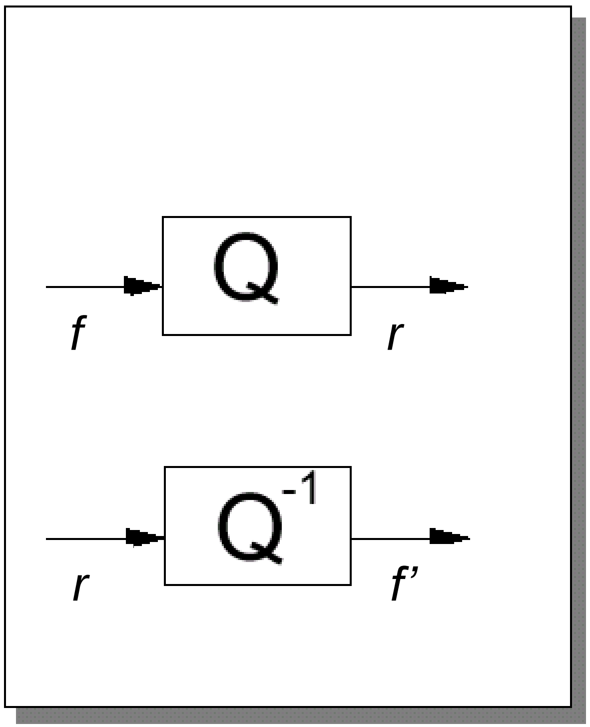
Chapter 4. Lossy Compression

- Quantization
- Discrete Cosine Transform (DCT)
- Motion Compensated Prediction

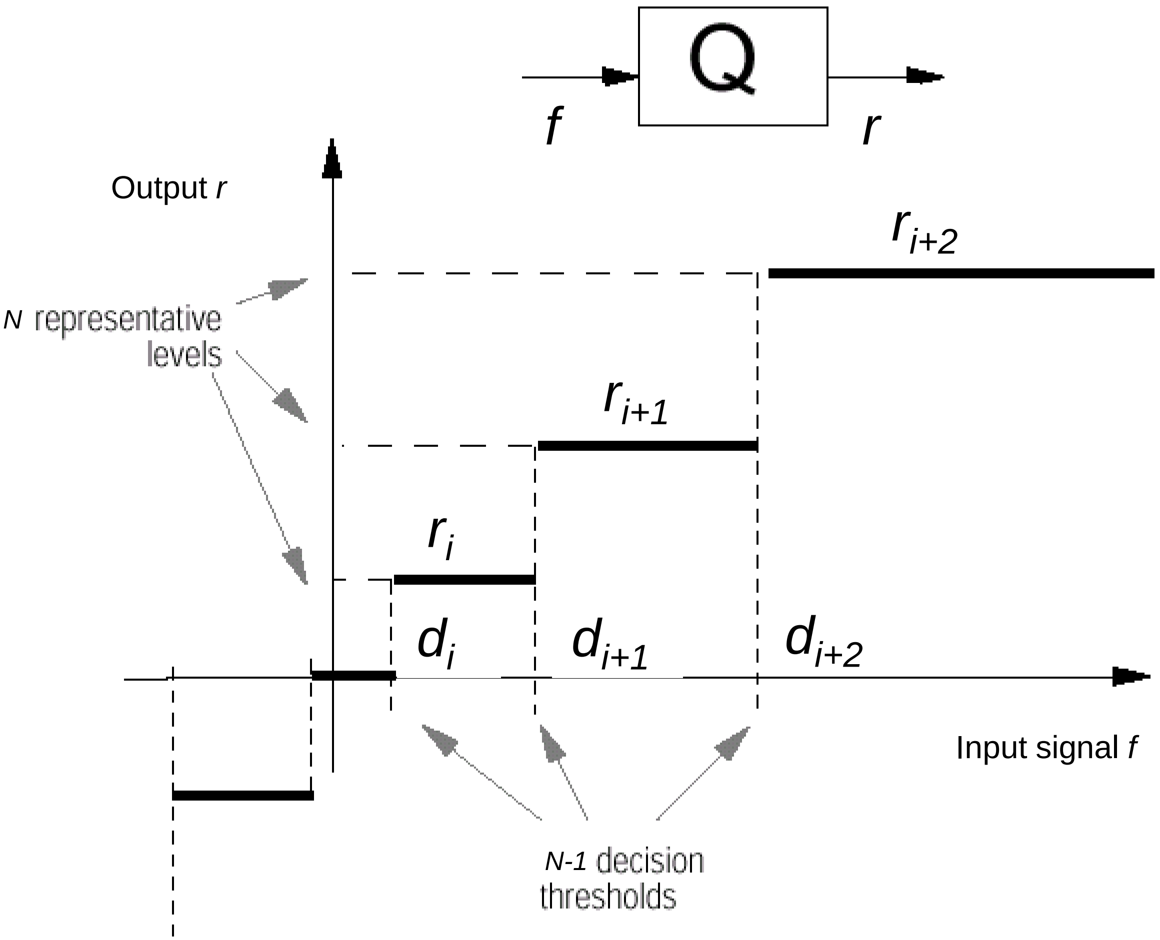
Quantization

Quantization

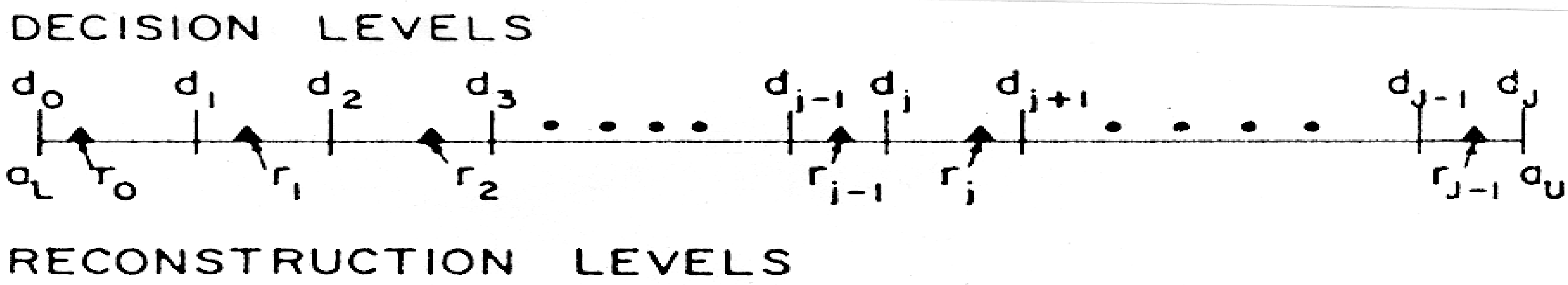
Quantization is the most important means of irrelevancy reduction.



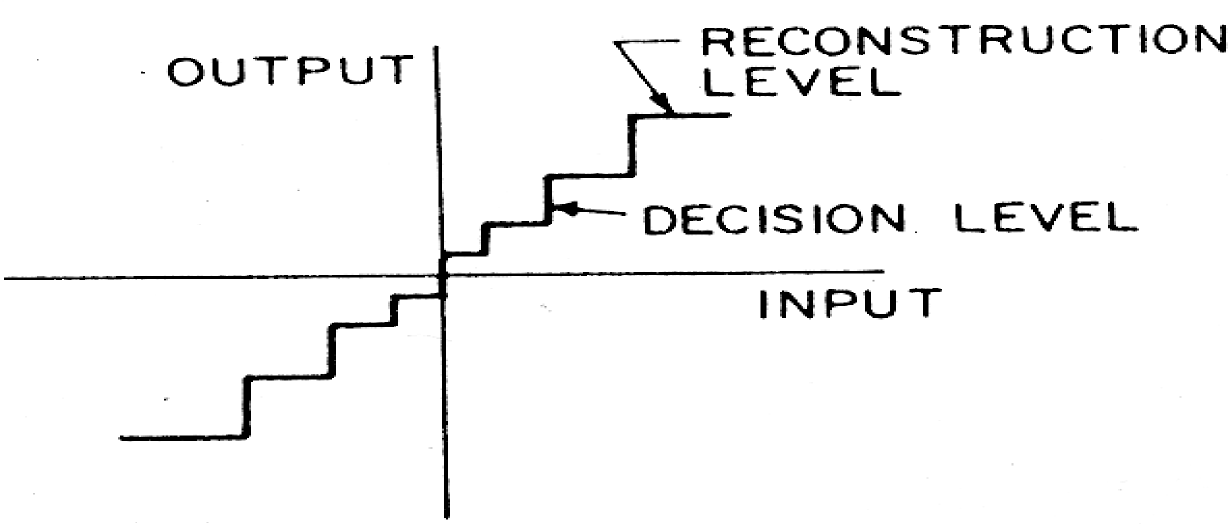
Input-output characteristic of a scalar quantizer



Quantization Levels

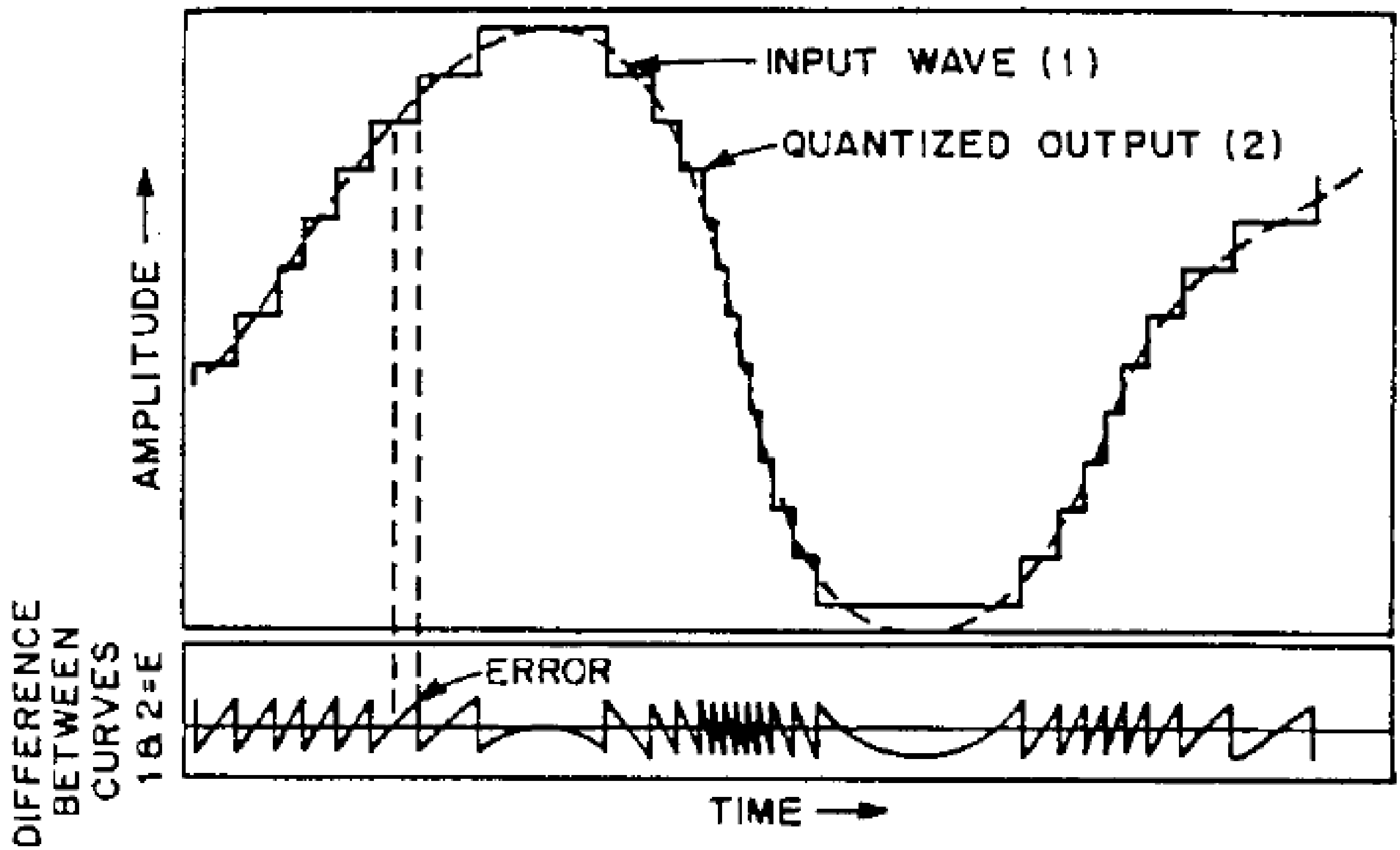


(a) LINE REPRESENTATION

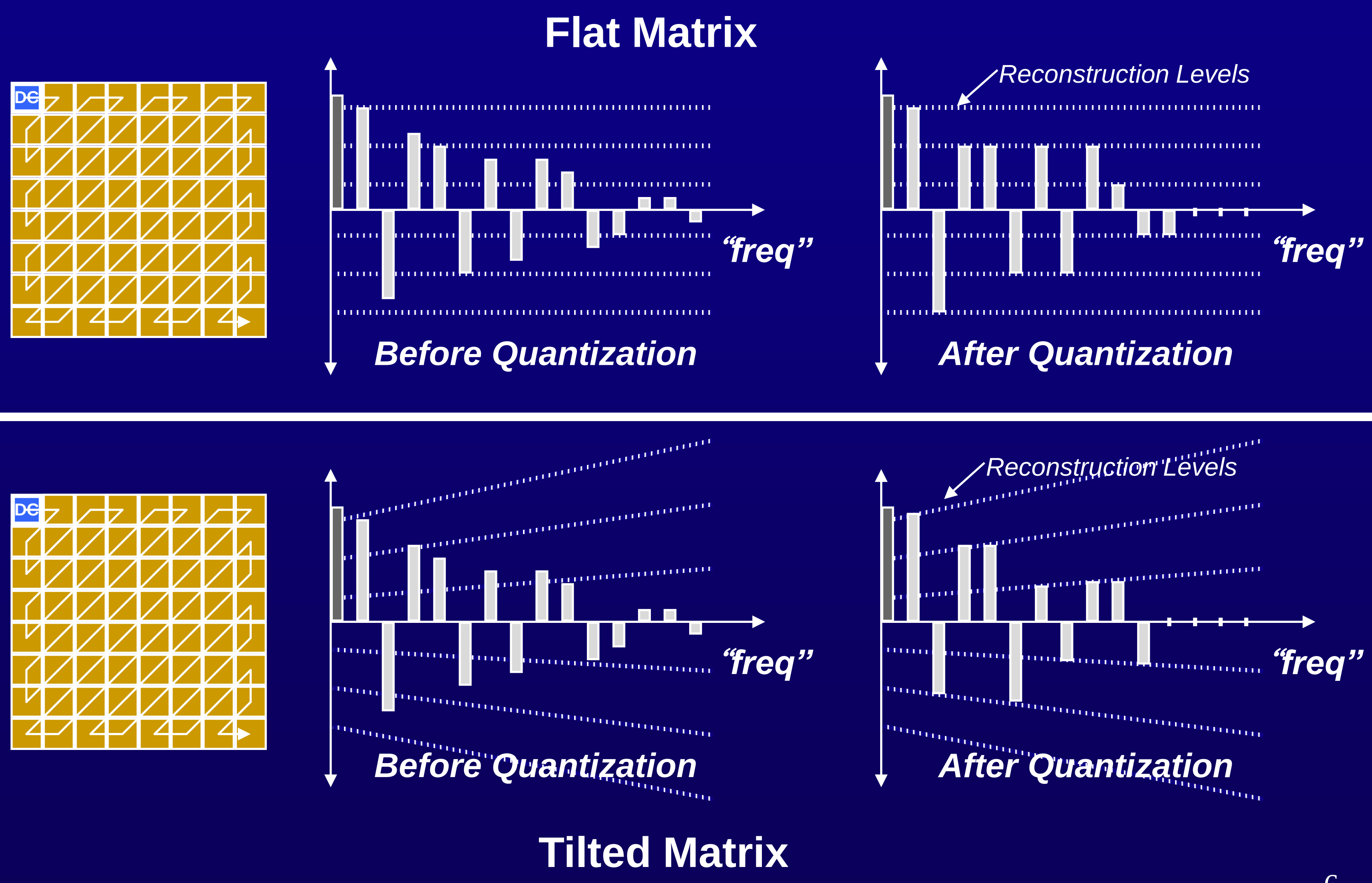


(b) STAIRCASE REPRESENTATION

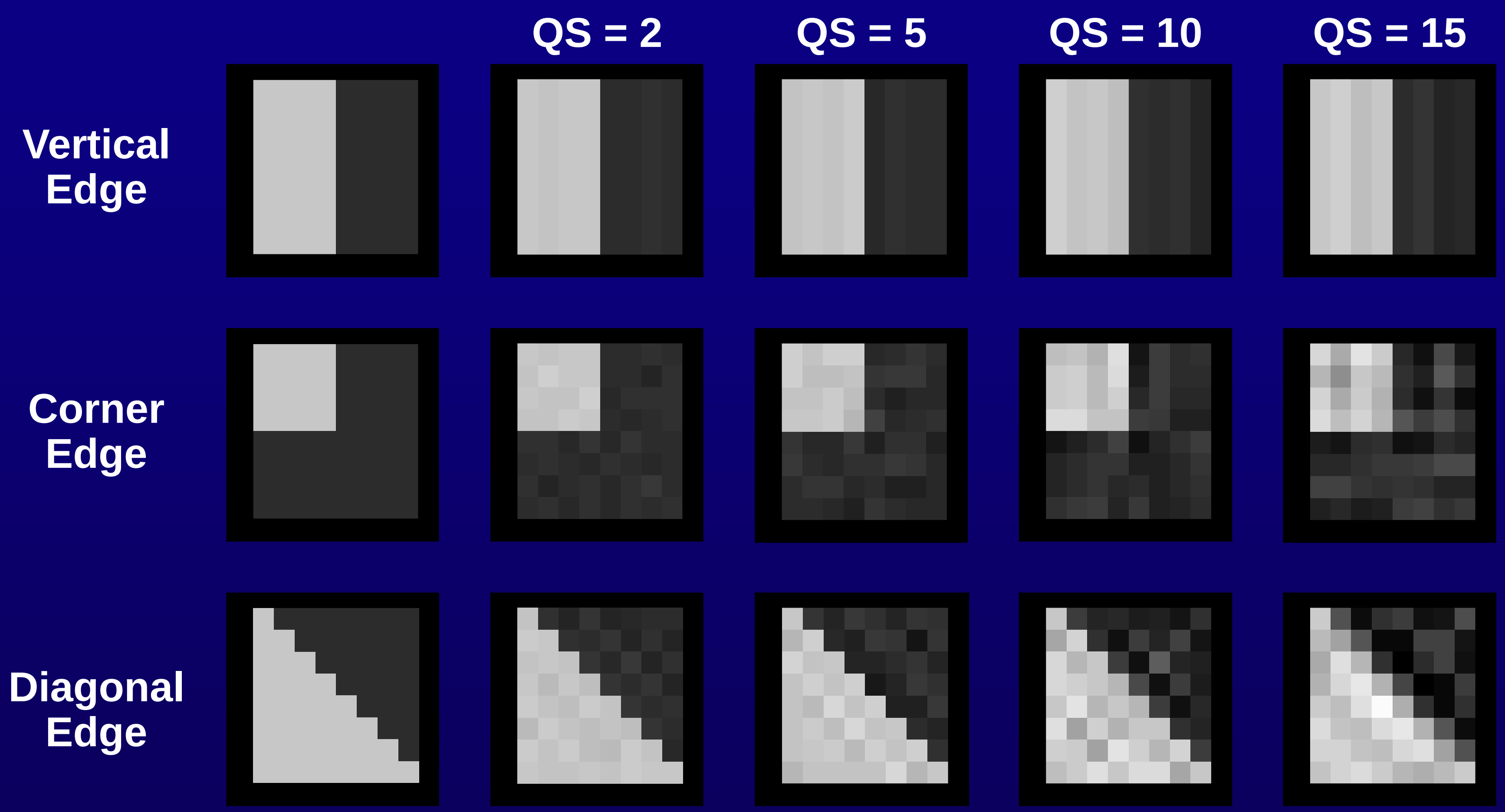
Quantization Errors



Quantization Errors



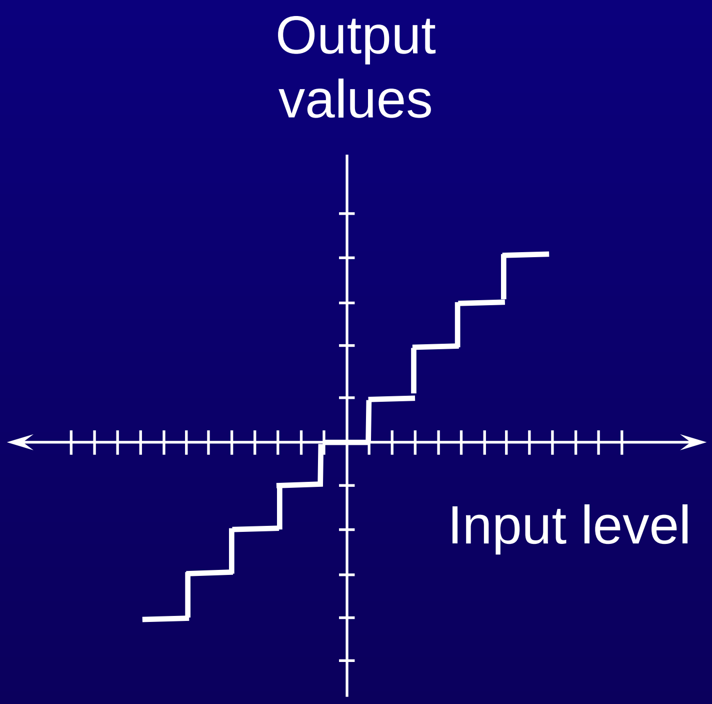
Quantization Artifacts



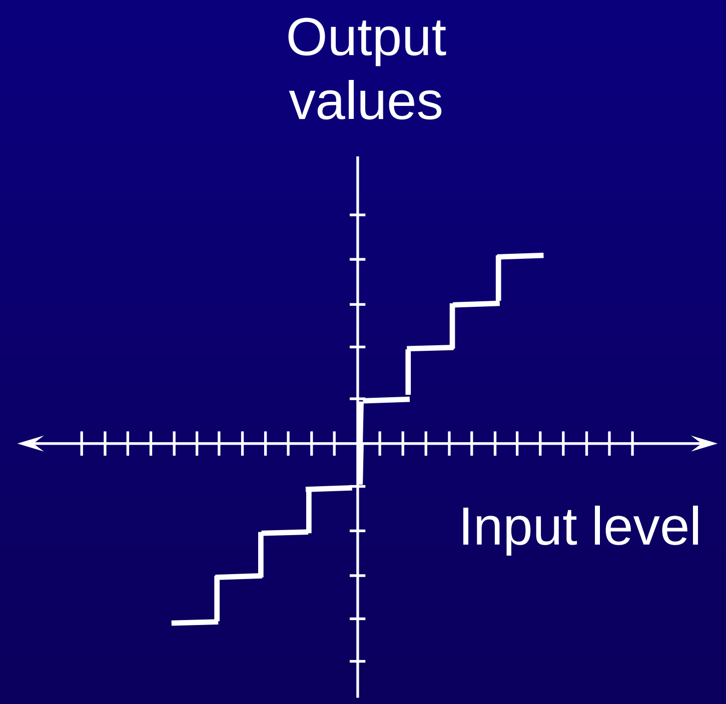
Shown after DCT, Quantization, Inverse Quantization and Inverse DCT using default Intra Quantization Matrix and Linear Quantizer Scale

Quantizer: Midtreader vs. Midrizer

Midtreader Quantizer



Midriser Quantizer

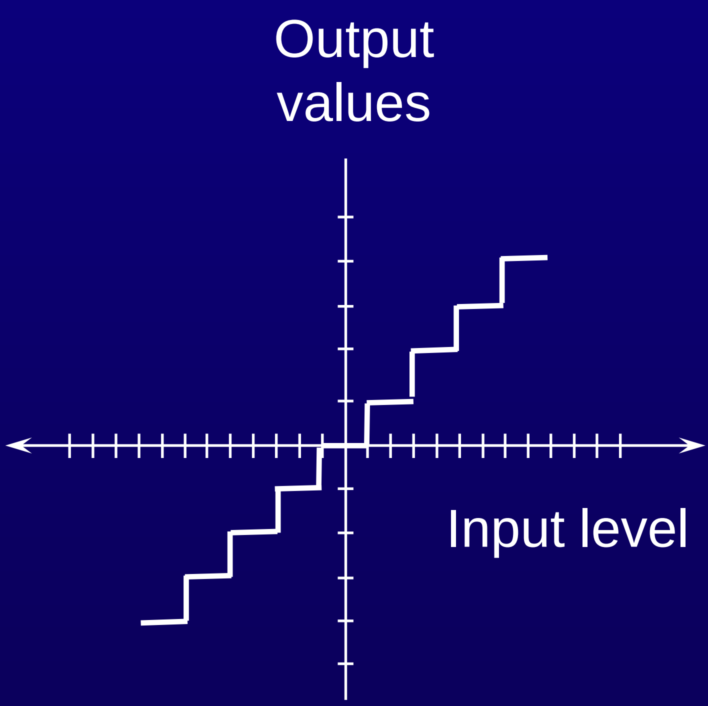


Non-uniform + residue -> midtread
Uniform -> midriser

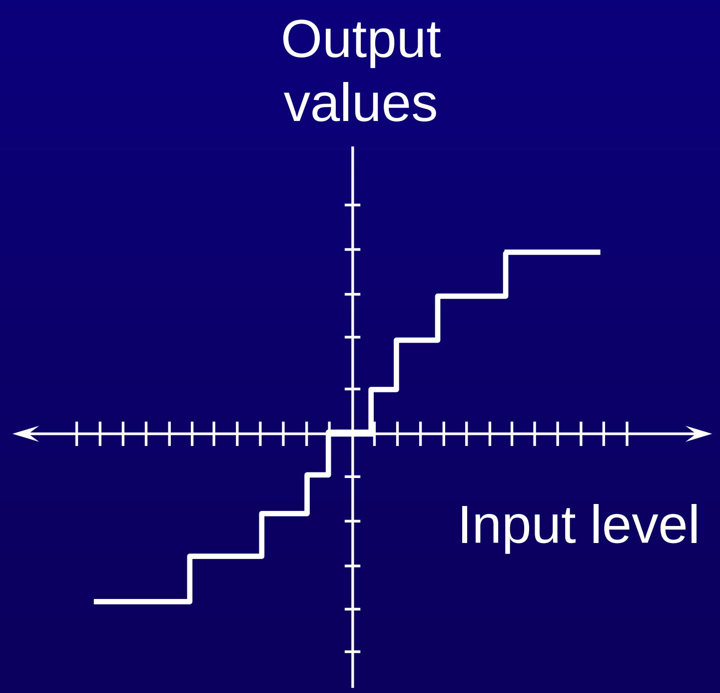
Quantizer: Uniform vs. Nonuniform

Ex. 四捨五入

Uniform Quantizer



Non-uniform Quantizer



Quantization Effect (1/2)

(a) **8** bits, or **256** gray level

(b) **7** bits, or **128** gray level

(c) **6** bits, or **64** gray level

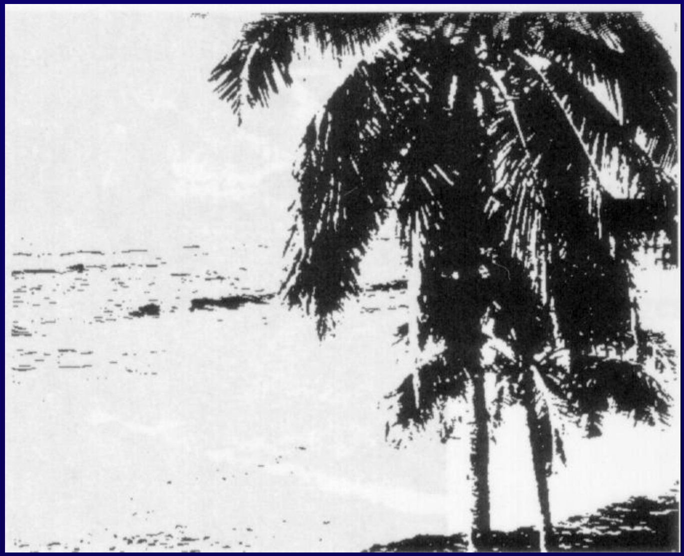
(d) **5** bits, or **32** gray level

Quantization Effect (2/2)

(e) **4** bits, or **16** gray level

(f) **3** bits, or **8** gray level.

