉, 바깥쪽에 위치한) 참조 샘플 라인은 $N \times N$ 크기의 현재 블록의 좌측(left) 및 좌하측(bottom-left)에 이웃하는 총 $2 \times N$ 개의 샘플($R^O_{(-1,1)}$, $R^O_{(-1,2)}$, ..., $R^O_{(-1,2N-1)}$, $R^O_{(-1,2N)}$)들, 현재 블록의 상측(top) 및 우상측(top-right)에 이웃하는 총 $2 \times N$ 개의 샘플($R^O_{(1,-1)}$, $R^O_{(2,-1)}$, ..., $R^O_{(2N-1,-1)}$, $R^O_{(2N,-1)}$)들 및 현재 블록의 좌상측(top-left)에 이웃하는 3개의 샘플($R^O_{(-1,-1)}$, $R^O_{(0,-1)}$, $R^O_{(-1,0)}$)들로 구성될 수 있다.
- [0223] 수학식 3을 참조하면, $R^O_{(x0,y0)}$ 는 2개의 참조 샘플 라인 중 바깥쪽에 위치한 참조 샘플 라인의 $(x0,y0)$ 위치의 참조 샘플을 지칭하고, $R^I_{(x1,y1)}$ 는 안쪽에 위치한 참조 샘플 라인의 $(x1,y1)$ 위치의 참조 샘플을 지칭한다. 그리고, 세타는 $\{R^O_{(x0,y0)}, R^I_{(x1,y1)}, t\}$ 로 구성될 수 있다.
- [0224] 도 12에서는 바깥 쪽에 위치한 참조 샘플 라인 내 1개의 샘플로서 $R^O_{(0,-1)}$ (1201)과 안쪽에 위치한 참조 샘플 라인 내 1 개의 샘플로서 $R^I_{(N+1,0)}$ (1202)이 선택되는 경우를 예시한다.
- [0225] 학습 단계에서, 위와 같이 현재 블록에 이웃하는 2개의 참조 샘플 라인 중 바깥 쪽에 위치한 참조 샘플 라인 내 1개의 샘플($R^O_{(x0,y0)}$)과 안쪽에 위치한 참조 샘플 라인 내 1 개의 샘플($R^I_{(x1,y1)}$)을 N 번 임의의 선택하여

정보 이득을 최대화하는 분할 파라미터가 결정될 수 있다.

- [0226] 그리고, 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 랜덤 트리의 리프 노드를 탐색하여, 리프 노드에 데이터의 분류 결과를 축적할 수 있다.
- [0227] 그리고, 앞서 설명한 방법과 같이, 인코더/디코더는 테스트 절차와 같은 방법으로 리프 노드까지 도달할 수 있고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스를 결정할 수 있다.
- [0228] 만약, 복수 개의 랜덤 트리로부터 랜덤 포레스트가 구성되는 경우, 인코더/디코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 랜덤 포레스트의 클래스로 결정할 수 있다.
- [0229] 아래의 도면을 참조하여, 2개의 이웃하는 참조 샘플 라인을 사용하는 분할 함수의 다른 일 예를 설명한다.
- [0230] 도 13은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.
- [0231] 도 13을 참조하면, 분할 함수 h 는 수학식 4와 같이 정의될 수 있다.

수학식 4

$$h(v, \theta) = \begin{cases} |R_{x_0, y_0}^o - R_{x_1, y_1}^l| < t_1 \\ |R_{x_2, y_2}^o - R_{x_1, y_1}^l| < t_2 \end{cases}$$

- [0232]
- [0233] 여기서, 분할 함수의 입력으로 2개의 참조 샘플 라인이 사용될 수 있으며, 앞서 설명한 도 12의 경우와 동일하게 참조 샘플 라인이 구성될 수 있다.
- [0234] 수학식 4를 참조하면, $R^o_{(x_0, y_0)}$ 와 $R^o_{(x_2, y_2)}$ 는 2개의 참조 샘플 라인 중 바깥쪽에 위치한 참조 샘플 라인의 (x_0, y_0) , (x_2, y_2) 위치의 참조 샘플을 지칭하고, $R^l_{(x_1, y_1)}$ 는 안쪽에 위치한 참조 샘플 라인의 (x_1, y_1) 위치의 참조 샘플을 지칭한다. 그리고, 세타는 $\{R^o_{(x_0, y_0)}, R^o_{(x_2, y_2)}, R^l_{(x_1, y_1)}, t_1, t_2\}$ 로 구성될 수 있다.
- [0235] 도 13에서는 바깥 쪽에 위치한 참조 샘플 라인 내 2개의 샘플로서 $R^o_{(-1, 2)}(1301)$ 및 $R^o_{(2, -1)}(1303)$ 과 안쪽에 위치한 참조 샘플 라인 내 1 개의 샘플로서 $R^l_{(N+1, 0)}(1302)$ 이 선택되는 경우를 예시한다.
- [0236] $R^o_{(x_0, y_0)}$ 과 $R^l_{(x_1, y_1)}$ 의 차분값이 제 1 임계값(t_1)보다 크거나 또는 $R^o_{(x_2, y_2)}$ 과 $R^l_{(x_1, y_1)}$ 의 차분값이 제 2 임계값(t_2)보다 큰 경우, 0의 값이 출력되어 우측 자식 노드로 이동될 수 있다. 반면에, 두 차분값이 모두 임계값보다 작은 경우, 1의 값이 출력되어 좌측 자식 노드로 이동될 수 있다.
- [0237] 학습 단계에서, 도 13의 예시와 같이 현재 블록에 이웃하는 2개의 참조 샘플 라인 중 바깥 쪽에 위치한 참조 샘플 라인 내 2개의 샘플($R^o_{(x_0, y_0)}$, $R^o_{(x_2, y_2)}$)과 안쪽에 위치한 참조 샘플 라인 내 1 개의 샘플($R^l_{(x_1, y_1)}$)을 N 번 임의의 선택하여 정보 이득을 최대화하는 분할 파라미터가 결정될 수 있다.
- [0238] 그리고, 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 랜덤 트리의 루트 노드에서부터 리프 노드까지 분할하여, 리프 노드에 데이터의 분류 결과를 축적할 수 있다.
- [0239] 그리고, 앞서 설명한 방법과 같이, 인코더/디코더는 테스트 절차와 같은 방법으로 리프 노드까지 도달할 수 있고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스를 결정할 수 있다.
- [0240] 만약, 복수 개의 랜덤 트리로부터 랜덤 포레스트가 구성되는 경우, 인코더/디코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 랜덤 포레스트의 클래스로 결정할 수 있다.
- [0241] 아래의 도면을 참조하여, 2개의 이웃하는 참조 샘플 라인을 사용하는 분할 함수의 다른 일 예를 설명한다.
- [0242] 도 14은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.
- [0243] 도 14을 참조하면, 분할 함수 h 는 수학식 5와 같이 정의될 수 있다.

수학식 5

$$h(v, \theta) = \begin{cases} |R_{x_0, y_0}^o - R_{x_1, y_1}^l| < t_1 \\ |R_{x_2, y_2}^o - R_{x_3, y_3}^l| < t_2 \end{cases}$$

[0244]

[0245] 여기서, 분할 함수의 입력으로 2개의 참조 샘플 라인이 사용될 수 있으며, 앞서 설명한 도 12의 경우와 동일하게 참조 샘플 라인이 구성될 수 있다.

[0246] 수학식 4의 분할 함수와 차이점을 위주로 설명하면, 2개의 참조 샘플 라인 중 바깥쪽에 위치한 참조 샘플 라인 뿐만 아니라 안쪽에 위치한 참조 샘플 라인에서도 2개의 참조 샘플(즉, $R^l_{x_1, y_1}$, $R^l_{x_3, y_3}$)이 선택될 수 있다. 그리고, 세타는 $\{R^o_{x_0, y_0}, R^o_{x_2, y_2}, R^l_{x_1, y_1}, R^l_{x_3, y_3}, t_1, t_2\}$ 로 구성될 수 있다.

[0247] 도 14에서는 바깥 쪽에 위치한 참조 샘플 라인 내 2개의 샘플로서 $R^o_{(-1, 2)}$ (1402) 및 $R^o_{(2, -1)}$ (1401)과 안쪽에 위치한 참조 샘플 라인 내 2개의 샘플로서 $R^l_{(2N, 0)}$ (1403) 및 $R^l_{(0, N)}$ (1404)이 선택되는 경우를 예시한다.

[0248] 위와 같이 2개의 참조 샘플 라인 중 안쪽에 위치한 참조 샘플 라인에서도 2개의 참조 샘플이 분할 파라미터로 결정되는 점을 제외하고, 앞서 도 13에서 설명한 방법과 동일한 방법으로 학습 단계와 테스트 단계가 수행될 수 있고, 부호화/복호화 단계에서 클래스가 결정될 수 있다.

[0249] 아래의 도면을 참조하여, 2개의 이웃하는 참조 샘플 라인을 사용하는 분할 함수의 다른 일 예를 설명한다.

[0250] 도 15는 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

[0251] 예를 들어, 분할 함수 h 는 수학식 6과 같이 정의될 수 있다.

수학식 6

$$h(v, \theta) = \{\theta < t\}$$

[0252]

[0253] 분할 함수 h 의 입력으로 2개의 참조 샘플 라인이 사용될 수 있으며, 앞서 설명한 도 12의 경우와 동일하게 참조 샘플 라인이 구성될 수 있다.

[0254] 여기서, 세타는 $\{\theta, t\}$ 로 구성될 수 있다. 또한, $\theta = \tan^{-1}(k_m)$ 으로 정의될 수 있고, k_m 은 $\Delta R^y_{(i, j)} / \Delta R^x_{(i, j)}$ 로 구성될 수 있다.

[0255] 도 15를 참조하면, $\Delta R^x_{(i, j)}$ 은 2개의 참조 샘플 라인 내 각 참조 샘플의 위치에서의 샘플값과 x 축 방향으로 우측에 인접한 참조 샘플의 샘플값과의 차이값을 나타내고, $\Delta R^y_{(i, j)}$ 은 2개의 참조 샘플 라인 내 각 참조 샘플의 위치에서의 샘플값과 y 축 방향으로 하단에 인접한 참조 샘플의 샘플값과의 차이값을 나타낸다.

[0256] 학습 단계에서, $\Delta R^x_{(i, j)}$ 과 $\Delta R^y_{(i, j)}$ 의 L2 놈(norm)(또는 변화율-크기($\text{gradient.magnitude}$))을 최대로하는 경우의 k_m 값이 결정되고, 그로부터 θ 값이 결정될 수 있다. Δx 와 Δy 의 L2 놈은 수학식 7과 같이 정의된다.

수학식 7

$$\text{gradient.magnitude} = \sqrt{\Delta x^2 + \Delta y^2}$$

[0257]

[0258] 그리고, 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 랜덤 트리의 루트 노드에서부터 리프 노드까지 분할하여, 리프 노드에 데이터의 분류 결과를 축적할 수 있다.

[0259] 그리고, 앞서 설명한 방법과 같이, 인코더/디코더는 테스트 절차와 같은 방법으로 리프 노드까지 도달할 수 있고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스를 결정할 수 있다.

[0260] 만약, 복수 개의 랜덤 트리로 랜덤 포레스트가 구성되는 경우, 인코더/디코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 랜덤 포레스트의 클래스로 결정하고, 현재 블록의 화면 내 예측 모드를 도출할 수 있다.

[0261] 아래의 도면을 참조하여, 2개의 이웃하는 참조 샘플 라인을 사용하는 분할 함수의 다른 일 예를 설명한다.

[0262] 도 16 및 도 17은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

[0263] 예를 들어, 분할 함수 h 는 수학식 8과 같이 정의될 수 있다.

수학식 8

[0264]
$$h(v, \theta) = D < t$$

[0265] 여기서, 세타는 $\{\theta, t\}$ 로 구성될 수 있다. D 는 아래의 수학식 9에 의해 계산될 수 있다.

수학식 9

[0266]
$$D = |D_0| + |D_1| + |D_2| + |D_3| + \dots + |D_{6N+1}|$$

[0267] 여기서, D_0 내지 $D_{(6N+1)}$ 은 각도 θ 방향으로 이웃하는 각 참조 샘플간의 샘플값의 차이값으로 정의된다.

[0268] 도 16을 참조하면, 각도 θ 가 0인 경우를 가정한다. 이때, D_0 내지 $D_{(6N+1)}$ 은 수평 방향으로 인접한 참조 샘플간의 샘플값의 차분으로 계산될 수 있다. 예를 들어, D_0 는 참조 샘플 $R^0_{(-1,-1)}(1601)$ 과 $R^0_{(0,-1)}(1602)$ 의 샘플값의 차이값으로 계산될 수 있다.

[0269] 도 17을 참조하면, 각도 θ 가 $\pi/2$ 인 경우를 가정한다. 이때, D_0 내지 $D_{(6N+1)}$ 은 수직 방향으로 인접한 참조 샘플간의 샘플값의 차분으로 계산될 수 있다. 예를 들어, D_0 는 참조 샘플 $R^0_{(-1,-1)}(1701)$ 과 $R^0_{(-1,0)}(1702)$ 의 샘플값의 차이값으로 계산될 수 있다.

실시예 2

[0271] 본 실시예에서는 랜덤 포레스트를 인코더에서 화면 내 예측 모드를 효율적으로 부호화하는 방법을 제안한다.

