DCT란 이산 코사인 변환 [離散-變換, discrete cosine transform, DCT]

직교 변환 부호화 방식의 하나. 1988년 ITU-T에서 원격 화상 회의 전화용 부호화 방식 H.261의 화상 압축 기술로서 채택되었으며 1989년부터 이산 코사인 변환(DCT) 전용 대규모 집적 회로(LSI)가 등장하여 널리 보급되었다. 고속 푸리에 변환(FFT)과 마찬가지로 시간축의 화상 신호를 주파수축으로 변환하는데, 변환을 위한 계수로서 이산적 코사인 함수를 사용하기 때문에 이산 코사인 변환이라고 부르게 되었다. 입력 신호의 성질에 따라 최적화한 변환에는 카루넨 루베 변환(KLT)이 있으나, 영상 신호의 경우에는 DCT가 KLT에 가까운 높은 부호화 능률을 얻을 수 있는 방법이다. DCT에서는 시간축의 화상 신호를 몇 개의 신호 전력이 큰 주파수 영역과 작은 영역으로 분해하여 변환하는데, 화상 신호의 전력은 저주파수 영역에 집중되어 있기 때문에 적절한 비트 배분으로 양자화하면 전체의 비트 수를 적게 하여 데이터를 압축할 수 있다. 그러면서도 재생 영상의 열화는 심하지 않기 때문에, 동화상 압축의 국제 표준인 엠페그(MPEG)에 채용되었으며 현재는 영상의 고능률 부호화와 압축 기술의 주류가 되었다.

> 직교 변환 부호화 [直交變換符號化, orthogonal transform coding]
> 영상 신호나 음성 신호의 고능률 부호화 방식의 하나. 입력 신호를 적절한 블록으로 분할하여 각 블록에 직교 변환을 한다. 변환된 신호 성분의 전력 크기에 따라 다른 비트 수를 할당하여 양자화함으로써 전체 비트 수를 감축하여 데이터를 압축하는 방식이다. 영상 신호의 전력은 저주파수 성분에 집중되어 있기 때문에 적절한 비트 배분에 의해 양자화하면 전체의 비트 수를 적게 할 수 있다. 그래서 직교 변환 부호화는 주로 화상의 고능률 부호화·압축 방식으로 연구되고 있다. 직교 변환 방식 중에서 많이 사용되는 방식은 고속 푸리에 변환(FFT), 이산 코사인 변환(DCT), 카루넨 루베 변환(KLT), 아다마르 변환(Hadamard transform), 경사 변환(slant transform) 등이다

(6) DCT 압축기술

1974년은 오늘날 멀티미디어 혁명을 가능케 한 기념비적인 발명이 있던 해이다. 미 텍사스대학의 라오 교수를 비롯한 3명의 연구진이 이산여현변환 (DC T: Discrete Cosine Transform)이라는 새로운 직교변환에 관한 논문을 IEEE학술지에 발표했던 것이다. 이 DCT는 특히 영상의 압축에 탁월한 성능을 갖는 것으로 오늘날 멀티미디어 관련 국제표준인 H.261, JPEG, MPEG의 핵심요소로 자리잡고 있다.

문자, 도형, 일반 데이터 등을 무손실 압축하면 완전 복구가 가능하지만 압축률은 평균적으로 2대1정도이다. 반면 영상 음성 음향 등의 데이터를 인간의 눈과 귀가 거의 느끼지 못할 정도로 작은 손실을 허용하면서 압축하면 10 대1이상의 압축률을 쉽게 얻을 수 있다.

동영상의 경우 화면간 중복성과 화면내 화소간 중복성이 많아 시각 특성을 잘 활용하면 MPEG영상 압축에서 볼 수 있듯이 30대1이상의 압축을 쉽게 얻을 수 있다. 정지영상은 화면내 화소의 중복성만이 있고, 한 화면이므로 화면간 중복성은 없어 JPEG에서 보듯이

MPEG보다는 다소 압축률이 낮다. 영상이 중복성이 높은 3차원(동영상) 혹은 2차원(정지영상) 데이터여서 압축도 크게 되는데 비해 음성과 음향은 중복성이 상대적으로 떨어지는 1차원 데이터여서 압축률도 영상에 비해 크게 떨어진다. 북미 이동통신용의 음성 압축방식인 VSELP에서는 8대1정도의 압축률이 얻어지고, 돌비 AC-3이나 MPEG 음향 압축에 있어서는 단일 채널의 경우 6대1, 채널간 중복성이 높은 스테레오나 다채널(예:극장영화 감상시의 5.1채널)의 경우 10대1정도의 압축이 얻어진다. 영상데이터를 효과적으로 압축하기 위한 목적으로 가장 널리 쓰이는 손실부호화 기법은 변환부호화이다. 이 방식의 기본구조는 공간적으로 높은 상관도를 가지면서 배열되어있는 데이터를 직교변환에 의하여 저주파 성분으로부터 고주파 성분에 이르기까지 여러 주파수 성분으로 나누어 성분별로 달리 양자화하는 것이다.

이때 각 주파수 성분간에는 상관도가 거의 없어지고 신호의 에너지가 저주파 쪽에 집중된다. 단순 PCM에 비해 같은 비트율에서 얻는 변환부호화의 이득은 각 주파수 성분의 분산치의 산술평균과 기하평균의 비와 같다. 즉 저주파쪽 으로 에너지의 집중이 심화될수록 압축효율이 높다.

공간상의 데이터에 대한 단순 PCM은 모든 표본을 같은 길이(예:m비트/표본) 의 비트로 표현하며 신호대 양자화 잡음비는 약 6m가 된다. 반면 직교변환에 의해 주파수 영역으로 바뀐 데이터는 에너지가 많이 모이는(즉 분산치가큰) 주파수 성분이 보다 많은 비트를 할당받아 그 주파수 성분을 보다 충실히 표현하도록 하고 있다. 분산치가 4배(즉 진폭이 2배) 될 때마다 1비트씩 더 할당받는데 이렇게 되면 모든 주파수 성분에서 동일한 양자화 에러 특성을 갖게 된다.

여러가지의 직교변환 가운데 이론적으로 영상신호의 에너지 집중특성이 가장 뛰어나 압축에 가장 효과적인 것은 카루넨-뢰브 변환(KLT)이다. 그러나 이것은 영상에 따라 변환함수가 새로 정의되어야 하므로 현실적으로 사용할수 없다.

이 KLT에 충분히 가까운 성능을 가지면서 구현 가능한 변환을 찾는것이 라오 교수팀의 목표였고 그 결과가 바로 앞에 말한 DCT이다.

현재 여러 국제표준에 핵심기술로 자리잡고 있는 DCT는 8×8크기의 화소를 하나의 블록으로 묶어 변환의 단위로 삼고 있다. 블록의 크기를 키울수록 압축효율은 높아지나 변환의 구현이 훨씬 어려워진다. 실험적으로 8×8이 성능과 구현의 용이성간 타협점으로 선택되었다.

DCT 변환계수의 양자화는 스칼라 양자화(SQ)와 벡터 양자화(VQ)가 가능하다. VQ는 보통 계수간 상관도가 높을 때 효과적이고 대신 SQ보다는 복잡도가 높다. DCT계수들끼리는 이미 상관도가 거의 없어 현재 국제표준 에서는 SQ를 채택하고 있다. 또 SQ도 다시 구현이 용이한 선형과 특성이 좋은 비선형 기법으로 나뉘는데 양자화된 계수가 다시 엔트로피 부호화(무손실)를 거치면 두 기법간 성능의 차가 작아진다. 현재 국제표준에서는 엔트로피 부호화가 뒤따르고 있어 H.261, JPEG, MPEG-1에서는 선형 기법만을 사용하였다. 그러나

MPEG-2에서는 약간의 성능개선을 위해 비선형 기법도 함께 채택했다.

또한 양자화된 DCT계수들의 통계적 특성을 이용한 무손실 압축을 위해 현재 국제표준에서는 런길이 부호화와 허프만 부호화를 결합하여 사용하고 있다. 영상의 압축은 이렇게 DCT, 양자화, 런길이 부호화, 허프만 부호화, 움직임보상 DPCM(동영상의 경우만 해당) 등 많은 기술이 결합되어 이루어지고 있다.