(g) **2** bits, or **4** gray level



(h) **1** bits, or **2** gray level.

Optimal Quantization (1/4)

- The optimal reconstruction levels, $\{r_j\}$, in minimum mean squares error (MMSE) sense:

$$D = E\left[e_Q^2\right] = E\left[\left(f - \hat{f}\right)^2\right]$$

$$= \int_{f=-\infty}^{\infty} p(f)\left(f - \hat{f}\right)^2 df = \sum_{j=0}^{J-1} \int_{f=d_j}^{d_{j+1}} p(f)(f - r_j)^2 df$$

d_i為判斷的threshold

Optimal Quantization (2/4)

- If J is large $\longrightarrow$ $p(f) \approx p(r_j)$

$$D = \sum_{j=0}^{J-1} p(r_j) \int_{dj}^{d_{j+1}} (f - r_j)^2 df$$

$$\approx \sum_{j=0}^{J-1} p(r_j) \left[(d_{j+1} - r_j)^3 - (d_j - r_j)^3 \right]$$

for optimal r_j: $\frac{\partial \mathrm{D}}{\partial r_j} = 0 \longrightarrow d_{j+1} - r_j = -d_j + r_j$

$\longrightarrow r_j = (d_j + d_{j+1})/2$

$$D_{min} = \frac{1}{12} \sum_{j=0}^{J-1} p(r_j)(d_{j+1} - r_j)^3$$

- If $p(f)$ is uniformly distributed:

$d_{j+1} - d_j = \Delta = 2^{-b}(b - bits)$, $p(r_j) = \frac{1}{J \times \Delta}$, $D = \frac{1}{12} \times J \times \frac{1}{J \times \Delta} \times \Delta^3 = \frac{\Delta^2}{12}$.

Optimal Quantization (3/4)

In general :

- To minimize *D*, with $d_0 = -\infty, d_L = \infty$.

$$\frac{\partial D}{\partial r_j} = 0, \qquad 1 \le j \le J-1.$$

$$\frac{\partial D}{\partial d_j} = 0, \qquad 1 \le j \le J.$$

- Max-Lloyd Quantizer:

$$r_j = \frac{\int_{f=d_j}^{d_{j+1}} f \cdot p(f) df}{\int_{f=d_j}^{d_{j+1}} p(f) df}, \ 0 \le j \le J-1, \ d_j = \frac{r_{j-1}+r_j}{2}, \ 1 \le j \le J-1.$$

Optimal Quantization (4/4)

- The integration can be replaced by summation if f is discrete valued.
- In practice, various distributions (e.g., uniform, Gaussian, or Laplacian) are used to model the source $p(f)$.
- If $p(f)$ is unknown, histogram can be used to obtain $p(f)$, after normalization.

Uniform and Optimal Quantization

- Uniform Quantization:

$$r_i = \frac{d_{i-1} + d_i}{2}, 1 \le i \le L_\infty$$

- The error e_Q is uniformly distributed with **zero mean** and **variance** $\frac{\boldsymbol{\Delta^2}}{\mathbf{12}}$ $(d_i - d_{i-1} = \Delta)$.
- Let the range of *f* be *A*. Its variance is

$$\sigma_f^2 = \frac{1}{A}\int_{-A/2}^{A/2} f^2 df = \frac{A^2}{12}$$

PSNR(peak)拿peak值替換A

- The signal-to-noise ratio for a uniform quantizer is

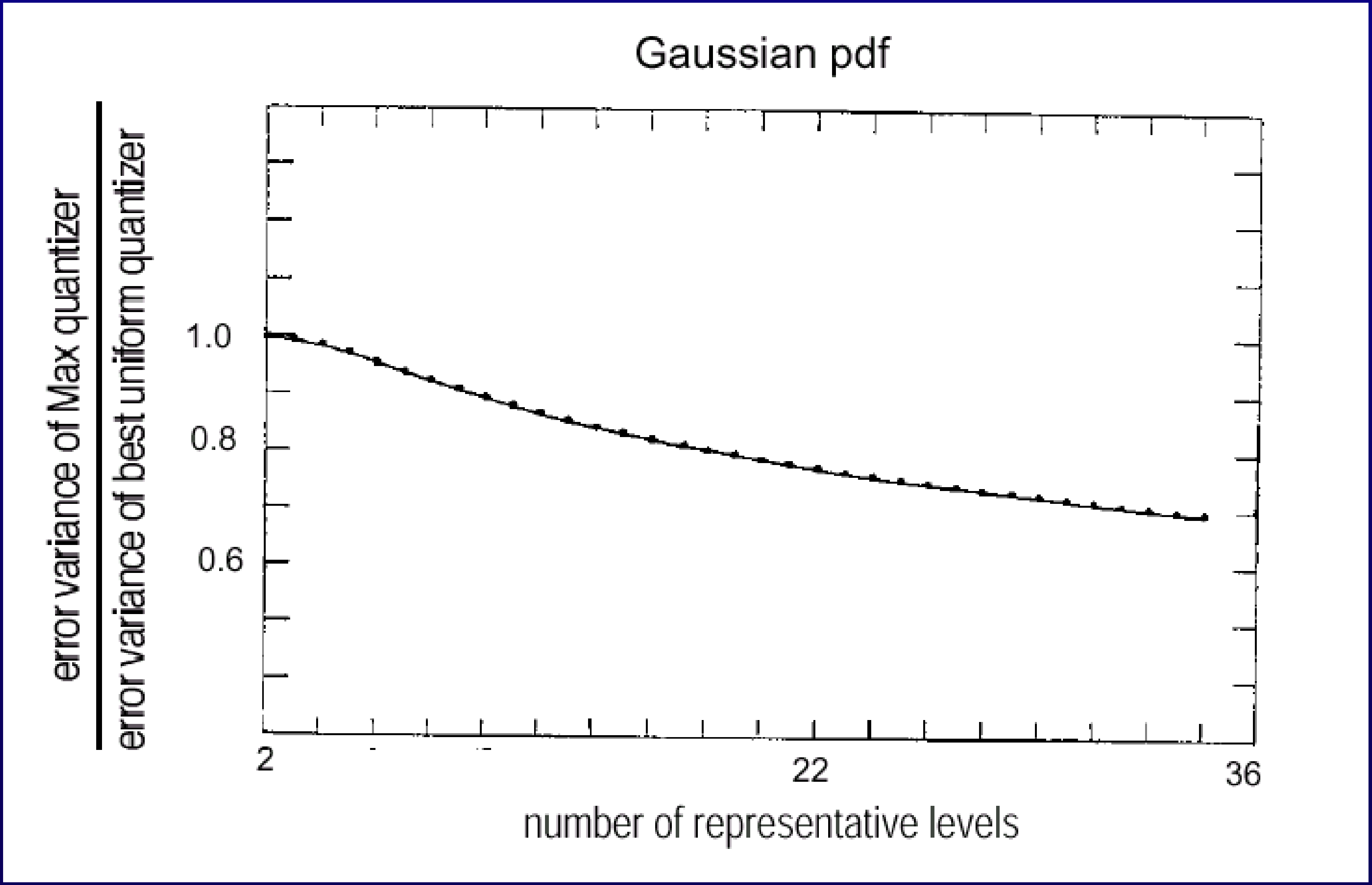
$$SNR = \frac{\text{Variance}}{\text{MSQE}} = \frac{A^2/12}{\frac{A^2}{2^{2b}}/12} = 2^{2b}$$

$$(SNR)_{\text{dB}} = 10\log_{10}(\text{SNR}) \approx 6b \text{ dB } (6\ bits)$$

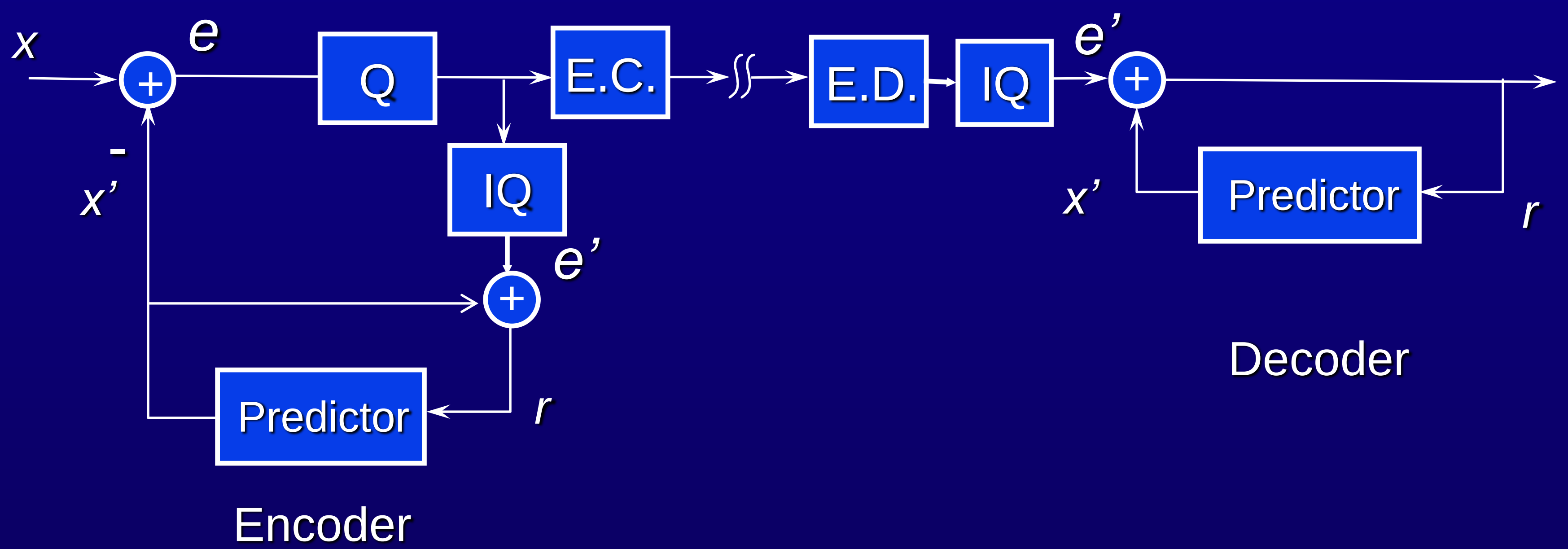
Max-Lloyd Quantizer (1/2)

Bits	Uniform d_i	Uniform r_i	Gaussian d_i	Gaussian r_i	Laplacian d_i	Laplacian r_i	Rayleigh d_i	Rayleigh r_i
1	−1.0000	−0.5000	−∞	−0.7979	−∞	−0.7071	0.0000	1.2657
	0.0000	0.5000	0.0000	0.7979	0.0000	0.7071	2.0985	2.9313
	1.0000		∞		∞		∞	
2	−1.0000	−0.7500	−∞	−1.5104	−∞	−1.8340	0.0000	0.8079
	−0.5000	−0.2500	−0.9816	−0.4528	−1.1269	−0.4198	1.2545	1.7010
	−0.0000	0.2500	0.0000	0.4528	0.0000	0.4198	2.1667	2.6325
	0.5000	0.7500	0.9816	1.5104	1.1269	1.8340	3.2465	3.8604
	1.0000		∞		∞		∞	
3	−1.0000	−0.8750	−∞	−2.1519	−∞	−3.0867	0.0000	0.5016
	−0.7500	−0.6250	−1.7479	−1.3439	−2.3796	−1.6725	0.7619	1.0222
	−0.5000	−0.3750	−1.0500	−0.7560	−1.2527	−0.8330	1.2594	1.4966
	−0.2500	−0.1250	−0.5005	−0.2451	−0.5332	−0.2334	1.7327	1.9688
	0.0000	0.1250	0.0000	0.2451	0.0000	0.2334	2.2182	2.4675
	0.2500	0.3750	0,5005	0.7560	0.5332	0.8330	2.7476	3.0277
	0.5000	0.6250	1.0500	1.3439	1.2527	1.6725	3.3707	3.7137
	0.7500	0.8750	1.7479	2.1519	2.3796	3.0867	4.2124	4.7111
	1.0000		∞		∞		∞	
4	−1.0000	−0.9375	−∞	−2.7326	−∞	−4.4311	0.0000	0.3057
	−0.8750	−0.8125	−2.4008	−2.0690	−3.7240	−3.0169	0.4606	0.6156
	−0.7500	−0.6875	−1.8435	−1.6180	−2.5971	−2.1773	0.7509	0.8863
	−0.6250	−0.5625	−1.4371	−1.2562	−1.8776	−1.5778	1.0130	1.1397
	−0.5000	−0.4375	−1.0993	−0.9423	−1.3444	−1.1110	1.2624	1.3850
	−0.3750	−0.3125	−0.7995	−0.6568	−0.9198	−0.7287	1.5064	1.6277
	−0.2500	−0.1875	−0.5224	−0.3880	−0.5667	−0.4048	1.7499	1.8721
	−0.1250	−0.0625	−0.2582	−0.1284	−0.2664	−0.1240	1.9970	2.1220
	0.0000	0.0625	0.0000	0.1284	0.0000	0.1240	2.2517	2.3814
	0.1250	0.1875	0.2582	0.3880	0.2644	0.4048	2.5182	2.6550
	0.2500	0.3125	0.5224	0.6568	0.5667	0.7287	2.8021	2.9492
	0.3750	0.4375	0.7995	0.9423	0.9198	1.1110	3.1110	3.2729
	0.5000	0.5625	1.0993	1.2562	1.3444	1.5778	3.4566	3.6403
	0.6250	0.6875	1.4371	1.6180	1.8776	2.1773	3.8588	4.0772
	0.7500	0.8125	1.8435	2.0690	2.5971	3.0169	4.3579	4.6385
	0.8750	0.9375	2.4008	2.7326	3.7240	4.4311	5.0649	5.4913
	1.0000		∞		∞		∞	

Max-Lloyd Quantizer (2/2)



Lossy Predictive Coding



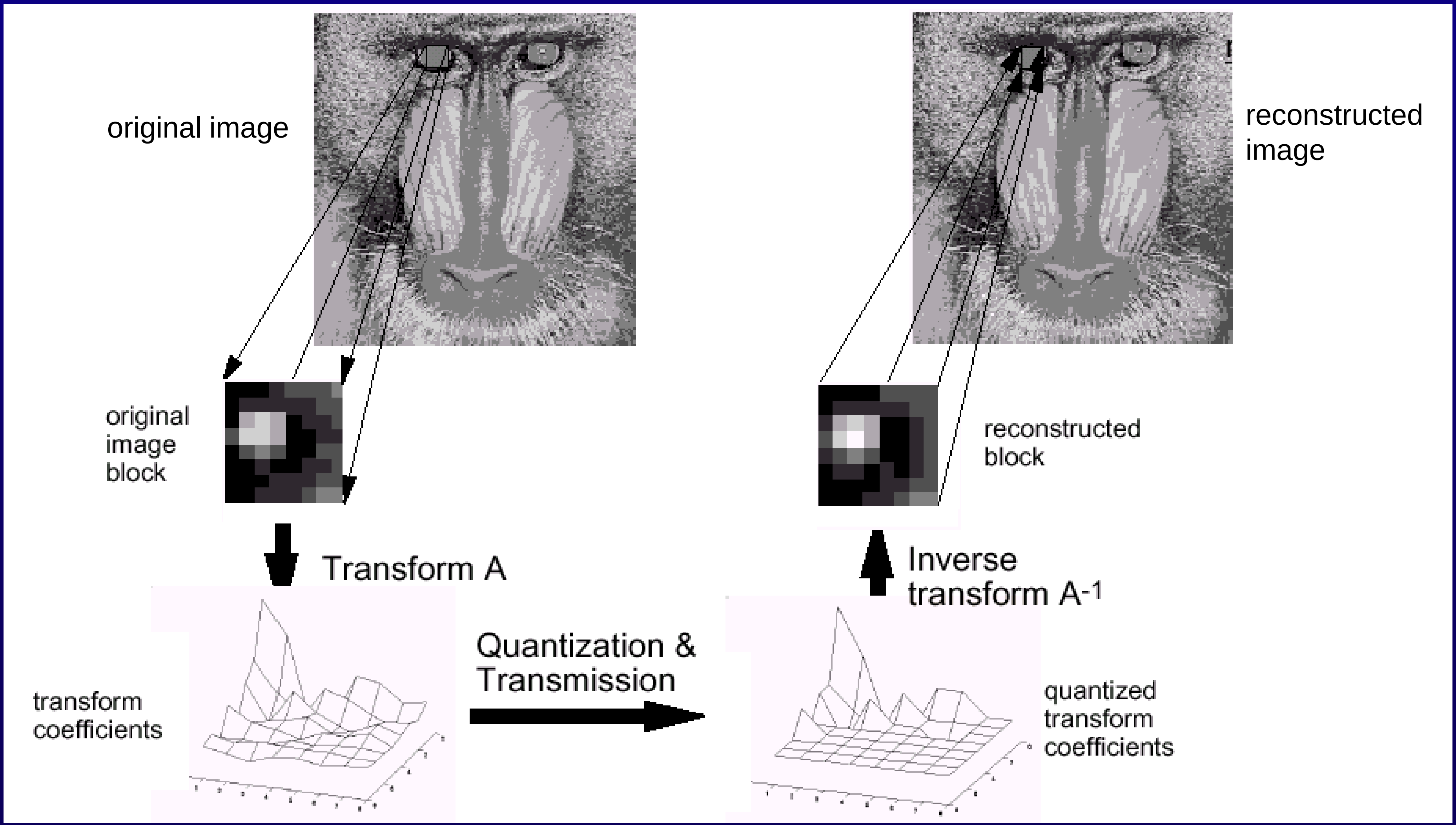
E.C.: Entropy Coder
E.D.: Entropy Decoder

Q: Quantizer
IQ: Inverse Quantizer

Discrete Cosine Transform (DCT)

Block-based DCT，切割block，
並使用8×8 4×4進行DCT

Transform Coding



Spatial Correlations in Images and Transform Coding

Correlation:

$$\rho_{1,h} = \frac{E[x(i,j)x(i-1,j)]}{E[x^2(i,j)]}$$

$$\rho_{1,v} = \frac{E[x(i,j)x(i,j-1)]}{E[x^2(i,j)]}$$

Autocovariance of an image is often modeled by

$$\mathrm{cov}(k,l) = \sigma^2 e^{-\alpha\sqrt{k^2+l^2}}$$

- Small when *k*, *l* are large
- Depend on image resolution

Pixel的座標差距（k,l）越大，
代表離越遠，自然關係不大
Resolution 越粗，cov越小
Resolution 越細，cov越大

Transform Coding: transform blocks of images to frequency (transform) domain, and code only the significant transform coefficients

Which transform to use?

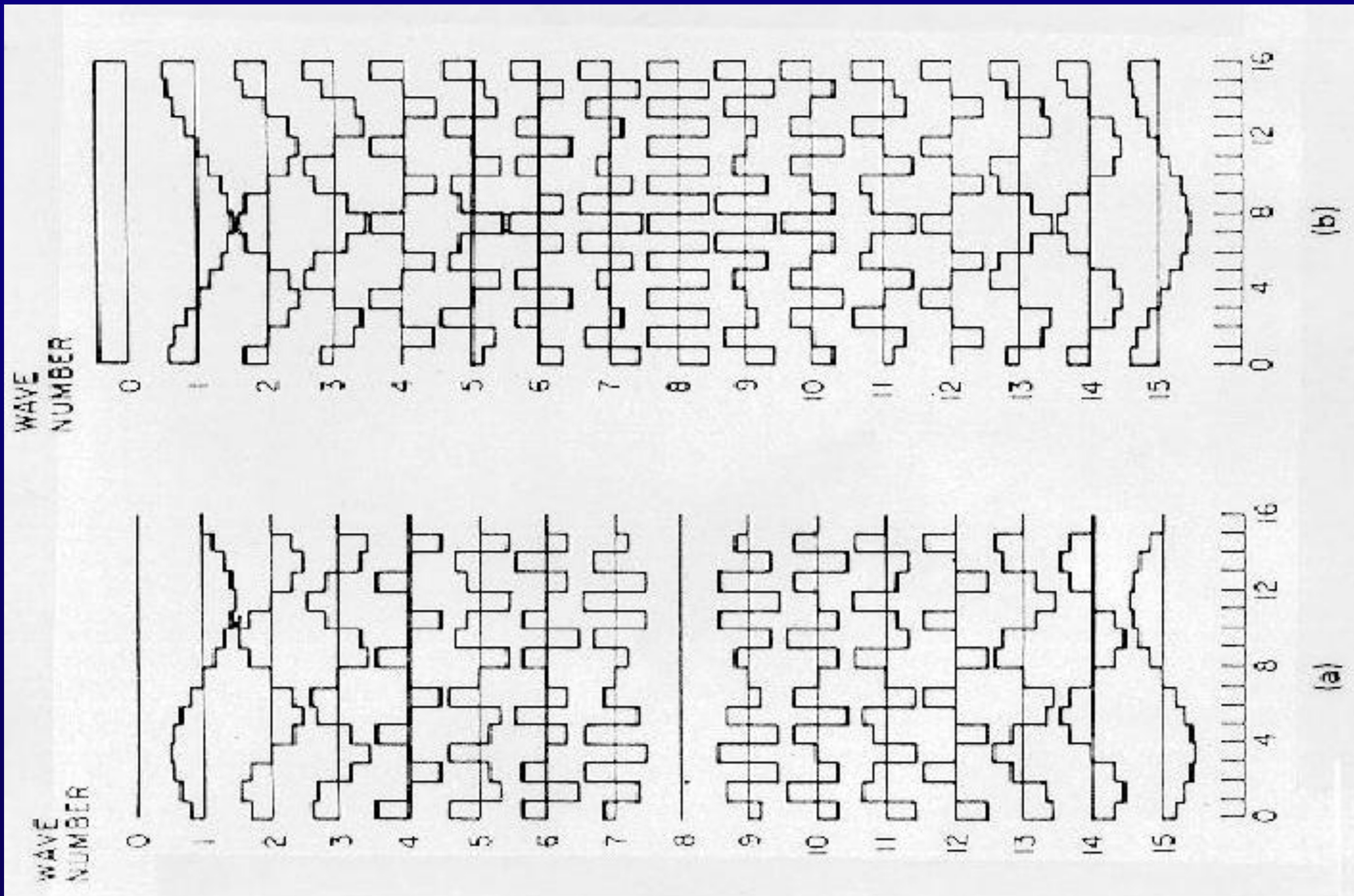
1-D Discrete Fourier Transform (DFT)

- 1-D DFT pair for a signal sequence *x(n)* of length *N*:

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{\frac{-j2\pi nk}{N}}, k = 0, \ldots N-1$$

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{\frac{-j2\pi nk}{N}}, n = 0, \ldots N-1$$

1-D 16-Point DFT Basis Vectors



Fourier basis functions. (a) Sine component. (b) Cosine component.

2-D Discrete Fourier Transform (1/2)

$$X(k_1,k_2)=\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1}x(n_1,n_2)e^{\frac{-j2\pi(n_1k_1+n_1k_1)}{N}},k_1,k_2=0,\ldots N-1$$

$$x(n)=\frac{1}{N^2}\sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1}X(k_1,k_2)e^{\frac{j2\pi(n_1k_1+n_1k_1)}{N}},n_1,n_2=0,\ldots N-1$$

- Either $\frac{1}{N^2}$ appears in forward or $\frac{1}{N}$ appears in each side.

2-D Discrete Fourier Transform (2/2)

- Fourier Transform of a real function results in complex numbers
- May result in artifacts due to discontinuity at block boundaries

Karhunen Loeve Transform (KLT) (1/3)

- Also called: eigenvector, principal component, analysis (PCA), or Hotelling [1933] transforms.
- Consider a population of random vectors and the mean vector (*E* is the expectation operator):

$$x = [x_1, x_2, \dots x_n]^T, m_x = E\{x\}$$

- The autocovariance matrix is defined as

$$C_x = E\{(x - m_x)(x - m_x)^T\}$$

- Given *M* vector samples, sample average is used to replace the expectation operator.

Karhunen Loeve Transform (KLT) (2/3)

$$m_x = \frac{1}{M}\sum_{k=1}^{M} x_k$$

$$C_x = \frac{1}{M}\left(\sum_{k=1}^{M} x_k\, x_k^T - m_x m_x^T\right)$$

$$\Sigma = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

$$\lambda_1 > \lambda_2 \cdots > \lambda_n$$

- Eigen analysis of C_x, $C_x = A^T \Sigma A = \sum_{i=1}^{n} \lambda_i\, a_i a_i^T$

  where A is an orthonormal matrix whose rows are formed from eigenvectors of C_x, i.e., Σ is the diagonal matrix consists of the "**ordered**" eigenvalues $\{\lambda_i\}$.

