목차

# 이전 내용의 보강

## PSNR

# 새로운 내용

## 이미지 에서의 압축의 원리 -근본적으로

## 동영상 에서의 압축의 원리 -근본적으로

### -> HEVC, H.264에서 각각 구체적으로 어떻게 알고리즘이 돌아가는지

### -> 어떻게 다이나믹한 크기로 압축이 되는지

## (개념적인 알고리즘을 상세하게)

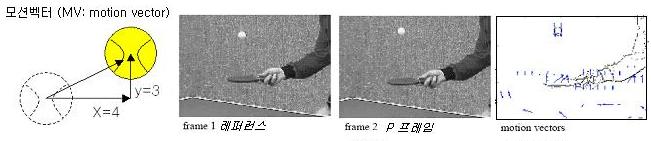
## (- 모션 벡터 예측 등등)

내용

## 영상 에서 압축 원리

* 모션 벡터 – 시간적 유사성

영상의 매 frame의 구성 요소의 움직임을 벡터로 표현한 것



좀 과장하여 예를 들면

탁구공을 그림으로 표현하면 3byte\* 50\*50=7500byte가 필요한데 모션벡터는 앞장에 (4,3)위치에 있다 라고하면

두바이트로 가능할수도 있게된다.  엄청난 압축이다^^.

**[출처]** [모션벡터 Motion Vector](http://blog.naver.com/msnayana/80106586501)|**작성자** [수수깡](http://blog.naver.com/msnayana)

대부분의 동영상 압축은 **Block-based Motion Estimation**(이하 **ME**)으로  수행

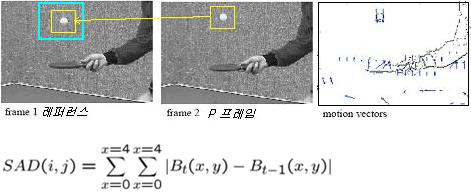
* 블럭기반움직임추정의 원리

아래그림은

현재 영상인 frame2에서  frame1에 같은 부분이 있는지 블럭을 나눠서 검색하는 과정으로

기본적으로 한 화면내의 블럭을 16 x 16으로 나눠서 하는데 통산적인 블럭은  8x8, 16x16, 32x32를  많이 사용한다.

하늘색 사각형은 **검색영역**이며  이것은 통상적으로  **Block Size의 ±7을 사용**하여 만든다.



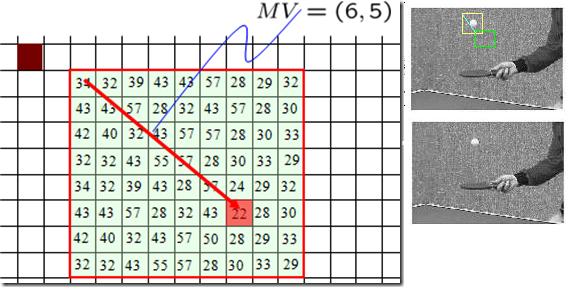
검색영역내에서 가장일치하는 영상을 찾는 방법은 아래와 같은 SAD방정식을 수행하여 찾는다

수식은 복잡하지만 의미는

Search Range를 매크로 블럭으로 탐색 하면서 Frame(t-1) - Cerrent block을 계산하여 그 ‘절대값’을 저장.

이렇게 각각 한픽셀씩 블럭을 이동해 가며 수행하고 나면

각 픽셀영역에 값이 쌓이고 그중에 가장 낮은 값을 찾으면 된다.



위 그림은

MV(6,5)지점인 숫자 22가 들어있는 좌표(6.5)의 지점이 모션벡터이다.

즉 Current Block과 가장 일치하는 이전 프레임의 좌표로 이전프레임에서 이동한 물체를 알수 있다

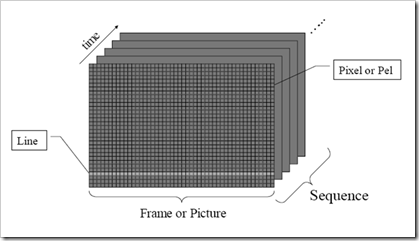
**[출처]** [모션벡터 Motion Vector](http://blog.naver.com/msnayana/80106586501)|**작성자** [수수깡](http://blog.naver.com/msnayana)

**#0. 들어가며..**

방학을 이용하여,**H.264 코덱을 공부하는 것을 목표**로 해서 동영상의 **인코딩, 디코딩**에 관해 배워 나가고 있습니다.