Transform Coding

- The goal of transform
  - To decorrelate the original signal
  - To redistribute among a small set of transform coefficients
- Image Transform

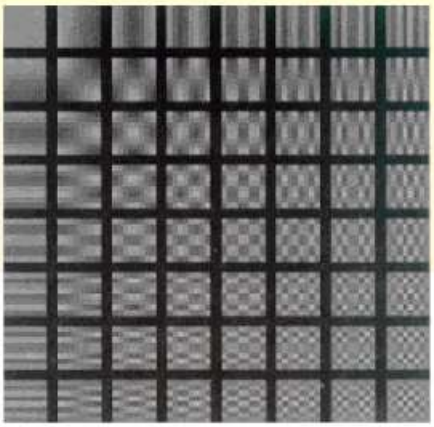
$$f(x,y) \rightarrow ?$$

An alternative mathematical representation of an image

  - Fourier transform
  - Discrete cosine transform
  - Walsh-Hadamard transform
  - Wavelet transform

DCT

- DCT summary
  - To decompose each input block into a series of waveforms, each with a particular spatial frequency
  - Most of the visually significant information is concentrated in just a few coefficients
  - Used in image/video compression



❖ For 8-by-8 DCT, the 64 basis functions

DCT(cont.)

- The Two-dimensional M by N DCT

$$0 \le p \le M-1$$

$$0 \le q \le N-1$$

$$B_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{mn} \cos\frac{\pi(2m+1)p}{2M} \cos\frac{\pi(2n+1)q}{2N}$$

$$\alpha_p = \begin{cases} 1/\sqrt{M}, & p=0 \\ \sqrt{2/M}, & 1 \le p \le M-1 \end{cases} \qquad \alpha_q = \begin{cases} 1/\sqrt{N}, & q=0 \\ \sqrt{2/N}, & 1 \le q \le N-1 \end{cases}$$

DCT(cont.)

- The Two-dimensional M by N IDCT

$$0 \le m \le M-1$$

$$0 \le m \le N-1$$

$$A_{mn} = \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} \alpha_p \alpha_q B_{pq} \cos\frac{\pi(2m+1)p}{2M} \cos\frac{\pi(2n+1)q}{2N}$$

$$\alpha_p = \begin{cases} 1/\sqrt{M}, & p=0 \\ \sqrt{2/M}, & 1 \le p \le M-1 \end{cases} \qquad \alpha_q = \begin{cases} 1/\sqrt{N}, & q=0 \\ \sqrt{2/N}, & 1 \le q \le N-1 \end{cases}$$

VISCOM

DCT(cont.)

- Example

Input

168	161	161	150	154	168	164	154
171	154	161	150	157	171	150	164
171	168	147	164	164	161	143	154
164	171	154	161	157	157	147	132
161	161	157	154	143	161	154	132
164	161	161	154	150	157	154	140
161	168	157	154	161	140	140	132
154	161	157	150	140	132	136	128

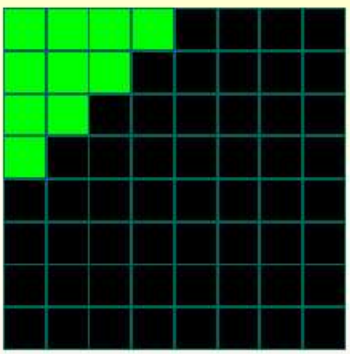
Output

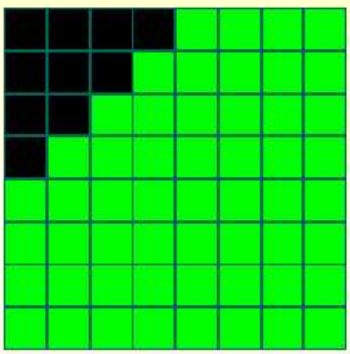
214	49	–3	20	-10	1	1	-6
34	-25	11	13	5	-3	15	-6
-6	-4	8	-9	3	-3	5	10
8	-10	4	4	-15	10	6	6
-12	5	-1	-2	-15	9	-5	-1
5	9	-8	3	4	-7	–14	2
2	-2	3	-1	1	3	-3	-4
-1	1	0	2	3	-2	-4	-2

VISCOM

DCT(cont.)

- Low pass filtering
- High pass filtering

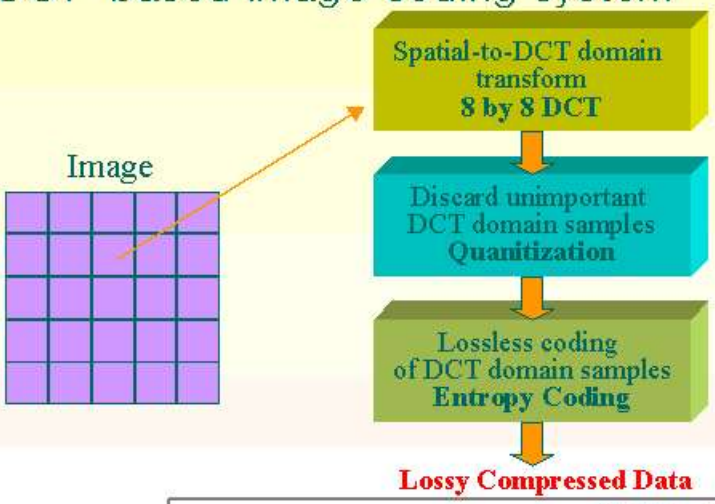




영상통신연구실

DCT(cont.)

- DCT-based image coding system



Program

- 블럭단위의 DCT함수 호출

```
for (y=0; y<m_nHeight ; y+=N) {
   for (x=0; x<m_nWidth ; x+=N) {
     for (l=0; l<N; l++) {
       for (k=0; k<N; k++) temp[k]=ptrSource[y+l][x+k];
         dct(temp);
         for (k=0; k<N; k++) z[l][k]=temp[k];
       }
     for (k=0; k<N; k++) {
       for (l=0; l<N; l++)    temp[l]=z[l][k];
           dct(temp);
           for (l=0; l<N; l++) z[l][k]=temp[l];
       }
```

행 방향 1차원 DCT

열 방향 1차원 DCT

VISCOM

Program(cont.)

DCT 함수

```
void CDCTfilter::dct(float temp[])
{
  int i, j;
  float y[N]={0,};

  for(i=0; i<N; i++){
    for(j=0; j<N; j++)
      y[i] += temp[j]*cos(((2*j+1)*i*M_PI)/(2*N));
    if(i==0) y[i] = (1/sqrt(N))*y[i];
    else  y[i] = (sqrt(2)/sqrt(N))*y[i];
  }
  for(i=0; i<N; i++)  temp[i]=y[i];
}
```

1차원 DCT

VISCOM

Program(cont.)

Filtering

```
for (l=0; l<N; l++) {
  for (k=0; k<N; k++) {
    if (l>N/4 || k>N/4)
      z[l][k]=0;
  }
}

for (l=0; l<N; l++) {
  for (k=0; k<N; k++) {
    if (l<N/4 || k<N/4)
      z[l][k]=0;
  }
}
```

Low Pass Filtering

High Pass Filtering

영상통신연구실

VISCOM

Program(cont.)

- 블럭단위의 IDCT함수 호출

```
for (k=0; k<N; k++) {
        for (l=0; l<N; l++) temp[l]=z[l][k];
        idct(temp);
        for (l=0; l<N; l++) z[l][k]=temp[l];
}

for (l=0; l<N; l++) {
        for (k=0; k<N; k++) temp[k]=z[l][k];
        idct(temp);
        for (k=0; k<N; k++) {
                p = temp[k];
                p >> ptrDest[y+l][x+k];
        }
}
```

열 방향 1차원 IDCT

행 방향 1차원 IDCT

VISCOM

Program(cont.)

- IDCT 함수

```
void CDCTfilter::idct(float temp[])
{
  int i, j;
  float k[N]={0,}, c;

  for(i=0; i<N; i++)
    for(j=0; j<N; j++){
      if(j==0) c = 1/sqrt(N);
      else c = sqrt(2)/sqrt(N);
      k[i] += c*temp[j]*cos(((2*i+1)*j*M_PI)/(2*N));
    }
  for(i=0; i<N; i++) temp[i]=k[i];
}
```

1차원IDCT

VISCOM

Program(cont.)

Color mode

```
for (y=0; y<m_nHeight ; y+=N) {
  for (x=0; x<m_nWidth ; x+=N) {
    for (l=0;l<N; l++) {
      for (k=0; k<N; k++){
        temp[k]=ptrSource[y+l][x+k];
        tempr[k] = temp[k].R;
        tempg[k] = temp[k].G;
        tempb[k] = temp[k].B;
      }
      dct(tempr);
      dct(tempg);
      dct(tempb);
      for (k=0; k<N; k++){
        zr[l][k]=tempr[k];
        zg[l][k]=tempg[k];
        zb[l][k]=tempb[k];
      }
    }
```

영상을 R,G,B 값으로 나누어서 읽어 온다

R,G,B 각각 DCT 함수 호출

R,G,B 각각의 DCT계수를 따로 저장

VISCOM

Program(cont.)