Karhunen Loeve Transform (KLT) (3/3)

- Basis vectors $\{a_i\}$ are the eigenvectors of the covariance matrix of the source.
- KLT projects the input vector onto the "orthonormal principal components (PCs)." $y = A(x - m_x)$

C_y= Σ=與上頁相同，
使yi與yj無關聯，i!=j

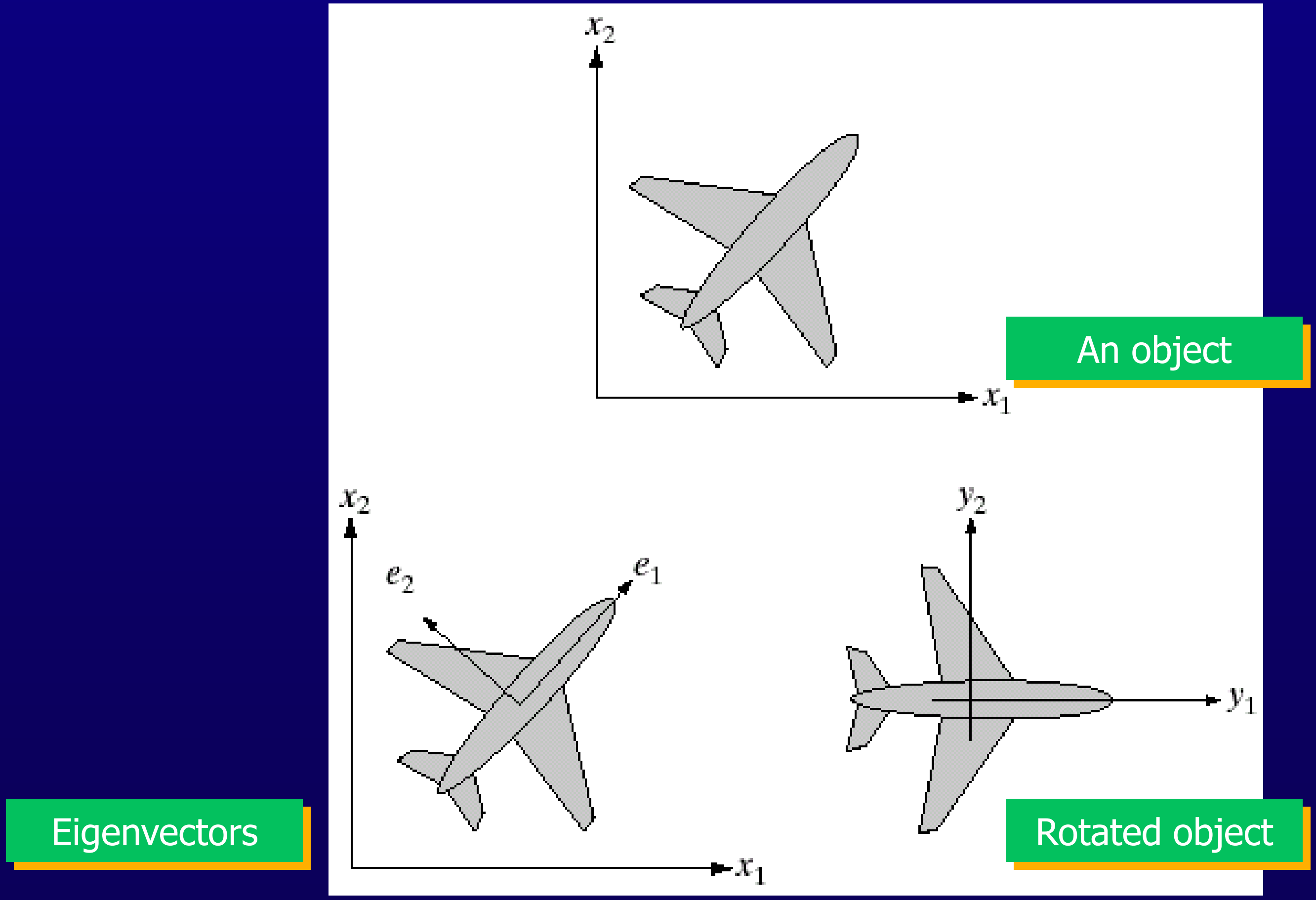
- C_y is a diagonal matrix whose elements along the main diagonal are $\{\lambda_i\}$. C_y and C_x have the same eigenvalues.
- KLT is the optimum transform for de-correlation. It is optimal in the sense that it minimizes the mean square error between x and the approximations x' (keeping only k largest $\{\lambda_i\}$).

保留most important使y1…yk最重要=>貼
近training samples=>依賴signal長怎樣

- **KLT basis vectors are signal dependent.**

Application of KLT (1/3)

- Object-orientation Estimation



Application of KLT (2/3)

- Face Image Compression:
  - Data source: CMU AMP Face Expression Database
  - 75 images/person, 64x64 pixels

Application of KLT (3/3)

- Represent original images with 6 parameters!

mean

eigenvectors




source-mean

-0.3694	-0.0362	0.0082	-0.0015	-0.0875	0.0308
0.1332	-0.2458	-0.0161	-0.0008	-0.0545	-0.1120
-0.1608	-0.0706	-0.0649	0.0349	0.2819	-0.0120

From DFT to DCT

- DCT has a higher compression ration than DFT
  - DCT avoids the generation of spurious spectral components

1-D DCT

$$F(k) = C(k) \sum_{n=0}^{N-1} f(n) \cos\left[\frac{(2n+1)k\pi}{2N}\right],$$

$$k = 0,1,\cdots,N-1,$$

$$f(n) = \sum_{k=0}^{N-1} C(k)F(k) \cos\left[\frac{(2n+1)k\pi}{2N}\right],$$

$$n = 0,1,\cdots,N-1,$$

where

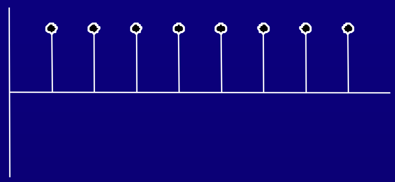
X(n) equally important，F(0)>=F(1)>=F(2)…

$$C(0) = \sqrt{\frac{1}{N}}, \qquad C(k) = \sqrt{\frac{2}{N}}, \qquad k = 1,2,\cdots,N-1$$

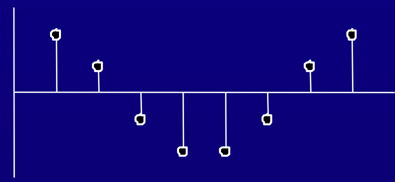
- The constants are often defined differently.

1-D 8-Point DCT Basis Vectors

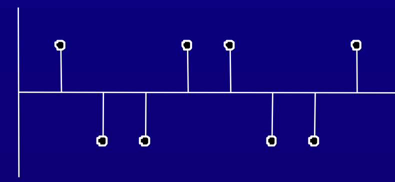
$\cos((2n+1)0\pi/16)$



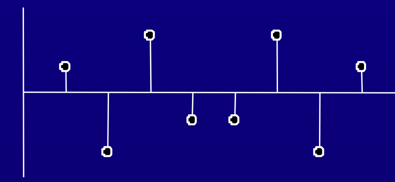
$\cos((2n+1)2\pi/16)$



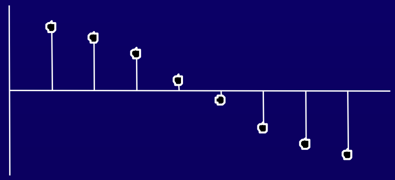
$\cos((2n+1)4\pi/16)$



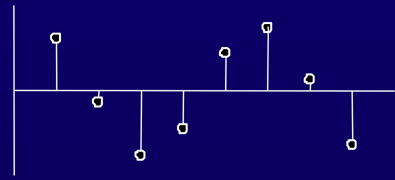
$\cos((2n+1)6\pi/16)$



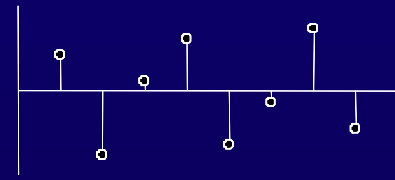
$\cos((2n+1)1\pi/16)$



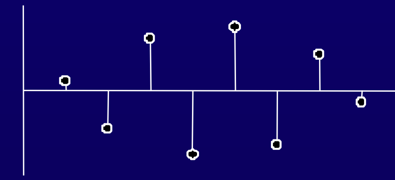
$\cos((2n+1)3\pi/16)$



$\cos((2n+1)5\pi/16)$

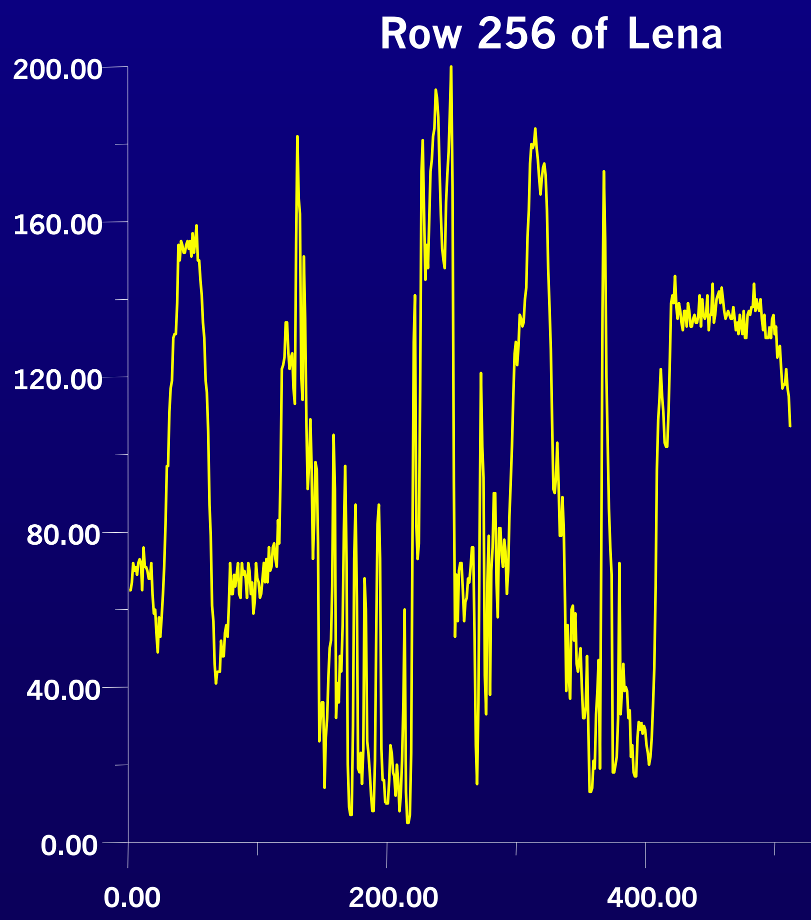


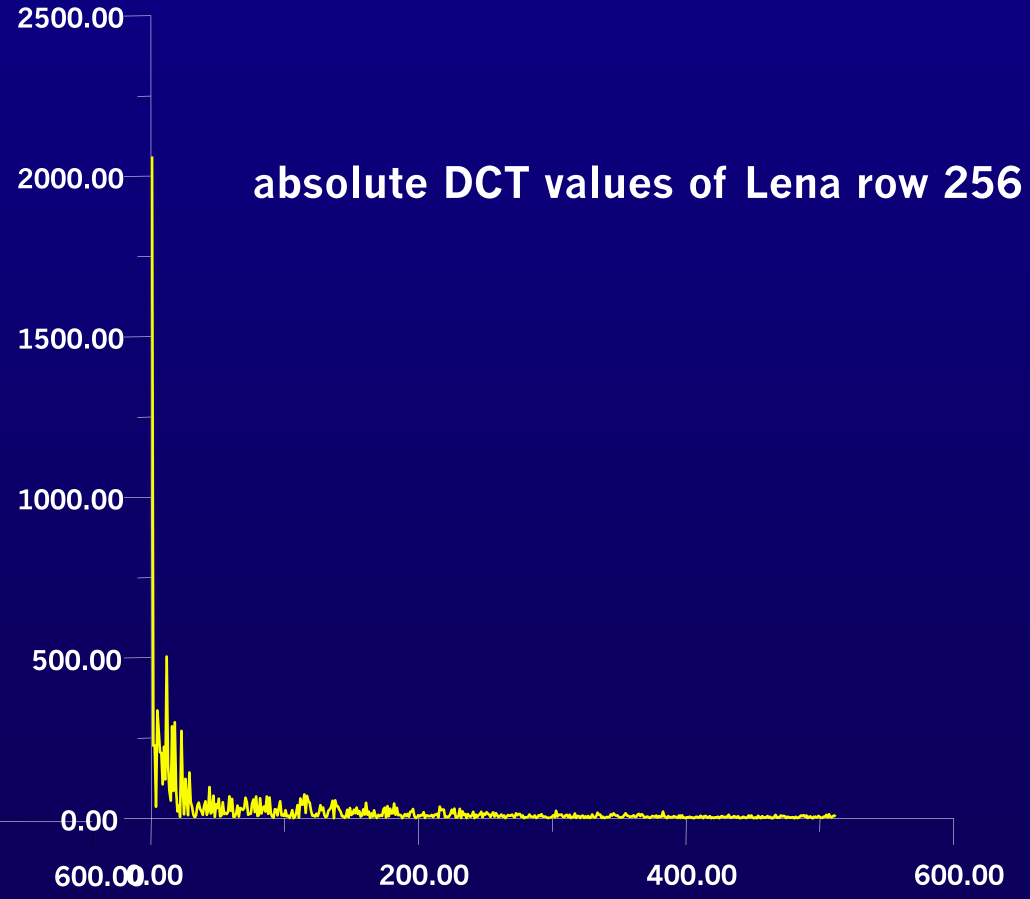
$\cos((2n+1)7\pi/16)$



Example of 1-D DCT

趨勢為$\lambda_1>\lambda_2\ldots$，重要的一樣在前面
v.s. KLT 是絕對$\lambda_1>\lambda_2\ldots$





2-D DCT

$$F(u,v) = C(u)C(v)\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\cos\left[\frac{(2m+1)u\pi}{2N}\right]\cos\left[\frac{(2n+1)v\pi}{2N}\right],$$
$$u,v = 0,1,\cdots,N-1,$$

$$f(m,n)$$
$$= \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} C(u)C(v)F(u,v)\cos\left[\frac{(2m+1)u\pi}{2N}\right]\cos\left[\frac{(2n+1)v\pi}{2N}\right],$$
$$m,n = 0,1,\cdots,N-1,$$

$$where\ C(u),C(v) = \sqrt{1/N}, \qquad for\ u,v = 0;$$
$$C(u),C(v) = \sqrt{2/N}, \qquad otherwise.$$

Two Examples of 2-D DCT

8x8 DCT

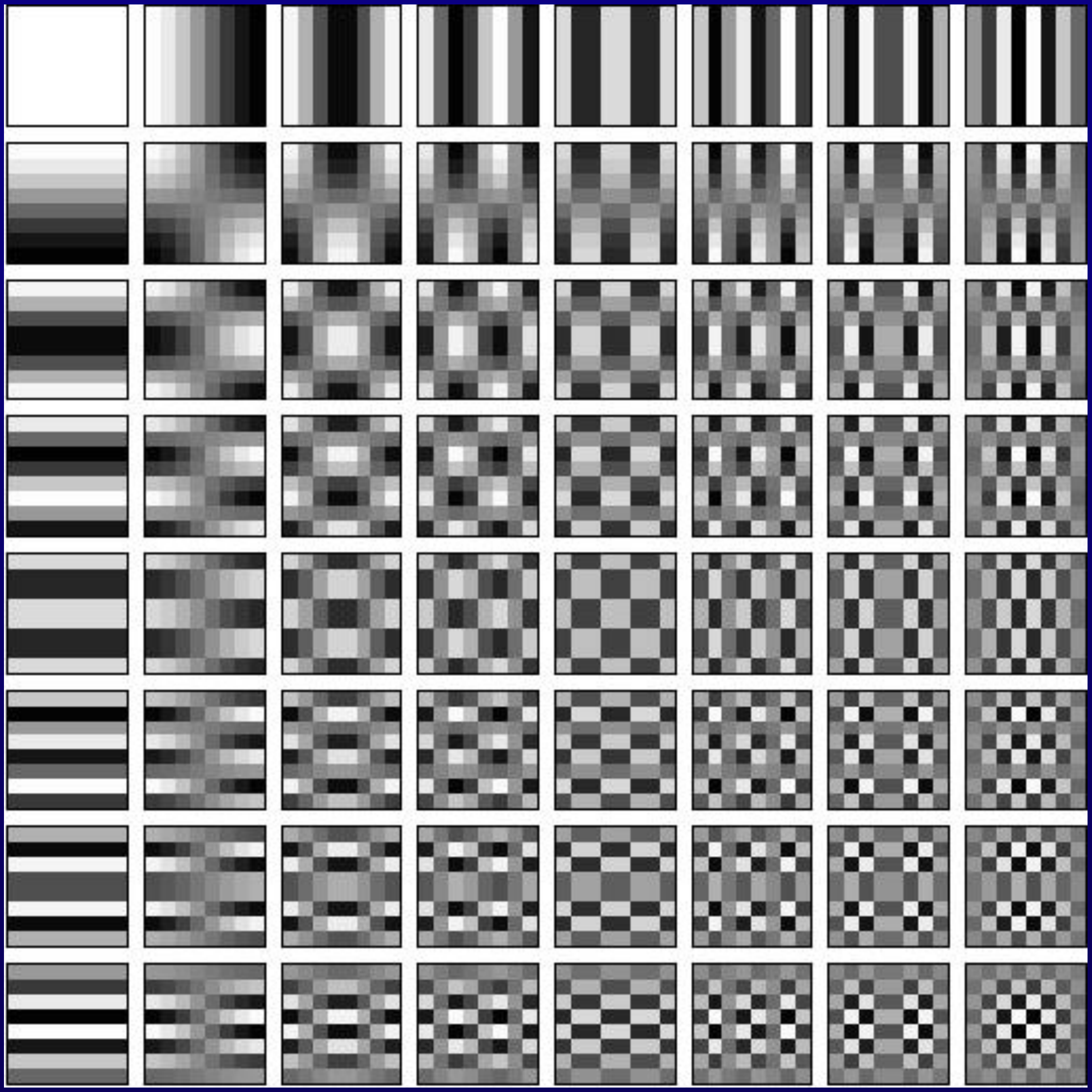
$$F(u,v) = \frac{1}{4}C(u)C(v)\sum_{m=0}^{7}\sum_{n=0}^{7} f(m,n)\cos\left[\frac{(2m+1)u\pi}{16}\right] cos\frac{(2n+1)v\pi}{16},$$
$$u,v = 0,1,\dots,N-1,$$

$$f(m,n) = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v)\cos\left[\frac{(2m+1)u\pi}{16}\right] cos\frac{(2n+1)v\pi}{16},$$
$$m,n = 0,1,\dots N-1,$$

where $C(u), C(v) = \frac{1}{2}$, for $u, v = 0$;

$C(u), C(v) = 1$, otherwise.

2-D 8-Point DCT Basis Functions



An Example of 8x8 DCT

Pixel domain

52	55	61	66	70	61	64	73
63	59	66	90	109	85	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	68	65	76	78	94

DCT ↓ ↑ **IDCT**

DC

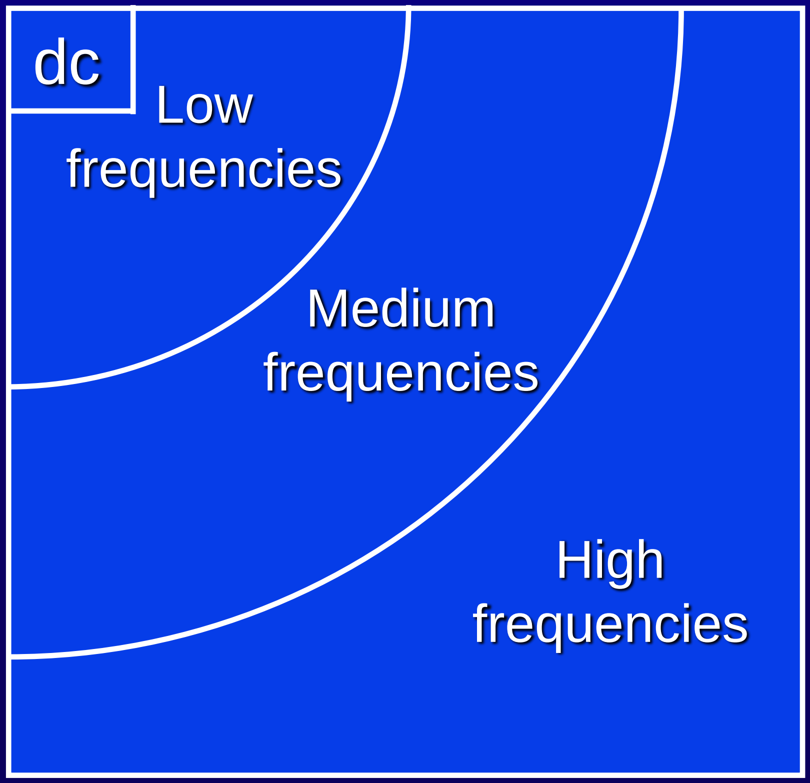
AC

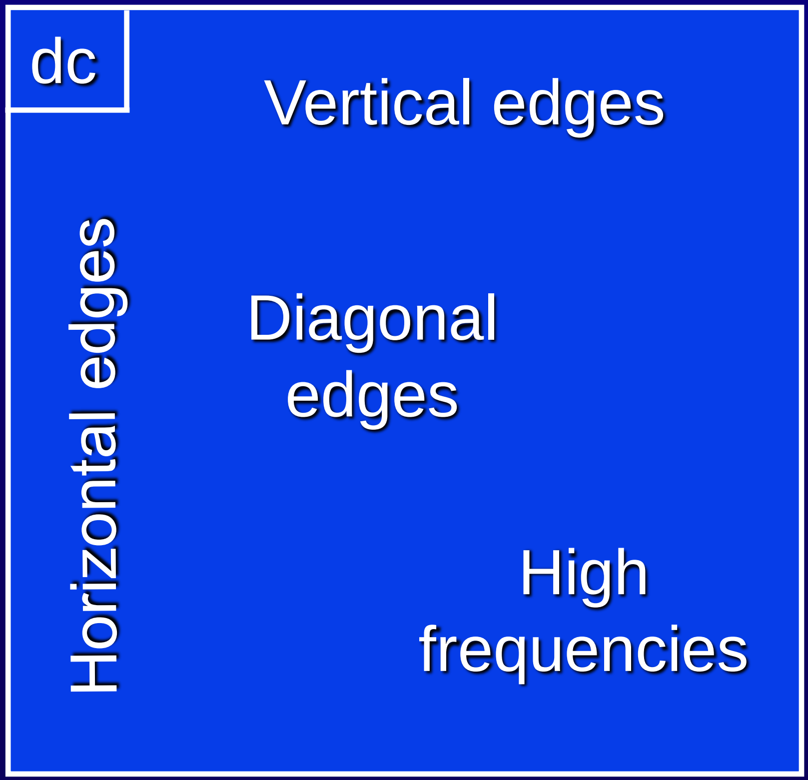
Frequency domain

609	- 29	- 62	25	55	- 20	- 1	3
7	- 21	- 62	9	11	- 7	- 6	6
- 46	8	77	- 25	- 30	10	7	- 5
- 50	13	35	- 15	- 9	6	0	3
11	- 8	- 13	- 2	- 1	1	- 4	1
- 10	1	3	- 3	- 1	0	2	- 1
- 4	- 1	2	- 1	2	- 3	1	- 2
- 1	- 1	- 1	- 2	- 1	- 1	0	- 1

DC: $F(0,0) = (1/8)\ \Sigma\Sigma\ f(m,n)$

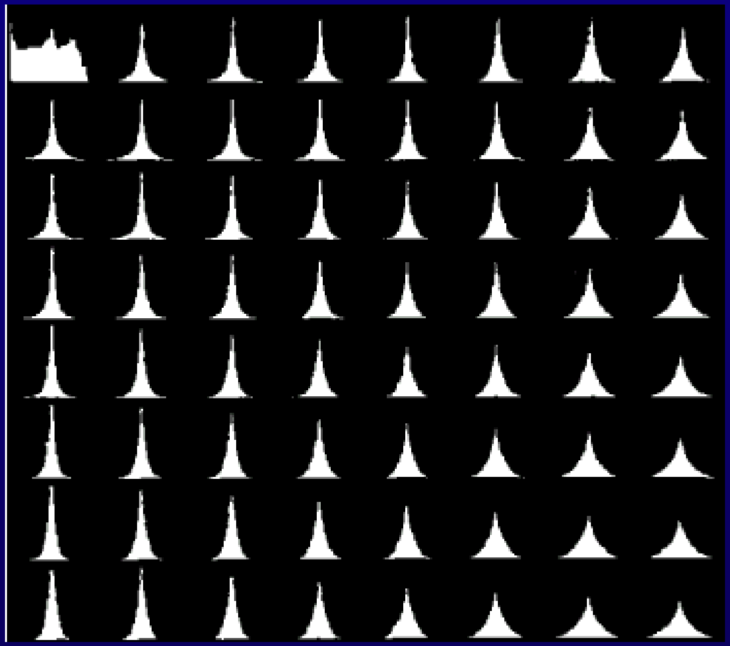
2-D DCT Coefficients





Amplitude Distribution of DCT Coefficients

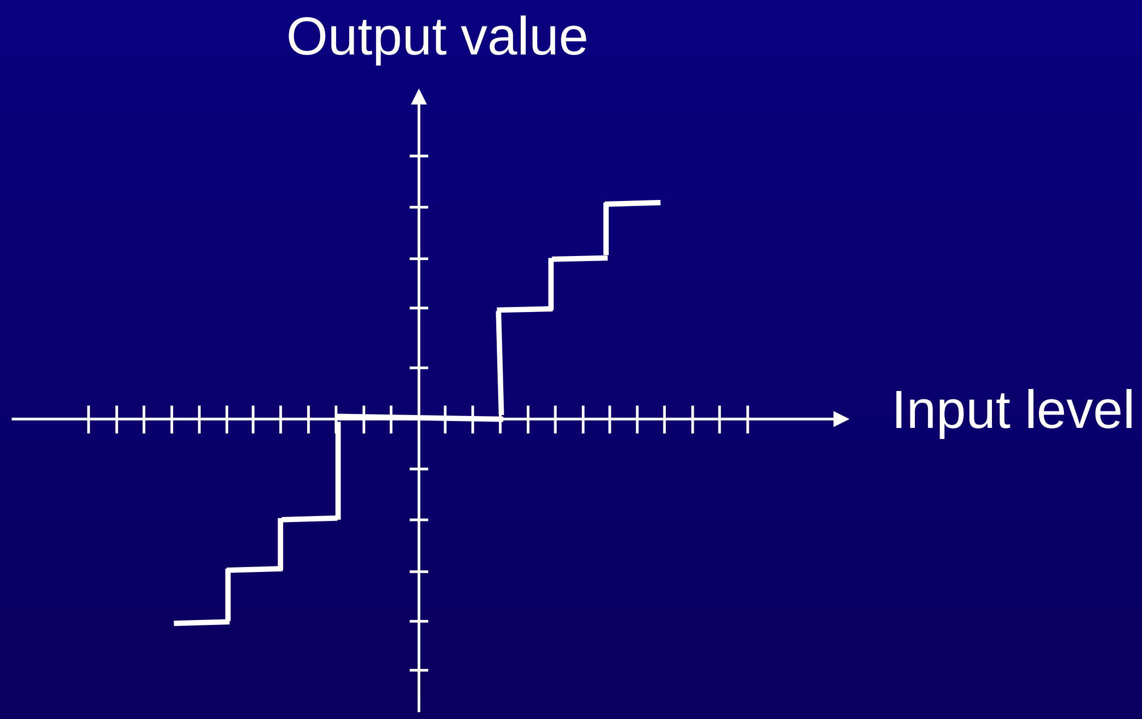
- Histogram for 8x8 DCT coefficient amplitudes measured for natural images:



- DC coefficient is typically uniformly distributed
- For the other AC coefficients, the distribution resembles a Laplacian pdf.