이 내용들은 제가 세미나를 통해 배운 내용을 나름대로 정리하여 잊어 버리지 않게 하는 것이 용도이고, 저도 세미나 도중 이해 가지 않는 부분이 있으므로, 제가 올린 내용이 틀릴 수 있습니다. 내용에 틀린 점이 있을 시 지적해 주시면 즉시 내용을 수정 하도록 하겠습니다.  
  
**#1. 동영상이란?**

동영상 압축의 기본을 알기 위해선 우선 동영상이 무엇인지 알아야 합니다.

[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile4.uf.tistory.com/image/1977E00B4A811B552614CD)   
-그림1. 동영상의 구조-

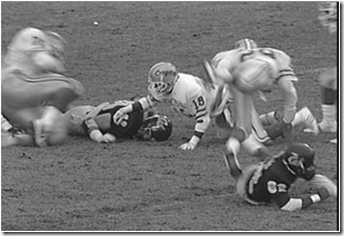
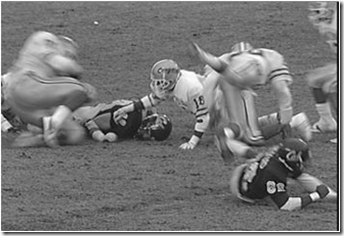
위의 그림에서 알 수 있듯이 **동영상은 정지된 사진(프레임)이 시간을 축으로 여러 장 모여 있는 것**입니다. 즉 주어진 시간 ~~동안 연속으로 촬영된 사진이라 볼 수 있습니다.(사실 여기서 소리도 포함 되야 하지만, 소리는 별도의 인코딩 과정을 거친 후 먹싱 되므로 소리는 빼고 생각해 보겠습니다.)   
  
그렇다면 동영상(RGB 비디오)의 데이터 크기는 어떻게 될까요? 프레임은 그림파일(Image)와 같습니다. 예를 들어 512×512의 동영상이 있다고 하면, 이 동영상의 한 프레임을 기준으로 가로 : 512, 세로 : 512개의 점(픽셀)으로 구성되어있으며, 1픽셀은 8bit(256 levels)입니다.  그 중에서 회색이 차지하는 비율만 생각해 보면, Width×Height×Bit(512×512×8)가 됩니다. 그리고 3개의 Color들로 구성 되어있으므로, RGB의 경우 Width×Height×Bit×Color(512×512×8×3)이 되어 6291456bit가 됩니다. 6291456(bit) == 6144(kbit) == 6(mbit) 입니다.  
  
512×512라는 비교적 작은 사이즈의 동영상도 무 압축일 경우, 한 프레임에 6메가라는 큰 사이즈가 되는 것입니다. 동영상서비스를 제공 할 경우 적어도 초당 24프레임을 보장해야 한다고 하니 1초의 동영상에~~**~~144메가~~**~~라는 큰 용량이 필요한 것입니다~~. 이것이 우리가 동영상 압축을 공부해야 하는 이유이기도 합니다.

**>> 동영상의 용량은 대체적으로 크기 때문에, 동영상 압축에 대한 공부가 필요하다**  
  
~~네트워크 기술이 많이 발달하였지만, 이런한 무 압축 비디오까지 실시간으로 전송 하는 데는 역부족 입니다. 더군다나 요즘 많이 보급된~~**~~FullHD의 경우 1920×1080의 해상도를 지원해야 하기 때문에, 어마어마한 데이터 양~~**~~이 될 수 밖에 없습니다. 또한 네트워크 대역폭이 이 정보를 전부 전송해 줄 수 있다고 하더라도, 동영상의 크기를 줄여, 그 남는 대역폭을 이용하여 여러 다른 서비스를 제공 해 줄 수도 있을 것입니다.~~

**#2. 동영상 압축 기법(동영상 압축을 위한 기본적인 방법)**

이러한 동영상을 압축 하려면 어떻게 해야 할까요?

**동영상 압축에는 크게 세 가지 기법이 사용 됩니다**. 첫번째는 **프레임 간의 상관관계** (temproal correlation)이고, 두번째는 **공간적인 상관관계**(spatial correlation)입니다. 세번째는 **저주파 성분에 민감한 시각의 특성을 이용**합니다 (DCT 변환).