[0272] 구체적으로, 랜덤 포레스트를 구성하는 랜덤 트리 내 분할 함수의 입력으로서, 현재 블록에 이웃하는 참조 샘플 뿐만 아니라 현재 블록의 예측 블록 내 샘플 값을 사용하여 화면 내 예측 모드를 결정(또는 예측)하는 방법을 제안한다.

[0273] 여기서, 예측 블록은 HEVC를 예로 들면 PU에 해당할 수 있으나, 본 발명이 이에 한정되는 것은 아니다. 즉, 본 실시예에서의 예측 블록은 동일한 예측 모드로 예측된 샘플들의 배열 또는 블록을 의미할 수 있으며, 코딩 블록, 코딩 유닛, 변환 블록, 변환 유닛 등으로 지칭될 수도 있다.

[0274] 본 실시예에서 제안하는 방법은 인코더에서 화면 내 예측 모드의 빠른 모드 결정(fast-mode decision)을 위해 사용할 수 있다. 예측 블록의 샘플을 이용하기 때문에 본 실시예에서 제안하는 방법은 인코더에서만 적용될 수 있다. 따라서, 본 실시예는 인코더를 기준으로 설명한다.

[0275] 예를 들어, HEVC에서 통상적으로 화면 내 예측 모드의 빠른 모드 결정(fast-mode decision)은 총 35개 화면 내 예측 모드 중 일부를 포함하는 모드 셋(mode set)을 먼저 선택하고(RMD: Rough Mode Decision), 선택된 모드들 중에서 비트율-왜곡 최적화(Rate-distortion optimization)를 통해 최적의 모드를 선택(FMD: Fine Mode Decision)하는 방법을 수행한다.

[0276] 본 실시예에서는 랜덤 포레스트를 이용하여 다음과 같은 방법을 사용할 수 있다.

[0277] - 인코더는 랜덤 포레스트 방법을 수행하여 추정된 하나 이상의 화면 내 예측 모드를 RMD 과정에서의 모드 셋에 추가하거나, RMD 과정에서의 모드 셋에서 선택된 기존의 화면 내 예측 모드를 대체할 수 있다. 이때, 인코더는 랜덤 포레스트 방법을 통해 선택되는 모드 중에서 각 모드의 확률값(즉, 클래스 별로 저장된 확률값)을 이용하여 내림 차순 순서(즉, 확률이 높은 순서)로 화면 내 예측 모드를 선택할 수 있다.

- [0278] - 인코더는 기존의 RMD 과정 대신에, 랜덤 포레스트 방법을 통해 추정된 화면 내 예측 모드들 중 확률값에 따라 내림 차순으로 하나 이상의 화면 내 예측 모드를 선택하여 모드 셋을 구성할 수 있다. 그리고, 인코더는 선택된 모드 셋에서 FMD 방법을 적용하여 최적의 화면 내 예측 모드를 선택할 수 있다.
- [0279] 빠른 모드 결정(Fast-mode decision)에 사용하는 랜덤 트리 내 분할 함수는 다양한 여러 방법에 의해 설계될 수 있다.
- [0280] 이하에서, 분할 함수를 설계하는 방법을 도 18 내지 도 22의 예시를 참조하여 설명한다. 다만, 본 발명이 이하에서 설명하는 예시에만 한정되는 것이 아님은 물론이다.
- [0281] 도 18을 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.
- [0282] 도 18을 참조하면, 분할 함수 h 는 수학식 10과 같이 정의될 수 있다.

수학식 10

- [0283]
$$h(v, \theta) = |R_{x0,y0} - P_{x1,y1}| < t$$
- [0284] $N \times N$ 크기의 현재 블록(HEVC를 예로 들면, PU)에 이웃하는 참조 샘플은 현재 블록의 좌측(left) 및 좌하측(bottom-left)에 인접한 총 $2 \times N$ 개의 샘플들, 현재 블록의 상측(top) 및 우상측(top-right)에 인접한 총 $2 \times N$ 개의 샘플들 및 현재 블록의 좌상측(top-left)에 인접한 1개의 샘플로 구성될 수 있다.
- [0285] 수학식 10을 참조하면, $R_{x0,y0}$ 는 현재 블록에 이웃하는 샘플 중 $(x0,y0)$ 좌표에 위치한 참조 샘플을 나타내고, $P_{x1,y1}$ 는 현재 블록의 예측 블록 내 $(x1,y1)$ 좌표에 위치한 예측 샘플을 나타낸다. 그리고, 세타는 $\{L^R, L^P, t\}$ 로 구성될 수 있다. L^R 은 $R_{x0,y0}$ 를 나타내며, L^P 는 $P_{x1,y1}$ 를 나타낸다.
- [0286] 다시 말해, 수학식 10의 분할 함수는 현재 블록에 이웃하는 샘플 중 1개의 참조 샘플, 현재 블록의 예측 블록 내 1개의 예측 샘플 및 임계값 t 의 파라미터로 구성될 수 있다.
- [0287] 도 18에서는 1개의 참조 샘플로서 $R_{(0,1)}$ (1801)과 1개의 예측 샘플로서 $P_{(2,1)}$ (1802)이 선택되는 경우를 예시한다.
- [0288] 학습 단계에서, 현재 블록에 이웃하는 참조 샘플 중 1개의 참조 샘플($R_{(x0,y0)}$)과 예측 블록 중 1 개의 예측 샘플($P_{(x1,y1)}$)을 N 번 임의의 선택하여 정보 이득을 최대화하는 분할 파라미터가 결정될 수 있다.
- [0289] 그리고, 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 랜덤 트리의 루트 노드에서부터 리프 노드까지 분할하여, 리프 노드에 데이터의 분류 결과를 축적할 수 있다.
- [0290] 그리고, 인코더는 테스트 절차와 같은 방법으로 리프 노드까지 도달할 수 있고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스를 결정할 수 있다.
- [0291] 만약, 복수 개의 랜덤 트리로 랜덤 포레스트가 구성되는 경우, 인코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 현재 블록의 화면 내 예측 모드로 결정할 수 있다.
- [0292] 아래의 도면을 참조하여, 빠른 모드 결정(Fast-mode decision)을 위한 랜덤 트리 내 분할 함수의 다른 일 예를 설명한다.
- [0293] 도 19는 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.
- [0294] 도 19를 참조하면, 분할 함수 h 는 수학식 11와 같이 정의될 수 있다.

수학식 11

- [0295]
$$h(v, \theta) = \begin{cases} |R_{x0,y0} - P_{x1,y1}| < t_1 \\ |R_{x2,y2} - P_{x1,y1}| < t_2 \end{cases}$$
- [0296] 수학식 11를 참조하면, $R_{(x0,y0)}$ 와 $R_{(x2,y2)}$ 는 현재 블록에 이웃하는 참조 샘플 중 $(x0,y0)$, $(x2,y2)$ 위치의

참조 샘플을 지칭하고, $P_{(x1,y1)}$ 는 예측 블록 내 $(x1,y1)$ 위치의 예측 샘플을 지칭한다. 그리고, 세타는 $\{L_1^R, L_2^R, L^P, t_1, t_2\}$ 로 구성될 수 있다. L_1^R 는 $R_{(x0,y0)}$ 를 나타내고, L_2^R 는 $R_{(x2,y2)}$ 를 나타내고, L^P 는 $R_{(x1,y1)}$ 를 나타낸다. 그리고, 앞서 도 18에서와 동일하게 현재 블록에 이웃하는 참조 샘플이 구성될 수 있다.

[0297] 도 19에서는 현재 블록에 이웃하는 참조 샘플 중 내 2개의 참조 샘플로서 $R_{(2,0)}$ (1901) 및 $R_{(N+1,0)}$ (1902)과 예측 블록 내 1 개의 예측 샘플로서 $P_{(N,1)}$ 이 선택되는 경우를 예시한다.

[0298] $R_{(x0,y0)}$ 과 $P_{(x1,y1)}$ 의 차분값이 제 1 임계값(t_1)보다 크거나 또는 $R_{(x2,y2)}$ 과 $P_{(x1,y1)}$ 의 차분값이 제 2 임계값(t_2)보다 큰 경우, 0의 값이 출력되어 우측 자식 노드로 이동될 수 있다. 반면에 두 차분값이 모두 임계값보다 작은 경우, 1의 값이 출력되어 좌측 자식 노드로 이동될 수 있다.

[0299] 학습 단계에서, 2개의 참조 샘플 샘플($R_{(x0,y0)}$, $R_{(x2,y2)}$)과 1 개의 예측 샘플($P_{(x1,y1)}$)을 N번 임의 선택하여 정보 이득을 최대화하는 분할 파라미터가 결정될 수 있다.

[0300] 그리고, 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 랜덤 트리의 루트 노드에서부터 리프 노드까지 분할하여, 리프 노드에 데이터의 분류 결과를 축적할 수 있다.

[0301] 그리고, 앞서 설명한 방법과 같이, 인코더는 테스트 절차와 같은 방법으로 리프 노드까지 도달할 수 있고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스를 결정할 수 있다.

[0302] 만약, 복수 개의 랜덤 트리도 랜덤 포레스트가 구성되는 경우, 인코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 현재 블록의 화면 내 예측 모드로 결정할 수 있다.

[0303] 아래의 도면을 참조하여, 빠른 모드 결정(Fast-mode decision)을 위한 랜덤 트리 내 분할 함수의 다른 일 예를 설명한다.

[0304] 도 20은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

[0305] 도 20을 참조하면, 분할 함수 h 는 수학식 12와 같이 정의될 수 있다.

수학식 12

$$h(v, theta) = \begin{cases} |R_{x0,y0} - P_{x1,y1}| < t_1 \\ |R_{x2,y2} - P_{x3,y3}| < t_2 \end{cases}$$

[0307] 여기서, 앞서 도 18에서와 동일하게 현재 블록에 이웃하는 참조 샘플이 구성될 수 있다.

[0308] 수학식 11의 분할 함수와 차이점을 위주로 설명하면, 참조 샘플 뿐만 아니라 예측 블록에서도 2개의 예측 샘플($P_{(x1,y1)}$, $P_{(x3,y3)}$)이 선택될 수 있다. 그리고, 세타는 $\{L_1^R, L_2^R, L_1^P, L_2^P, t_1, t_2\}$ 로 구성될 수 있다. L_1^R 는 $R_{(x0,y0)}$ 를, L_2^R 는 $R_{(x2,y2)}$ 를, L_1^P 는 $P_{(x1,y1)}$ 를, L_2^P 는 $P_{(x3,y3)}$ 를 나타낸다.

[0309] 도 20에서는 2개의 참조 샘플로서 $R_{(1,0)}$ (2001) 및 $R_{(0,1)}$ (2002)과 예측 블록 내 2 개의 샘플로서 $P_{(N,1)}$ (2003) 및 $P_{(N,N)}$ (2004)이 선택되는 경우를 예시한다.

[0310] 예측 블록에서도 2개의 예측 샘플이 분할 파라미터로 결정되는 점을 제외하고, 앞서 도 19에서 설명한 방법과 동일한 방법으로 학습 단계와 테스트 단계가 수행될 수 있고, 부호화 단계에서 클래스가 결정될 수 있다.

[0311] 아래의 도면을 참조하여, 빠른 모드 결정(Fast-mode decision)을 위한 랜덤 트리 내 분할 함수의 다른 일 예를 설명한다.

[0312] 도 21은 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

[0313] 도 21을 참조하면, 분할 함수 h 는 수학식 13와 같이 정의될 수 있다.

수학식 13

$$h(v, \theta) = \{\theta < t\}$$

[0314]

[0315] 수학식 13을 참조하면, 세타는 $\{\theta, t\}$ 로 구성될 수 있고, $\theta = \arctan(\Delta y / \Delta x)$ 를 이용하여 계산될 수 있다.

[0316] 도 21을 참조하면, Δx 는 예측 블록 내 현재 예측 샘플(2101)의 샘플값과 x 축 방향으로 인접한 예측 샘플(2102)의 샘플값과의 차이값을 나타내고, Δy 는 예측 블록 내 현재 예측 샘플(2101)의 샘플값과 y 축 방향으로 하단에 인접한 예측 샘플(2103)의 샘플값과의 차이값을 나타낸다.

[0317] 학습 단계에서, 현재 예측 샘플에서 Δx 와 Δy 의 L2 놈(norm)(또는 변화율-크기($\text{gradient.magnitude}$))을 최대로 하는 경우의 θ 값이 결정될 수 있다. Δx 와 Δy 의 L2 놈은 앞서 설명한 수학식 7에 의해 계산될 수 있다.

[0318] 이와 같이, 인접한 픽셀과의 픽셀값의 변화량을 이용함으로써 예측 블록의 특성을 분할 파라미터에 반영할 수 있다.

[0319] 그리고, 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 랜덤 트리의 루트 노드에서부터 리프 노드까지 분할하여, 리프 노드에 데이터의 분류 결과를 축적할 수 있다.

[0320] 그리고, 앞서 설명한 방법과 같이, 인코더는 테스트 절차와 같은 방법으로 리프 노드까지 도달할 수 있고, 리프 노드에 저장된 클래스 중에서 최대 확률을 가지는 클래스를 결정할 수 있다.

[0321] 만약, 복수 개의 랜덤 트리로서 랜덤 포레스트가 구성되는 경우, 인코더는 각 랜덤 트리에서 결정된 클래스 중 가장 많이 결정된 클래스를 현재 블록의 화면 내 예측 모드로 결정할 수 있다.