User Interface

2 by 2, 4 by 4, 8 by 8, 16 by 16, 32 by 32 block size의 Low pass/High pass filter 구현

VISCOM

Test

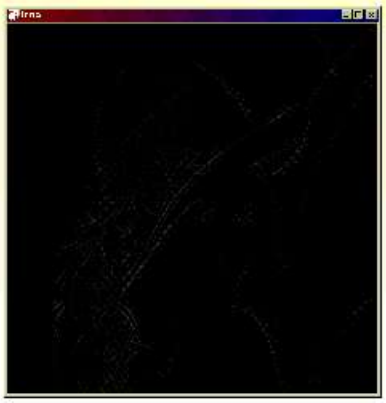
- 원영상(Gray)
- 원영상(Color)

영상통신연구실

VISCOM

Test(cont.)

- 8 by 8 gray DCT low pass filtering
- 8 by 8 gray DCT high pass filtering



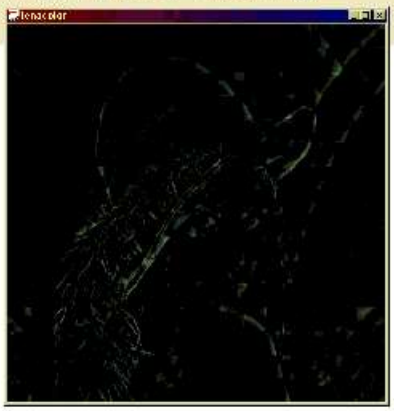
영상통신연구실

VISCOM

Test(cont.)

- 32 by 32 color DCT low pass filtering

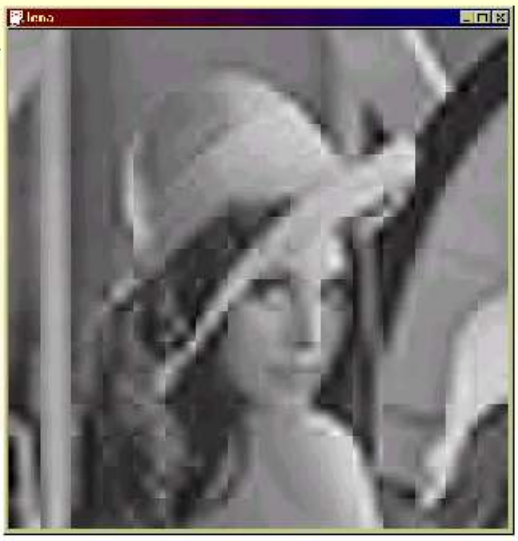
- 32 by 32 color DCT High pass filtering



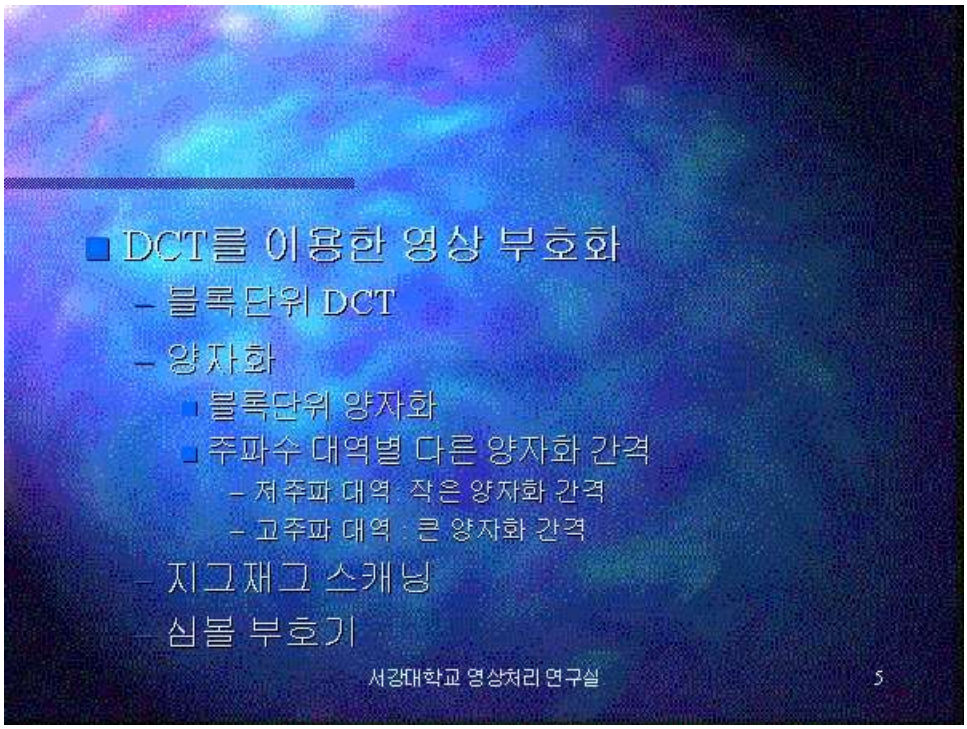
VISCOM

Test(cont.)

32 by 32 low pass filtering, 3개의 최저주파 대역으로만 복원, 심한 영상왜곡



영상통신연구실



Variable Length Coding-DCT 압축의 마지막 단계

이미 설명된 대로, 효과적인 코딩 또는 다른 항목에서의 데이터 압축은 낮은 entropy를 가진 데 이터에만 적용될 수 있다. 그러므로, 뒤의 모든 데이터 압축 시스템 개념은 데이터를 코딩하기 전 에 낮은 entropy를 갖는 형태로 데이터의 형태(form)가 바뀐다.

DCT 코딩과 양자화는 디지털비 디오 데이터의 entropy를 낮추기 위한 과정이다. DCT 압축 시스템에서의 마지막 과정으로, Variable Length Coding이 데이터 흐름을 효과적으로 부호화하고 데이터 압축을 완성하는데 사 용된다.

Variable Length Coding은 높은 확률로 발생하는 값(레벨)을 짧은 코드로 나타내고 낮은 확률로 발생하는 값(레벨)을 보다 긴 코드로 지정하므로 써 데이터 흐름의 전체 bit수를 감소시키 는 것을 의미한다.

DPCM을 보기로 들어보면, 이것은 entropy의 레벨에 관한 bit의 총량을 감소시킨다. 이전에 언급 된 대로, VLC에서 코드 워드는 각 값이 발생하는 확률의 감소로 인하여 "0","1","01","10"...처럼 간단히 지정될 수 없다. 만일 이와 같은 코드가 사용된다면 디코더는 각 코드가 어디서 시작되는 지를 알 수 없게 된다.

실제로, 코드 워드가 구성되므로 각 코드는 어떤 보다 긴 코드의 첫 번째 부분과 같지 않다.(표.1) 이 방법에서, 어떠한 조합이 전송될 때 디코더는 각 코드워드의 시작을 알 수 있게 된다. 여러개 의 방법이 Variable Length Code를 구성하는데 사용 가능한데, 가장 일반적인

것이 Huffman Code이다.

그림 25는 허프만 코드를 사용하여 데이터 흐름이 어떻게 부호화 되었는가를 보여준 다. 어떤 데이터 흐름의 임의의 확률 0.4, 0.22, 0.20, 0.06, 0.05, 0.04, 0.03으로 각각 0, 1, 2, 3, 4, 5, 6 이라는 7개의 값을 가졌다고 가정해 보자.

허프만 코드를 구성하기 위하여, 먼저 그림 25-(b)에서 처럼 그들이 발생 확률이 감소하는 순서로 이들 값을 배열한다. 그러면 발생 확률이 가장 낮은 2 개의 값(여기서는 "5"와 "6") 을 발견하게 되는데 이것을 그림 25-(b)에서처럼 branch로 연결한다. 그 다음 이들이 발새 알 값의 확률을 계산하고(이것은 각 값의 확률의 합이다.) 이것을 이들의 branch가 묶여지는 곳에 표시한다.