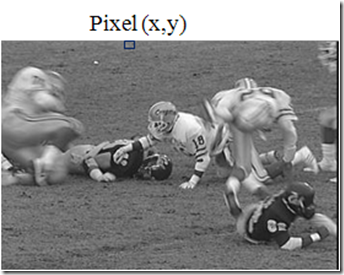
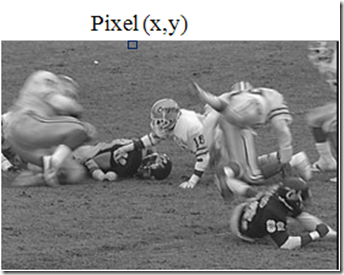
[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile23.uf.tistory.com/image/20420C0F4A812801212C5D) [](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile22.uf.tistory.com/image/1838EF114A81280102FEFF)

                        -그림2. 프레임 9-                                                   -그림3. 프레임 10-

먼저 **프레임간의 상관관계**를 알아보도록 하겠습니다. 그림 2와 3은 각각 같은 동영상(football\_sif.yuv)의 9번과 10번 프레임입니다. 마치 같은 그림처럼 닮아있습니다.  **근접한 프레임은 거의 비슷한 이미지로 이루어져 있다는 아이디어를 이용**하여, **영상의 크기를 줄일 수 있습니다**.

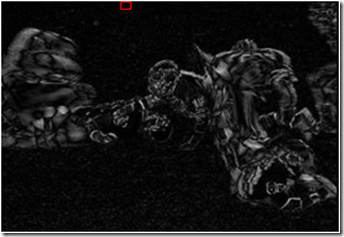
[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile2.uf.tistory.com/image/166290134A812802776392)  
-그림 4. Image의 **공간적인 중복**-

그림 4에서 보는 것과 같이 **한 장의 이미지에도 인접한 픽셀간의 정보는 거의 중복 된다는 것**을 알 수 있습니다. 이방법으로도 영상의 크기를 줄 일 수 있을 것입니다. - JPEG또한 사용  
  
Temporal correlation(프레임간의 상관관계)은  그림 5,6처럼 현재 프레임(t)의 하나의 픽셀(x,y)은 이전 프레임(t-1)의 비슷한 위치(x,y)에 있을 가능성이 높고, 또한 두 필셀 사이에는 유사성이 높다는 것입니다.

[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile23.uf.tistory.com/image/1243180C4A812802AC19BD) [](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile4.uf.tistory.com/image/14134D0B4A8128043F21E4)

                       -그림 5. Frame(t)-                                                -그림 6. Frame(t-1)-

이렇게 프레임(t)에서 프레임(t-1)과의 차를 계산해 보면 중복 데이터가 사라지고, 차이인 그림 7이 나오게 됩니다.

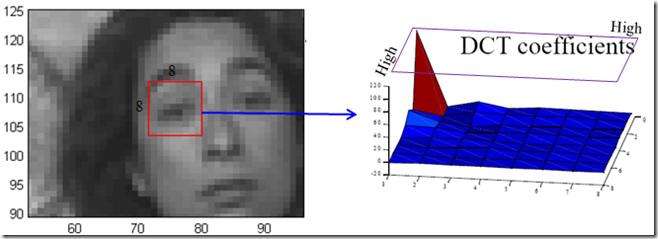
[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile21.uf.tistory.com/image/1438BF0C4A81280474D8F3)  
      -그림 7. Frame(t) 와 Frame(t-1)의 Difference -   
  
  
이번에 간단한 동영상의 정의와, 동영상을 압축하는 기본적인 방법을 알아 보았습니다. 다음에는 RGB와 YUV를 알아보고, 위에서 간단히 소개한 압축의 기법들을 자세히 소개하는(주파수 도메인(DCT)을 이용한 방법, Motion Estimation 등) 순으로 진행 할 예정이며, 기본적인 지식을 갖춘 후, H.264에 대해서도 알아 보도록 하겠습니다.