[0322] 아래의 도면을 참조하여, 빠른 모드 결정(Fast-mode decision)을 위한 랜덤 트리 내 분할 함수의 다른 일 예를 설명한다.

[0323] 도 22는 본 발명이 적용되는 실시예로서, 분할 함수를 결정하는 방법을 예시하는 도면이다.

[0324] 도 22을 참조하면, 분할 함수 h 는 수학식 14와 같이 정의될 수 있다.

수학식 14

$$h(v, \theta) = D < t$$

[0325]

[0326] 수학식 14를 참조하면, 세타는 $\{\theta, t\}$ 로 구성될 수 있다. D 는 수학식 15에 의해 계산될 수 있다.

수학식 15

$$D = |D_0| + |D_1| + |D_2| + |D_3| + \dots + |D_{N(N-1)}|$$

[0327]

[0328] 여기서, D_0 내지 $D_{N(N-1)}$ 은 각도 θ 방향으로 이웃하는 각 예측 샘플간의 샘플값의 차이값으로 정의된다.

[0329] 도 22(a)를 참조하면, 각도 θ 가 0인 경우를 가정한다. 이때, D_0 내지 $D_{N(N-1)}$ 은 수평 방향으로 인접한 예측 샘플간의 샘플값의 차분으로 계산될 수 있다. 예를 들어, D_0 는 예측 샘플 $P_0(2201)$ 과 $P_1(2202)$ 의 예측 샘플값의 차이값으로 계산될 수 있다.

[0330] 도 22(b)를 참조하면, 각도 θ 가 $\pi/2$ 인 경우를 가정한다. 이때, D_0 내지 $D_{N(N-1)}$ 은 수직 방향으로 인접한 예측 샘플간의 샘플값의 차분으로 계산될 수 있다. 예를 들어, D_0 는 참조 샘플 $P_0(2203)$ 과 $P_4(2204)$ 의 예측 샘플값의 차이값으로 계산될 수 있다.

[0331] 실시예 3

[0332] 앞서 실시예 1에서 제안한 방법은 랜덤 포레스트를 통해 결정되는 각 클래스 마다 하나의 화면 내 예측 모드가 매핑될 수 있다. 이 경우, 화면 내 예측 모드를 시그널링하지 않고도 화면 내 예측 모드를 복호화할 수 있다는

장점이 있다.

- [0333] 반면, 본 실시예에서는 각 클래스마다 하나 이상의 화면 내 예측 모드를 매핑하여 예측 모드 결정의 정확도를 향상시키는 방법을 제안한다.
- [0334] 클래스에 매핑되는 화면 내 예측 모드를 1개의 특정 예측 모드가 아닌 복수 개의 예측 모드들의 집합으로 고려함으로써, 정확하게 화면 내 예측 모드를 추정할 수 있으며, 이를 통해 부호화 효율을 향상시킬 수 있다. 또한, 복수 개의 예측 모드가 매핑되는 클래스를 결정하고, 결정된 클래스 범위 내에서 현재 블록의 화면 내 예측 모드를 지시하기 위한 인덱스를 시그널링함으로써, 화면 내 예측 모드를 표현하기 위해 사용되는 비트 수를 줄일 수 있다.
- [0335] 도 23은 본 발명이 적용될 수 있는 실시예로서, 클래스에 복수 개의 화면 내 예측 모드를 매핑하는 방법을 예시하는 도면이다.
- [0336] 도 23을 참조하면, 현재 블록의 화면 내 예측 모드는 기존의 HEVC에서와 같은 경우를 가정한다(앞서 도 6 참조). 예를 들어, 3개, 4개 또는 5개의 인접한 화면 내 예측 모드를 분류하고, 이를 각 클래스에 매핑할 수 있다.
- [0337] 이 경우, 인코더/디코더는 랜덤 포레스트를 이용하여 클래스를 결정하고, 인코더는 결정된 클래스 내에서 현재 블록의 화면 내 예측 모드를 지시하는 인덱스(index) 정보를 디코더에게 시그널링할 수 있다. 그리고, 디코더는 결정된 클래스 내에서 인코더로부터 수신한 인덱스 정보를 이용하여 현재 블록의 화면 내 예측 모드를 결정할 수 있다.
- [0338] 복수 개의 화면 내 예측 모드가 대응되는(또는 매핑되는) 클래스를 구성하는 방법은 다양한 여러 방법에 의해 수행될 수 있다. 다음은 클래스를 구성하기 위해 사용될 수 있는 방법들을 예시한다. 각각의 방법들이 독립적으로 사용될 수도 있고, 복수 개의 방법이 조합되어 사용될 수도 있다.
- [0339] 클래스를 구성하는 방식은 다음과 같은 방식 중 하나 혹은 그 조합으로 구성할 수 있다. 아래 예에서 구분한 클래스는 인덱스 i 로 표현한다.
- [0340] - 앞서 도 23의 예시에서와 같이, 유사한 방향을 가지는 예측 모드의 집합(또는 인접한 예측 모드의 집합)을 하나의 클래스로 결정하는 방법
- [0341] 예를 들어, 대표 모드(m_i)를 선택하여 m_{i-n} 내지 m_{i+n} 예측 모드를 하나의 클래스로 구성할 수 있다. 여기서, m 은 예측 모드의 번호(HEVC의 경우를 예로 들면, 0번 내지 34번 모드 중 어느 하나)를 나타내며, i 는 분류되는 클래스의 인덱스를 나타낸다.
- [0342] - 통계적으로 자주 선택되는 모드 집합과 그렇지 않은 모드 집합을 각각 하나의 클래스로 구성하는 방법
- [0343] - MPM 모드와 non-MPM 모드를 나누어 클래스를 구성하는 방법
- [0344] - 방향성 예측 모드(Angular prediction mode)와 비방향성 예측 모드(non-angular prediction mode)를 나누어 클래스를 구성하는 방법
- [0345] - 현재 블록의 예측 모드로 결정될 확률이 높은 예측 모드의 경우 복수 개의 클래스에 포함시키는 방법
- [0346] 예를 들어, 일반적으로 영상에서 자주 선택되는 예측 모드(플래너 모드(planar mode), 수직 방향 모드(vertical mode), 수평 방향 모드(horizontal mode), DC 모드 등) 중 적어도 어느 하나 이상의 예측 모드를 복수 개의 클래스에 포함되도록 클래스를 구성할 수 있다. 또는, MPM 모드 중 적어도 어느 하나 이상의 예측 모드를 복수 개의 클래스에 포함되도록 클래스를 구성할 수 있다.
- [0347] 각 클래스에 포함되는(또는 매핑되는) 예측 모드의 수는 동일할 수도 있고, 클래스 별로 서로 다를 수도 있다. 그리고, 동일한 랜덤 포레스트를 이용하여 클래스를 결정할 수도 있고, 클래스 별로 독립적인 서로 다른 랜덤 포레스트를 적용할 수도 있다.
- [0348] 랜덤 포레스트를 구성하는 랜덤 트리 내 분할 함수 및 분할 파라미터를 설계하는 방법, 랜덤 트리 또는 랜덤 포레스트 내에서 클래스를 결정하는 방법 등은 앞서 실시예 1에서 설명한 방법과 동일하게 적용될 수 있다.
- [0349] 실시예 4
- [0350] 본 실시예에서는 랜덤 포레스트에 기반하여 계층적으로 화면 내 예측 모드를 부호화/복호화하는 방법을 제안한

다.

- [0351] 구체적으로, 화면 내 예측 모드의 예측 모드를 계층적으로 분할하여, 하나 이상의 랜덤 포레스트를 사용하여 각 계층에서 클래스를 결정하여 화면 내 예측 모드를 복호화하는 방법을 제안한다.
- [0352] 본 실시예에서 계층적으로 사용되는 각 랜덤 포레스트는 동일한 형태의 랜덤 트리로 구성될 수도 있고, 서로 다른 형태의 랜덤 트리로 구성될 수도 있다.
- [0353] 이전 계층에서 랜덤 포레스트를 통해 결정되는 클래스에 따라 현재 계층에서 사용되는 랜덤 포레스트의 종류가 결정될 수 있다.
- [0354] 인코더/디코더는 제 1 결정 트리(또는 제 1 랜덤 포레스트)에서 리프 노드를 탐색하여 현재 블록에 적용될 수 있는 복수 개의 인트라 예측 모드로 구성된 인트라 예측 모드 그룹을 결정할 수 있다. 그리고, 인코더/디코더는 상기 인트라 예측 모드 그룹에 따라 정해지는 제 2 결정 트리(또는 제 2 랜덤 포레스트)에서 리프 노드를 탐색하여 상기 인트라 예측 모드 그룹 내에서 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택할 수 있다. 구체적인 예시를 아래의 도면을 참조하여 설명한다.
- [0355] 도 24는 본 발명의 일 실시예에 따른 랜덤 포레스트에 기반한 계층 구조의 화면 내 예측 모드의 결정 방법을 예시하는 도면이다.
- [0356] 도 24를 참조하면, 먼저 예측 모드를 방향성 예측 모드(angular prediction mode)인지 비방향성 예측 모드(non-angular prediction mode)인지 랜덤 포레스트 1(RF 1)(2401)을 이용하여 분류할 수 있다. 즉, 랜덤 포레스트 1(2401)은 방향성 모드와 비방향성 모드로 구분되는 클래스를 결정하는 랜덤 트리의 집합에 해당된다.
- [0357] 만약, 랜덤 포레스트 1(2401)에 의해 현재 블록의 화면 내 예측 모드가 방향성 모드로 결정되는 경우, 인코더/디코더는 랜덤 포레스트 2(RF 2)(2402)를 이용하여 현재 블록의 화면 내 예측 모드를 최종적으로 복호화할 수 있다.
- [0358] 만약, 랜덤 포레스트 1(2401)에 의해 현재 블록의 화면 내 예측 모드가 비방향성 모드로 결정되는 경우, 인코더/디코더는 랜덤 포레스트 3(RF 3)(2403)을 이용하여 현재 블록의 화면 내 예측 모드를 최종적으로 부호화/복호화할 수 있다.
- [0359] 이 경우, 앞서 설명한 실시예 1 또는 실시예 3 방법을 이용하여 각 랜덤 포레스트의 클래스를 분류할 수 있다.
- [0360] 도 24에서 예시하는 방법 이외에 다양한 여러 방법을 이용하여 화면 내 예측 모드의 복호화 단계를 계층적으로 분할하고, 각 계층에서 서로 다른 랜덤 포레스트를 이용하여 화면 내 예측 모드를 부호화/복호화할 수 있다.
- [0361] 실시예 5
- [0362] 본 실시예에서는 랜덤 포레스트를 이용하여 MPM 모드를 구성하는 방법을 제안한다.
- [0363] 만약, MPM 모드 셋(mode set)이 N개의 부호화모드로 구성되는 경우를 가정하면, 랜덤 포레스트를 이용하여 다음과 같은 방법으로 MPM 모드 셋이 구성될 수 있다.
- [0364] - 랜덤 포레스트의 결과로 랜덤 트리의 각 리프 노드에서 화면 내 예측 모드 별 확률이 얻어질 수 있다. 이때, 가장 높은 확률값을 가지는 상위 N개의 화면 내 예측 모드가 MPM 모드 셋으로 결정할 수 있다.
- [0365] - 랜덤 포레스트에 의해 화면 내 예측 모드 별 확률이 얻어지는 경우, 가장 높은 확률값을 가지는 상위 (N-m)개의 화면 내 예측 모드와 미리 정의된 m개의 예측 모드(또는 미리 정의된 방법에 의해 결정되는 m개의 예측 모드)의 조합으로 N개의 MPM 모드 셋이 결정될 수 있다.
- [0366] 랜덤 포레스트를 통해 랜덤 트리의 각 리프 노드에서 화면 내 예측 모드 별 확률값을 구하는 방법을 아래의 도면을 참조하여 설명한다.
- [0367] 도 25는 본 발명이 적용될 수 있는 실시예로서, 화면 내 예측 모드 별 확률값을 결정하는 방법을 예시하는 도면이다.
- [0368] 도 25를 참조하면, 각 랜덤 트리의 리프 노드 마다 화면 내 예측 모드가 대응되는 클래스의 확률 값이 저장될 수 있다. 여기서, 클래스 별 확률 값은 테스트 샘플 데이터를 이용하여 측정될 수 있다.
- [0369] 각 리프 노드에서 화면 내 예측 별로 확률 값을 구하여, 확률이 높은 화면 내 예측 모드를 선택하여 MPM 모드 셋을 구성하거나 기존의 MPM 모드 셋의 전부 또는 일부에 추가하여 MPM 모드 셋을 구성할 수 있다.