한번 이들 값이 연결되면, 이들을 하나의 값으로 간주하고 (junction A) 이 계산에서 주어진 확률로 발생한다고 하자(이 경우 junction A에서 0.07로 표시된 다.) 그 다음에 낮은 확률을 가진 '4'와 '5'를 볼 수 있는데 같은 방법으로 하여 junction B로 지정 한다.(그림 25-(c))

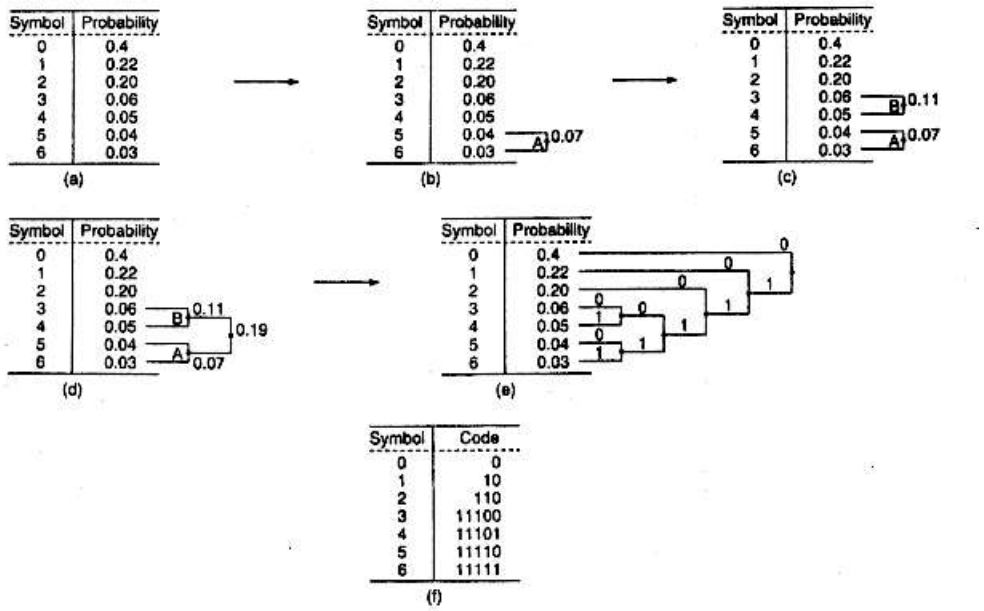


Fig 25. An example of how a Variable Length Code is constructed using the Huffman Code.

그 다음 junction A와 junction B를 연결하여 보면 확률이 값 '2'(0.20)보다 낮음을 알 수 있다.(그 림 25-(d)) 이 과정을 branch가 마지막 값에 연결될 때까지 계속한다.(그림 25-(e)) 코딩을 끝내 기 위해 같은 junction의 두 branch가 반드시 다른 코드를 갖도록 branch에

'0'과 '1'을 지정한다. 그림 25-(e)는 그 보기이다.

Introduction

- Topics
  - Unitary transform
  - Discrete Fourier Transform (DFT)
  - Discrete Cosine Transform (DCT)
  - Discrete Sine Transform (DST)
  - Discrete Walsh Transform (DWT)
  - Discrete Hadamard Transform (DHT)
  - Haar transform
  - Slant transform
  - Karhunen-Loeve(KL) transform

Unitary Transforms

- Unitary Transformation for 1-Dim. Sequence
  - Series representation of $\{u(n), 0 \le n \le N-1\}$

$$\mathbf{v} = A\mathbf{u} \quad \Rightarrow \quad v(k) = \sum_{n=0}^{N-1} a(k,n)u(n), \quad 0 \le k \le N-1$$

$$\mathbf{u} = A^{*}\mathbf{v} \quad \Rightarrow \quad u(n) = \sum_{n=0}^{N-1} a^{*}(k,n)v(k), \quad 0 \le n \le N-1$$

$$where \quad A^{-1} = A^{*T} \quad (unitary \ \ matrix)$$

  - Basis vectors : $\mathbf{a}_{\mathbf{k}}^{*} = \{a^{*}(k,n), 0 \le n \le N-1\}^{T}$
  - Energy conservation :

$$\mathbf{v} = A\mathbf{u} \quad \Rightarrow \quad \|\mathbf{v}\|^2 = \|\mathbf{u}\|^2$$

$$(\because \ \|\mathbf{v}\|^2 = \sum_{k=0}^{N-1} |v(k)|^2 = \mathbf{u}^{*\mathbf{T}} A^{*T} A\mathbf{u} = \mathbf{u}^{*\mathbf{T}}\mathbf{u} = \sum_{n=0}^{N-1} |u(n)|^2 = \|\mathbf{u}\|^2)$$

Unitary Transformation for 2-Dim. Sequence

- Definition :

$$v(k,l)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}u(m,n)a_{k,l}(m,n),\quad 0\le k,l\le N-1$$

$$u(m,n)=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}v(k,l)a_{k,l}^{*}(m,n),\quad 0\le m,n\le N-1$$

- Basis images : $\{a_{k,l}^{*}(m,n)\}$
- Orthonormality and completeness properties
  - Orthonormality :

$$\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}a_{k,l}(m,n)a_{k',l'}^{*}(m,n)=\delta(k-k',l-l')$$

  - Completeness :

$$\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}a_{k,l}(m,n)a_{k,l}^{*}(m',n')=\delta(m-m',n-n')$$

Unitary Transformation for 2-Dim. Sequence

- Separable Unitary Transforms
  - $a_{k,l}(m,n)=a_k(m)b_l(n)$

$$v(k,l)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}a_k(m)u(m,n)a_l(n)\leftrightarrow V=AUA^{T}$$

$$u(m,n)=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}a_k^{*}(m)v(k,l)a_l^{*}(n)\leftrightarrow U=A^{*T}VA^{*}$$

  - separable transform reduces the number of multiplications and additions from $O(N^4)$ to $O(N^3)$
- Energy conservation

$$\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}|u(m,n)|^2=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}|v(k,l)|^2$$

1-dim. DFT

Definition

$$f(n) = \begin{cases} f(n) & , n = 0, \cdots, N-1 \\ 0 & , otherwise \end{cases}$$

- **Inverse DFT**

$$f(n) = \begin{cases} \frac{1}{N} \sum_{k=0}^{N-1} F(k) W_N^{-kn} & , n = 0, \cdots, N-1 \\ 0 & , otherwise \end{cases}$$

- **Forward DFT**

$$F(k) = \sum_{n=0}^{N-1} f(n) W_N^{kn}, \quad k = 0, \cdots, N-1$$

where $W_N \equiv e^{-j\frac{2\pi}{N}}$

1-dim. DFT (cont.)

DFS and DFT

- **Discrete Fourier Series for periodic signals (DFS)**

$$\tilde{g}(n) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{G}_{DFS}(k) e^{j2\pi \frac{k}{N} n}, \quad \tilde{G}_{DFS}(k) = \sum_{n=0}^{N-1} \tilde{g}(n) e^{-j2\pi \frac{k}{N} n}$$

where $\tilde{g}(n) = \sum_{k=-\infty}^{\infty} g(n - kN)$

- **DFT for one period of periodic signals**

$$g(n) = \frac{1}{N} \sum_{k=0}^{N-1} G_{DFT}(k) e^{j2\pi \frac{k}{N} n} R_N(n), \quad G_{DFT}(k) = \sum_{n=0}^{N-1} g(n) e^{-j2\pi \frac{k}{N} n}$$

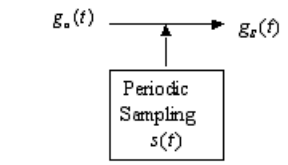
- **DTFT and DFT for one period of periodic signals**

$$g(n) = \int_{-\frac{1}{2}}^{\frac{1}{2}} G_{DTFT}(f) e^{j2\pi f n} df, \quad G_{DTFT}(f) = \sum_{n=-\infty}^{\infty} g(n) e^{-j2\pi f n}$$

$$G_{DFT}(k) = G_{DTFT}(f)\big|_{f = k/N}$$

1-dim. DFT (cont.)

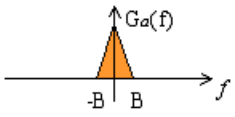
Periodic Sampling , CTFT and DTFT

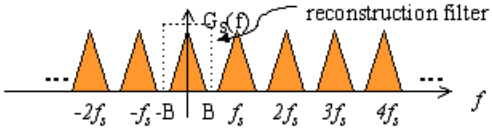


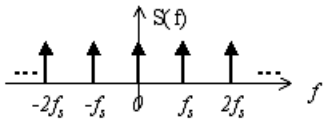
$$g_S(t) = g_a(t) \bullet s(t)$$
$$where \quad s(t) = \sum_m \delta(t - m\Delta t)$$

$\Downarrow$ (Fourier Transform)

$$G_S(f) = G(f) \otimes S(f)$$



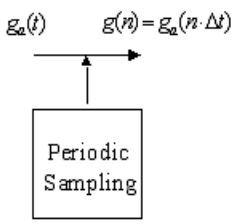




$$G_S(f) = \frac{1}{\Delta t}\sum_{l=-\infty}^{\infty} G_a(f - l \cdot \frac{1}{\Delta t})$$
$$= \Delta f \sum_{l=-\infty}^{\infty} G_a(f - \Delta f \cdot l) \quad (\Delta f = f_S)$$

1-dim. DFT (cont.)