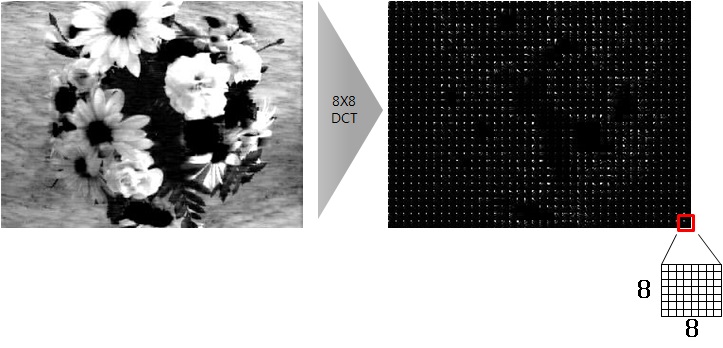
**#1. Discrete Cosine Transform 개요**

**Discrete Cosine Transform**(이하 DCT)는 무엇일까요? 번역해 보자면 이산 코사인 변환이라고 합니다. 이전 시간에 배운 ME는 프레임간 유사성을 이용하여 영상을 압축 할 수 있는 기법이었는데요. DCT는 해당하는 프레임(Picture)만을 이용해서 영상을 압축 할 수 있도록 하는 기법이라 할 수 있습니다. 그렇기 때문에 **사진을 압축하는 JPEG에서 쓰이는 방식**입니다. 동영상 코덱에서는 H.261, H.263, H.264.. MPEG등 에서 이 방식을 사용하여 영상을 압축하게 됩니다.

**>> DCT는 우리말로 [이산 코사인 변환] 으로 유사성을 이용하는 것이 아니라, 하나의 프레임 안에서 압축하는 것, 따라서 영상(H.261, H.263, H.264.. MPEG등) 및 JPEG에서 쓰이기도 함**

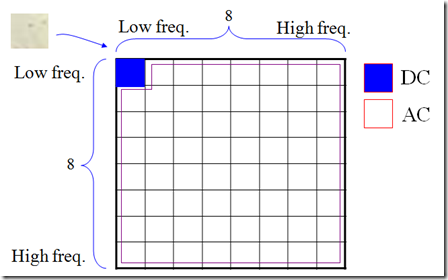
대체 DCT란 놈이 어떤 것인지 좀더 자세히 알아보도록 하죠. 혹시 신호처리를 알고 계신 분은 좀더 쉬울 수 있겠습니다. 저는 신호처리를 모르는 관계로 처음 배울 때 잘 이해가 가지 않더군요.

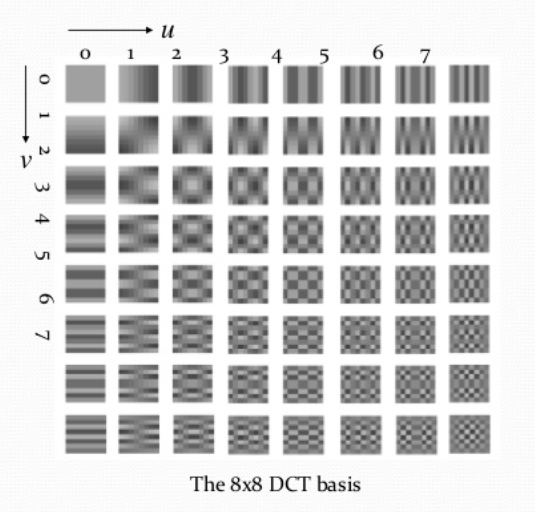
[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile22.uf.tistory.com/image/145D020E4A97DF396D2449)  
- Fig1. DCT 변환 -