- [0370] 전술한 실시예 1 내지 실시예 5는 독립적으로 적용될 수도 있고, 실시예 1 내지 실시예 5 중의 일부 또는 전부가 조합되어 적용될 수도 있다.
- [0371] 도 26은 본 발명의 일 실시예에 따른 인트라 예측 방법을 예시하는 도면이다.
- [0372] 인코더/디코더는 이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색한다(S2601).
- [0373] 전술한 바와 같이, 인코더/디코더는 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 결정 트리의 루트 노드에서부터 리프 노드까지 자식 노드를 선택하여 리프 노드를 탐색할 수 있다.
- [0374] 또한, 전술한 바와 같이 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력할 수 있다.
- [0375] 또한, 전술한 바와 같이, 분할 파라미터는 결정 트리(또는 랜덤 트리)의 각 노드에서 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)로 분할하였을 때 계산되는 불확실성의 감소, 즉 정보 이득(information gain)을 최대화하는 방향으로 학습될 수 있다.
- [0376] 앞서 도 10 및 수학적식 2에서 설명한 바와 같이, 분할 파라미터는 현재 블록에 이웃하는 참조 샘플 중에서 2개의 참조 샘플을 이용하여 결정될 수 있다.
- [0377] 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조할 수 있다. 이때, 앞서 도 12 내지 14 및 수학적식 3 내지 5에서 설명한 바와 같이, 분할 파라미터는 다중 참조 샘플 라인 중 현재 블록에 인접한 2개의 참조 샘플 라인의 복수 개의 참조 샘플을 이용하여 결정될 수 있다.
- [0378] 또한, 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우, 분할 파라미터는 다중 참조 샘플 라인 중에서 복수 개의 참조 샘플을 이용하여 결정될 수 있다.
- [0379] 또한, 앞서 도 16 및 수학적식 6에서 설명한 바와 같이, 분할 파라미터는 상기 다중 참조 샘플 라인의 특정 참조 샘플의 샘플값과 수평 방향으로 특정 참조 샘플의 우측에 인접한 참조 샘플의 샘플값과의 차이값과, 특정 참조 샘플의 샘플값과 수직 방향으로 특정 참조 샘플의 하단에 인접한 참조 샘플의 샘플값과의 차이값을 이용하여 결정될 수 있다.
- [0380] 또한, 앞서 도 16, 도 17 및 수학적식 8에서 설명한 바와 같이, 분할 파라미터는 다중 참조 샘플 라인의 각 참조 샘플의 샘플값과 특정 각도 방향으로 인접한 참조 샘플의 샘플값과의 차이값의 합을 이용하여 결정될 수 있다.
- [0381] 인코더/디코더는 리프 노드와 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택한다(S2602).
- [0382] 전술한 바와 같이, 결정 트리의 각 리프 노드에는 클래스의 확률 정보가 저장되어 있을 수 있으며, 결정 트리에서 탐색을 통해 출력되는 클래스는 리프 노드에 저장된 클래스 중 가장 확률이 높은 클래스로 결정될 수 있다.
- [0383] 또한, 랜덤 포레스트는 하나 이상의 결정 트리(또는 랜덤 트리)로 구성될 수 있다. 복수 개의 결정 트리로 구성된 랜덤 포레스트(random forest)의 각 결정 트리를 탐색하여 복수 개의 클래스가 선택되는 경우, 인코더/디코더는 복수 개의 클래스 중에서 가장 많이 선택된 클래스를 랜덤 포레스트의 클래스로 결정할 수 있다.
- [0384] 전술한 바와 같이, 인코더/디코더는 랜덤 포레스트를 구성하는 랜덤 트리의 루트 노드로부터 리프 노드까지 분할(또는 탐색)하여, 도달된 리프 노드의 클래스 정보(또는 화면 내 예측 모드)를 저장하고, 이를 최종 클래스 결정(또는 현재 블록의 화면 내 예측 모드 결정)의 투표에 사용할 수 있다.
- [0385] 랜덤 포레스트가 복수 개의 랜덤 트리로 구성된 경우, 각각의 랜덤 트리에서 결정된 클래스(또는 화면 내 예측 모드)가 투표(voting)에 사용되며, 랜덤 포레스트 내에서 가장 많은 투표를 얻은 클래스(또는 화면 내 예측 모드)가 최적의 클래스(또는 현재 블록의 화면 내 예측 모드)로 결정될 수 있다.
- [0386] 만약, 랜덤 포레스트가 단일의 랜덤 트리로 구성된 경우, 랜덤 트리에서 결정된 클래스(또는 화면 내 예측 모드)가 최적의 클래스(또는 현재 블록의 화면 내 예측 모드)로 결정될 수 있다.
- [0387] 또한, 각 클래스 마다 하나 이상의 화면 내 예측 모드를 매핑하여 예측 모드 결정의 정확도를 향상시킬 수

있다. 다시 말해, 인코더/디코더는 클래스(class)에 복수 개의 인트라 예측 모드가 매핑되는 경우, 클래스 내에서 현재 블록의 인트라 예측 모드를 지시하는 인덱스(index) 정보를 복호화할 수 있다.

- [0388] 즉, 복수 개의 예측 모드가 매핑되는 클래스를 결정하고, 결정된 클래스 범위 내에서 현재 블록의 화면 내 예측 모드를 지시하기 위한 인덱스를 시그널링함으로써, 화면 내 예측 모드를 표현하기 위해 사용되는 비트 수를 줄일 수 있다.
- [0389] 인코더/디코더는 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출한다(S2603).
- [0390] 전술한 바와 같이, 인코더/디코더는 클래스 마다 하나의 화면 내 예측 모드가 매핑된 경우, 결정된 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0391] 반면, 인코더/디코더는 클래스(class)에 복수 개의 인트라 예측 모드가 매핑되는 경우, 클래스 내에서 현재 블록의 인트라 예측 모드를 지시하는 인덱스(index) 정보를 복호화하고, 인덱스 정보를 이용하여 선택된 클래스 내에서 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0392] 복수 개의 결정 트리로 구성된 랜덤 포레스트(random forest)의 각 결정 트리를 탐색하여 복수 개의 클래스가 선택되는 경우, 인코더/디코더는 복수 개의 클래스 중에서 가장 많이 선택된 클래스를 랜덤 포레스트의 클래스로 결정하고, 이를 이용하여 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0393] 그리고, 인코더/디코더는 현재 블록의 인트라 예측 모드가 하나 이상의 계층으로 분할된 경우, 각 계층 별로 결정되는 랜덤 포레스트 내 결정 트리의 리프 노드에서 클래스를 선택함으로써, 계층적으로 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0394] 다시 말해, 인코더/디코더는 제 1 결정 트리에서 리프 노드를 탐색하여 현재 블록에 적용될 수 있는 복수 개의 인트라 예측 모드로 구성된 인트라 예측 모드 그룹을 결정하고, 상기 인트라 예측 모드 그룹에 따라 정해지는 제 2 결정 트리에서 리프 노드를 탐색하여 상기 인트라 예측 모드 그룹 내에서 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택할 수 있다.
- [0395] 이때, 계층적으로 사용되는 각 랜덤 포레스트 또는 결정 트리는 동일한 형태의 결정 트리로 구성될 수도 있고, 서로 다른 형태의 결정 트리로 구성될 수도 있다.
- [0396] 이전 계층에서 랜덤 포레스트를 통해 결정되는 클래스에 따라 현재 계층에서 사용되는 랜덤 포레스트 또는 결정 트리의 종류가 결정될 수 있다.
- [0397] 인코더/디코더는 인트라 예측 모드에 기반하여 현재 블록의 예측 샘플을 생성한다(S2604).
- [0398] 앞서 도 5 및 도 6에서 설명한 바와 같이, 인코더/디코더는 현재 블록의 인트라 예측 모드를 도출하고, 현재 블록에 이웃하는 샘플들(neighboring samples)을 이용하여 예측에 사용할 참조 샘플들을 구성할 수 있다. 그리고, 현재 블록에 이웃하는 샘플들 중 일부가 아직 디코딩되지 않았거나, 이용 가능하지 않을 경우, 인코더/디코더는 이용 가능한 샘플들로 이용 가능하지 않은 샘플들을 대체(substitution)하여 예측에 사용할 참조 샘플들을 구성할 수 있다. 그리고, 인코더/디코더는 인트라 예측 모드에 기반하여 참조 샘플의 필터링을 수행할 수 있다. 그리고, 인코더/디코더는 인트라 예측 모드와 참조 샘플들에 기반하여 현재 블록에 대한 예측 샘플을 생성할 수 있다.
- [0399] 도 27은 본 발명의 일 실시예에 따른 인트라 예측부를 예시하는 도면이다.
- [0400] 도 27에서는 설명의 편의를 위해 인트라 예측부를 하나의 블록으로 도시하였으나, 인트라 예측부는 인코더 및/또는 디코더에 포함되는 구성으로 구현될 수 있다.
- [0401] 도 27를 참조하면, 인트라 예측부는 앞서 도 5 내지 도 26에서 제안된 기능, 과정 및/또는 방법을 구현한다. 구체적으로, 인트라 예측부는 리프 노드 탐색부(2701), 클래스 선택부(2702), 예측 모드 도출부(2703) 및 예측 샘플 생성부(2704)를 포함하여 구성될 수 있다.
- [0402] 리프 노드 탐색부(2701)는 이진 트리(binary tree) 구조를 가지는 결정 트리(decision tree)에서 각 노드에 저장된 분할 함수(split function)를 이용하여 자식 노드(child node)를 선택하는 과정을 루트 노드(root node)로부터 순차적으로 수행함으로써, 리프 노드(leaf node)를 탐색한다.
- [0403] 전술한 바와 같이, 리프 노드 탐색부(2701)는 앞서 도 11에서 설명한 방법과 같은 방법으로, 학습 절차를 통해 분할 파라미터가 결정된 결정 트리의 루트 노드에서부터 리프 노드까지 자식 노드를 선택하여 리프 노드를 탐색

할 수 있다.

- [0404] 또한, 전술한 바와 같이 분할 함수는 분할 파라미터(split parameter)로 구성되고, 상기 현재 블록에 이웃하는 참조 샘플을 입력으로 하여, 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)에 대응되는 값을 출력할 수 있다.
- [0405] 또한, 전술한 바와 같이, 분할 파라미터는 결정 트리(또는 랜덤 트리)의 각 노드에서 좌측 자식 노드(left child node) 또는 우측 자식 노드(right child node)로 분할하였을 때 계산되는 불확실성의 감소, 즉 정보 이득(information gain)을 최대화하는 방향으로 학습될 수 있다.
- [0406] 앞서 도 10 및 수학적식 2에서 설명한 바와 같이, 분할 파라미터는 현재 블록에 이웃하는 참조 샘플 중에서 2개의 참조 샘플을 이용하여 결정될 수 있다.
- [0407] 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조할 수 있다. 이때, 앞서 도 12 내지 14 및 수학적식 3 내지 5에서 설명한 바와 같이, 분할 파라미터는 다중 참조 샘플 라인 중 현재 블록에 인접한 2개의 참조 샘플 라인의 복수 개의 참조 샘플을 이용하여 결정될 수 있다.
- [0408] 또한, 현재 블록의 예측 샘플을 생성하기 위해 다중 참조 샘플 라인을 참조하는 경우, 분할 파라미터는 다중 참조 샘플 라인 중에서 복수 개의 참조 샘플을 이용하여 결정될 수 있다.
- [0409] 또한, 앞서 도 16 및 수학적식 6에서 설명한 바와 같이, 분할 파라미터는 상기 다중 참조 샘플 라인의 특정 참조 샘플의 샘플값과 수평 방향으로 특정 참조 샘플의 우측에 인접한 참조 샘플의 샘플값과의 차이값과, 특정 참조 샘플의 샘플값과 수직 방향으로 특정 참조 샘플의 하단에 인접한 참조 샘플의 샘플값과의 차이값을 이용하여 결정될 수 있다.
- [0410] 또한, 앞서 도 16, 도 17 및 수학적식 8에서 설명한 바와 같이, 분할 파라미터는 다중 참조 샘플 라인의 각 참조 샘플의 샘플값과 특정 각도 방향으로 인접한 참조 샘플의 샘플값과의 차이값의 합을 이용하여 결정될 수 있다.
- [0411] 클래스 선택부(2702)는 리프 노드에서 하나 이상의 인트라 예측 모드가 매핑되는 클래스(class)를 선택한다.
- [0412] 전술한 바와 같이, 결정 트리의 각 리프 노드에는 클래스의 확률 정보가 저장되어 있을 수 있으며, 결정 트리에서 탐색을 통해 출력되는 클래스는 리프 노드에 저장된 클래스 중 가장 확률이 높은 클래스로 결정될 수 있다.
- [0413] 또한, 랜덤 포레스트는 하나 이상의 결정 트리(또는 랜덤 트리)로 구성될 수 있다. 복수 개의 결정 트리로 구성된 랜덤 포레스트(random forest)의 각 결정 트리를 탐색하여 복수 개의 클래스가 선택되는 경우, 클래스 선택부(2702)는 복수 개의 클래스 중에서 가장 많이 선택된 클래스를 랜덤 포레스트의 클래스로 결정할 수 있다.
- [0414] 전술한 바와 같이, 클래스 선택부(2702)는 랜덤 포레스트를 구성하는 랜덤 트리의 루트 노드로부터 리프 노드까지 분할(또는 탐색)하여, 도달된 리프 노드의 클래스 정보(또는 화면 내 예측 모드)를 저장하고, 이를 최종 클래스 결정(또는 현재 블록의 화면 내 예측 모드 결정)의 투표에 사용할 수 있다.
- [0415] 랜덤 포레스트가 복수 개의 랜덤 트리로 구성된 경우, 각각의 랜덤 트리에서 결정된 클래스(또는 화면 내 예측 모드)가 투표(voting)에 사용되며, 랜덤 포레스트 내에서 가장 많은 투표를 얻은 클래스(또는 화면 내 예측 모드)가 최적의 클래스(또는 현재 블록의 화면 내 예측 모드)로 결정될 수 있다.
- [0416] 만약, 랜덤 포레스트가 단일의 랜덤 트리로 구성된 경우, 랜덤 트리에서 결정된 클래스(또는 화면 내 예측 모드)가 최적의 클래스(또는 현재 블록의 화면 내 예측 모드)로 결정될 수 있다.
- [0417] 또한, 각 클래스마다 하나 이상의 화면 내 예측 모드를 매핑하여 예측 모드 결정의 정확도를 향상시킬 수 있다. 다시 말해, 클래스 선택부(2702)는 클래스(class)에 복수 개의 인트라 예측 모드가 매핑되는 경우, 클래스 내에서 현재 블록의 인트라 예측 모드를 지시하는 인덱스(index) 정보를 복호화할 수 있다.
- [0418] 즉, 복수 개의 예측 모드가 매핑되는 클래스를 결정하고, 결정된 클래스 범위 내에서 현재 블록의 화면 내 예측 모드를 지시하기 위한 인덱스를 시그널링함으로써, 화면 내 예측 모드를 표현하기 위해 사용되는 비트 수를 줄일 수 있다.
- [0419] 예측 모드 도출부(2703)는 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출한다.
- [0420] 전술한 바와 같이, 예측 모드 도출부(2703)는 클래스마다 하나의 화면 내 예측 모드가 매핑된 경우, 결정된 클래스를 이용하여 현재 블록의 인트라 예측 모드를 도출할 수 있다.