Periodic Sampling , CTFT and DTFT (cont.)



$$\begin{cases} g(n) = \int_{-\frac{1}{2}}^{\frac{1}{2}} G(f) e^{j2\pi fn} df \\ g_a(n\Delta t) = \int_{-\infty}^{\infty} G_a(f) e^{j2\pi fn\Delta t} df \\ \quad = \int_{-\frac{1}{2}}^{\frac{1}{2}} \left[\sum_{l=-\infty}^{\infty} G_a(\frac{f}{\Delta t} - l\frac{1}{\Delta t}) e^{j2\pi fn} \right] df / \Delta t \\ \quad = \int_{-\frac{1}{2}}^{\frac{1}{2}} \left[\sum_{l=-\infty}^{\infty} G_a(\Delta f \cdot (f - l)) e^{j2\pi fn} \right] df / \Delta t \end{cases}$$

$$G(f) = \Delta f \sum_{l=-\infty}^{\infty} G_a(\Delta f \cdot f - \Delta f \cdot l))$$

$$(cf) \quad G_S(f) = \Delta f \sum_{l=-\infty}^{\infty} G_a(f - \Delta f \cdot l)$$

1-dim. DFT (cont.)

Calculation of DFT : Fast Fourier Transform Algorithm (FFT)

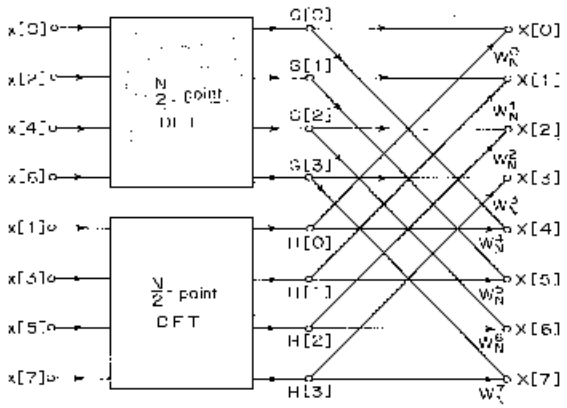
- **Decimation-in-time algorithm**

$$
\begin{aligned}
G(k) &= \sum_{n=0}^{N-1} g(n) e^{-j2\pi \frac{kn}{N}} \\
&= \sum_{n=0}^{N/2-1} g(2n) e^{-j2\pi \frac{k(2n)}{N}} + \sum_{n=0}^{N/2-1} g(2n+1) e^{-j2\pi \frac{k(2n+1)}{N}} \\
&= \sum_{n=0}^{N/2-1} g(2n) e^{-j2\pi \frac{kn}{N/2}} + \sum_{n=0}^{N/2-1} \left\{ g(2n+1) e^{-j2\pi \frac{k}{N}} \right\} e^{-j2\pi \frac{kn}{N/2}} \\
&= \begin{cases} \sum_{n=0}^{N/2-1} g(2n) e^{-j2\pi \frac{kn}{N/2}} + e^{-j2\pi \frac{k}{N}} \cdot \sum_{n=0}^{N/2-1} g(2n+1) e^{-j2\pi \frac{kn}{N/2}}, & 0 \le k \le N/2 - 1 \\ \sum_{n=0}^{N/2-1} g(2n) e^{-j2\pi \frac{kn}{N/2}} - e^{-j2\pi \frac{(k-N/2)}{N}} \cdot \sum_{n=0}^{N/2-1} g(2n+1) e^{-j2\pi \frac{kn}{N/2}}, & N/2 \le k < N \end{cases}
\end{aligned}
$$

1-dim. DFT (cont.)

FFT (cont.)

- **Decimation-in-time algorithm (cont.)**



1-dim. DFT (cont.)

FFT (cont.)

Decimation-in-frequency algorithm (cont.)

$$
\begin{aligned}
G(k) &= \sum_{n=0}^{N-1} g(n) e^{-j2\pi \frac{kn}{N}} \\
&= \sum_{n=0}^{N/2-1} g(2n) e^{-j2\pi \frac{kn}{N}} + \sum_{n=N/2}^{N-1} g(n) e^{-j2\pi \frac{kn}{N}} \\
&= \sum_{n=0}^{N/2-1} \left\{ g(n) + g(n+N/2) e^{-j2\pi \frac{k}{N}\frac{N}{2}} \right\} e^{-j2\pi \frac{kn}{N}} \\
&= \sum_{n=0}^{N/2-1} \left\{ g(n) + g(n+N/2)(-1)^k \right\} e^{-j2\pi \frac{kn}{N}} \\
&= \begin{cases} \sum_{n=0}^{N/2-1} \left\{ g(n) + g(n+N/2) \right\} e^{-j2\pi \frac{k'n}{N/2}}, & k = 2k' \\ \sum_{n=0}^{N/2-1} \left\{ g(n) - g(n+N/2) \right\} e^{-j2\pi \frac{k'n}{N/2}} \cdot e^{-j2\pi \frac{n}{N}}, & k = 2k'+1 \end{cases}
\end{aligned}
$$

1-dim. DFT (cont.)

FFT (cont.)

Decimation-in-frequency algorithm (cont.)

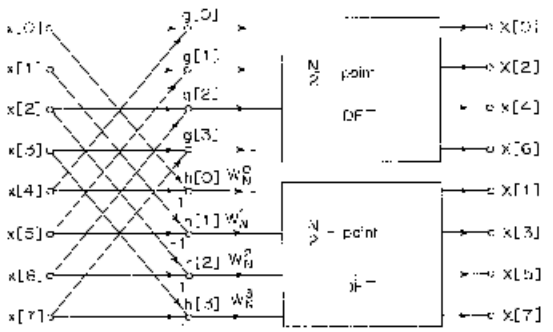


Figure 9.17 Flow graph of decimation-in-frequency decomposition of an N-point DFT computation into two (N/2)-point DFT computations (N = 8).

2-Dim. DFT

Definition

$$f(m,n)=\begin{cases} f(m,n) & ,\ 0\le m,n\le N-1 \\ 0 & ,\ otherwise \end{cases}$$

- **Inverse DFT**

$$f(m,n)=\frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}F(k,l)W_N^{-km}W_N^{-ln},\quad 0\le m,n\le N-1$$

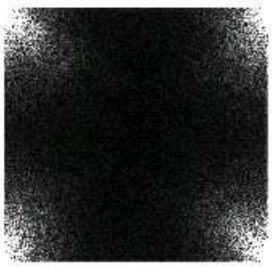
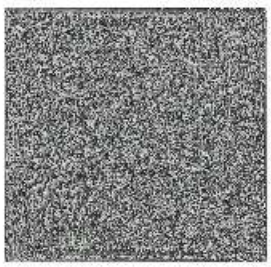
$$where\ \ W_N\equiv e^{-j\frac{2\pi}{N}}$$

- **Forward DFT**

$$F(k,l)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}f(m,n)W_N^{km}W_N^{ln},\quad 0\le k,l\le N-1$$

2-Dim. DFT (cont.)

example

(a) Original Image (b) Magnitude (c) Phase

2 – dim DFT of a 512 × 512 Lena image

2-Dim. DFT (cont.)

Properties of 2D DFT

- **Separability**

$$F(k,l) = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} W_N^{km} \left(\frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f(m,n) W_N^{ln} \right), \quad k,l = 0, \cdots, N-1$$

$$f(m,n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} W_N^{-km} \left(\frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} F(k,l) W_N^{-ln} \right), \quad m,n = 0, \cdots, N-1$$

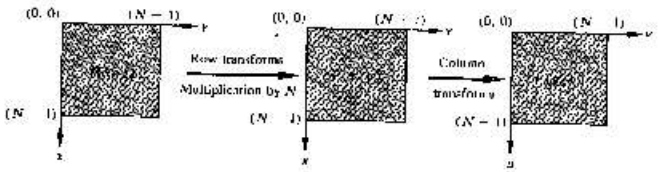


Figure 3.7 Computation of the 2-D Fourier transform as a series of 1-D transforms.