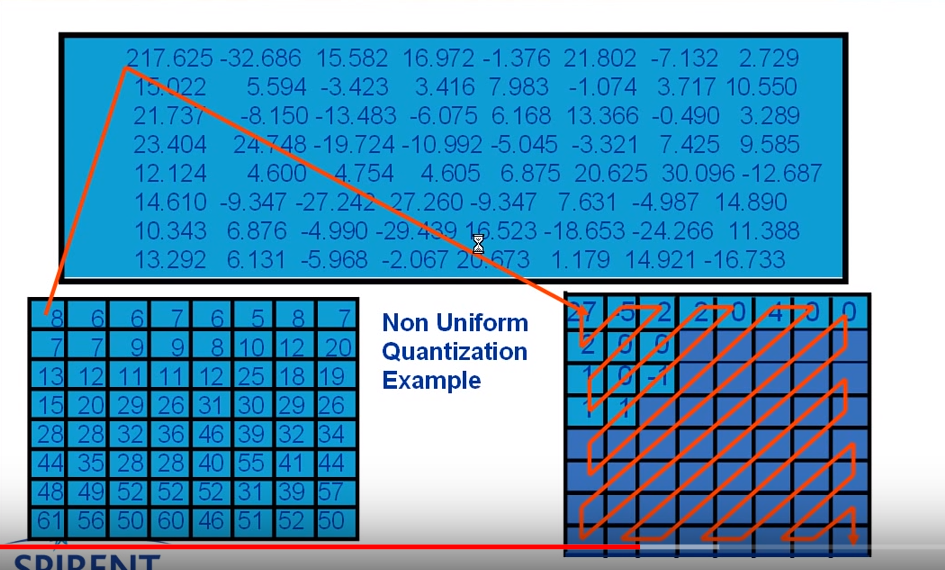
영상을 DCT 변환하면 Fig1에서 처럼**공간 영역(Spatial Domain)에서 주파수 영역(Frequency Domain)으로 변하게 됩니다.** 그림을 잘 살펴보면 주파수가 낮은 쪽으로 데이터(신호)가 몰려 있는 것을 볼 수 있습니다. 이와 같이 DCT를 하면 신호성분이 낮은 주파수에 몰리게 되는 **‘에너지 집중 현상’**이 나타납니다.

**#2. DCT를 하는 이유**

[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile3.uf.tistory.com/image/2070D3144B53138B7988E3)  
- Fig2. DCT Coefiicients -



u: 수평방향 주파수, v: 수직방향 주파수



그럼 코덱은 압축을 위해서 왜 DCT를 수행 하는 것일까요? DCT는 공강영역을 주파수 영역으로 변환한다고 하였습니다. 영상을 관찰하다 보면 **인접한 픽셀간에는 비슷한 색상인 경우가 많이 있습니다**. 즉 8×8 블럭을 DCT하였을 경우 64개의 픽셀중에 같은 색상이 낮은 주파수(DC)로 몰리게 됩니다. 그리고 색상의 변화가 있는 경우 높은 주파수(AC)으로 위치하게 됩니다.

DCT를 수행하게 되면 대부분의 이미지는 Fig1에서 와 같은 결과가 나오게 됩니다. 빨간색으로 우뚝 솟은 부분(DC)이 제일 큰 신호 값이며, 파란색은 울퉁불퉁 하긴 하지만 높은 주파수 영역으로 갈 수록 평평한 것을 볼 수 있습니다. 즉 위에서 설명 했듯이 **8×8 블럭의 색상 평균값이 DC성분이 되며, 나머지 평균값과 틀린 부분이 AC 성분**이 되는 것입니다. 눈으로 보아도 DC에 값이 많이 몰린 것을 알 수 있죠? 이 의미가 바로 인접한 픽셀간은 거의 비슷한 색상으로 이루어 져 있기 때문에 DC의 값이 크다라고 생각하시면 됩니다.

그런데 **사람의 눈 또한 낮은 주파수 성분(DC)에는 민감하게 반응 하지만, 높은 주파수 성분(AC)에는 민감하지 않기 때문에 높은 주파수 영역(그림에서 파란색)을 좀 생략 한다고 해도 화질의 차이**를 잘 느끼지 못하게 됩니다. 이것이 압축이 되는 원리 입니다.

그림1 에서 보는 것과 같이 DCT변환 전에는 8×8 블럭 이므로 64개의 데이터가 필요 하지만  DCT를 하여 높은 주파수 성분(AC)을 좀 버려서(화질의 차이는 크지 않으므로)  DC에서 가까운 10개의 데이터만 가지게 된다면, 그만큼 압축이 되는 것 입니다. 사실 DCT는 이렇게 주파수 성분으로 바꾸는 작업만을 이야기 하며 다음시간에 이야기할 Quantization과 Entropy coding과정을 거쳐야 실질적인 압축이 일어납니다.

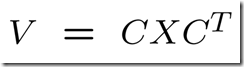
**#3. Discrete Cosine Transform(DCT)**

위의 개념이 이해가 가시나요? 어떻게 영상을 주파수 영역으로 변환하는지에 관해 알아 보겠습니다.