Threshold Coding (1/2)

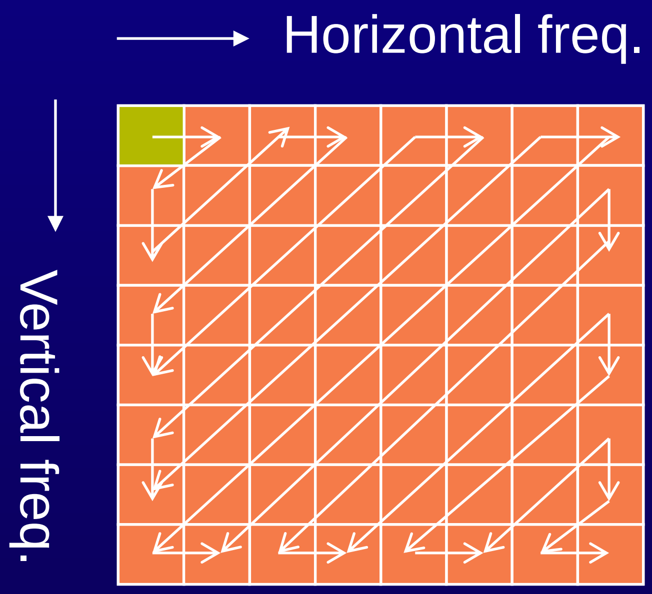
- Transform coefficients that fall below a threshold are discarded
- Implementation by uniform quantizer with threshold characteristics:



- Positions of non-zero transform coefficients are transmitted in addition to their amplitude values

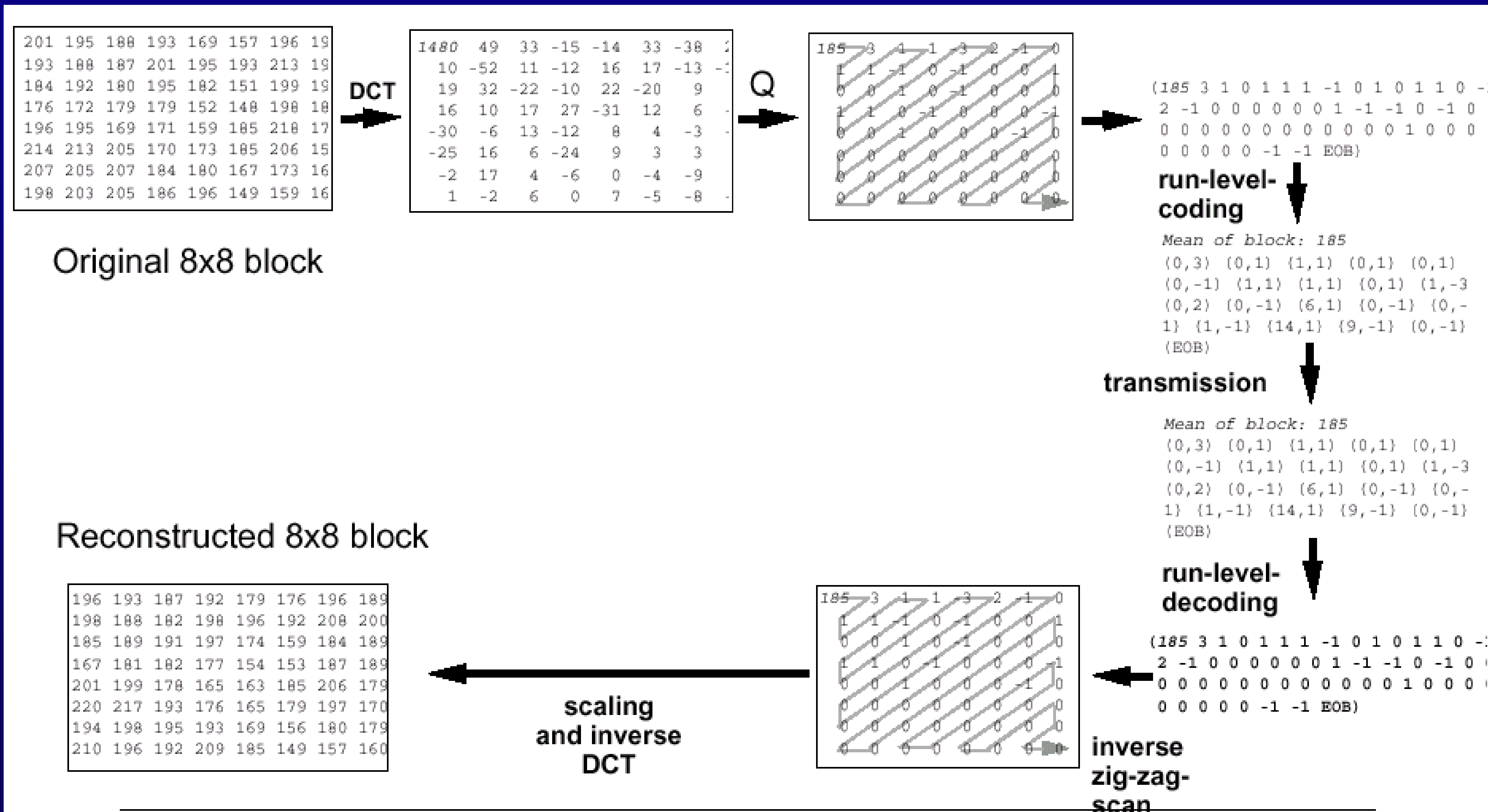
Threshold Coding (2/2)

- Efficient encoding of the position of non-zero transform coefficients: zig-zag-scan+run-level-coding

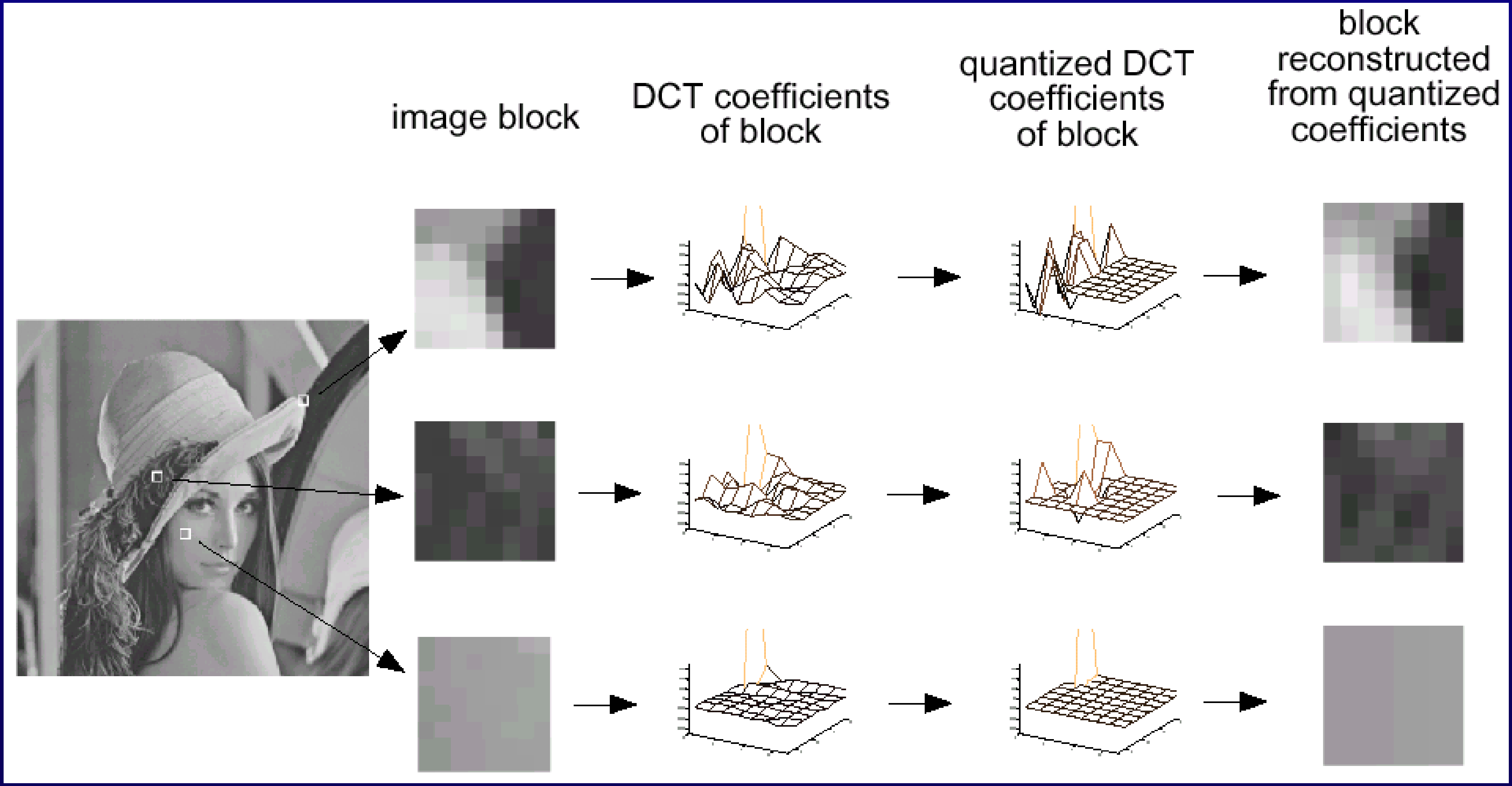


DCT coefficients

DCT + Quantization + Run-Level-Coding



Details in A Block vs. DCT Coefficients Transmitted



Typical DCT Coding Artifacts

- DCT coding with increasingly coarse quantization, block size 8x8

quantizer step-size for AC coefficient: 25

quantizer step-size for AC coefficient: 100

quantizer step-size for AC coefficient: 200

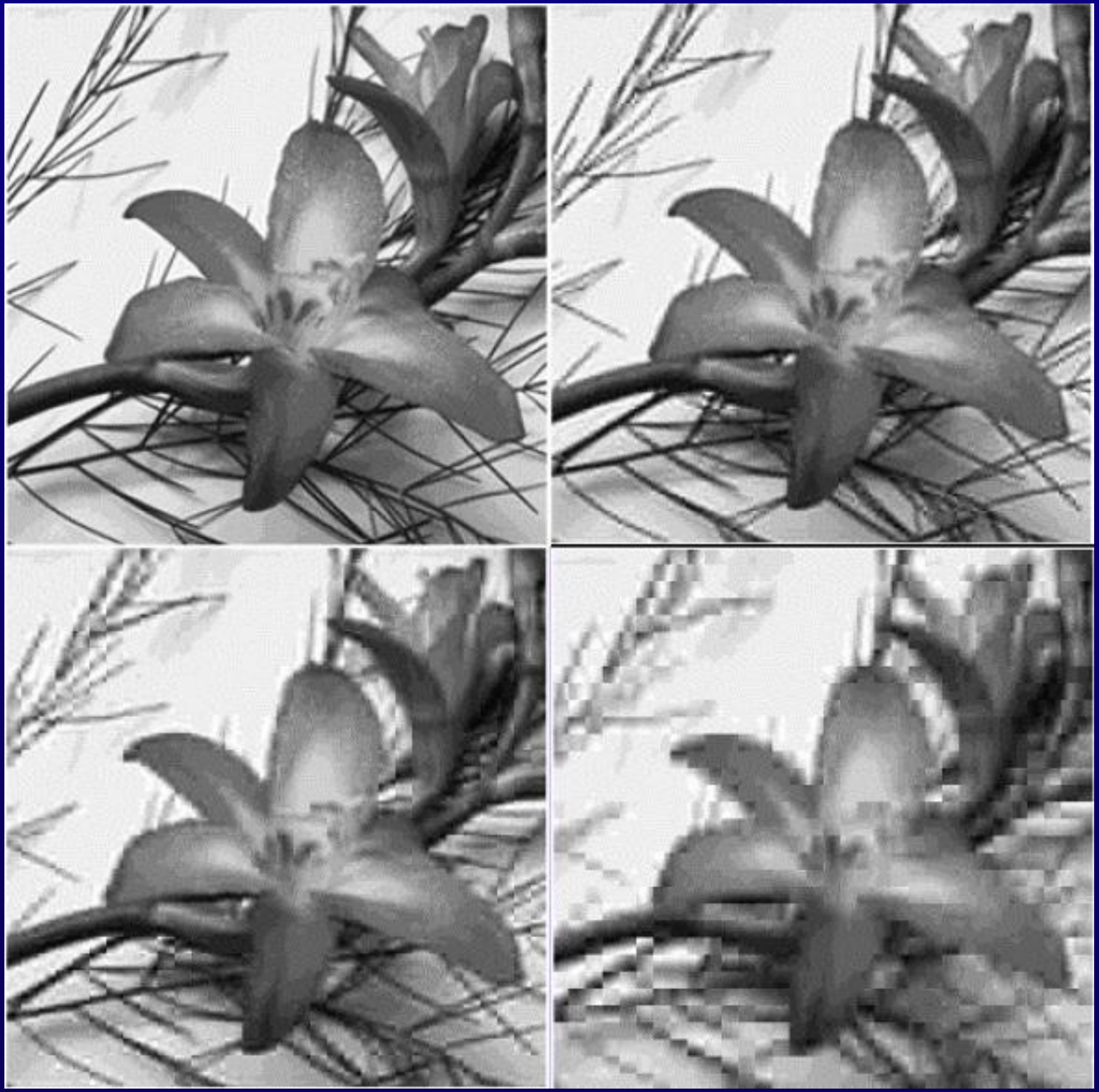
coefficient 25，代表差距為+-12.5，依此類推，因此coefficient 越大誤差越大

Image Approximated by Different Number of DCT Coefficients

original

with 16/64 coefficients

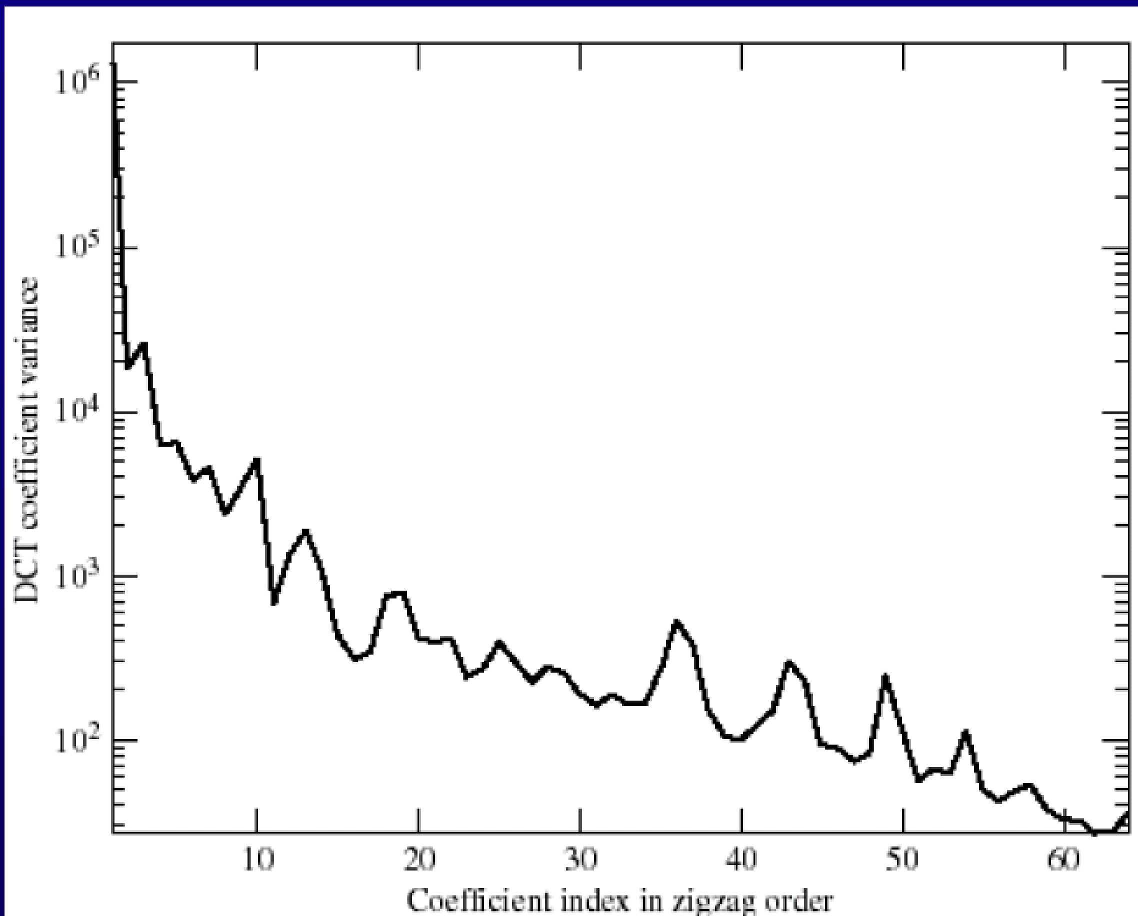
64為8×8來的，取coefficient 最重要的k個保留，因此保留越少重建回來越慘

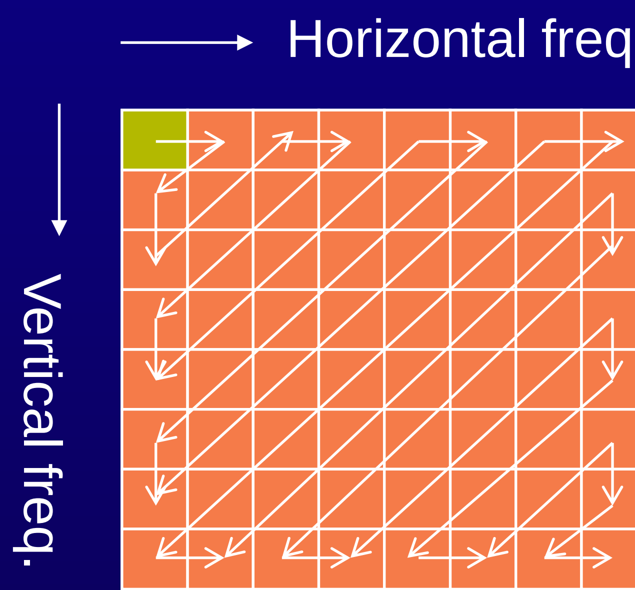


with 8/64 coefficients

with 4/64 coefficients

Energy Distribution of DCT Coefficients in Typical Images





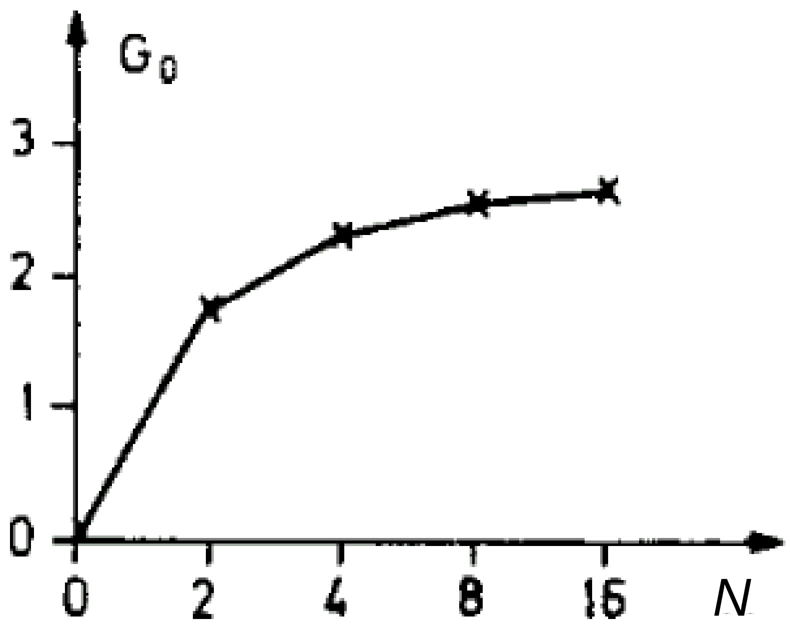
DCT coefficients

可以看到趨勢往下，越重要的covariance 越大，反正越小

Influence of DCT Block Size

- Efficiency as a function of block-size *NxN*, measured for *b*-bit quantization in the original domain and equivalent quantization in the transform domain

$$G_0 = \frac{\text{Memoryless entropy of original signal}}{\text{mean entropy of transform coefficients}}$$

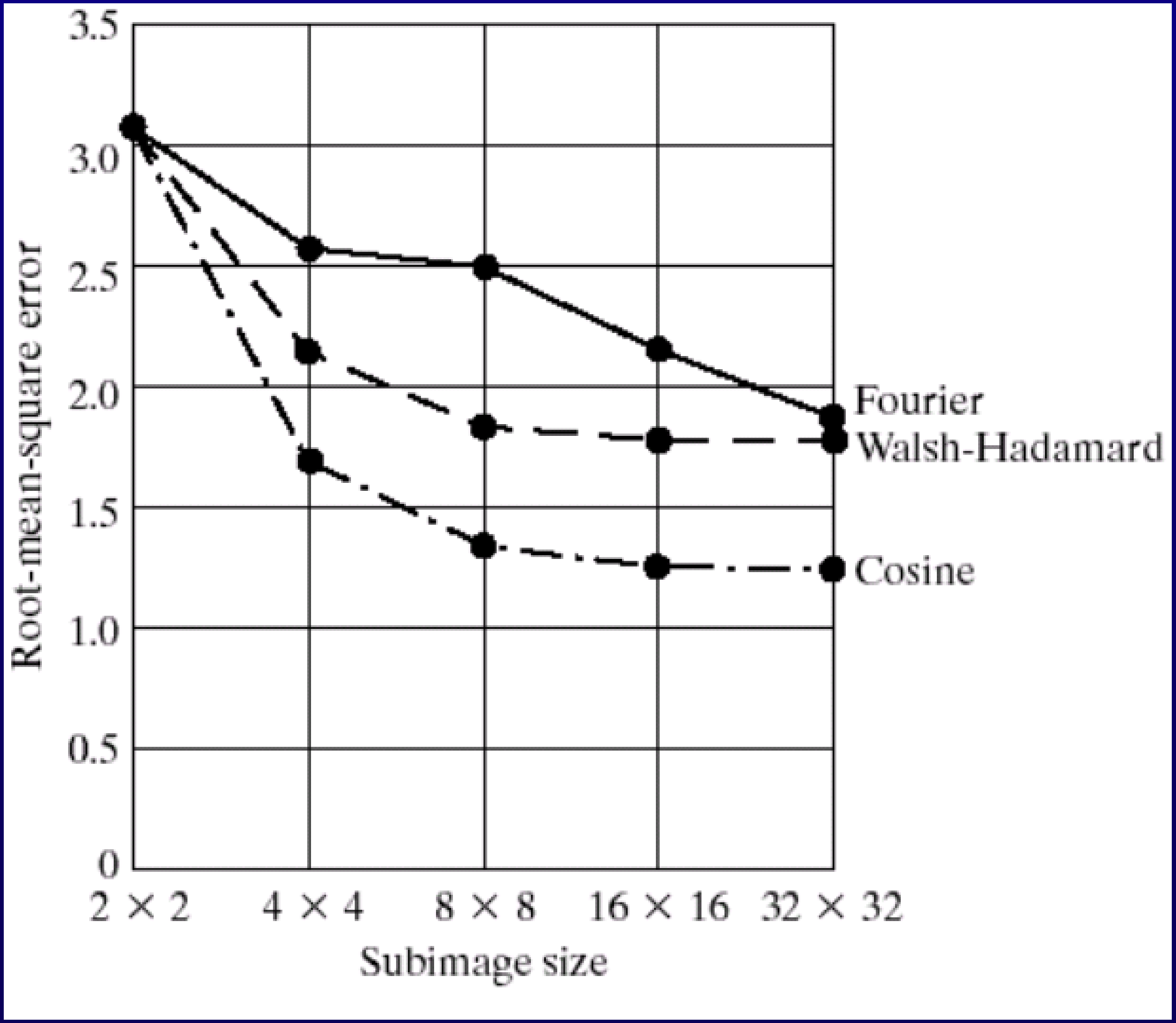


block size越大壓縮越大，但16×16比8×8複雜多了=>tradeoff

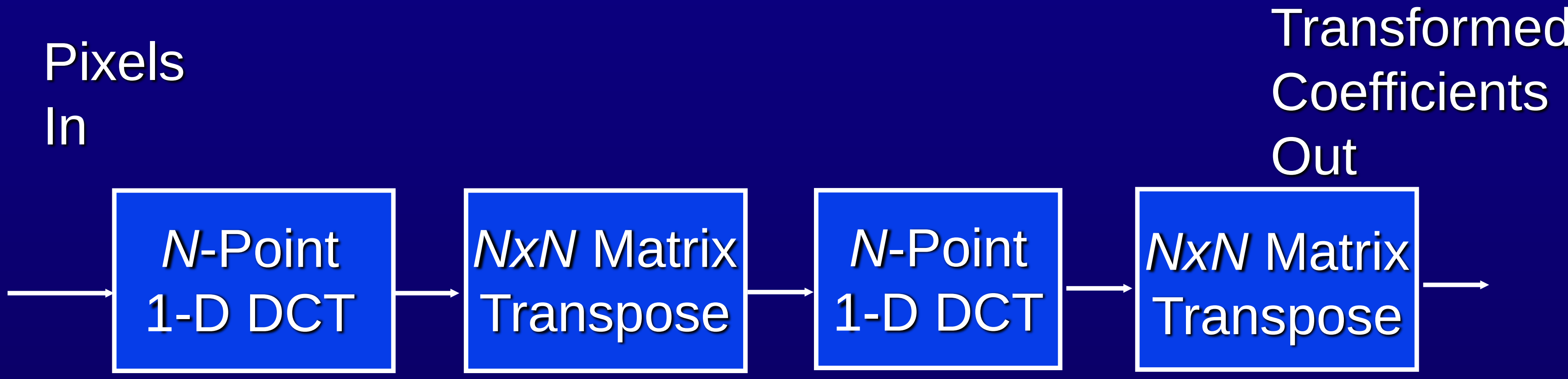
- Block size 8x8 is a good compromise

Influence of Block Size

- 75% resulting coefficients are truncated to compare the performance

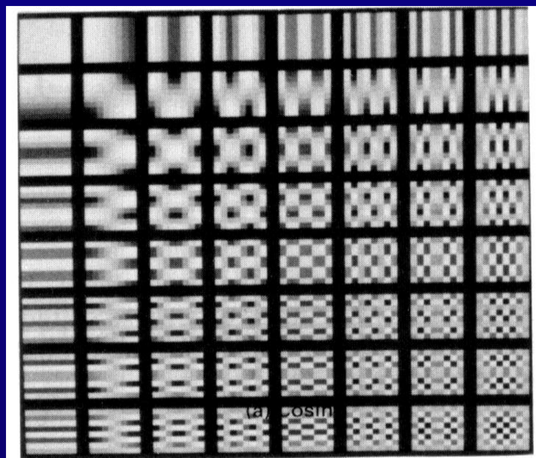


Implementation of 2-D DCT Using Row-Column Decomposition

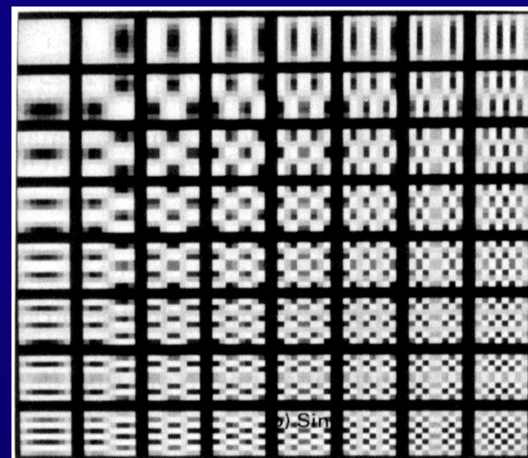


Comparison of Various Transforms (1/4)