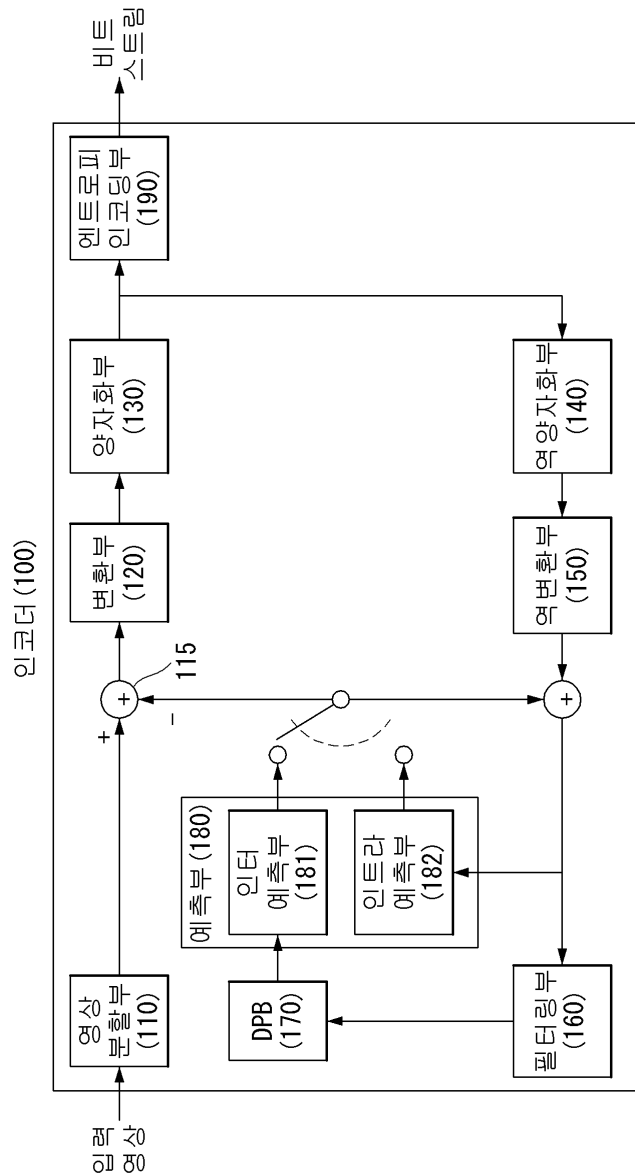
- [0421] 반면, 예측 모드 도출부(2703)는 클래스(class)에 복수 개의 인트라 예측 모드가 매핑되는 경우, 클래스 내에서 현재 블록의 인트라 예측 모드를 지시하는 인덱스(index) 정보를 복호화하고, 인덱스 정보를 이용하여 선택된 클래스 내에서 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0422] 복수 개의 결정 트리로 구성된 랜덤 포레스트(random forest)의 각 결정 트리를 탐색하여 복수 개의 클래스가 선택되는 경우, 예측 모드 도출부(2703)는 복수 개의 클래스 중에서 가장 많이 선택된 클래스를 랜덤 포레스트의 클래스로 결정하고, 이를 이용하여 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0423] 그리고, 예측 모드 도출부(2703)는 현재 블록의 인트라 예측 모드가 하나 이상의 계층으로 분할된 경우, 각 계층 별로 결정되는 랜덤 포레스트 내 결정 트리의 리프 노드에서 클래스를 선택함으로써, 계층적으로 현재 블록의 인트라 예측 모드를 도출할 수 있다.
- [0424] 다시 말해, 예측 모드 도출부(2703)는 제 1 결정 트리에서 리프 노드를 탐색하여 현재 블록에 적용될 수 있는 복수 개의 인트라 예측 모드로 구성된 인트라 예측 모드 그룹을 결정하고, 상기 인트라 예측 모드 그룹에 따라 정해지는 제 2 결정 트리에서 리프 노드를 탐색하여 상기 인트라 예측 모드 그룹 내에서 하나 이상의 인트라 예측 모드가 매핑되는 클래스를 선택할 수 있다.
- [0425] 이때, 계층적으로 사용되는 각 랜덤 포레스트 또는 결정 트리는 동일한 형태의 결정 트리로 구성될 수도 있고, 서로 다른 형태의 결정 트리로 구성될 수도 있다.
- [0426] 이전 계층에서 랜덤 포레스트를 통해 결정되는 클래스에 따라 현재 계층에서 사용되는 랜덤 포레스트 또는 결정 트리의 종류가 결정될 수 있다.
- [0427] 예측 샘플 생성부(2704)는 인트라 예측 모드에 기반하여 현재 블록의 예측 샘플을 생성한다.
- [0428] 앞서 도 5 및 도 6에서 설명한 바와 같이, 예측 샘플 생성부(2704)는 현재 블록의 인트라 예측 모드를 도출하고, 현재 블록에 이웃하는 샘플들(neighboring samples)을 이용하여 예측에 사용할 참조 샘플들을 구성할 수 있다. 그리고, 현재 블록에 이웃하는 샘플들 중 일부가 아직 디코딩되지 않았거나, 이용 가능하지 않을 경우, 인코더/디코더는 이용 가능한 샘플들로 이용 가능하지 않은 샘플들을 대체(substitution)하여 예측에 사용할 참조 샘플들을 구성할 수 있다. 그리고, 예측 샘플 생성부(2704)는 인트라 예측 모드에 기반하여 참조 샘플의 필터링을 수행할 수 있다. 그리고, 예측 샘플 생성부(2704)는 인트라 예측 모드와 참조 샘플들에 기반하여 현재 블록에 대한 예측 샘플을 생성할 수 있다.
- [0429] 이상에서 설명된 실시예들은 본 발명의 구성요소들과 특징들이 소정 형태로 결합된 것들이다. 각 구성요소 또는 특징은 별도의 명시적 언급이 없는 한 선택적인 것으로 고려되어야 한다. 각 구성요소 또는 특징은 다른 구성요소나 특징과 결합되지 않은 형태로 실시될 수 있다. 또한, 일부 구성요소들 및/또는 특징들을 결합하여 본 발명의 실시예를 구성하는 것도 가능하다. 본 발명의 실시예들에서 설명되는 동작들의 순서는 변경될 수 있다. 어느 실시예의 일부 구성이나 특징은 다른 실시예에 포함될 수 있고, 또는 다른 실시예의 대응하는 구성 또는 특징과 교체될 수 있다. 특허청구범위에서 명시적인 인용 관계가 있지 않은 청구항들을 결합하여 실시예를 구성하거나 출원 후의 보정에 의해 새로운 청구항으로 포함시킬 수 있음은 자명하다.
- [0430] 본 발명에 따른 실시예는 다양한 수단, 예를 들어, 하드웨어, 펌웨어(firmware), 소프트웨어 또는 그것들의 결합 등에 의해 구현될 수 있다. 하드웨어에 의한 구현의 경우, 본 발명의 일 실시예는 하나 또는 그 이상의 ASICs(application specific integrated circuits), DSPs(digital signal processors), DSPDs(digital signal processing devices), PLDs(programmable logic devices), FPGAs(field programmable gate arrays), 프로세서, 콘트롤러, 마이크로 콘트롤러, 마이크로 프로세서 등에 의해 구현될 수 있다.
- [0431] 펌웨어나 소프트웨어에 의한 구현의 경우, 본 발명의 일 실시예는 이상에서 설명된 기능 또는 동작들을 수행하는 모듈, 절차, 함수 등의 형태로 구현될 수 있다. 소프트웨어 코드는 메모리에 저장되어 프로세서에 의해 구동될 수 있다. 상기 메모리는 상기 프로세서 내부 또는 외부에 위치하여, 이미 공지된 다양한 수단에 의해 상기 프로세서와 데이터를 주고 받을 수 있다.
- [0432] 본 발명은 본 발명의 필수적 특징을 벗어나지 않는 범위에서 다른 특정한 형태로 구체화될 수 있음은 당업자에게 자명하다. 따라서, 상술한 상세한 설명은 모든 면에서 제한적으로 해석되어서는 아니 되고 예시적인 것으로 고려되어야 한다. 본 발명의 범위는 첨부된 청구항의 합리적 해석에 의해 결정되어야 하고, 본 발명의 등가적 범위 내에서의 모든 변경은 본 발명의 범위에 포함된다.

산업상 이용가능성

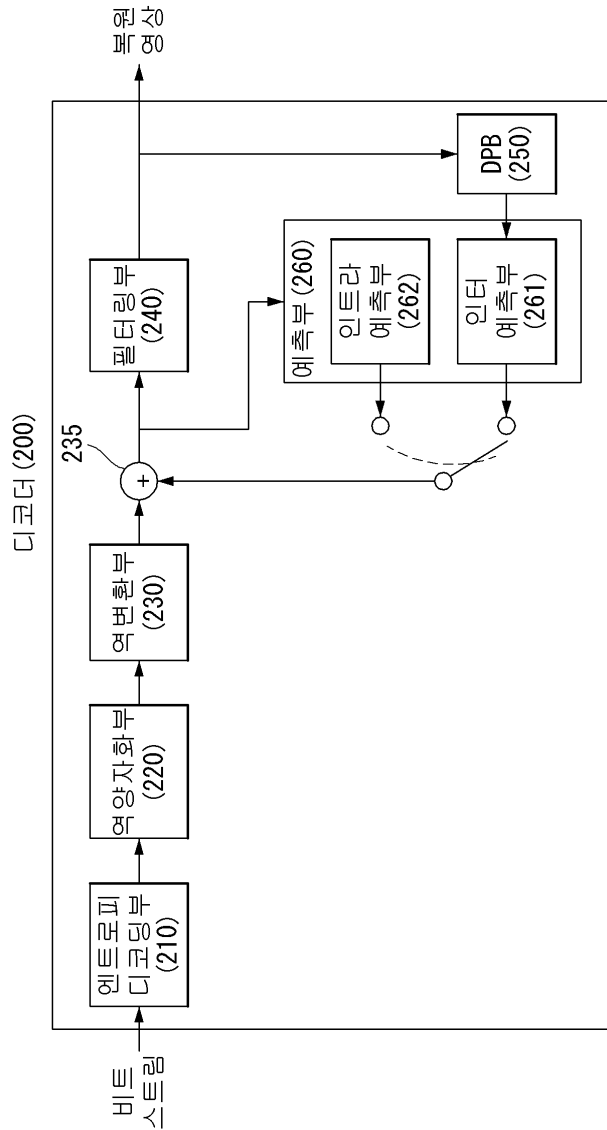
[0433] 이상, 전술한 본 발명의 바람직한 실시예는, 예시의 목적을 위해 개시된 것으로, 당업자라면 이하 첨부된 특허 청구범위에 개시된 본 발명의 기술적 사상과 그 기술적 범위 내에서, 다양한 다른 실시예들을 개량, 변경, 대체 또는 부가 등이 가능할 것이다.