DCT도 ME와 마찬가지로 블럭 단위로 수행하게 됩니다. 그 이유는 블럭 사이즈가 커지면 시간이 엄청 걸리기도 하지만, 전체 프레임을 한번에 처리하기 위해서는 어마어마한 메모리도 필요 하게 됩니다. 그래서 일반적으로 JPEG, H.261, H.263, MPEG에서는 8×8 블럭 단위로 DCT를 수행하게 됩니다. (H.264는 기본 4×4에 변형이 가능합니다.)

그럼 어떻게 공간 영역을 주파수 영역으로 바꿀 수 있을까요?

아래는 2D-DCT를 수행하기 위한 수식입니다.

[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile25.uf.tistory.com/image/120D03284B53138B554881)  
- Fig3. 2D-DCT -

V는 DCT를 수행하여 주파수 도메인으로 바뀐 , C는 DCT수행에 필요한 메트릭스(Fig4 참조), X는 원본 블럭, CT는 C 메트릭스의 가로 세로를 변환한 메트릭스 입니다.

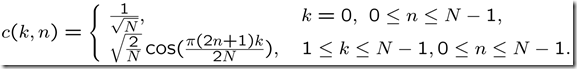
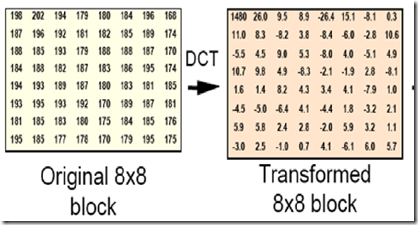
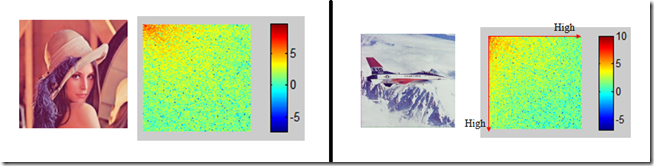
[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile24.uf.tistory.com/image/1206E7144B53138C2E0A89)  
- Fig4. N×N Cosine Transfrom Matrix   C={c(k,n))} -

Fig4는 DCT 수행에 필요한 Matrix를 어떻게 생성하는지를 나타냅니다. k는 행을 n은 열을 나타내고 N은 Matrix의 크기를 나타냅니다. 우리는 8×8 DCT를 수행할 것이므로 N은 8이 됩니다. 이렇게 생성된 Matrix를 이용하여 V=CXCT 라는 수식으로 DCT가 된 데이터를 가져올 수 있습니다. 아래 Fig5는 DCT후의 데이터 변화를 나타냅니다. **숫자로 확인하여도 DC성분에 값이 집중된 것을 알 수 있습니다**.

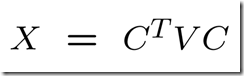
[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile25.uf.tistory.com/image/1967DF0E4B53138C3044A0)  
- Fig5. DCT후의 데이터 변화 -

아래에 있는 Fig6에서도 마찬가지로**높은 값들은 DC쪽에 집중되어 분포**하게 되는 것을 알 수 있습니다.

[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile2.uf.tistory.com/image/19025F0C4B53138D0D8EE0)   
-Fig6. Matlab으로 확인해본 DCT후의 데이터 분포-

**#4. Inverse DCT는 어떻게?**

DCT를 하여 주파수 도메인으로 변환 할 수 있다는 것을 알았습니다. 그럼 다시 공간 도메인으로 변경 하려면 어떻게 해야 할까요?

[](http://crynut84.tistory.com/script/powerEditor/pages/http:/cfile4.uf.tistory.com/image/127599114B53138E26AFA4)  
- Fig7. 2D-Inverse DCT -