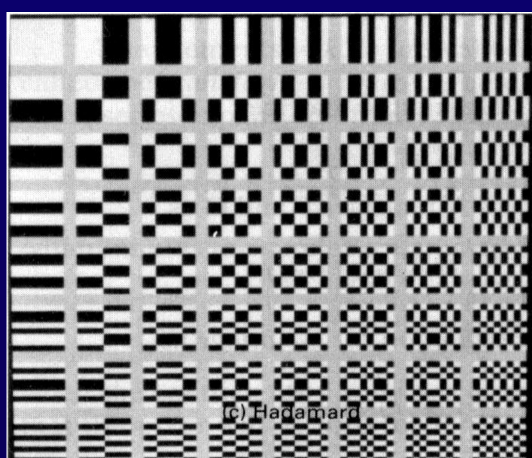
Basis Functions of 2-D Transform ($M = N = 8$)



Cosine

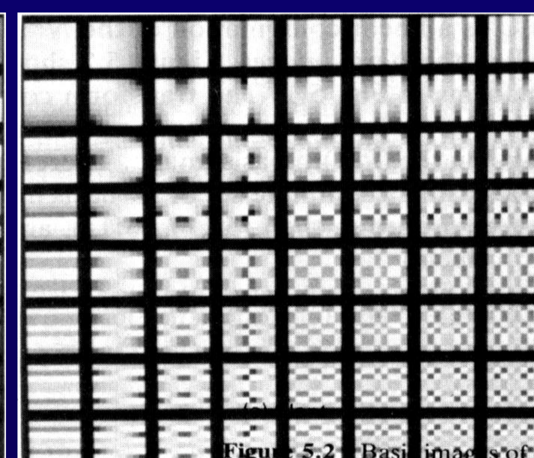


Sine



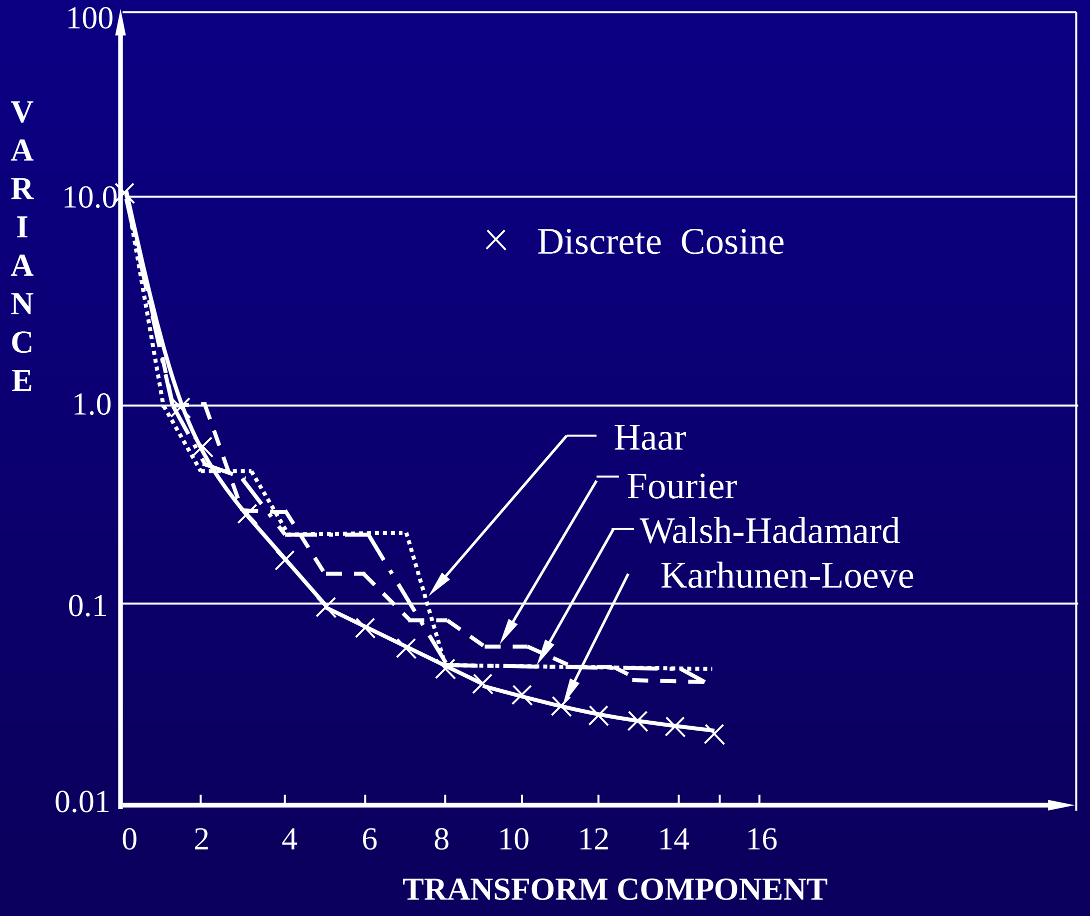
Hadamard

Haar



Slant

Comparison of Various Transforms (2/4)



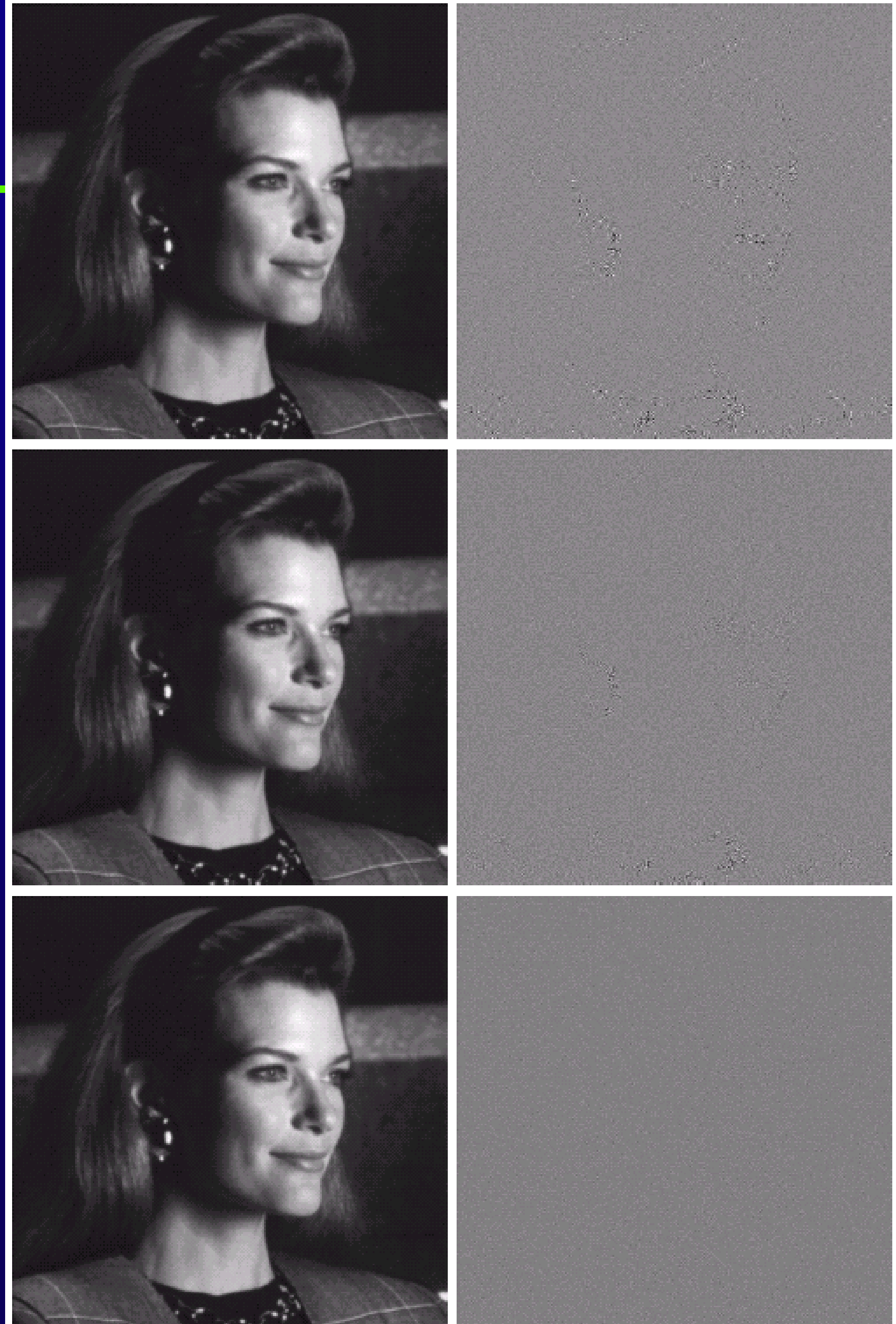
Comparison of Various Transforms (3/4)

DFT

Hadamard

50% of coefficients are truncated

DCT



DCT Discussions

- Simulations have shown that for practical images, DCT outperforms WHT and DFT in the energy compaction, and is close to KLT optimal decomposition.
- Block size: 8x8 or 16x16
- Complexity
- Performance: for typical CCIR601 pictures

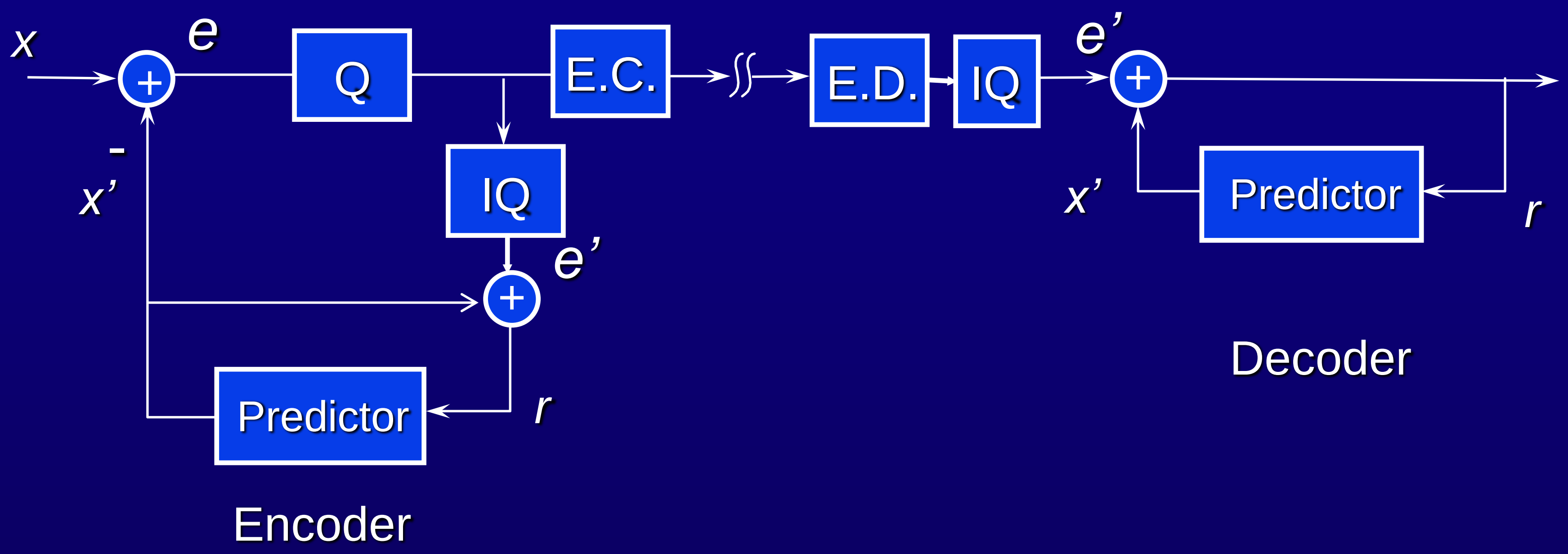
  Excellent when >= 2 b/pel

  Good when >= 0.8 b/pel

  Blocking artifacts ~< 0.5 b/pel
- Adjust threshold and quantizer to match local statistics and human visual criteria
- Effect of transmission errors

Motion Compensated Prediction

Predictive Coding

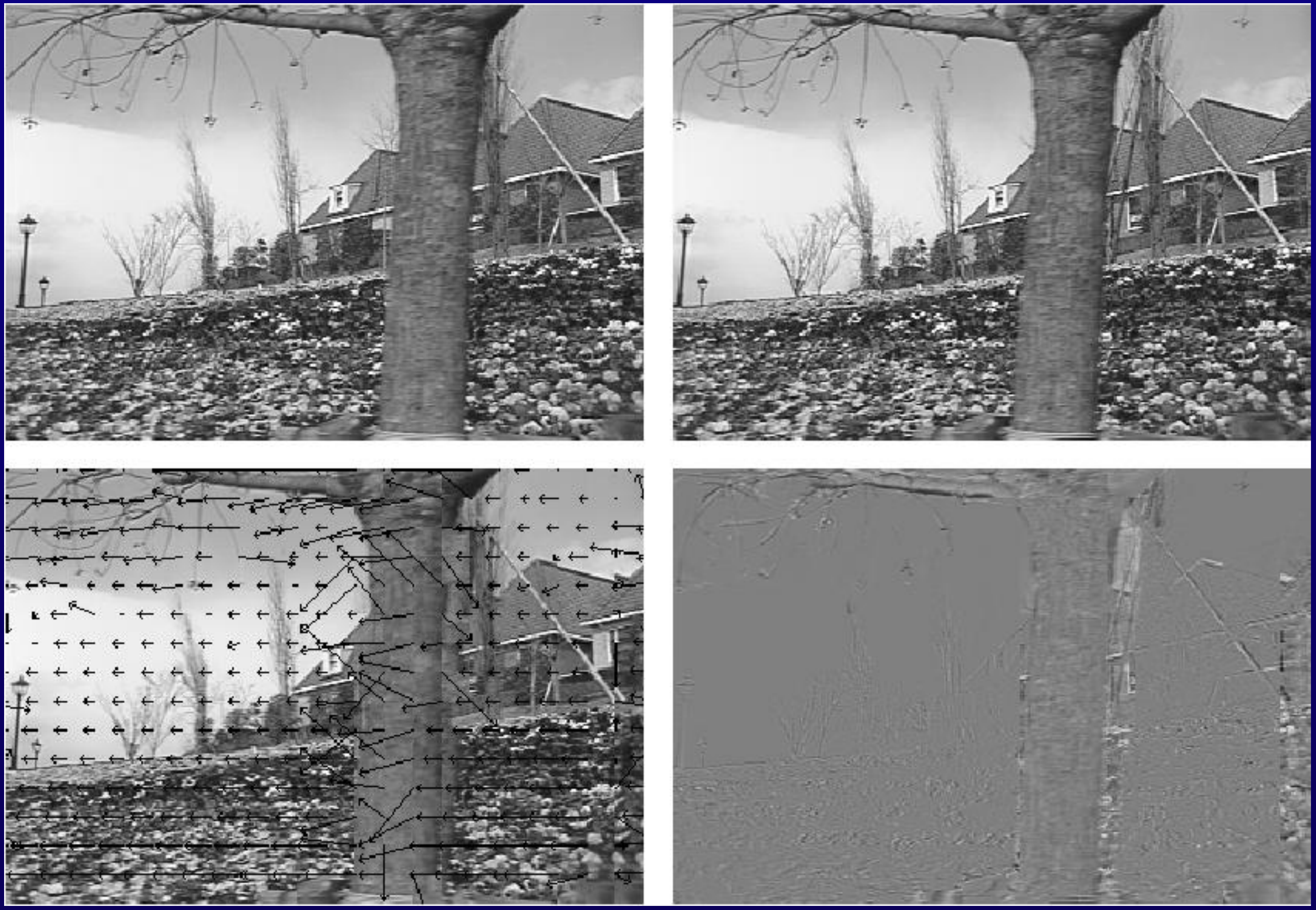


E.C.: Entropy Coder
E.D.: Entropy Decoder

Q: Quantizer
IQ: Inverse Quantizer

Motion Compensated Prediction Example

Frame 1 Frame 2



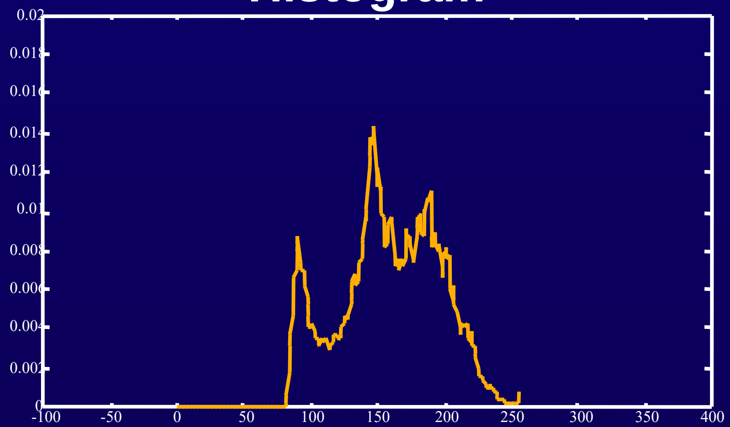
Frame2 with MV MC prediction error frame

Gray-Scale Statistics of Prediction Errors

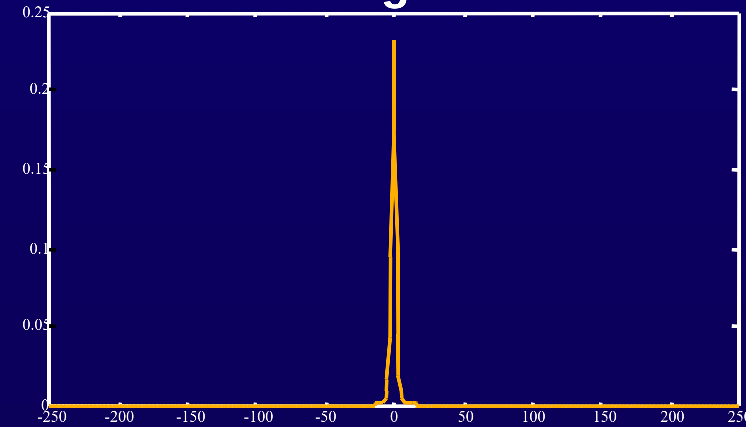
One Frame of Original Image Pair

Prediction Error

Histogram



Histogram



Motion Compensated Prediction with Different ME Precisions

- There is 40% improvement even at high motion-estimation error of +/- 1 pixel.
- Different motion precision achieve different performance.

$$R = \frac{1}{2}\log_2 \frac{\sigma_e^2}{D}$$

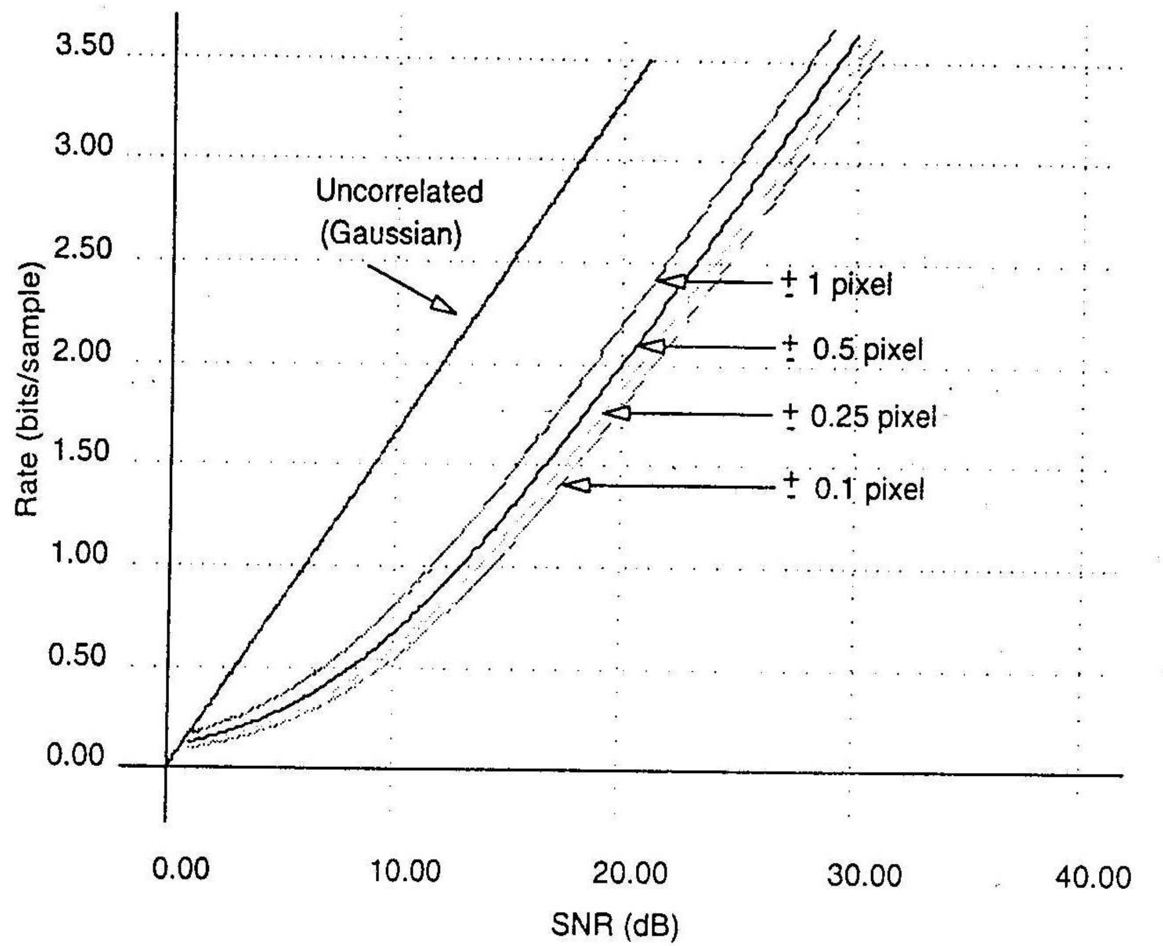


Figure 4.6 Rate-distortion performance for motion-compensated prediction with no intraframe coding.

snr 越小越好

R-D Function (Inter vs. Intra)

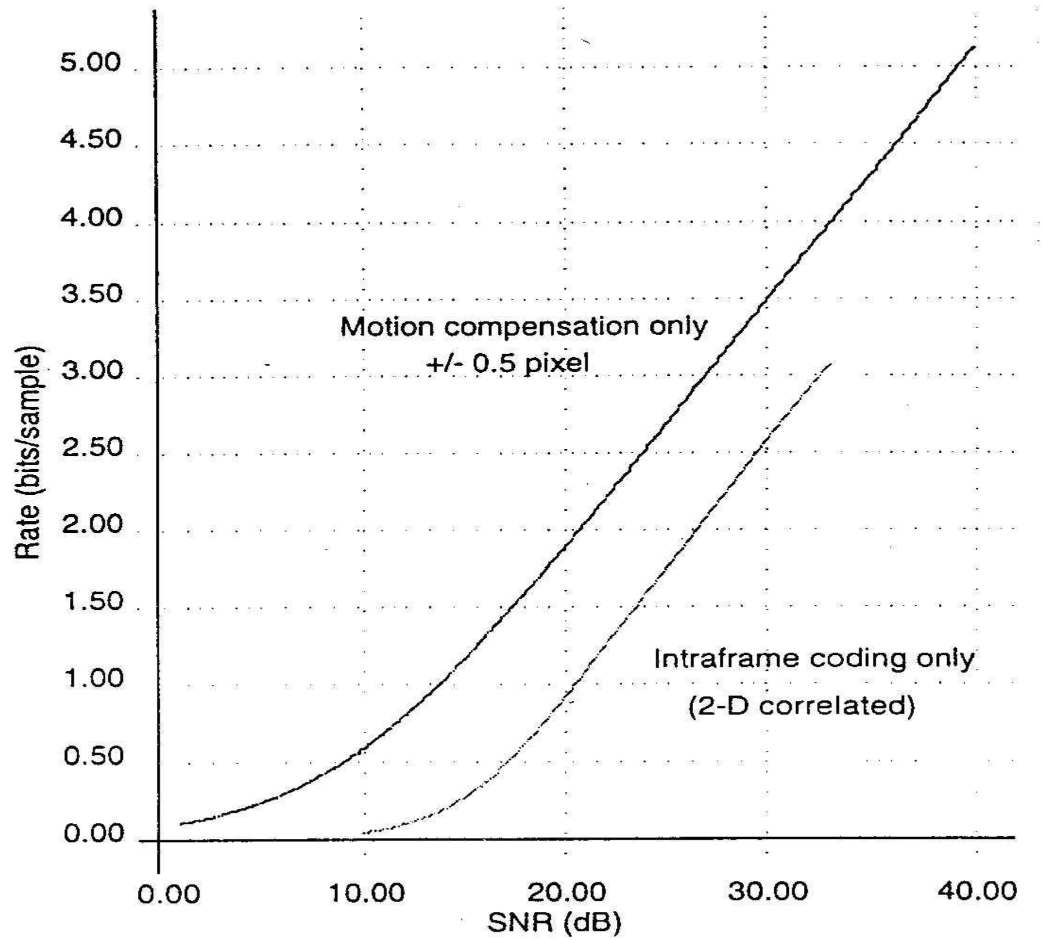


Figure 4.7 Rate-distortion performance for motion-compensated prediction versus intraframe coding.

R-D Function of Hybrid-coding

- Hybrid coding can get better results.