도면

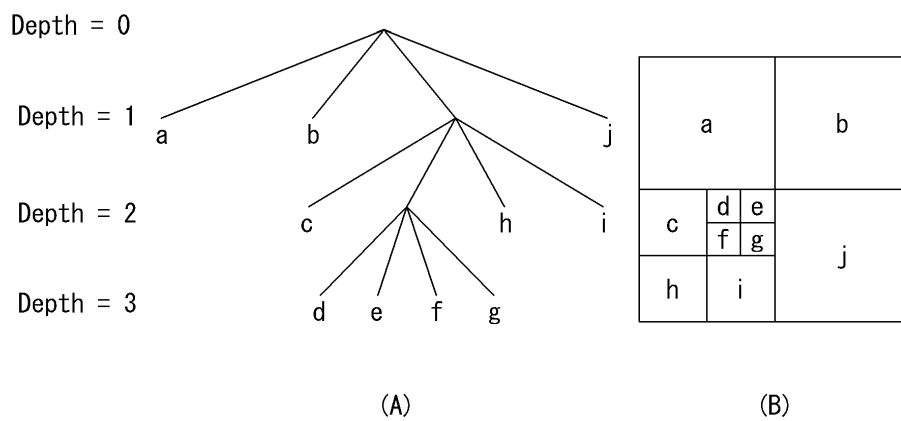
도면1



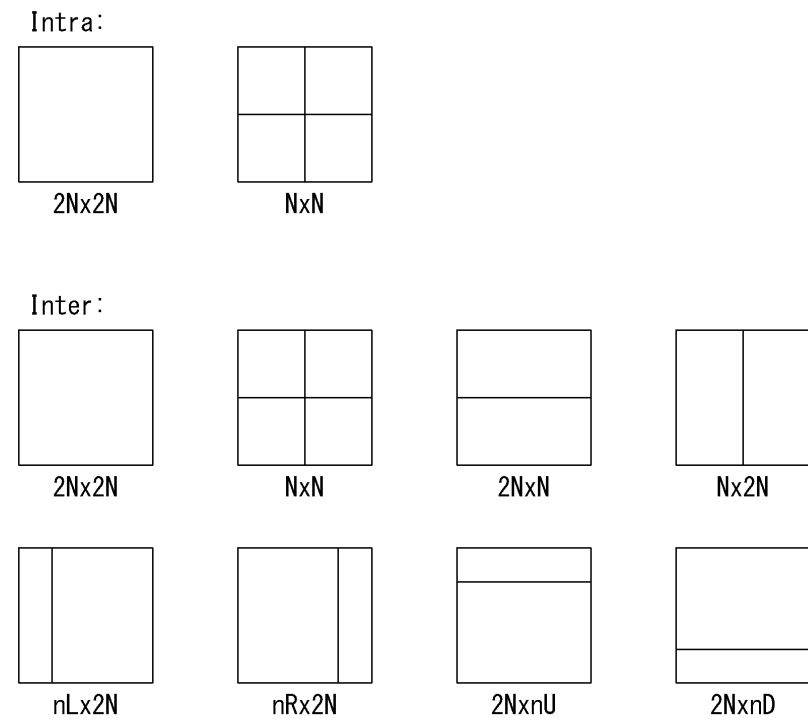
도면2



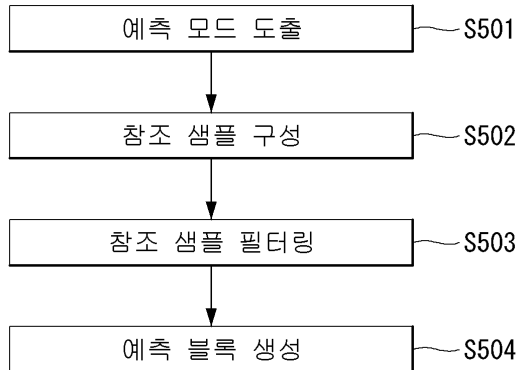
도면3



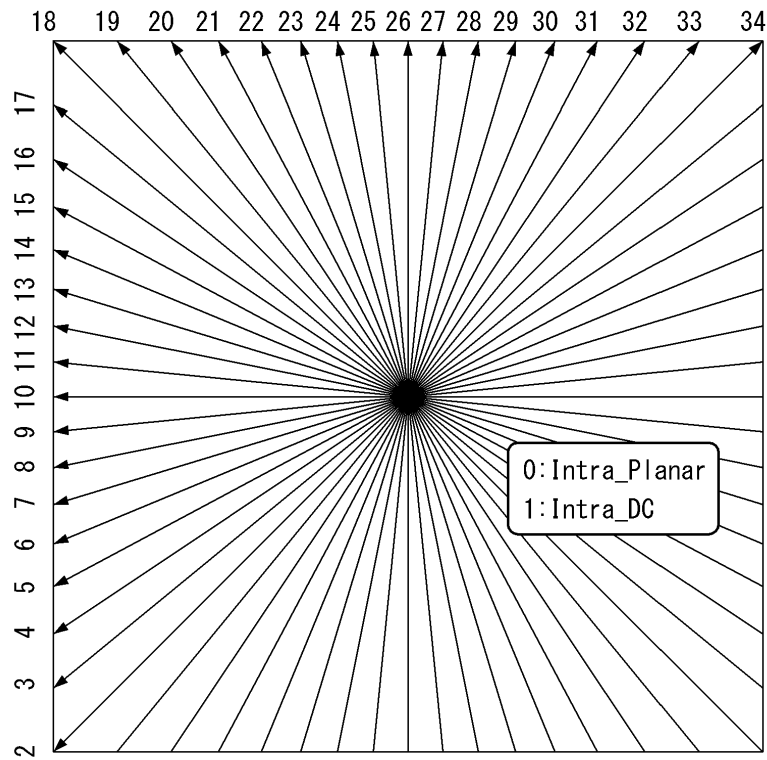
도면4



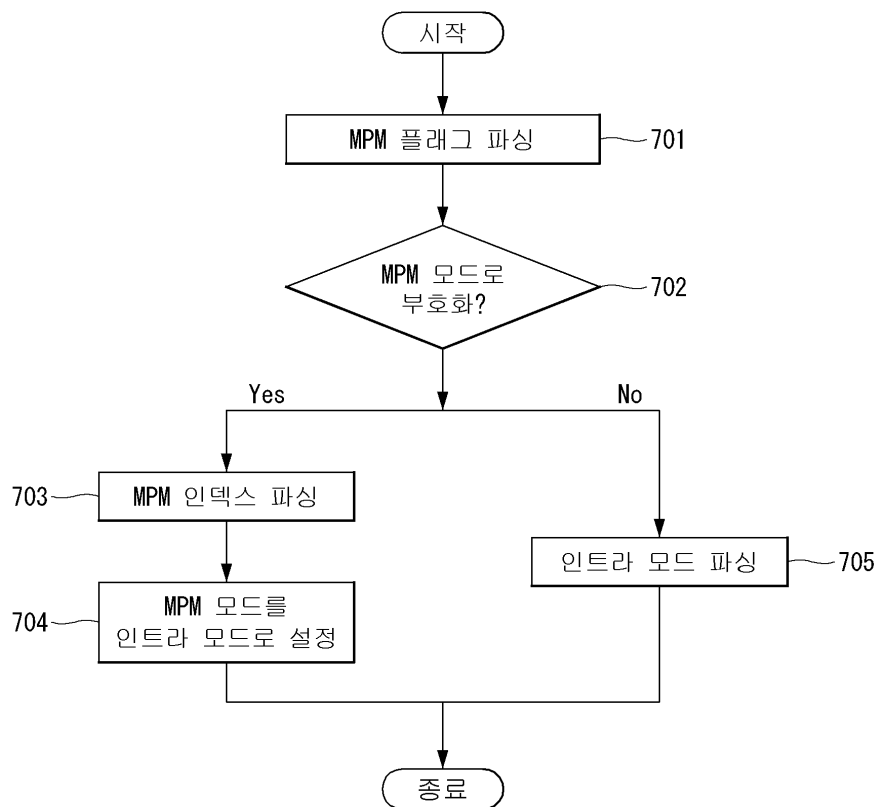
도면5



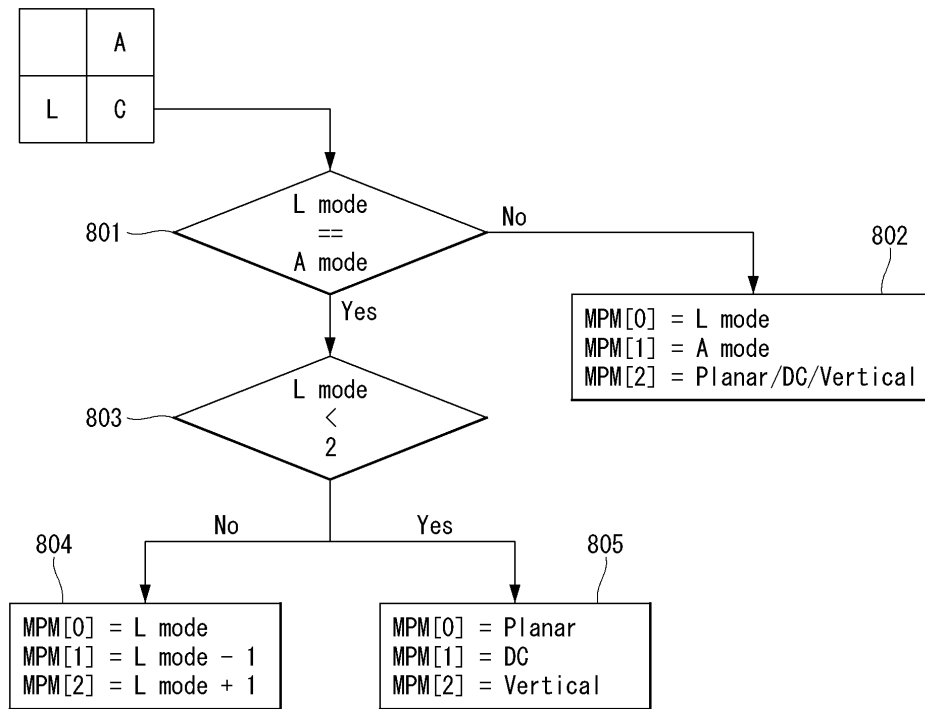
도면6



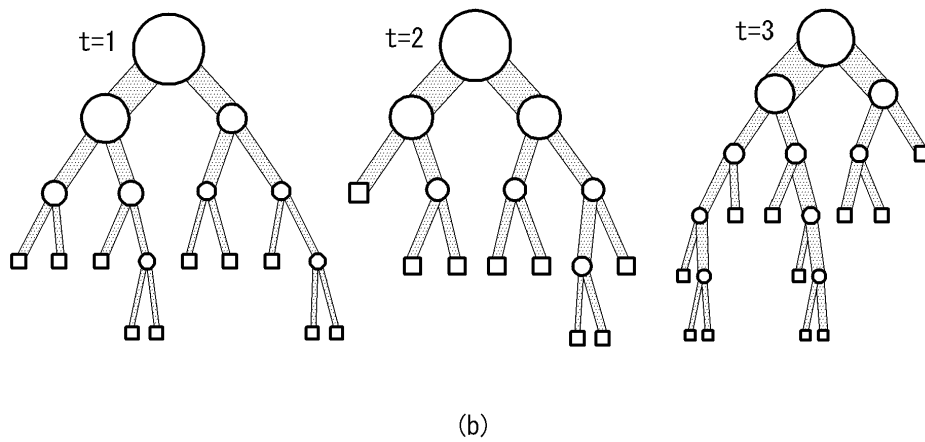
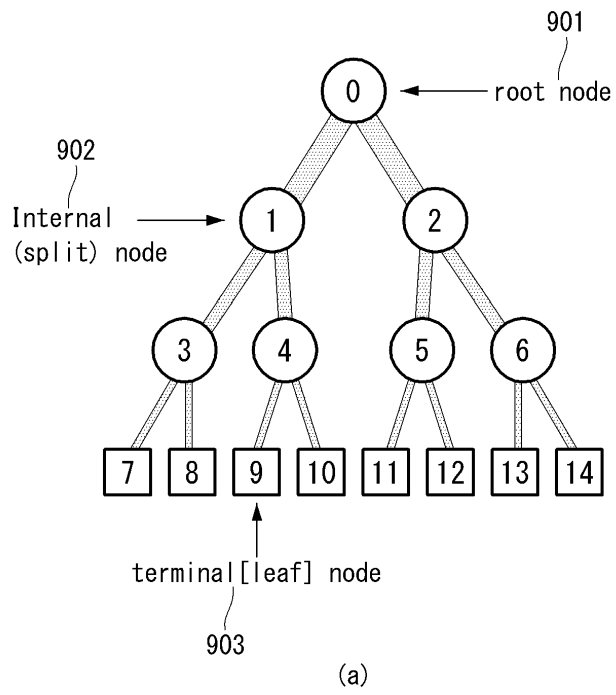
도면7



도면8



도면9

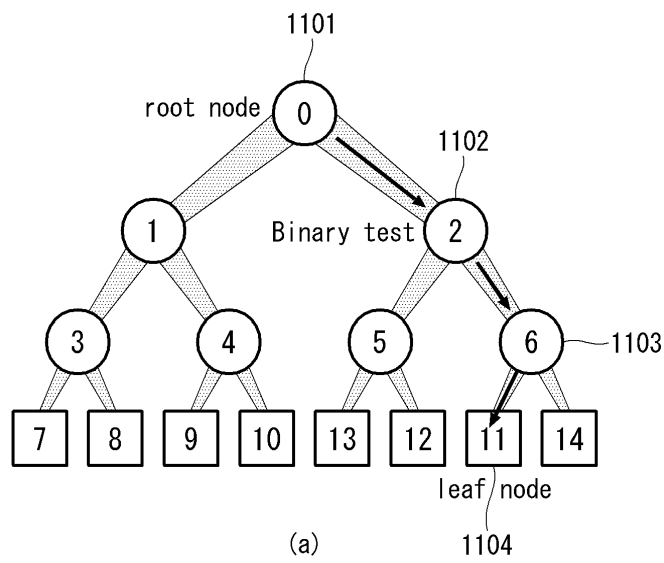


도면10

1001

$R_{0,0}$	$R_{1,0}$	$R_{2,0}$	$\cdots$	$R_{N,0}$	$R_{N+1,0}$	$\cdots$	$\cdots$	$R_{2N,0}$
$R_{0,1}$	$P_{1,1}$	$P_{2,1}$	$\cdots$	$P_{N,1}$	-1	$\cdots$	$\cdots$	-1
$R_{0,2}$	$P_{1,2}$				-1			-1
$\vdots$	$\vdots$				-1			-1
$R_{0,N}$	$P_{1,N}$	$\cdots$		$P_{N,N}$	-1			-1
$R_{0,N+1}$	-1	-1	-1	-1	-1			-1
$\vdots$	$\vdots$							-1
$\vdots$	$\vdots$							-1
$R_{0,2N}$	-1	-1	-1	-1	-1	-1	-1	-1