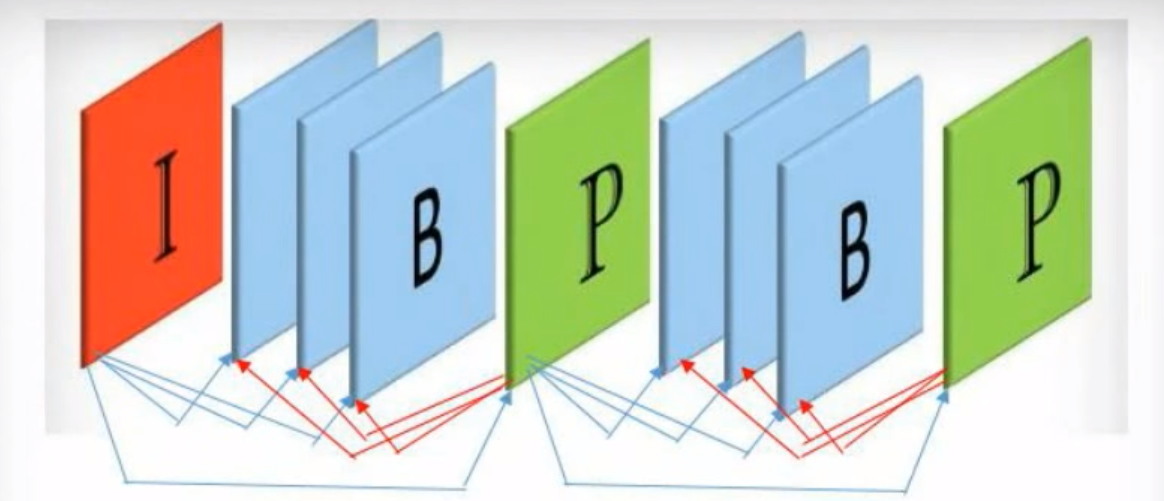
위와 같은 공식으로 원래 블럭을 복원 해 낼 수 있습니다. 물론 **복원한 블럭은 오리지날과 100% 동일**하게 됩니다.

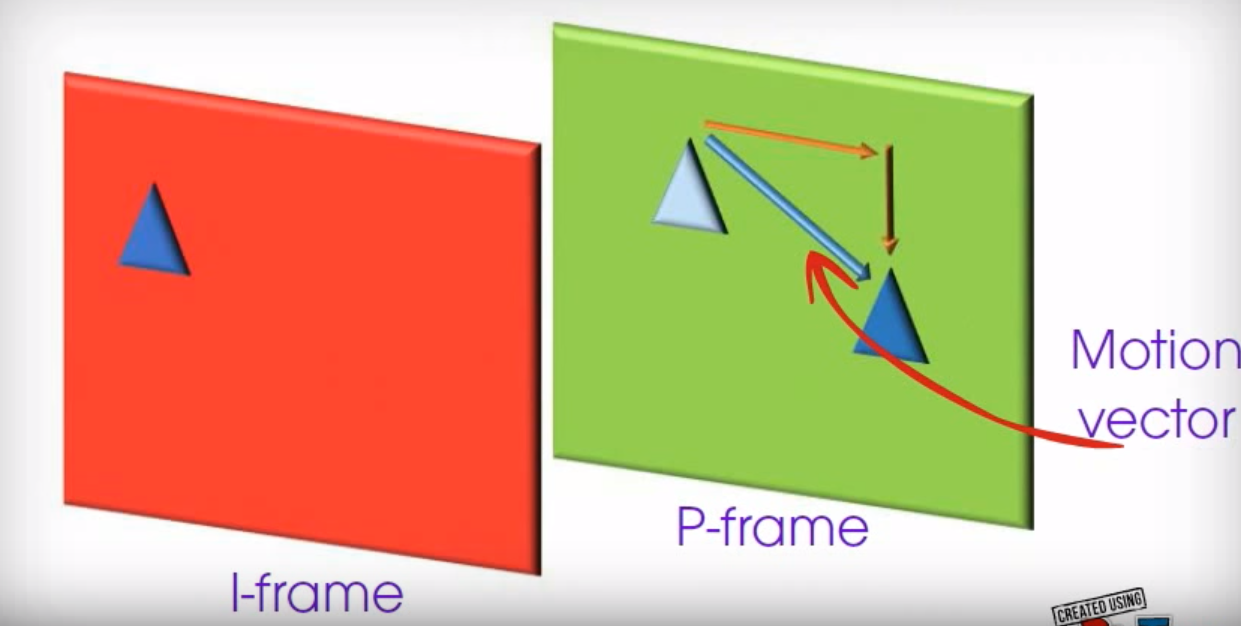
**#5. 끝마치며**

이번 시간에는 코덱의 인코딩 과정중 하나인 DCT의 개념과 DCT를 왜 해야하는지 알아 보았으며, DCT된 블럭을 원래대로 되돌리는 것 까지 알아 보았습니다. 다음 시간에는 **Quantization에 관해 알아보고 전체적인 인코딩 프로세싱**도 살짝 짚어 보려구 합니다.