2-Dim. DFT (cont.)

Properties of 2D DFT

- **Translation**

$$f(m,n) \exp\left[j \frac{2\pi}{N} k_0 m \right] \exp\left[j \frac{2\pi}{N} l_0 n \right] \Leftrightarrow F(k-k_0, l-l_0)$$

$$f(m-m_0, n-n_0) \Leftrightarrow F(k,l) \exp\left[-j \frac{2\pi}{N} k m_0 \right] \exp\left[-j \frac{2\pi}{N} l n_0 \right]$$

- **Conjugate symmetry**

  For real $f(m,n)$

$$F(k,l) = F^*(N-k, N-l)$$
$$|F(k,l)| = |F(N-k, N-l)|$$

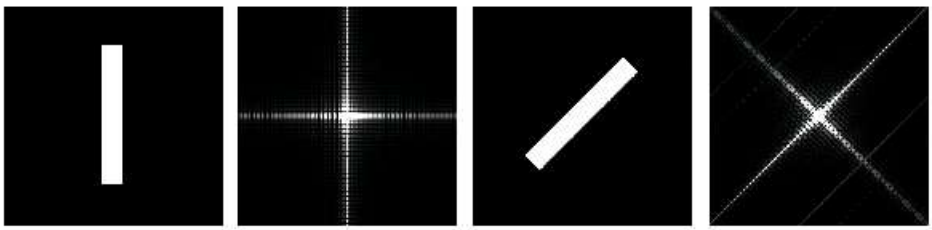
2-Dim. DFT (cont.)

Properties of 2D DFT (cont.)

- **Rotation**

$$m = r\cos\theta,\ n = r\sin\theta, \quad k = \omega\cos\phi,\ l = \omega\sin\phi$$

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \phi + \theta_0)$$



(a) a sample image (b) its spectrum (c) rotated image (d) resulting spectrum

2-Dim. DFT (cont.)

Properties of 2D DFT

- **Circular convolution and DFT**

$$f(m,n) * g(m,n) = \sum_p \sum_q f(p,q) g_C(m-p, n-q)$$

$$f(m,n) * g(m,n) \Leftrightarrow F(k,l)G(k,l)$$

$$f(m,n)g(m,n) \Leftrightarrow F(k,l) * G(k,l)$$

- **Correlation**

$$R_{fg}(m,n) = f(m,n) \circ g(m,n) = \sum_p \sum_q f^*(p,q) g_C(m+p, n+q)$$

$$f(m,n) \circ g(m,n) \Leftrightarrow F^*(k,l)G(k,l)$$

$$f^*(m,n)g(m,n) \Leftrightarrow F(k,l) \circ G(k,l)$$

2-Dim. DFT (cont.)

Calculation of 2-dim. DFT

- **Direct calculation**

$$F(k,l)=\sum_{m=0}^{N_1-1}\sum_{n=0}^{N_2-1} f(m,n)e^{-j\left(\frac{2\pi}{N}\right)km}e^{-j\left(\frac{2\pi}{N}\right)ln}, \quad \begin{matrix}0\le k\le N_1-1,\\ 0\le l\le N_2-1\end{matrix}$$

  - Complex multiplications & additions : $o(N_1^2N_2^2)$

- **Using separability**

$$F(k,l)=\sum_{m=0}^{N_1-1}\left(\sum_{n=0}^{N_2-1} f(m,n)e^{-j\left(\frac{2\pi}{N}\right)ln}\right)e^{-j\left(\frac{2\pi}{N}\right)km}, \quad \begin{matrix}0\le k\le N_1-1,\\ 0\le l\le N_2-1\end{matrix}$$

  - Complex multiplications & additions : $o(N_1N_2(N_1+N_2))$

- **Using 1-dim FFT**

  - Complex multiplications & additions : ???

2-dim. DCT

Definition

- **Inverse DCT**

$$f(x,y)=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)C(u,v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

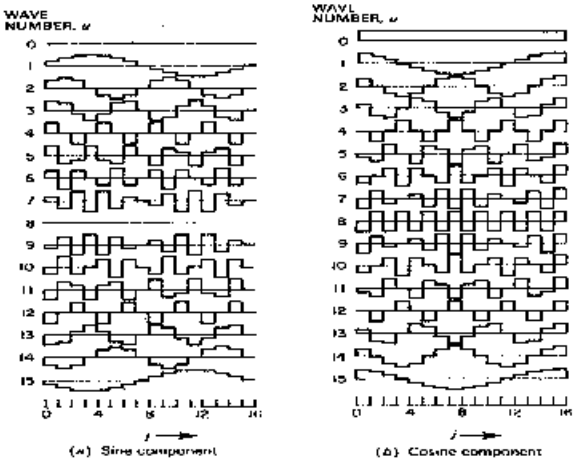
$$\alpha(u)=\begin{cases}\sqrt{\frac{1}{N}} & for \quad u=0\\ \sqrt{\frac{2}{N}} & for \quad u=1,2,\cdots,N-1\end{cases}$$

- **Forward DCT**

$$C(u,v)=\alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

2-dim. DCT (cont.)

- Basis Functions for 1-dim. DCT (N=16)



Fast algorithm of 1-dim. DCT

- $\tilde{g}(x) = g(2x)$

  $\tilde{g}(N-x-1) = g(2x+1) \quad for \quad 0 \le x \le (\frac{N}{2}) - 1$

- $G(u) = \alpha(u)\sum_{x=0}^{N-1} g(x)\cos[\frac{\pi(2x+1)u}{2N}]$

  $= \alpha(u)(\sum_{x=0}^{N/2-1} g(2x)\cos[\frac{\pi(4x+1)u}{2N}] + \sum_{x=0}^{N/2-1} g(2x+1)\cos[\frac{\pi(4x+3)u}{2N}])$

  $= \alpha(u)(\sum_{x=0}^{N/2-1} \tilde{g}(x)\cos[\frac{\pi(4x+1)u}{2N}] + \sum_{x=0}^{N/2-1} \tilde{g}(N-x-1)\cos[\frac{\pi(4x+3)u}{2N}])$

  $= \alpha(u)\sum_{x=0}^{N-1} \tilde{g}(x)\cos[\frac{\pi(4x+1)u}{2N}]$

  $= \mathrm{Re}[\alpha(u)e^{-ju\pi/2N}\sum_{x=0}^{N-1} \tilde{g}(x)e^{-j2\pi ux/N}]$

  $= \mathrm{Re}[\alpha(u)W_{2N}^{u/2}\,DFT\{\tilde{g}(x)\}_N]$

Discrete Sine Transform (DST)

1-dim. DST

Definition

- **Inverse DST**

$$u(n)=\sqrt{\frac{2}{N+1}}\sum_{n=0}^{N-1}v(k)\sin\frac{\pi(k+1)(n+1)}{N+1},\quad 0\le n\le N-1$$

- **Forward DST**

$$v(k)=\sqrt{\frac{2}{N+1}}\sum_{n=0}^{N-1}u(n)\sin\frac{\pi(k+1)(n+1)}{N+1},\quad 0\le k\le N-1$$