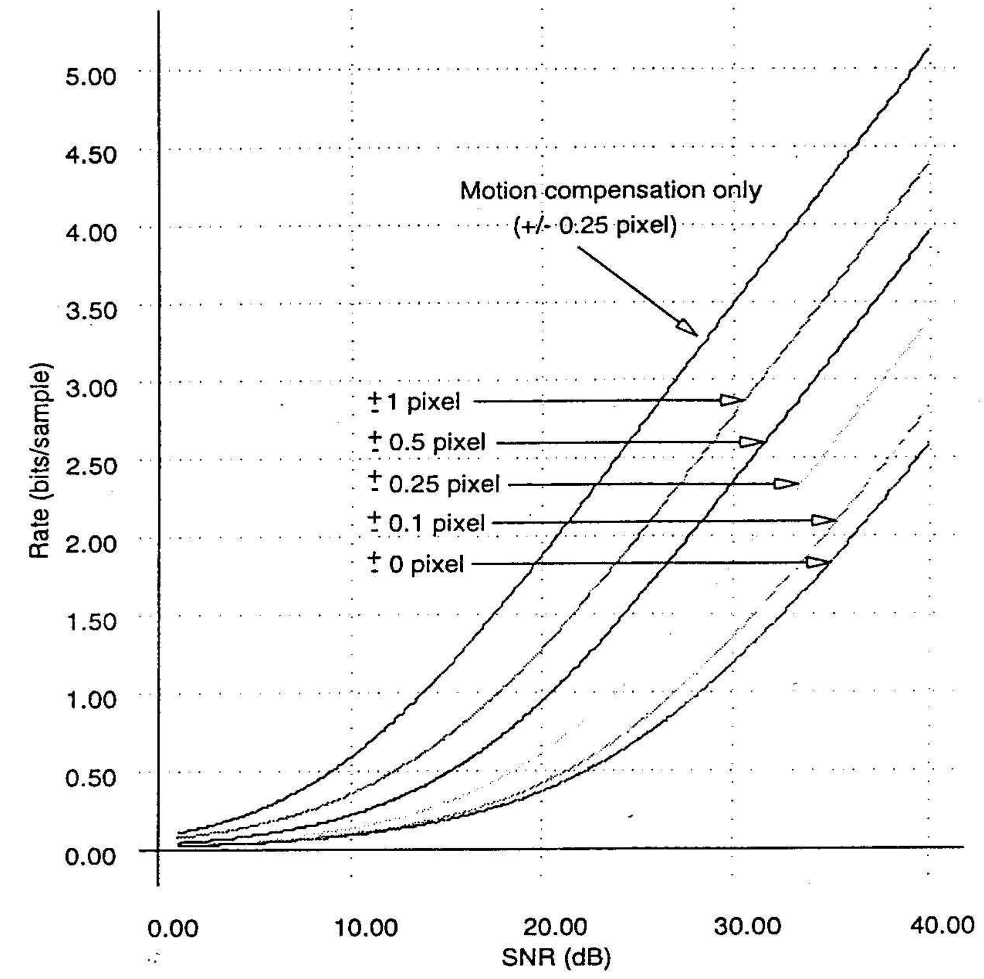
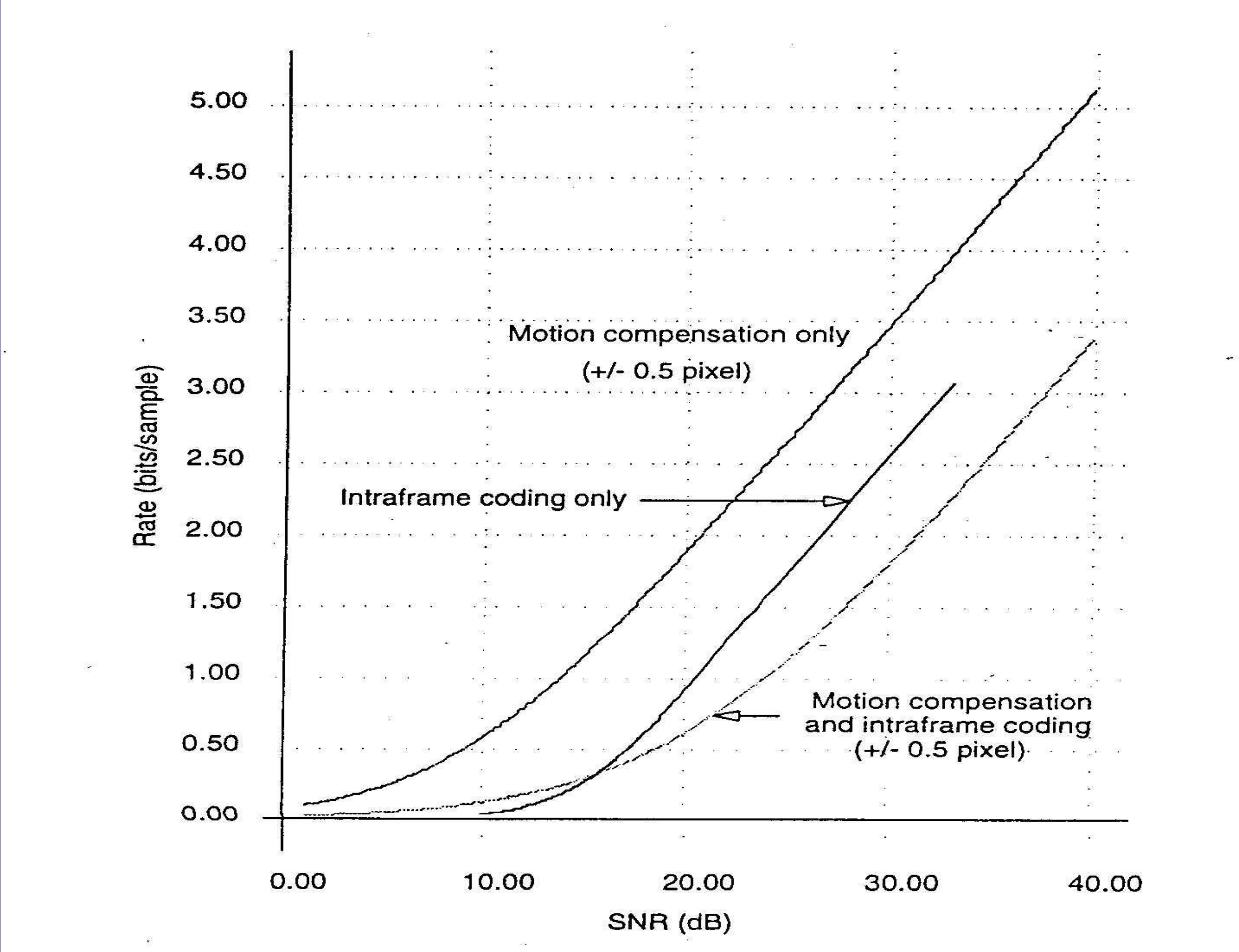


Figure 4.8 Rate-distortion functions for hybrid video coding.

R-D Function (Hybrid vs. Inter vs. Intra)



Block-Matching Motion Estimation

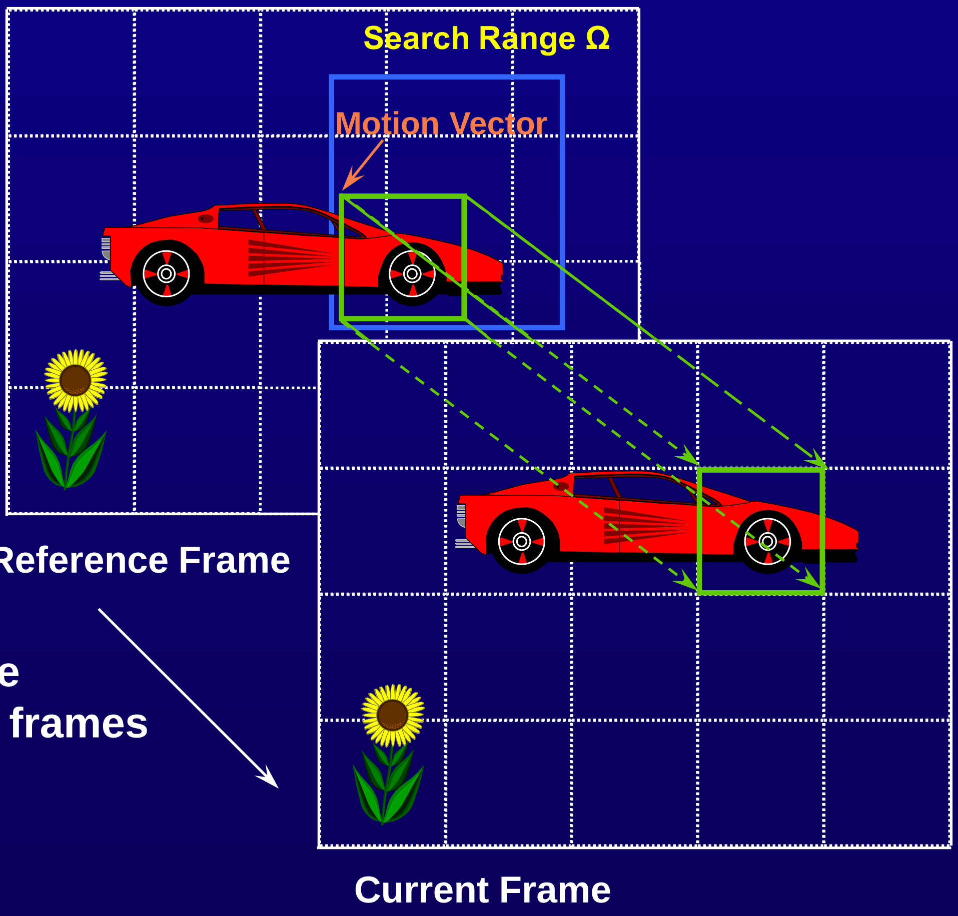
$X_t(p,q)$

the block at location *(p,q)* in *t*-th frame

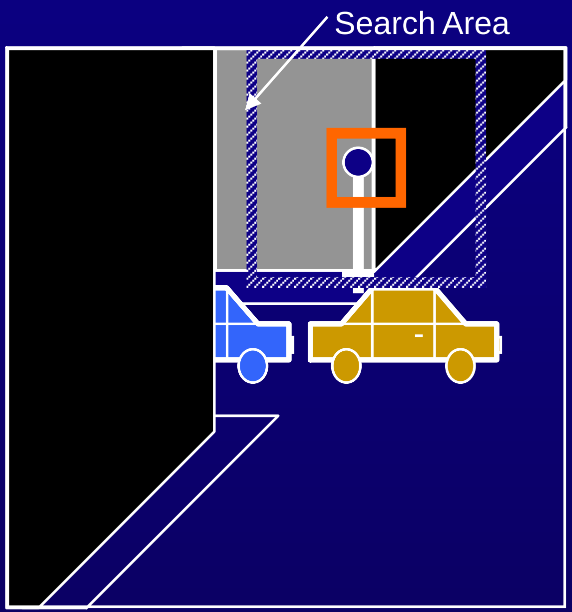
Motion Vector

$V_t(p.q) = (Vec_i, Vec_j)$

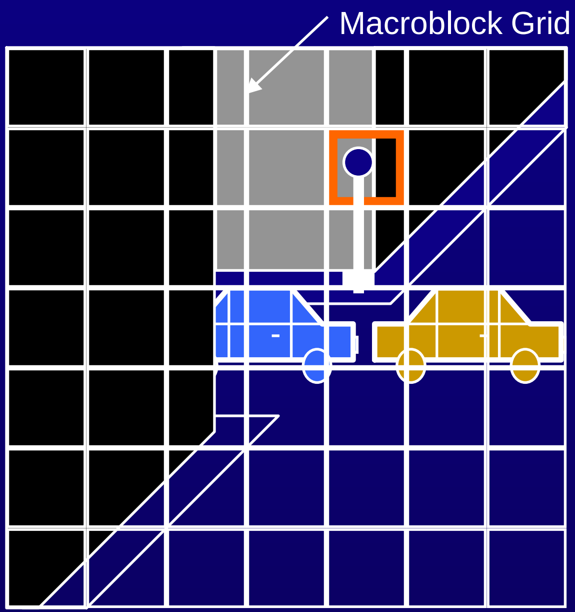
the location in the search range Ω
that has the maximum correlation value
between blocks in temporally adjacent frames



Example of Forward Motion Estimation

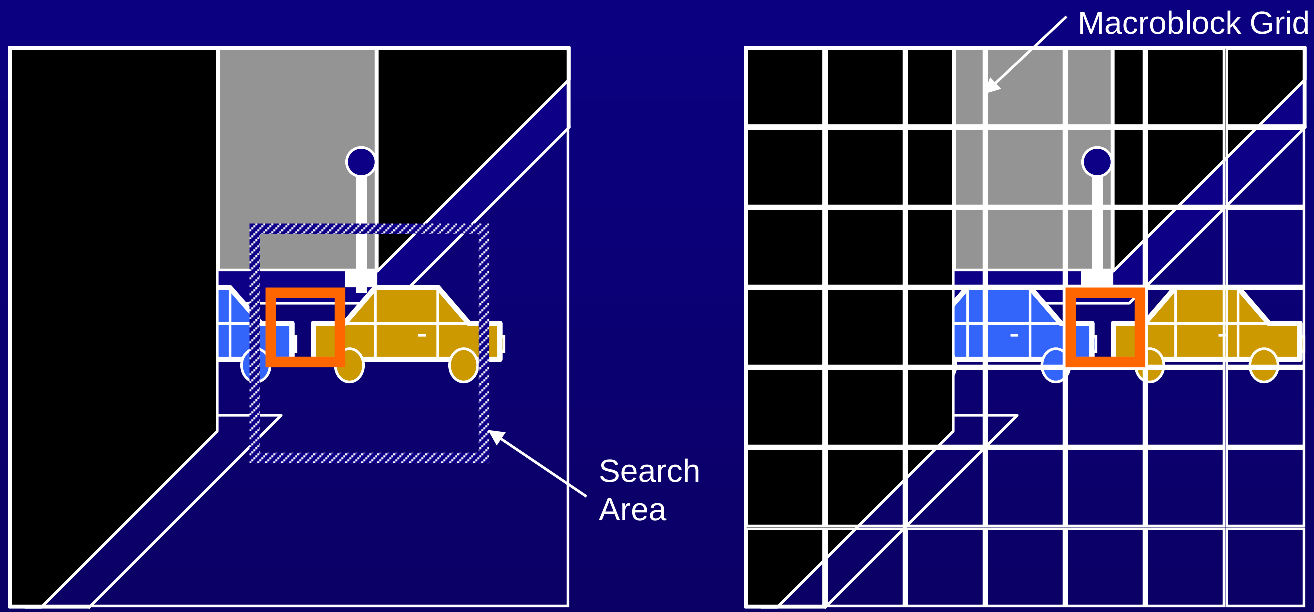


Previous I or P Picture.
Within the search area, a
good match is found
for this still object.



Current P Picture.
Current MB is shown
with heavy outline. Since
a match is found, this
MB is intercoded.

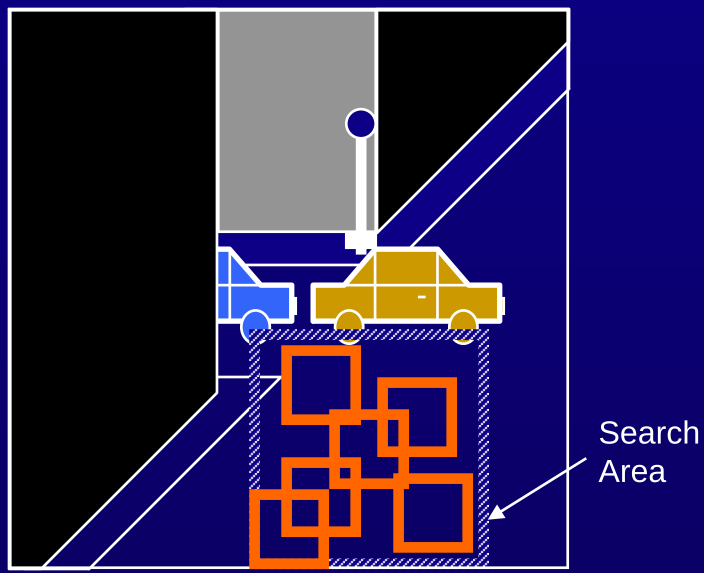
Example of Forward Motion Estimation



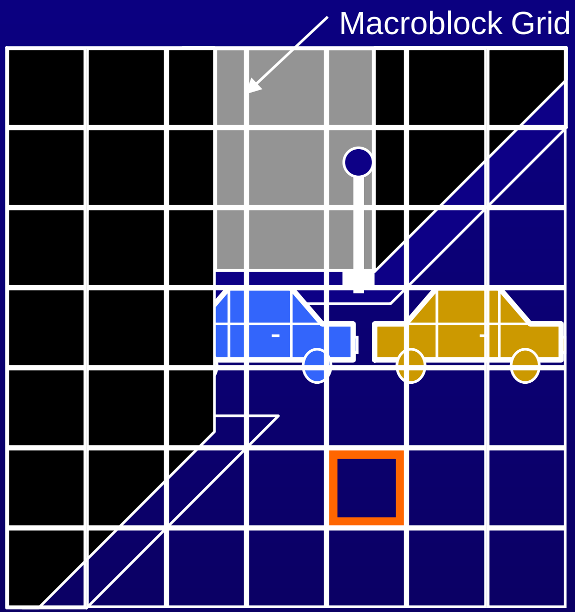
Previous I or P Picture.
Within the search area, a good match is found for this moving object. Encoder sends appropriate forward motion vector.

Current P Picture.
Current MB is shown with heavy outline. Since a match is found, this MB is intercoded.

Example of Forward Motion Estimation

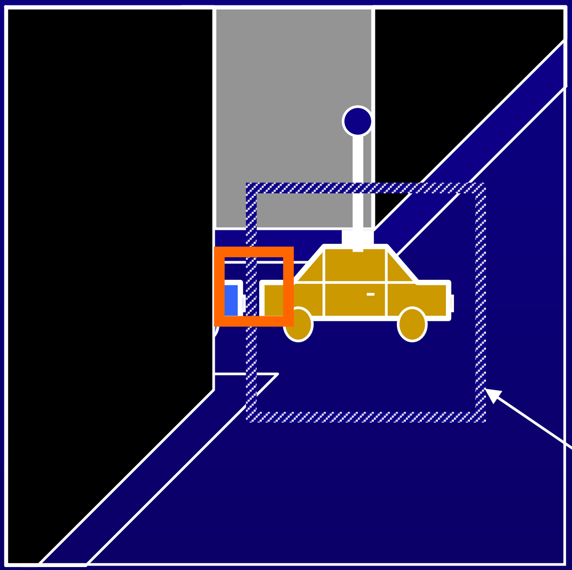


Previous I or P Picture.
Within the search area, many good matches are found. Encoder must pick one and send appropriate motion vector.

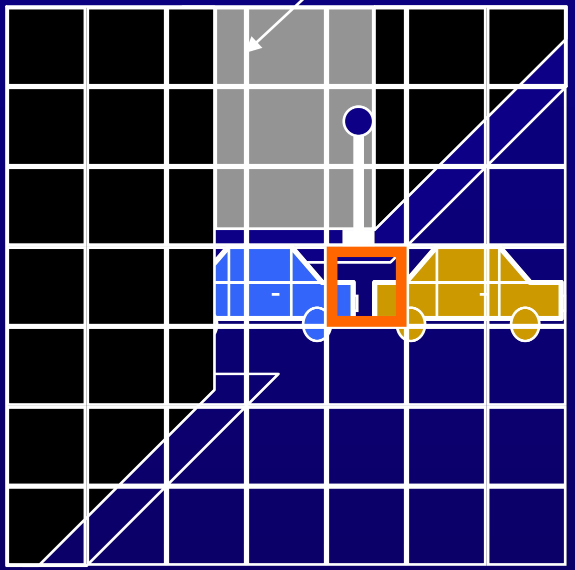


Current P Picture.
Current MB is shown with heavy outline. Since a match is found, this MB is intercoded.

Example of Forward Motion Estimation

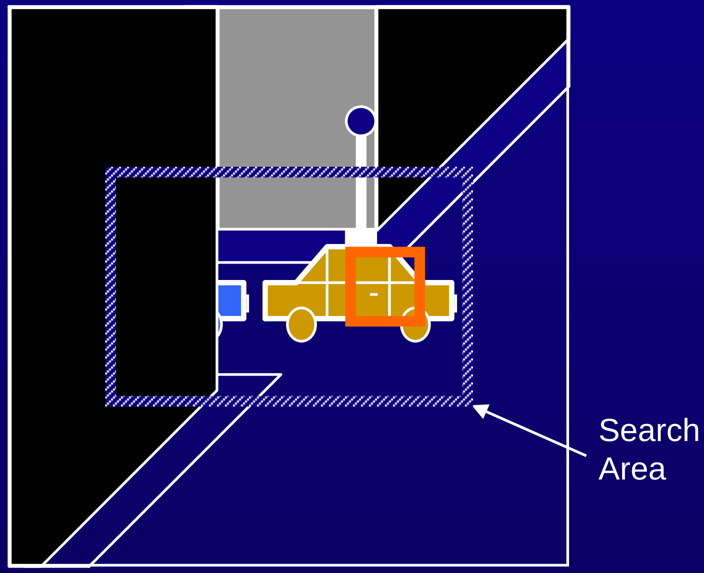


Previous I or P Picture.
Within the search area, no good match is found. Note that a good match would be found with a larger search area. Search area is an important encoder design parameter.

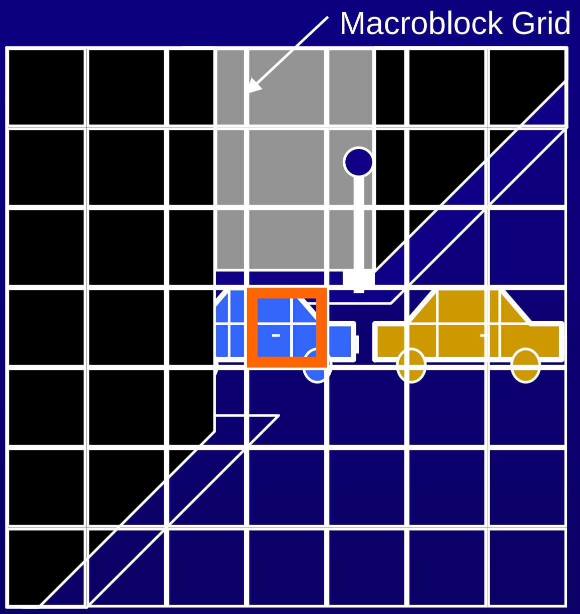


Current P Picture.
Current MB is shown with heavy outline. Since no match is found, this MB is intracoded.

Example of Forward Motion Estimation

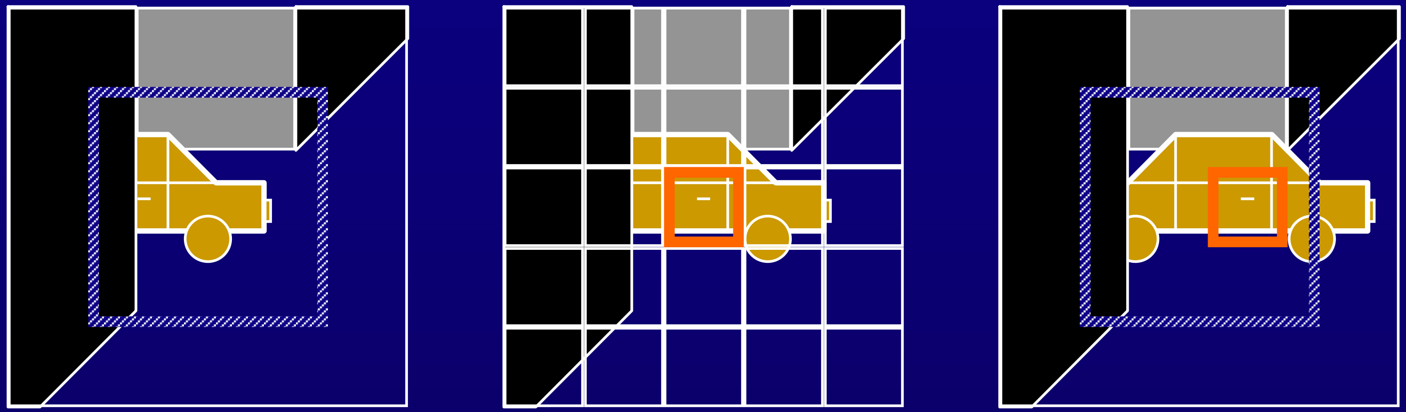


Previous I or P Picture.
Within the search area, a good match is found, but within a different object. There is no requirement that motion vectors represent true motion of objects.



Current P Picture.
Current MB is shown with heavy outline. Since a match is found, this MB is intercoded.

Example of Backward Motion Estimation



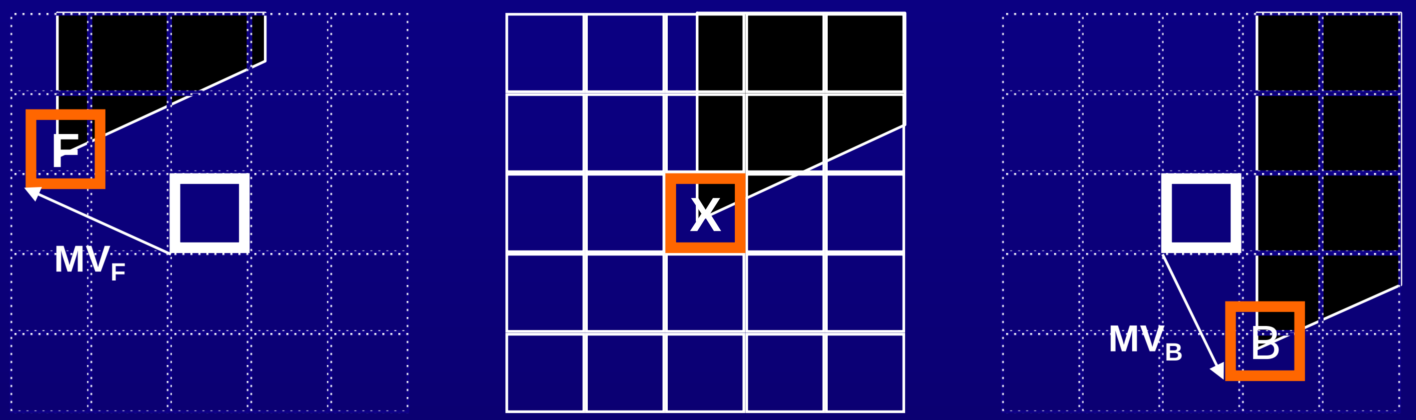
Previous I or P Picture.
Searching here finds no good match because some features are partially hidden.

Current B Picture.
Current MB is shown with heavy outline.

Next I or P Picture.
Searching here finds a good match because features are now uncovered.

forward 前面的當ref. Backward 後面的當ref.