제가 전체 그림을 제시하고 세부 강좌를 만들었어야 하는데, 세부적인 설명부터 하다보니 좀 오해가 생기는 부분이 있는 것 같아서요. 그럼 좋은 주말 저녁 되세요^^

출처: <http://crynut84.tistory.com/61> [Life is Dynamic]

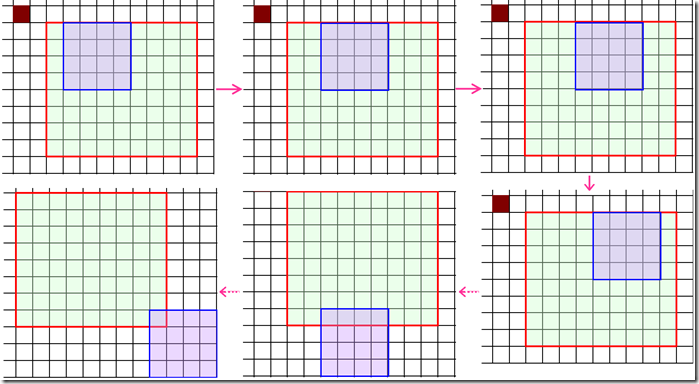


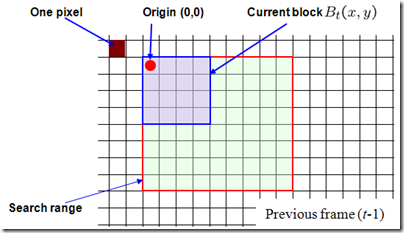
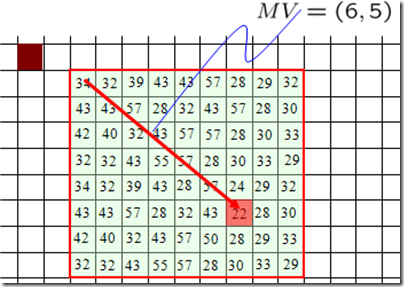


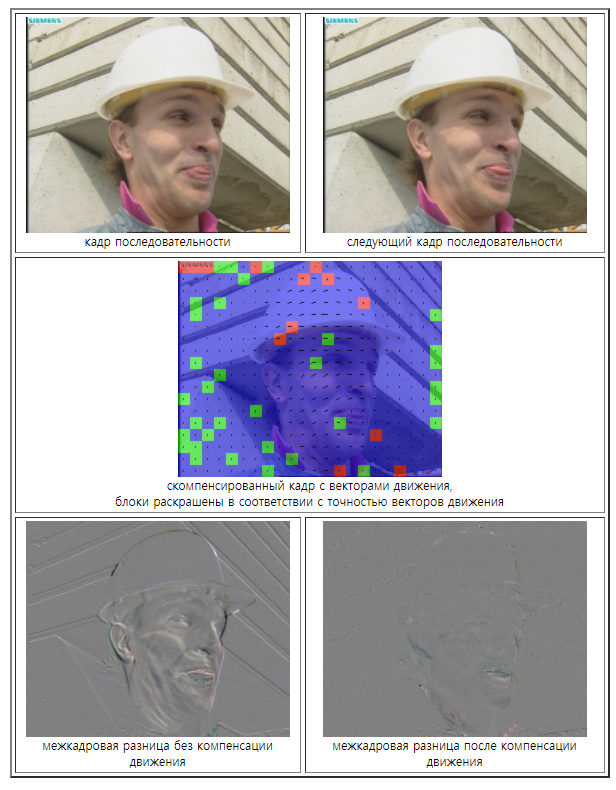
I 프레임 - Infra Frame 의 약자로, 쉽게 말해 키 프레임 입니다. 이것은 JPEG 같은 방식으로 소스로부터 직접 압축되어 온 전체 그림이죠. 가장 화질도 좋지만 가장 용량도 큽니다.  
  
  
  
P 프레임 - Previous 또는 Predicted Frame 이라 불리며, 이전에 나온 키 프레임의 정보를 바탕으로 구성된 프레임 입니다. 화질/용량 둘 다 중간급입니다.  
  
  
  
B 프레임 - Bidirectional Frame 의 약자로, 전후의 I/P 프레임의 정보를 바탕으로 구성된 프레임 입니다. 화질/용량이 다 최하급입니다.

<http://egloos.zum.com/shiyoul/v/4360145>

모션벡터





Process