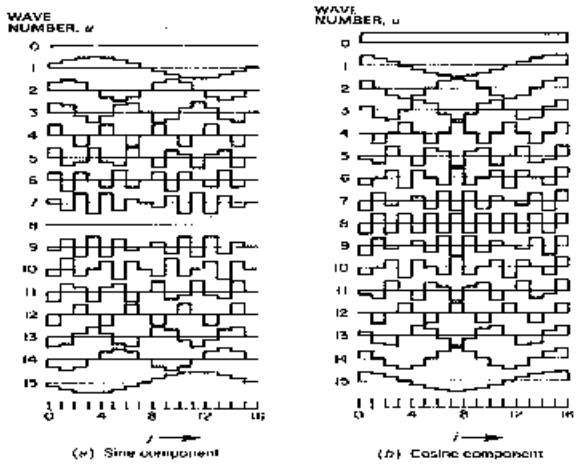
Reference for fast algorithm of DST

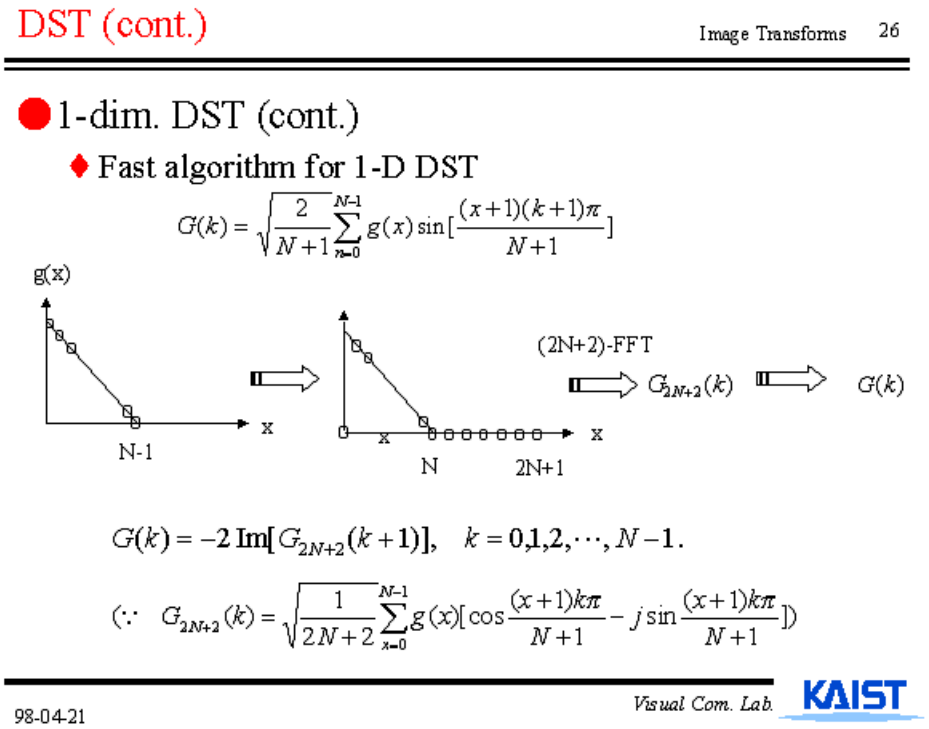
- P.Yip and K. R. Rao, "A Fast Computational Algorithm for the Discrete Sine Transform," IEEE Trans. On Communicatins, Vol. COM-28, No. 2, Feb., 1980

DST (cont.)

1-dim. DST (cont.)

Basis Functions for 1-dim. DST (N=16)





DCT (Discrete Cosinr Transform), 이산 여현 방식

---

1. 개요

o JPEG 손실 부호화 과정의 핵심으로서,

o 사람 눈의 특성이 고주파 성분에는 덜 민감하고, 저주파 성분에 더 민감하다는 사실을 이용함.

2. JPEG 손실 부호화 과정

o 원영상 --> [FDCT 변환] --> [양자화기] --> [엔트로피 부호화] --> 압축된 영상
8x8블럭 ↑ ↑
[양자화 테이블] [부호표]

o FDCT (Forward Discrete Cosine Transform) 정의식

[생략]

o 2차원 DCT 변환 계수 행렬 F(u,v)의 좌상 위치에 있는 계수를 DC 계수라 하고, 공간 블록의 평균 밝기에 비례함.

o 나머지 계수는 AC 계수라 하고, 양자화하면 '0'값이 많이 나옴.

o 양자화기를 거쳐 나온 값을 지그재그 스캔하여 엔트로피 부호화(Run Length Coding, Huffman Coding)로 압축함.

3. DCT 방식의 특징

o 원 신호를 표준 영상 값을 사용하여 저주파 성분과 고주파 성분으로 나누어 줌.

o DCT 변환의 기저 벡터, 즉 변환 행렬 C 는 실수이며 직교 변환임.
* 직교 변환에는 DFT (Discrete Fourier Transform), DCT, 하다마드 변환 등 있음.

o 입력 데이터 성분간의 자기 상관성이 높은 경우에, DCT 방식은 아주 우수한 에너지 집중 (Energy Compaction) 특징을 갖음.

- DCT 변환은 에너지 집중도가 가장 높은 K-L (Karhunen-Loeve) 변환을 대치할 수 있는 실용적인 대안임.
* K-L 변환은 입력 데이터의 통계적 분석이 전제됨.

MPEG2K 강좌 (JPEG)

Baseline System

baseline이란 JPEG의 가장 기초가 되는 구조랍니다. 이것을 바탕으로 여러 MODE로 새로운 기법을 추가하여 만든답니다.그러므로 가장 중요한 부분이죠.

Source : 8 bit , 12 bit 의 소스 이미지를 사용합니다. gray level image의 경우는 그 대로 사용하면 되겠고 Color image의 경우는 각 컬러 성분별로 코딩을 해주죠. Color는

Luminance(밝기)와 Chrominance(색상)으로 나누는데 그 이유는 사람의 눈이 색상에 비해 밝기 정보에 더 민감하기 때문에 압축을 할 때 색상 성분을 밝기 성분보다 적게 포함시키고자 하는 목적 때문이죠.. 움..

Encoding 순서 : Level-shifting -> 2D-DCT -> Zigzag Scan (DC의 경우 DPCM) -> Quantization -> VLC

Level-shifting : 8bit의 이미지의 픽셀이 가지는 값의 범위를 보면 0 ~ 255이겠죠. 이것을 그냥 DCT변환 해 버리면 값의 범위가 무지하게 늘어난답니다. 압축이 목적이다보니 무조건 크면 안좋죠.. 그렇기 때문에 128정도의 shifting을 해주면 값의 범위가 -128 ~ 127 로 변하게 되겠죠.. 이것을 DCT변환 해주면 안했을 때보다 값의 범위가 적어진답니다. 값의 범위가 적어지면 그만큼 압축할 때 적은 Bit만 있으면 됩니다.

2D-DCT : 움 DCT는 다덜 알고 있을테니. 간단히 말하면 이미지의 각 픽셀에서의 값을 보면 전체적으로 Uniform하다고(?)볼 수도 있겠죠.. 그런데 Transform coding (DCT)등을 하면 값이 낮은 주파수 쪽으로 몰리게 되죠.. 이것을 Energy compaction(에너지 집중) 이라고 하는데 이렇게 되면 낮은 주파수부분에 이미지에 대한 정보가 대부분 들어 있기 때문에 고주파 부분은 잘라 버려도 되게 됩니다. 그러면 당연히 보내야 할 비트가 줄어들게 되죠..

Zigzag Scan

100	20	9	1				
20	15	-4	2				
9	-4						
1	2						
1	-1				0	0	0
-1					0	0	0
1					0	0	0

1	2	6	7	15	16		
3	5	8	14				
4	9	3					
10	12						
11					0	0	0
					0	0	0
					0	0	0

움.. 설명하려면 그림이 필요하겠군요.. 위에서 위의 그림은 DCT후의 Block이랍니다. 보시다시피 빨간색 Pixel이 DC가 되고 값이 가장 크게 됩니다. 에너지 집중 현상.. ^^ 글구 이것을 지그 재그 방식으로 읽어온답니다. 이렇게 해야 저주파 부분부터 고주파 부분으로 순서적으로 읽어올 수 있어서 고주파 부분을 없앨 수가 있겠죠. 아래 그림이 그것을 나타냅니다.

DPCM : DC성분은 따로 코딩한답니다. 보다시피 값이 커서리 앞 Block의 DC값과의 차이를 압축해서 보내죠. AC의 경우는 위의 지그재그 방식으로 읽어서 RUN/Level 코딩을 한답니다.

Quantization : 값이 크므로 어떤 값으로 나누어준답니다. 그 값은 JPEG에서 Quantization Table로 미리 정해 놓았죠.. 즉 Out = IN/Q 형식이 되겠죠.

VLC : Runlength 코딩을 하는데 말 그대로 읽어온 것이 순차적으로 표현되어 있기 때문에 0는 빼고 읽는거죠.. 그 대신에 .. 그림적인 설명 또 필요.. ㅠ.ㅠ

(39 -3 2 1 -1 1 0 0 0 0 -1 EOB)

움.. 지그재그 스캔 후의 값이 위와 같으면 39는 DC이므로 따로 코딩하고 -3은 0뒤에 나온 것이 아니므로 (0, -3)이 되며 그 뒤의 숫자도 마찬가지입니다. 0뒤에 나온 -1은 0가 4개 이므로 (4, -1)이 되겠죠. 참.. EOB는 Block의 끝을 나타내는데 0가 끝날 때까지 계속 나오면 그냥 EOB라해서 끝으로 보죠. 이 (R/S)로 표현된 것을 Huffam Code와 Category를 나누어서 압축한답니다. 자세한 건 관련 책들을 보세용..