Example of Bidirectional Motion Estimation



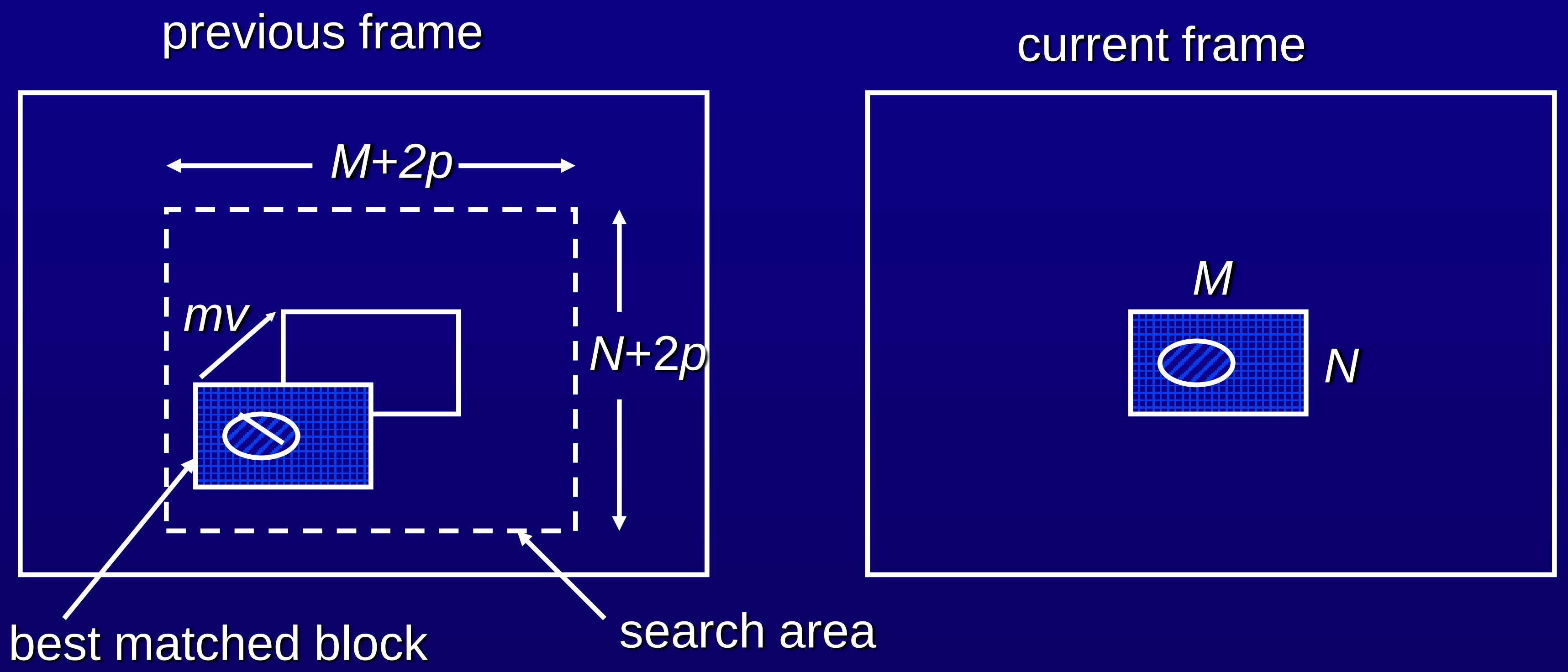
Previous I or P Picture | **Current B Picture** | **Next I or P Picture**

Define:
- X = Current MB
- F = “Best” MB in previous I or P Picture
- B = “Best” MB in next I or P Picture
- MV_F = MV corresponding to F’s displacement from X
- MV_B = MV corresponding to B’s displacement from X

Compute: “Goodness” of F, B and (F+B)/2 as predictors for X

Decide:
- If F is best, send MV_F — Forward Prediction
- If B is best, send MV_B — Backward Prediction
- If (F+B)/2 is best, send MV_F and MV_B — Interpolated Prediction

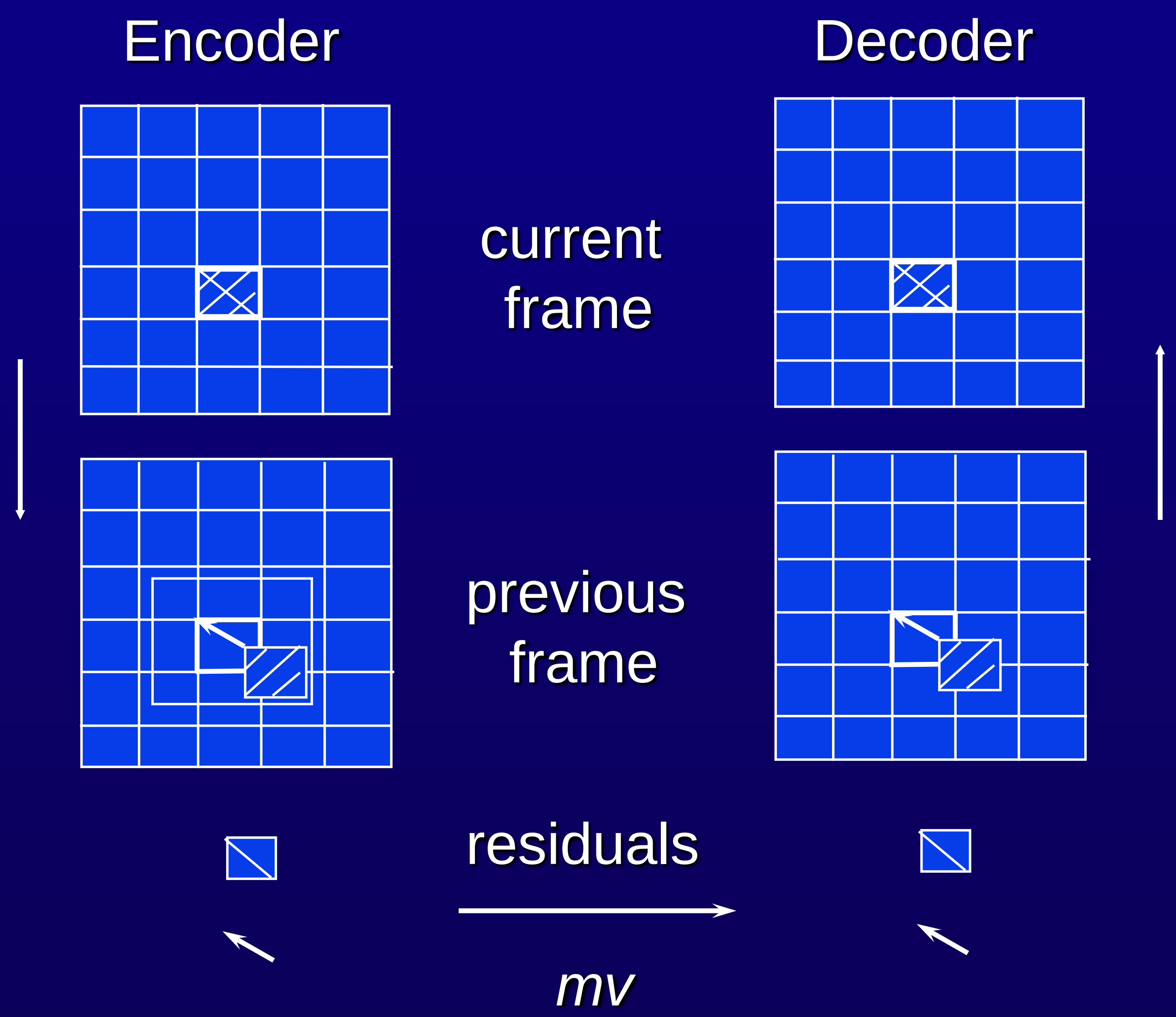
Block Matching Algorithm (BMA)



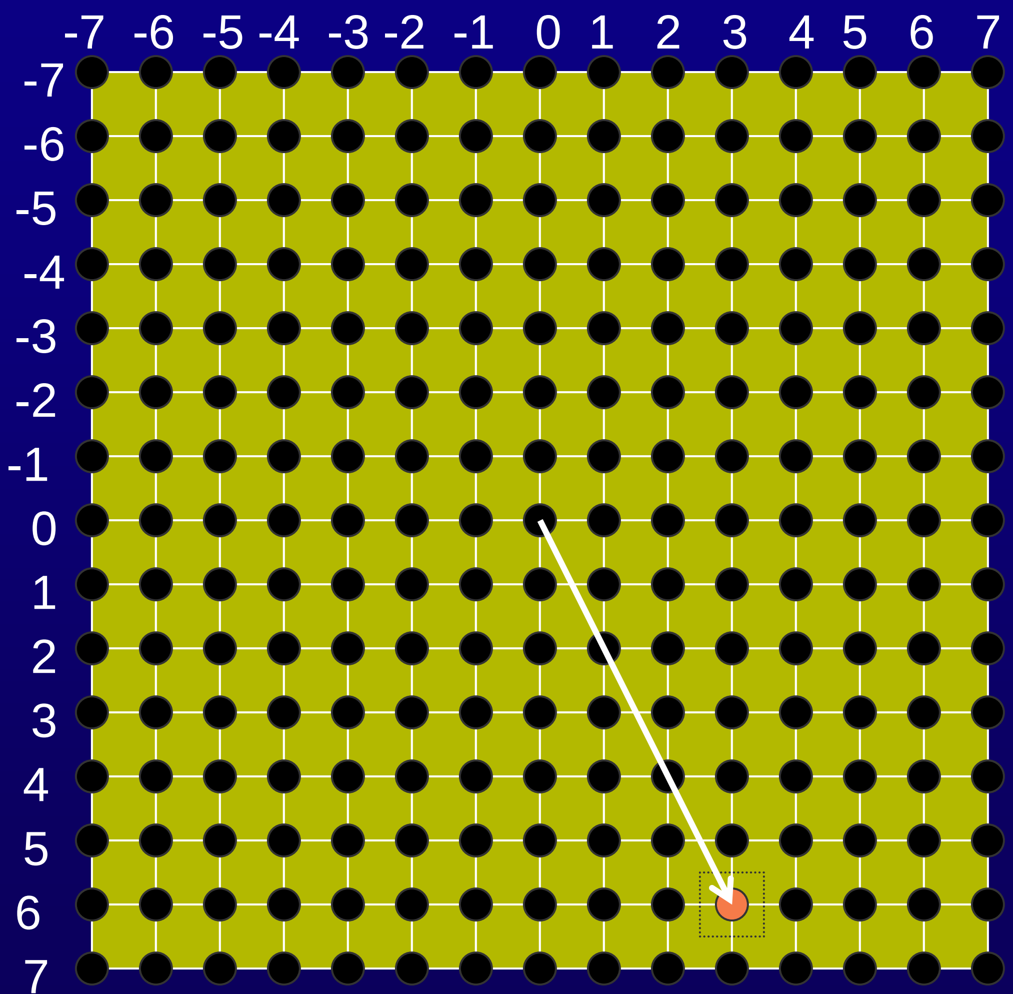
- one motion vector (*mv*) per block
- matching criteria: mean-absolute-differences (*MAD*), mean-squared-differences (*MSD*), cross-correlation-function (*CCF*)

通常左右尋找best matched block 範圍會比上下多(所以通常非對稱)~因為人類的習性，CCF越大越好，另外兩個越小越好。MAD計算最簡單

Motion Compensated Prediction (MCP) Using BMA



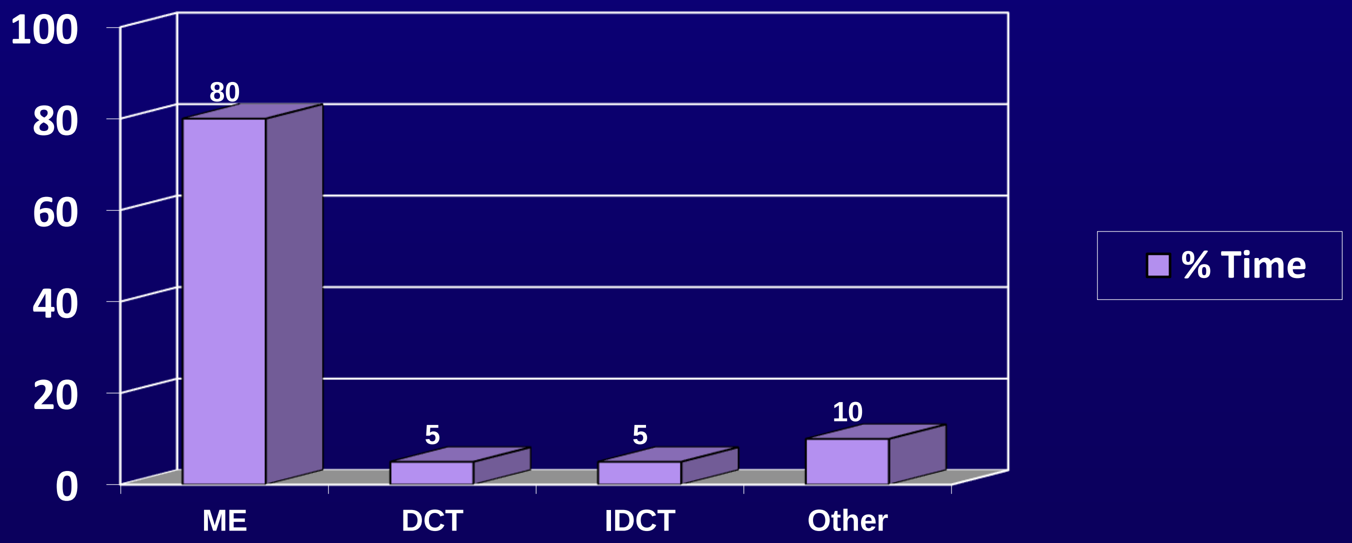
Full-Search BMA Using MAD



Two-Dimensional Full Search Procedure Search Range -7 to +7 Motion Vector (3,6) in this case

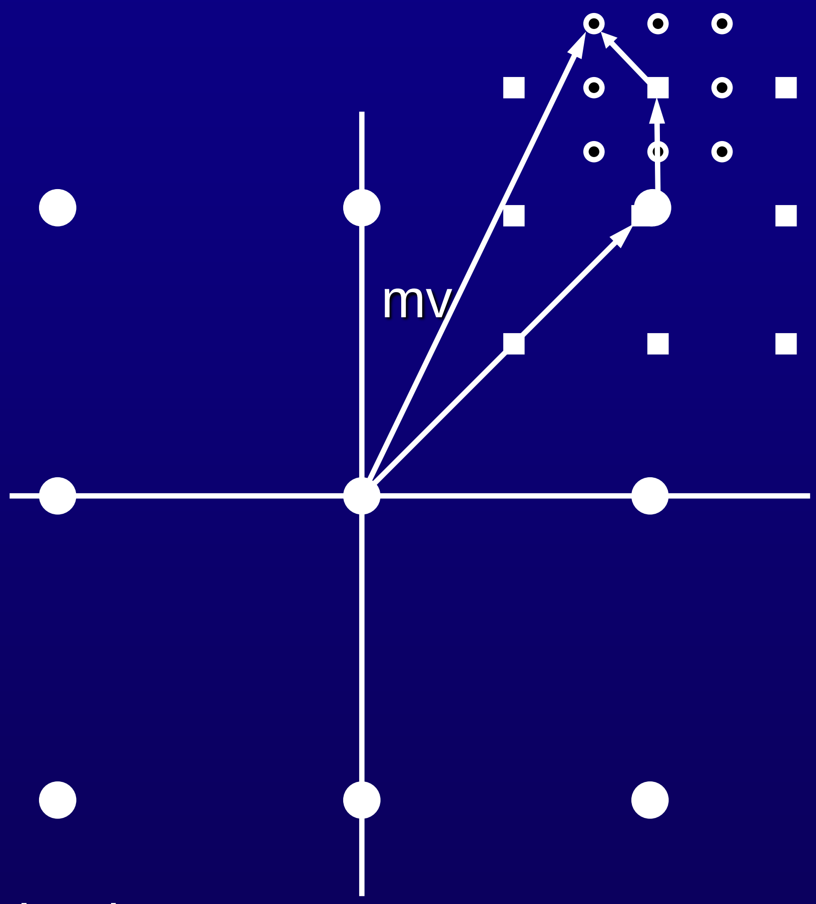
Need for Fast Motion Estimation

- Motion estimation is the most time-consuming operation in a typical video encoder



Fast Search Methods for BMA

- 2-D logarithmic search
- three step search
- conjugate direction search
- ...

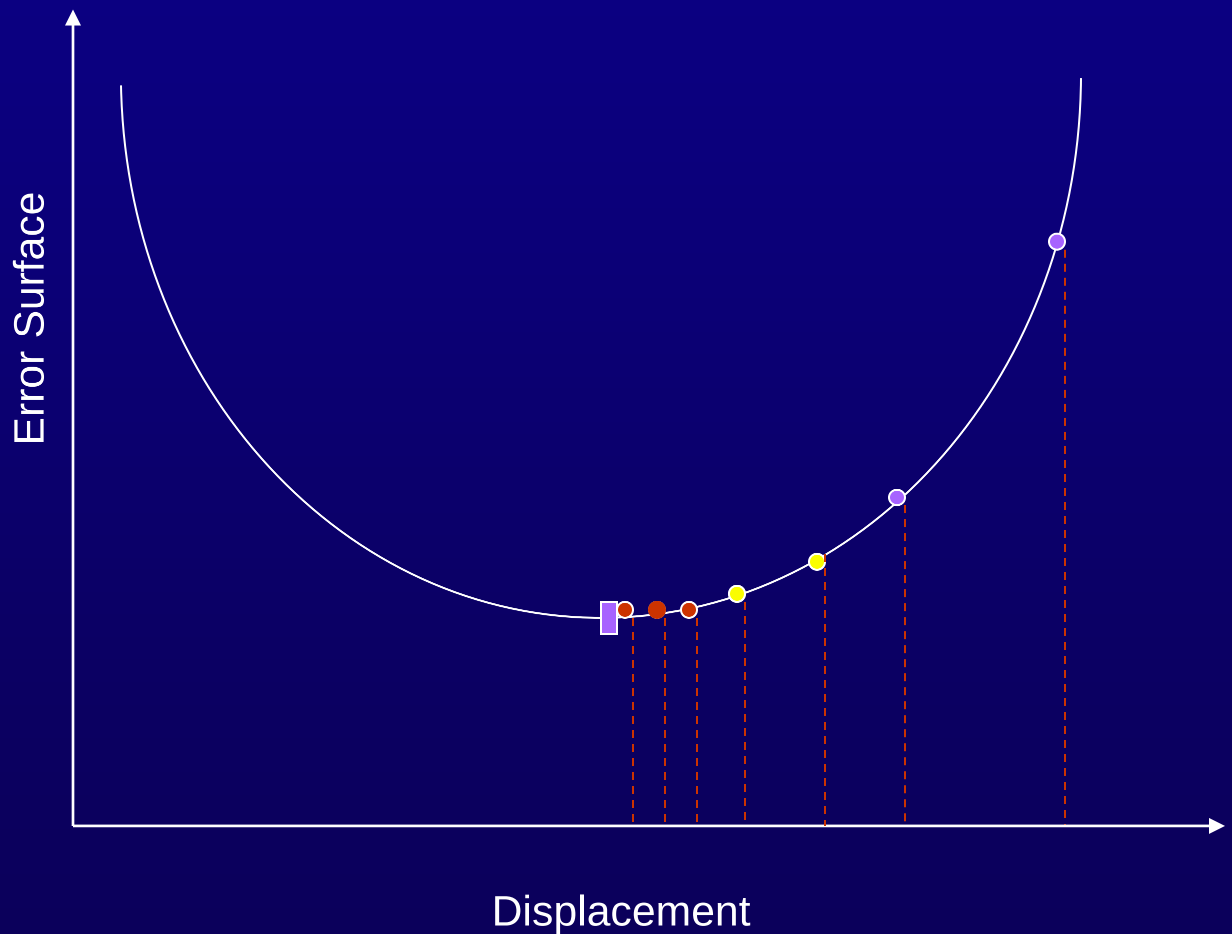


fast search是為了加速ME(佔
整體時間大部分，所以要加速)

- greatly reduces the computation load
- sub-optimal solution, may be trapped at local minima

Fast Search: Strategy I

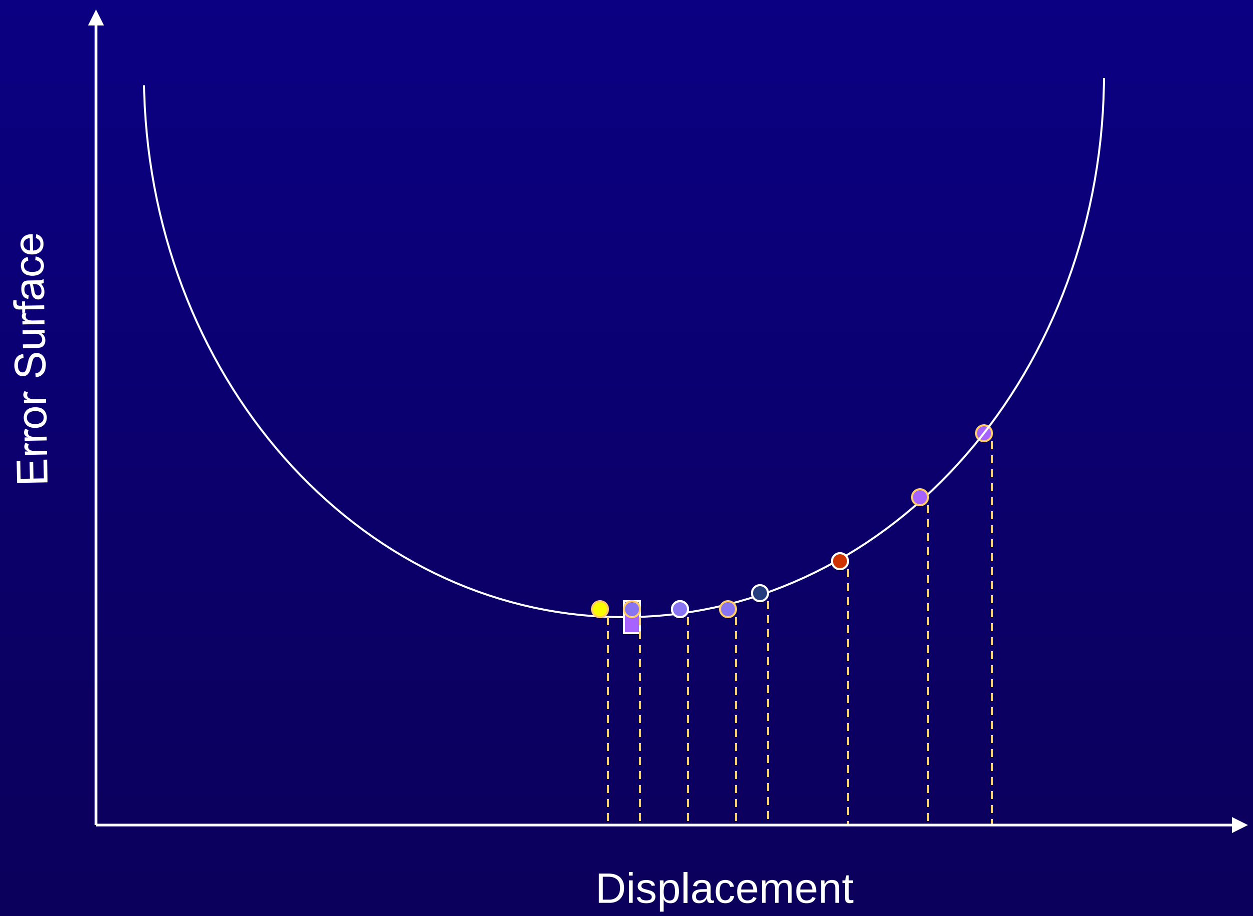
- Pick a large step size around the starting point.
- At every stage, reduce the step size and move in the direction of the best match.



保證可以找到某個解，
但不確定為最佳解，
若為smooth才會得
到最佳解

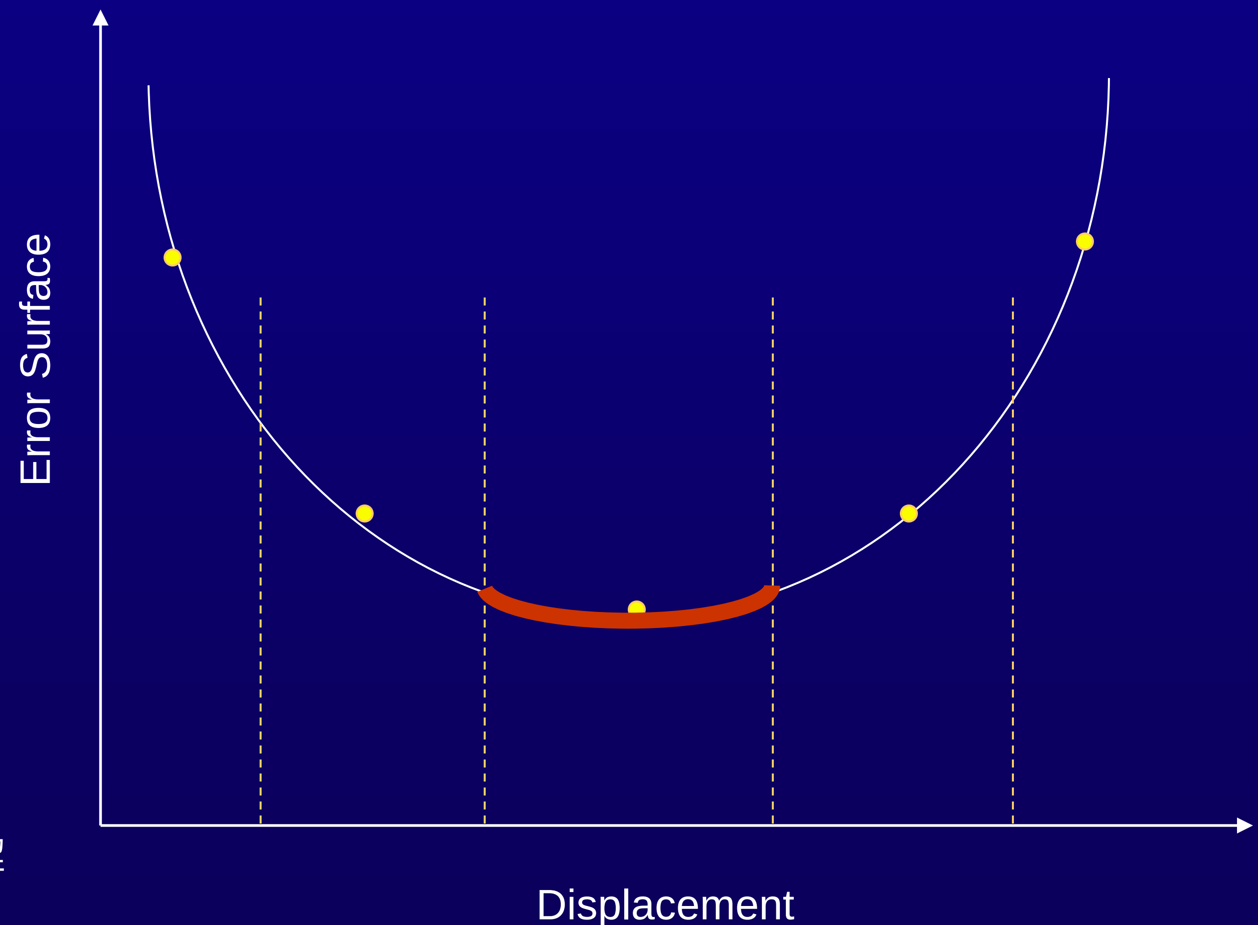
Fast Search: Strategy II

- Move in direction of best match.
- Reduce step size only on overshoot.