도면11



210	20	30	40	30	0	0	0	0
255								
255								
255								
200								
210								
220								
225								
225								

(b)

도면12

$$h(v, \theta) = |R_{x0, y0}^O - R_{x1, y1}^I| < t$$

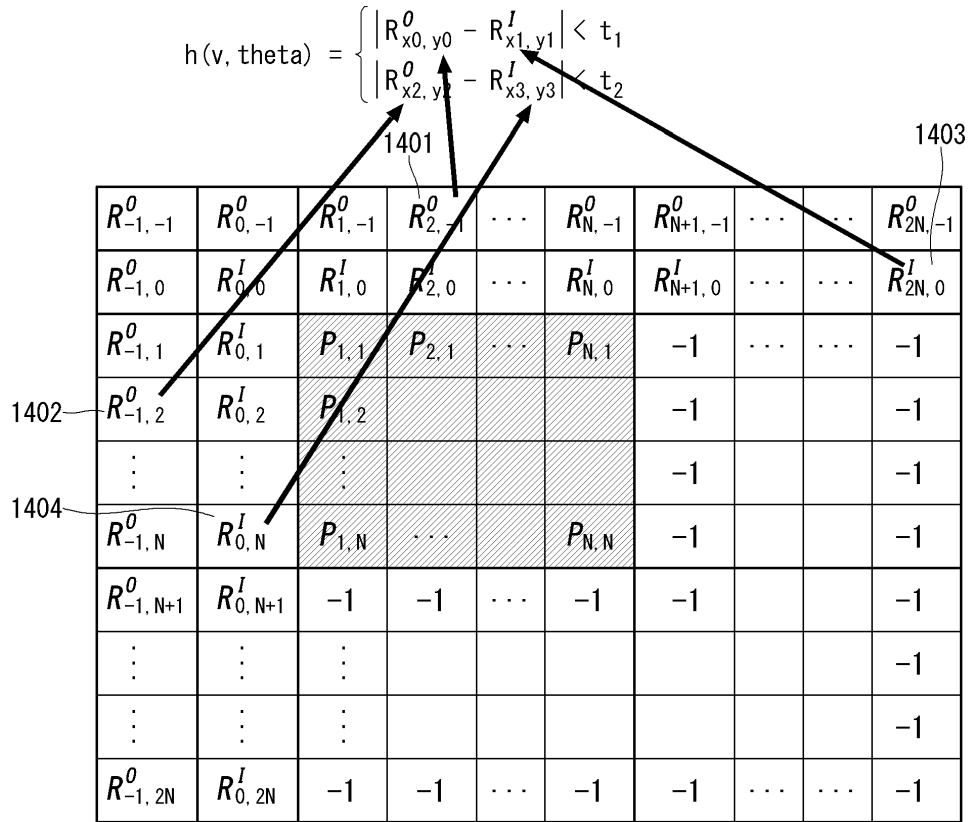
$R_{-1, -1}^O$	$R_{0, -1}^O$	$R_{1, -1}^O$	$R_{2, -1}^O$	...	$R_{N, -1}^O$	$R_{N+1, -1}^O$	...	...	$R_{2N, -1}^O$
$R_{-1, 0}^O$	$R_{0, 0}^I$	$R_{1, 0}^I$	$R_{2, 0}^I$	...	$R_{N, 0}^I$	$R_{N+1, 0}^I$	...	...	$R_{2N, 0}^I$
$R_{-1, 1}^O$	$R_{0, 1}^I$	$P_{1, 1}$	$P_{2, 1}$	...	$P_{N, 1}$	-1	...	...	-1
$R_{-1, 2}^O$	$R_{0, 2}^I$	$P_{1, 2}$				-1			-1
$\vdots$	$\vdots$	$\vdots$				-1			-1
$R_{-1, N}^O$	$R_{0, N}^I$	$P_{1, N}$	...		$P_{N, N}$	-1			-1
$R_{-1, N+1}^O$	$R_{0, N+1}^I$	-1	-1	...	-1	-1			-1
$\vdots$	$\vdots$	$\vdots$							-1
$\vdots$	$\vdots$	$\vdots$							-1
$R_{-1, 2N}^O$	$R_{0, 2N}^I$	-1	-1	...	-1	-1	...	...	-1

도면13

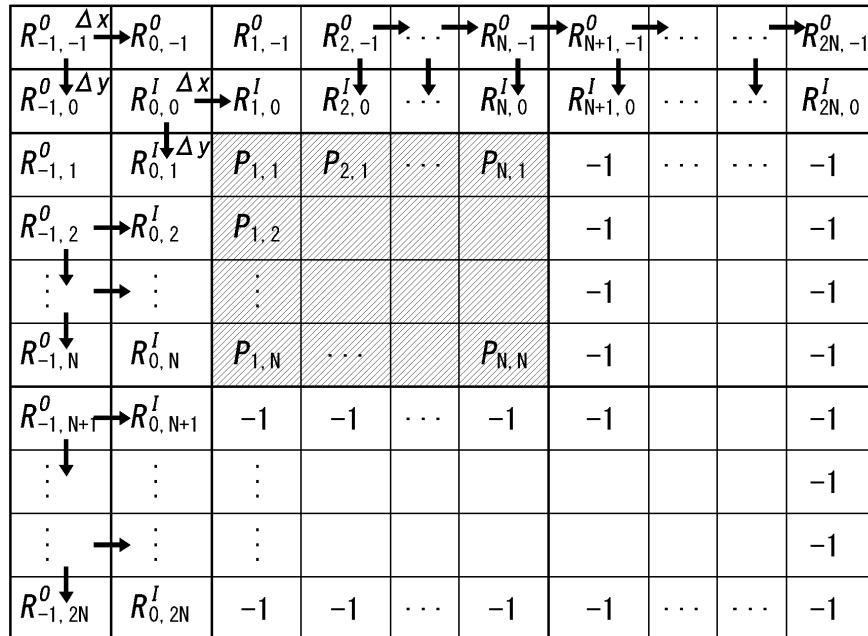
$$h(v, \theta) = \begin{cases} |R_{x0, y0}^O - R_{x1, y1}^I| < t_1 \\ |R_{x2, y2}^O - R_{x1, y1}^I| < t_2 \end{cases}$$

$R_{-1, -1}^O$	$R_{0, -1}^O$	$R_{1, -1}^O$	$R_{2, -1}^O$	...	$R_{N, -1}^O$	$R_{N+1, -1}^O$	...	...	$R_{2N, -1}^O$
$R_{-1, 0}^O$	$R_{0, 0}^I$	$R_{1, 0}^I$	$R_{2, 0}^I$	...	$R_{N, 0}^I$	$R_{N+1, 0}^I$	...	...	$R_{2N, 0}^I$
$R_{-1, 1}^O$	$R_{0, 1}^I$	$P_{1, 1}$	$P_{2, 1}$	...	$P_{N, 1}$	-1	...	...	-1
$R_{-1, 2}^O$	$R_{0, 2}^I$	$P_{1, 2}$				-1			-1
$\vdots$	$\vdots$	$\vdots$				-1			-1
$R_{-1, N}^O$	$R_{0, N}^I$	$P_{1, N}$	...		$P_{N, N}$	-1			-1
$R_{-1, N+1}^O$	$R_{0, N+1}^I$	-1	-1	...	-1	-1			-1
$\vdots$	$\vdots$	$\vdots$							-1
$\vdots$	$\vdots$	$\vdots$							-1
$R_{-1, 2N}^O$	$R_{0, 2N}^I$	-1	-1	...	-1	-1	...	...	-1

도면14



도면15



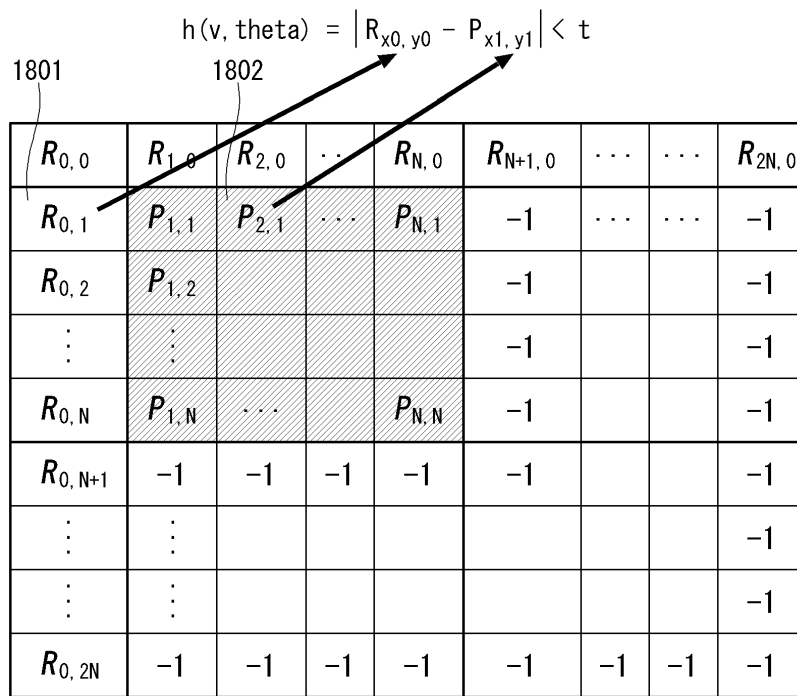
도면16

1601	1602									
$R_{-1,-1}^O \xrightarrow{D_0} R_{0,-1}^O \xrightarrow{D_1} R_{1,-1}^O \xrightarrow{D_2} R_{2,-1}^O \cdots R_{N,-1}^O R_{N+1,-1}^O \cdots \xrightarrow{D_{2N}} R_{2N,-1}^O$										
$R_{-1,0}^O \rightarrow R_{0,0}^I \rightarrow R_{1,0}^I \rightarrow R_{2,0}^I \cdots R_{N,0}^I R_{N+1,0}^I \cdots \rightarrow R_{2N,0}^I$										
$R_{-1,1}^O \rightarrow R_{0,1}^I$	$P_{1,1}$	$P_{2,1}$	$\cdots$	$P_{N,1}$	-1	$\cdots$	$\cdots$	-1		
$R_{-1,2}^O$	$R_{0,2}^I$	$P_{1,2}$			-1			-1		
$\vdots$	$\vdots$	$\vdots$			-1			-1		
$R_{-1,N}^O$	$R_{0,N}^I$	$P_{1,N}$	$\cdots$	$P_{N,N}$	-1			-1		
$R_{-1,N+1}^O$	$R_{0,N+1}^I$	-1	-1	$\cdots$	-1	-1		-1		
$\vdots$	$\vdots$	$\vdots$						-1		
$\vdots$	$\vdots$	$\vdots$						-1		
$R_{-1,2N}^O \rightarrow R_{0,2N}^I$	-1	-1	$\cdots$	-1	-1	$\cdots$	$\cdots$	-1		

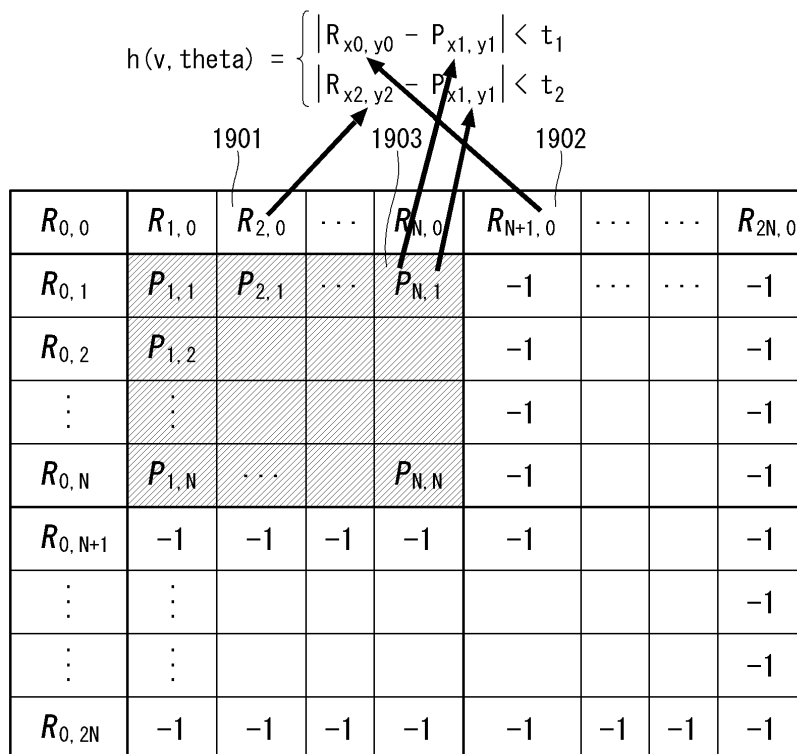
도면17

1701	$R_{-1,-1}^O$	$R_{0,-1}^O$	$R_{1,-1}^O$	$R_{2,-1}^O$	$\cdots$	$R_{N,-1}^O$	$R_{N+1,-1}^O$	$\cdots$	$\cdots$	$R_{2N,-1}^O$
1702	$R_{-1,0}^O \downarrow^{D_0} R_{0,0}^I$	$R_{1,0}^I$	$R_{2,0}^I$	$\cdots$	$R_{N,0}^I$	$R_{N+1,0}^I$	$\cdots$	$\cdots$	$R_{2N,0}^I \downarrow$	
	$R_{-1,1}^O \downarrow^{D_1} R_{0,1}^I$	$P_{1,1}$	$P_{2,1}$	$\cdots$	$P_{N,1}$	-1	$\cdots$	$\cdots$	-1	
	$R_{-1,2}^O \downarrow^{D_2} R_{0,2}^I$	$P_{1,2}$				-1			-1	
	$\vdots$	$\vdots$				-1			-1	
	$R_{-1,N}^O$	$R_{0,N}^I$	$P_{1,N}$	$\cdots$	$P_{N,N}$	-1			-1	
	$R_{-1,N+1}^O$	$R_{0,N+1}^I$	-1	-1	$\cdots$	-1	-1		-1	
	$\vdots$	$\vdots$	$\vdots$						-1	
	$\vdots$	$\vdots$	$\vdots$						-1	
	$R_{-1,2N}^O \downarrow^{D_{2N}} R_{0,2N}^I$	-1	-1	$\cdots$	-1	-1	$\cdots$	$\cdots$	-1	