Decoder : Encoding을 반대로 하면 되겠죠... ^^;;
움.. 설명이 쉬울줄 알았는데 힘들군요.. 빨리 만들려구 하니까 잘 안되네요 나중에 시간 나면 자세하고 정확하게 고치도록 하죠 ^^;;

이산 코사인 변환 (Discrete Cosine Transformation) 데이터 무손실

그림 2는 자동차의 옆의 섹션을 보여주고 있다. 그리고 헤드라이트는 8 x 8 블록으로 되어 있다. 이 블록은 칼라 수치 매트릭스에 표시되고 이것은 이산 코사인 변환 (DCT)을 위해서 사용된다. DCT 는 아주 빈번히 나오는 이미지 부분을 인간의 눈으로는 인식하지 못할 정도로 나누어서 압축을 하게된다. DCT는 어떠한 신호도 다른 주파수의 사인 신호를 더해서 (중첩) 나타낼 수 있는 퓨리에 변환에 기초를 두고 있다. 이 퓨리에 변환은 주파수와 진폭의 분포 값으로 이미지 안에서 픽셀의 값을 만들어내게 된다. 이 이야기는 이미지의 안에서 크고, 일반적인 영역은 좀더 낮은 주파수의 부분으로 표현된다는 이야기이다. 반대로 세세한 부분은 높은 영역을 가지게 된다. 우리의 구체적인 예제에서는, DCT 변환은 보여지는 8 x 8 매크로 블록은 8 x 8 계수 매트릭스로 바뀐다는 것이다. 이 계수 매트릭스의 왼쪽 윗부분의 값들은 가장 낮은 주파수의 부분을 포함하게 된다. 여기, 위치 0,0 의 계수는 일반적으로 DC 계수라고 불려진다. 여기에 남은 63개의 계수들은 AC 계수들이라고 불려진다 (AC = Amplitude Coefficient : 진폭 계수). 일반적으로 두 연속되는 8 x 8 블록들의 DC 계수 사에는 강한 연관성이 있게 마련이다. 그리고 DC 계수들은 다른 전 처리기들에 의해서 인코딩이 된다. 그리고 남은 63개의 AC 계수들은 설정 패턴대로 정렬되게 된다.

DCT는 특히 DC 계수 안에서 가장 낮은 계수의 블록의 시그널 에너지를 집중시킨다. 보다 높은 AC 계수들은 일반적으로 0 이나 거의 0이다. 왜냐하면 이미지의 비쥬얼 정보의 주된 부분은 낮은 주파수 영역 안의 연속적으로 분포하는 범위의 값 안에 존재하기 때문이다. 이산 변환 후에, 계수들은 부가적인 압축의 개선을 이루기 위해서 양자화가 된다.

이미지 처리와 압축

아날로그 이미지는 컴퓨터에서 직접 처리하거나 저장할 수 없음
표본화(Sampling) 및 양자화(Quantization) 과정을 거쳐 디지털 이미지로 변환
(1) 표본화 (Sampling)
아날로그 이미지의 연속적인 위치 데이터를 불연속적인 디지털 데이터로 변환하는 과정

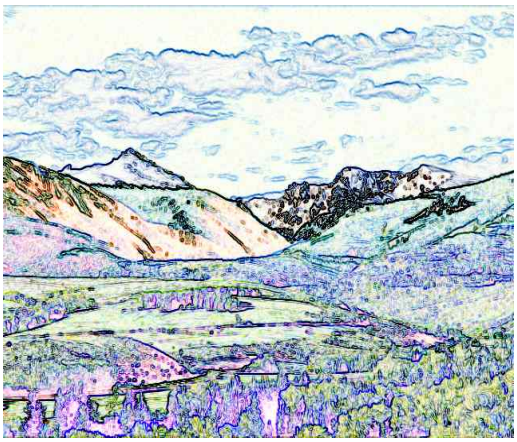
표본화를 거친 이미지

(2) 양자화 (Quantization)
연속적인 색상 데이터를 불연속적인 디지털 데이터로 변환하는 과정
각 화소의 밝기 또는 색을 컴퓨터에서 인지할 수 있는 숫자로 표현하는 과정
표현할 수 있는 색상의 수가 2G일 경우 G비트 양자화라고 하며, 일반적인 흑백 사진의 경우 256레벨(8비트), X선 이미지의 경우 1024레벨(10비트) 정도임

양자화를 거친 이미지

기본 이미지에 임의의 변형을 가하여 특수한 효과를 얻는 기법
특수 효과 뿐만 아니라 잡음이나 왜곡 등 손상된 이미지의 품질을 원상태로 복원시키기도 함
윤곽선 추출(Edge Detection)
이미지의 그레이 레벨(gray level)이 급격하게 변하는 부분을 감지하여 표시하는 필터
Sobel 알고리즘, Kirsch 알고리즘 등 여러 가지 알고리즘이 존재

원본 이미지



윤곽선을 추출한 이미지

윤곽선 추출 필터의 적용

평균값 필터(Average Filter)
이미지의 각 픽셀에서 일정한 주위의 픽셀값의 평균치를 구하여 현재 픽셀값을 대체시키는 필터
잡음이 감소하고 경계선이 흐릿해지는 특징이 있음

원본 이미지

평균값 필터로 처리한 이미지

평균값 필터의 적용

밝기 조절 필터(Brightness Filter)
픽셀의 값을 전체적으로 일정 값만큼 곱하여 밝기를 조절하는 효과를 주는 필터

히스토그램 평준화(Histogram Equalization)
이미지에서 명암도에 따른 픽셀의 수를 고르게 분포시키는 기법
히스토그램 평준화를 수행하면 이미지 히스토그램이 고르게 분산되는 것을 볼 수 있음

이미지 데이터의 양을 줄이는 방법
한 화소당 데이터의 양을 줄이는 방법

이미지를 구성하는 화소의 수를 줄이는 방법
데이터를 압축하는 방법
(1) GIF 압축
RLE(Run Length Encoding) 방식을 응용한 LZW(Lempel-Ziv-Welch) 알고리즘을 사용

A	1	B	8	A	1				

RLE 압축 방식

수평으로 같은 색을 갖는 이미지의 경우 압축 효과가 크다.

154 bytes 213 bytes 318 bytes

501 bytes 1,148 bytes 8,236 bytes
이미지의 특성에 따른 GIF 압축률

(2) JPEG(Joint Photographic Experts Group) 압축
특별히 컬러 사진의 압축을 위하여 고안되었으며, 1992년 국제 표준으로 확정됨
손실(Lossy) 압축은 JPEG에서 일반적으로 쓰이는 방식이며, 무손실(Lossless) 압축은 X-레이 등 픽셀 하나 하나가 중요한 경우 사용
24비트 컬러를 사용하며 압축 특성으로 인한 색번짐이 나타날 수 있음

확대된 GIF 그림 / 425 bytes 확대된 JPG 그림 / 5,219 bytes
JPG에서의 색 번짐

JPEG 압축 과정
RGB모델에서 YIQ모델로 변환
YIQ모델 : Y는 밝기, I는 색상, Q는 순도의 정보를 가짐
인간의 시각은 밝기 정보에 더 민감하게 반응

RGB모델을 YIQ모델로 변환

YIQ의 매크로 블록(Macroblock)화
Y는 16 * 16, I와 Q는 8 * 8의 크기로 나눔
매크로 블록을 8 * 8 블록화
JPEG 압축은 전체 이미지를 8 * 8 픽셀 블록 단위로 나누어 압축을 수행
DCT 변환
2차원 평면 공간의 컬러 정보를 2차원의 주파수 정보로 푸리에 변환(Fourier Transform)하는 과정
1개의 DC 계수와 63개의 AC 계수를 얻음
양자화(Quantization)

인간이 구별하기 힘든 범위 내에서 DCT 계수를 반올림
이 과정에서 인간의 눈이 잘 인식하지 못하는 높은 주파수의 DCT계수들은 거의 0 이 됨
가장 큰 데이터 압축이 일어나는 동시에 가장 데이터 손실이 많은 과정
지그재그 스캐닝(Zig-zag Scanning)
DCT 계수 지그재그로 읽어 일차원 형태로 배열
낮은 주파수의 계수는 앞쪽에, 높은 주파수의 계수는 뒤쪽에 위치

DCT계수 및 지그재그 스캐닝

엔트로피 코딩(Entropy Coding)
무손실 압축을 사용하여 최종 압축을 수행
일반적으로 허프만 코딩(Huffman coding)을 많이 사용