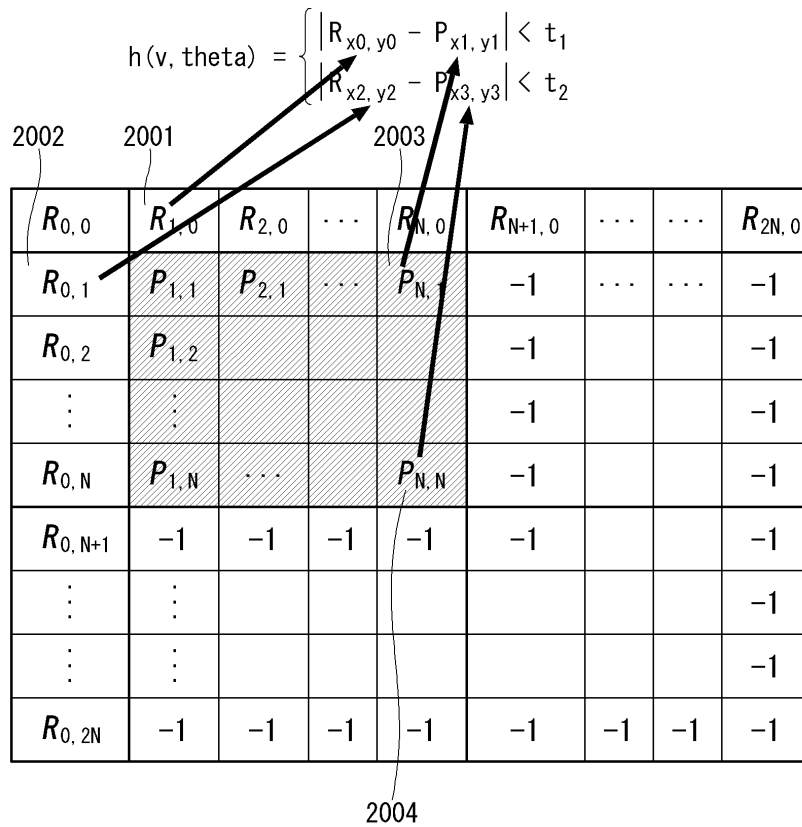
도면18



도면19

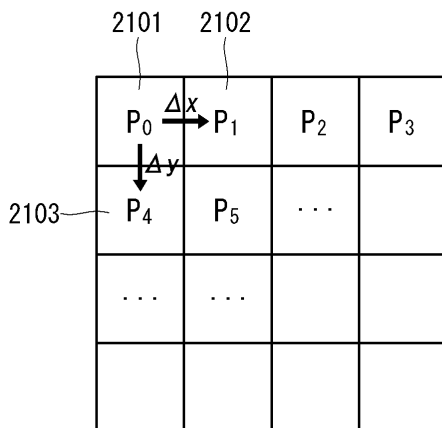


도면20

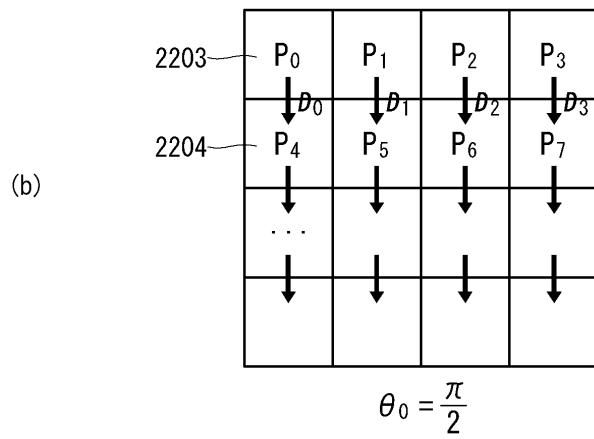
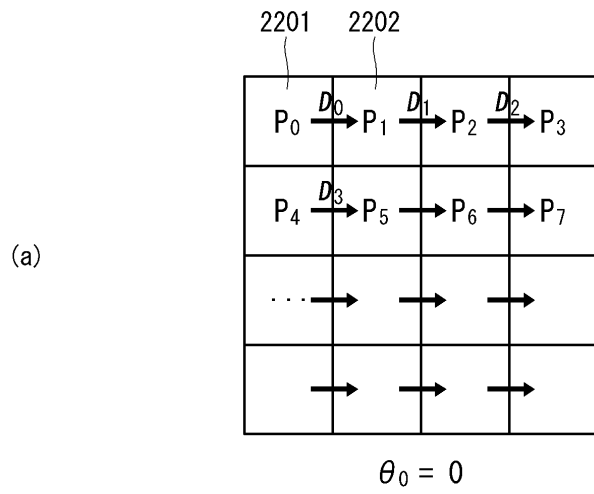


도면21

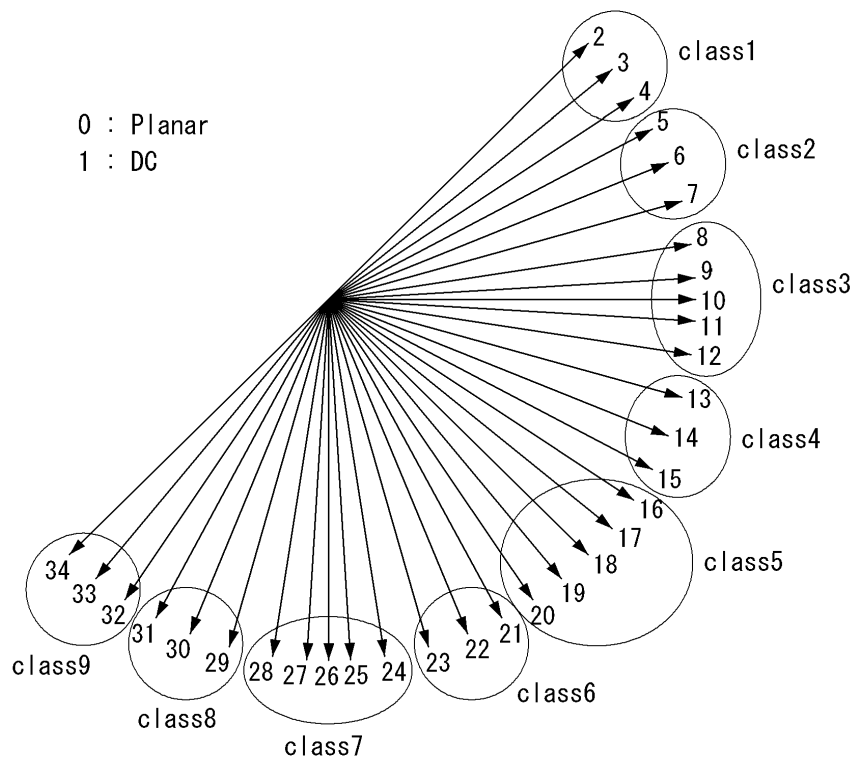
Gradient Magnitude가 최대가 되는 θ 선택



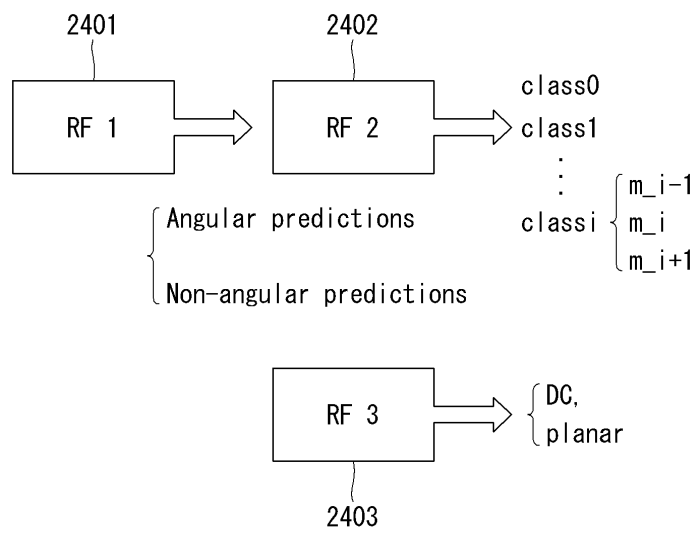
도면22



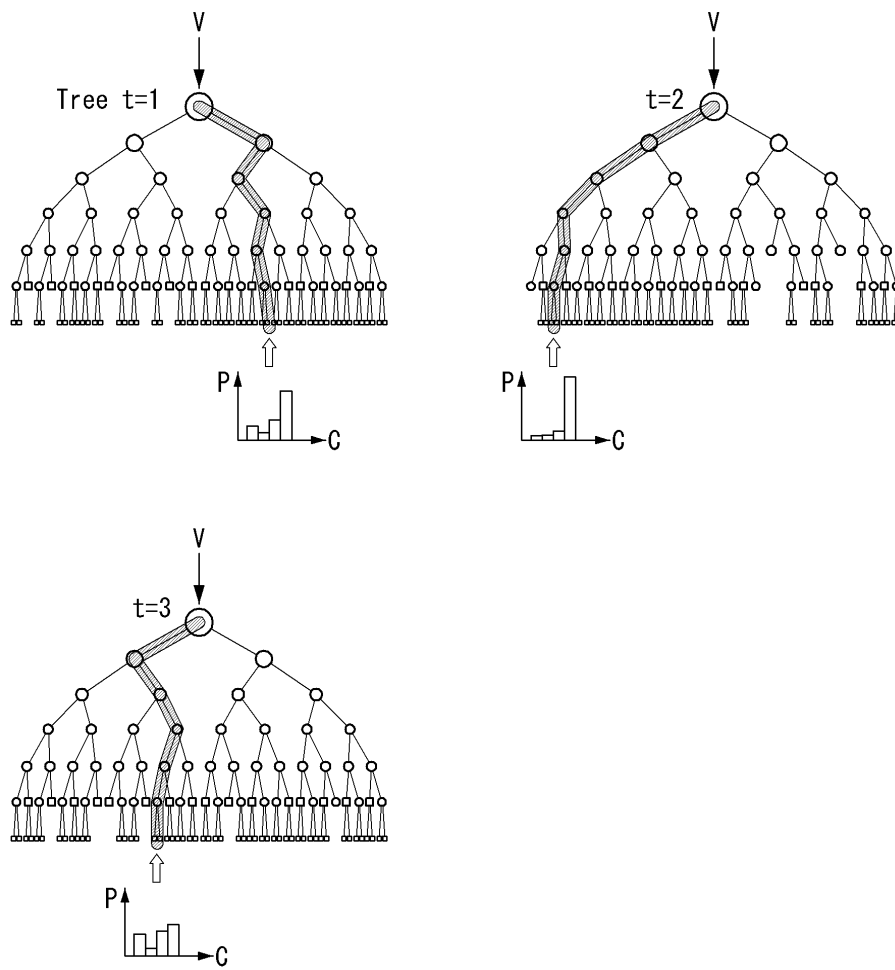
도면23



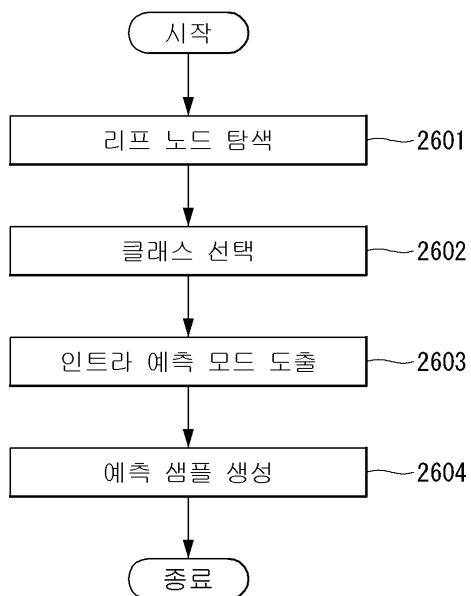
도면24



도면25



도면26



도면27