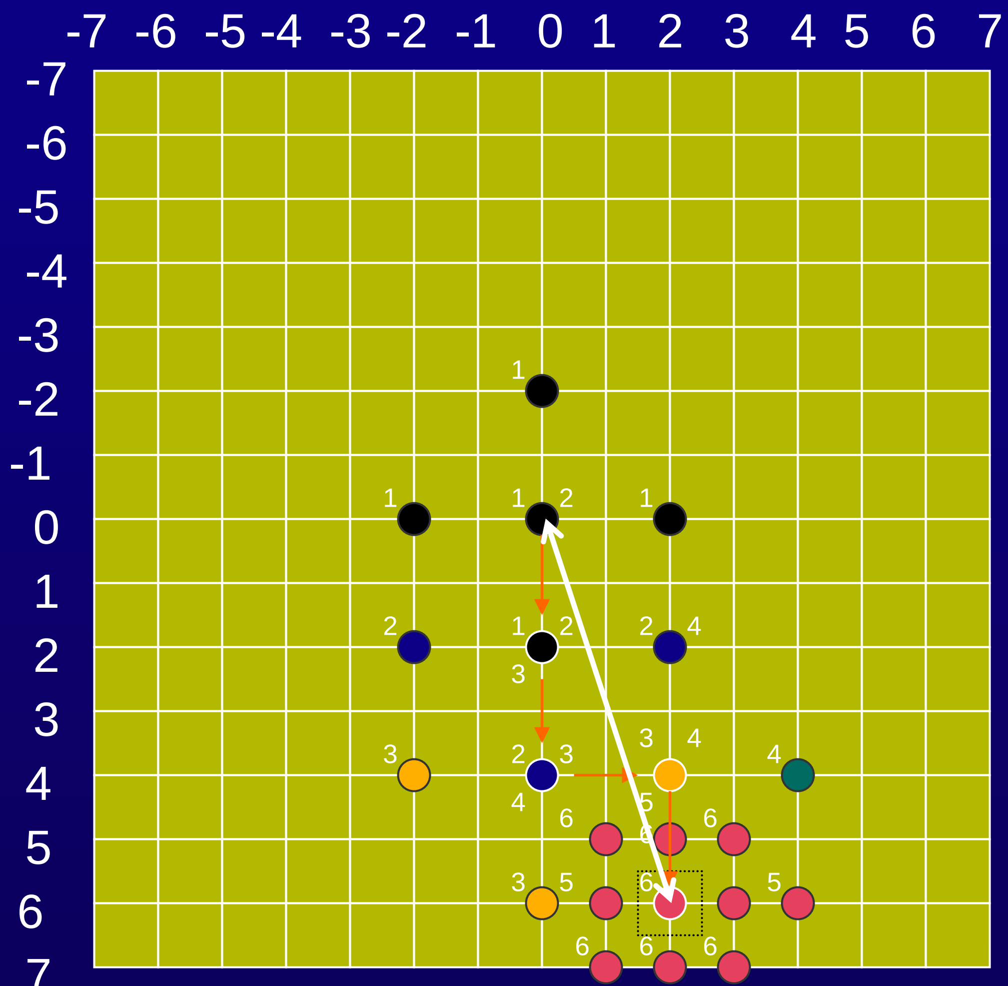
Fast Search: Strategy III

- Divide search space into many regions.
- Pick a center point for each region.
- Do a full search over region with best center.



三個方法都會卡在局部解
Full search為最常用的

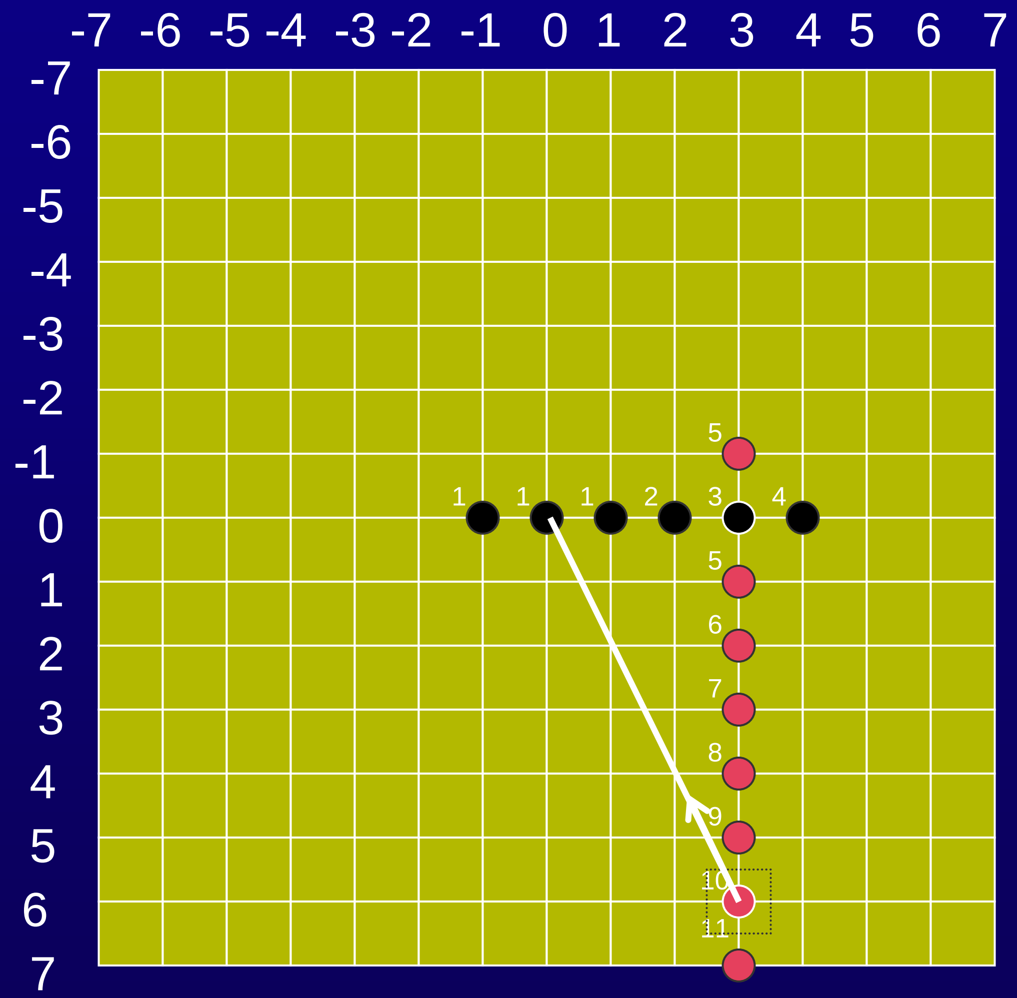
Fast ME: 2-D Logarithm Search



找sub-optimal

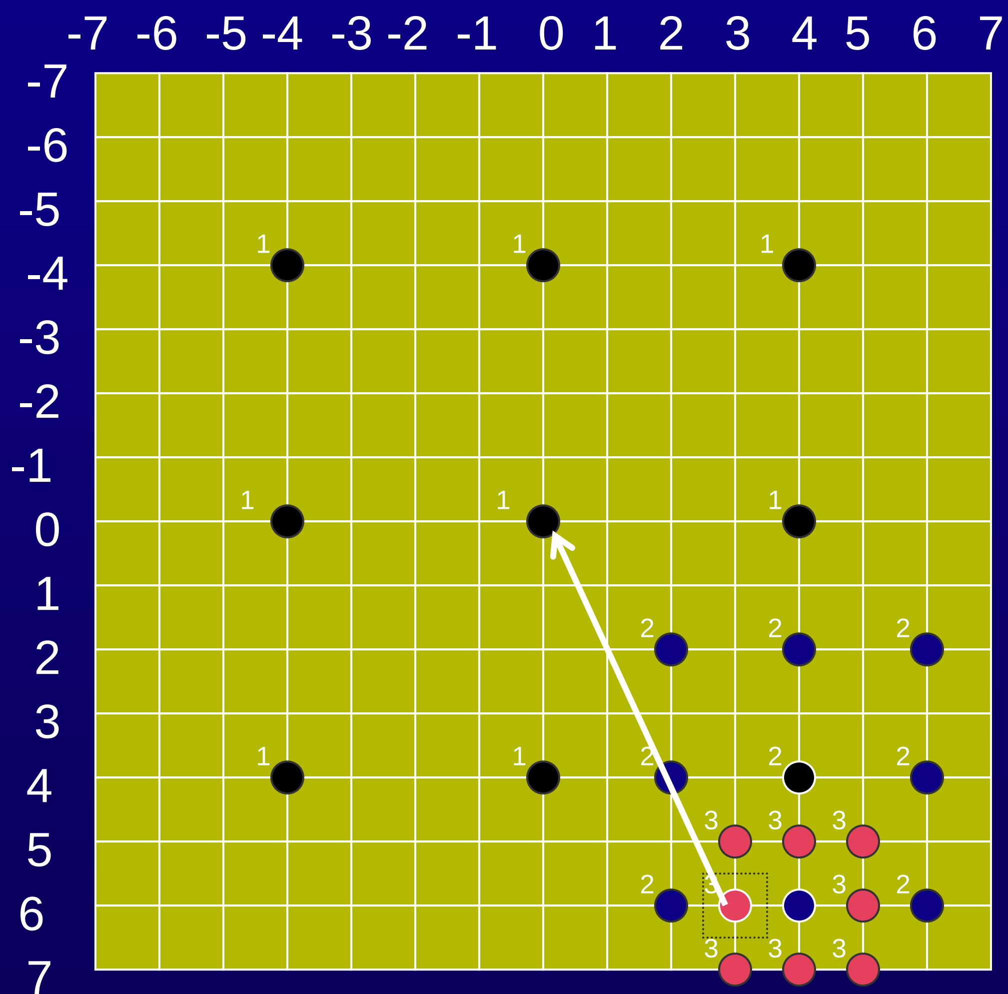
2-D Logarithmic Search Procedure Search Range -7 to +7 Motion Vector (3,6) in this case

Fast ME: One-At-A-Time Search (OTS)



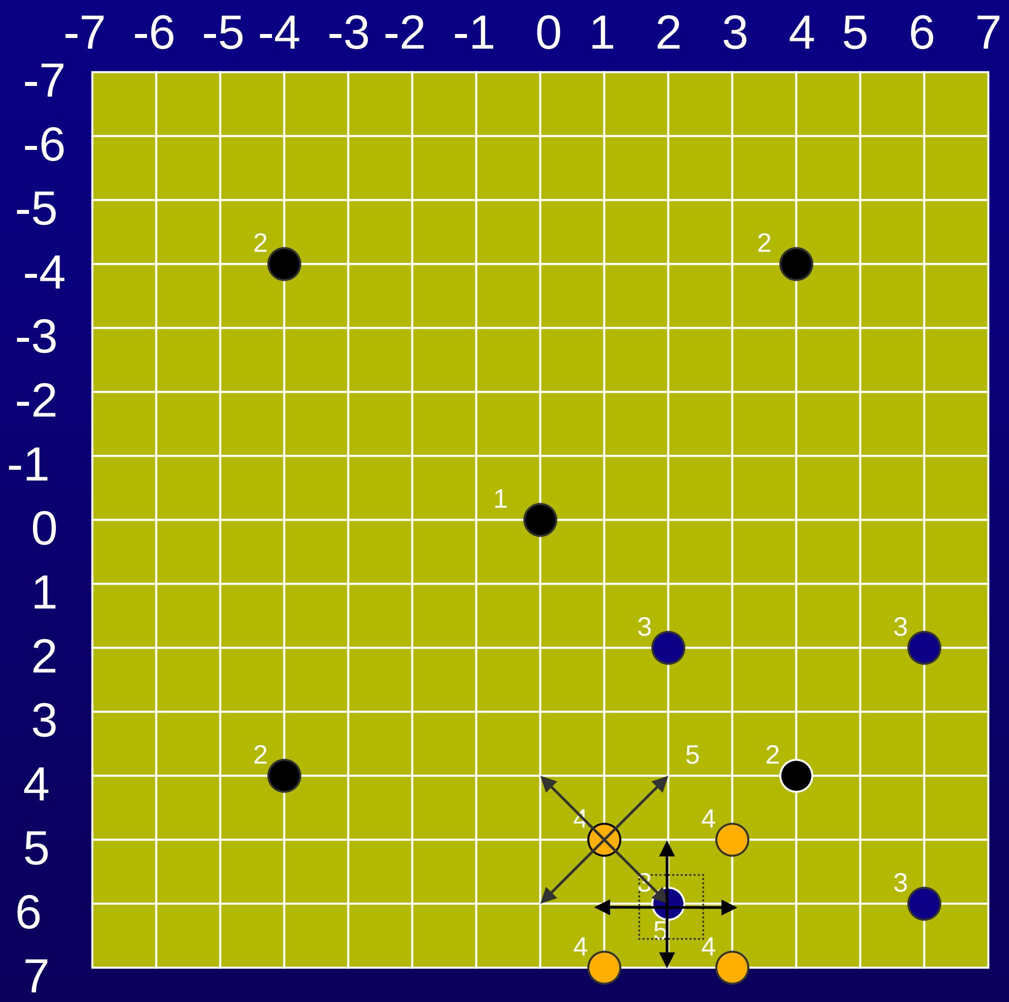
One-at-a-time Search (OTS) Procedure Search Range -7 to +7 Motion Vector (3,6) in this case

Fast ME: Three-Step Search (TSS)



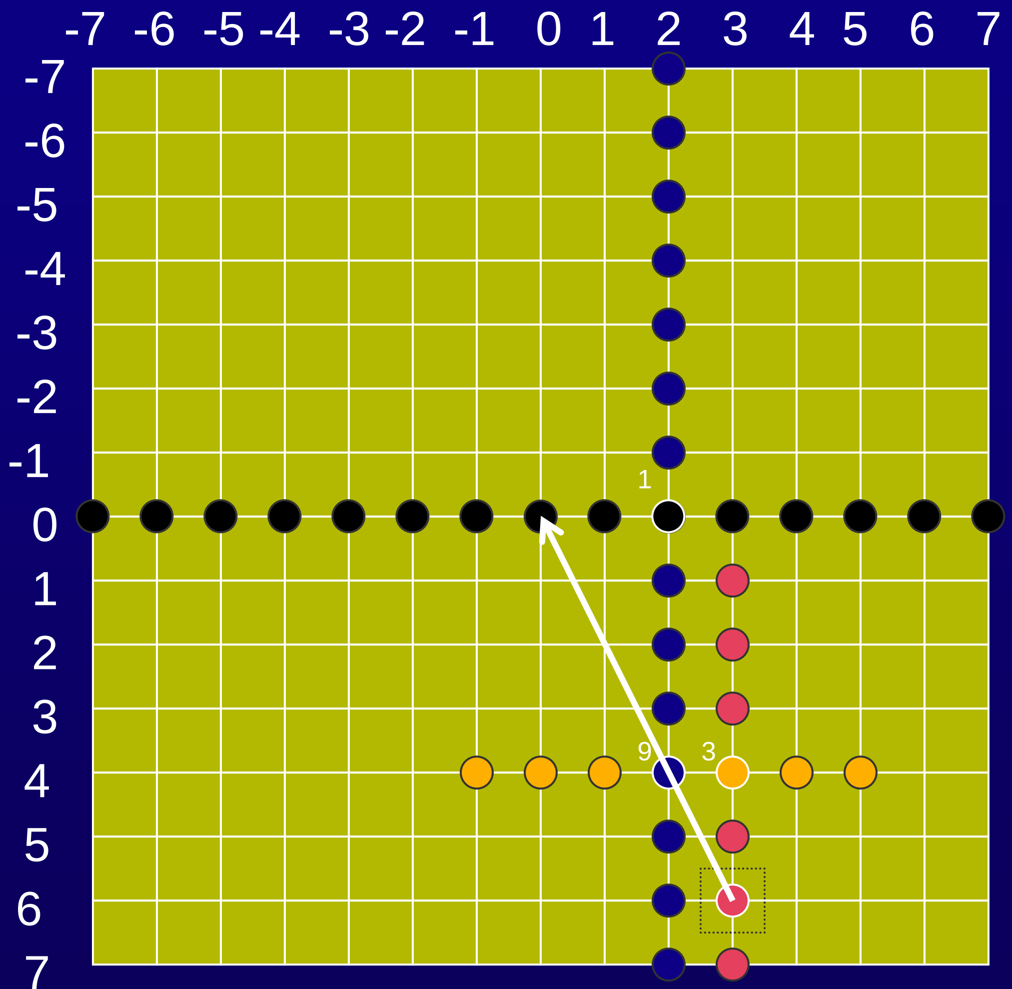
Three-Step Hierarchical Search Procedure Search Range -7 to +7 Motion Vector (3,6) in this case

Fast ME: Cross-Search Algorithm (CSA)



Cross-Search Procedure Search Range -7 to +7 Motion Vector (3,6) in this case

Fast ME: 1-D Full Search (1DFS)



One-Dimensional Full Search Procedure Search Range -7 to +7 Motion Vector (3,6) in this case

Computational Complexity of Fast Search Algorithms

Algorithm	Max number of search points	w		
		4	8	16
FSA	$(2w+1)^2$	82	289	1089
2D log	$2+7\log_2 w$	16	23	30
TSS	$1+8\log_2 w$	17	25	33
CDS	$3+2w$	11	19	35
CSA	$5+4\log_2 w$	13	17	21

Block Matching Criteria

- Cross-Correlation Function (CCF)

$$CCF(i,j) = \frac{\sum_{k=1}^{N}\sum_{l=1}^{N} x_t(k,l) \cdot x_{t-1}(k+i,l+j)}{[\sum_{k=1}^{N}\sum_{l=1}^{N} x_t^2(k,l)]^{1/2}[\sum_{k=1}^{N}\sum_{l=1}^{N} x_{t-1}^2(k+i,l+j)]^{1/2}}$$

$$(Vec_i, Vec_j) = \arg\max_{i,j} CCF(i,j)$$

- Mean Squared Errors (MSE)

$$MSE(i,j) = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}[x_t(k,l) - x_{t-1}(k+i,l+j)]^2$$

$$(Vec_i, Vec_j) = \arg\min_{i,j} MSE(i,j)$$

Block Matching Criteria (Cont.)

轉到YCbCr才能做，且只做Y

- Mean Absolute Errors (MAE)

$$MAE(i,j) = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}|x_t(k,l) - x_{t-1}(k+i,l+j)|$$

$$(Vec_i, Vec_j) = \arg\min_{i,j} MAE(i,j)$$

- Pel Difference Classification (PDC)

$$T(k,j,i,j) = 1, if\,|x_t(k,l) - x_{t-1}(k+i,l+j)| \leq \text{Threshold}$$

$$G(i,j) = \sum_{k=1}^{N}\sum_{l=1}^{N} T(k,l,i,j)$$

$$(Vec_i, Vec_j) = \arg\max_{i,j} G(i,j)$$

Motion Vector Prediction

$$V_t(p,q) = E\left[\hat{V}_t(p,q)\right] + \Delta\, V_t(p,q), |\Delta V_t(p,q)| \le \omega$$

- Temporal Prediction
  - Inter-Frame Prediction
- Spatial Prediction
  - Inter-Block Prediction
  - Median Vector Prediction
  - Boundary Match Prediction
  - Side Match Prediction

$E[\hat{V}_t(p,q)]$: ***Predicted Motion Vector***

$\Delta V_t(p,q)$: ***Refined Displacement***

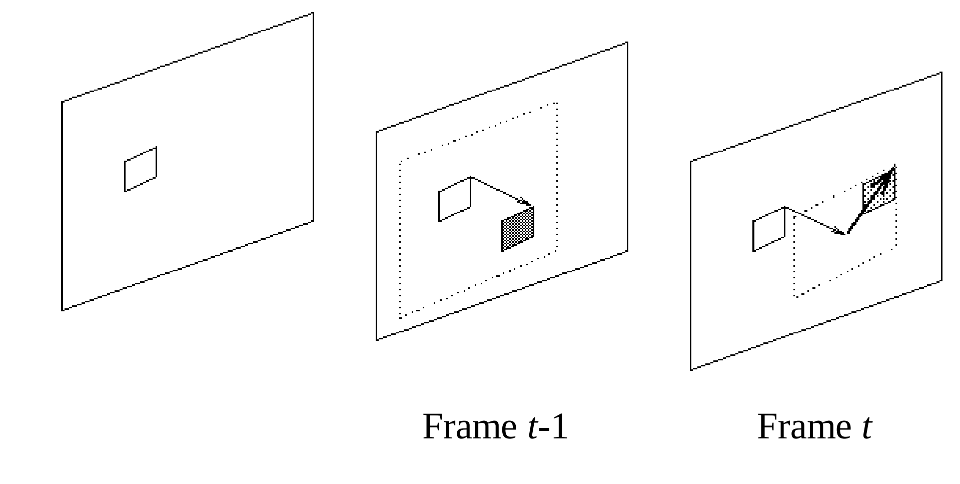
ω : ***Reduced Search Range***

(e.g., one-quarter of search range Ω ***)***

Motion Vector Prediction (Cont.)

Inter-Frame Prediction

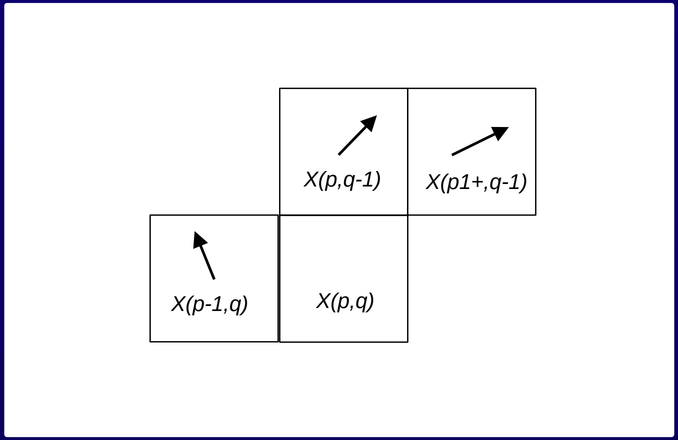
$$E[\hat{V}_t(p,q)] = V_{t-1}(p,q)$$



Median Vector Prediction

$$E[\hat{V}_t(p,q)] = \arg\underset{v \in V_{cs}}{\mathrm{Median}}\ V$$

$$V_{cs} = \{V_t(p,q-1), V_t(p-1,q), V_t(p+1,q-1)\}$$



Predictive ME Using Boundary Matching

$[X(p,q)]_l^c = (x_{l1}, x_{l2}, \ldots, x_{ln})^T$ **column vector**

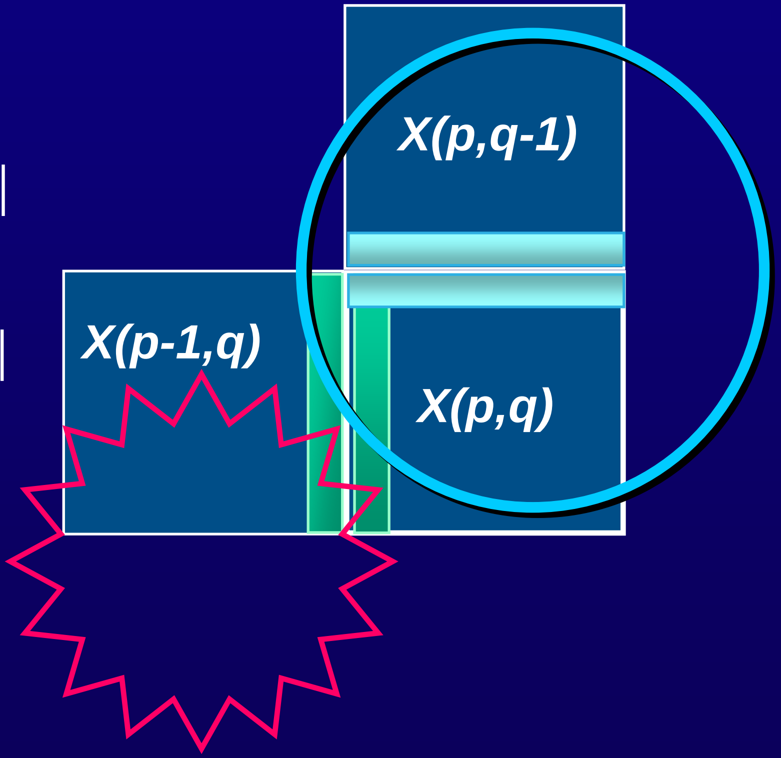
$[X(p,q)]_l^c = (x_{l1}, x_{l2}, \ldots, x_{ln})^T$ **row vector**

$$d_{bm,V_t(p,q-1)} = \sum_{j-1}^{N} \left| [X(p,q)]_{1,j}^r - [X(p,q-1)]N_{N,j}^r \right|$$

$$d_{bm,V_t(p-1,q)} = \sum_{j-1}^{N} \left| [X(p,q)]_{1,j}^r - [X(p-1,q)]N_{N,j}^r \right|$$

$$E[\hat{v}_t(p,q)] = arg \min_{v \in V_{cs}} d_{bm,v}$$

$$V_{cs} = \{V_t(p,q-1), V_t(p-1,q)\}$$



Predictive ME Using Side Matching

$$d_{smU} = \sum_{j=1}^{N} \left| [X_{MC}(p,q)]_{1,j}^{r} - [X(p,q-1)]N_{N,j}^{r} \right|$$

$$d_{smL} = \sum_{j=1}^{N} \left| [X_{MC}(p,q)]_{1,j}^{c} - [X(p-1,q)]N_{N,j}^{c} \right|$$

$$d_{smR} = \sum_{j=1}^{N} \left| [X_{MC}(p,q)]_{N,j}^{c} - [X(p+1,q)]N_{1,j}^{c} \right|$$

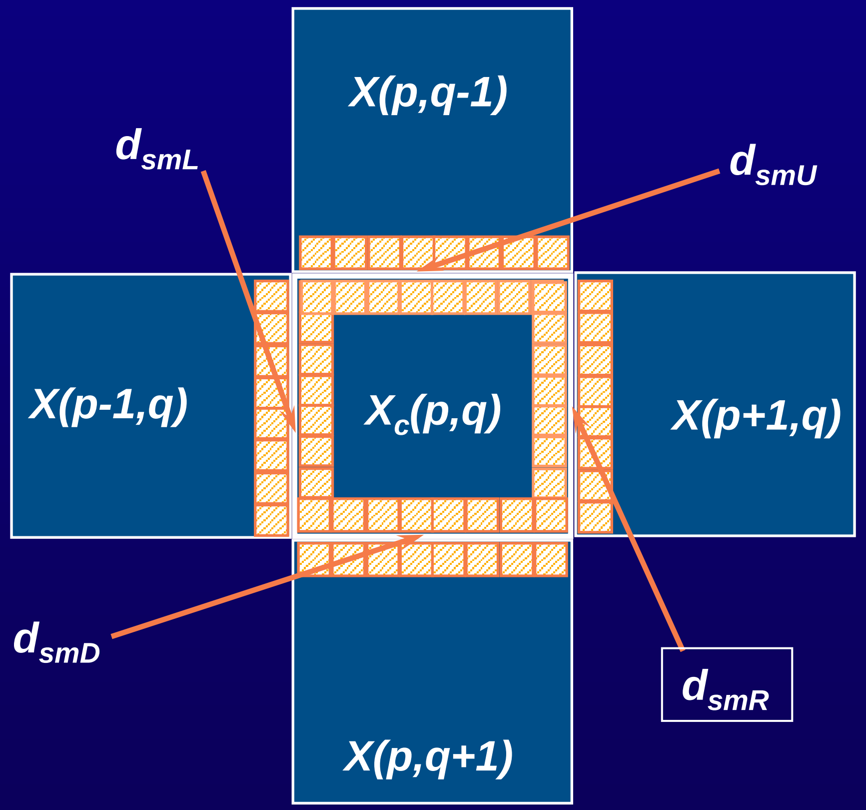
$$d_{smD} = \sum_{j=1}^{N} \left| [X_{MC}(p,q)]_{N,j}^{r} - [X(p,q+1)]N_{1,j}^{r} \right|$$

$$d_{sm} = d_{smU} + d_{smL} + d_{smR} + d_{smD}$$

$$E[\hat{v}_t(p,q)] = arg \min_{v \in V_{cs}} d_{sm,v}$$

$$V_{cs} = \{V_t(p,q-1), V_t(p-1,q), V_t(p+1,q-1)\}$$

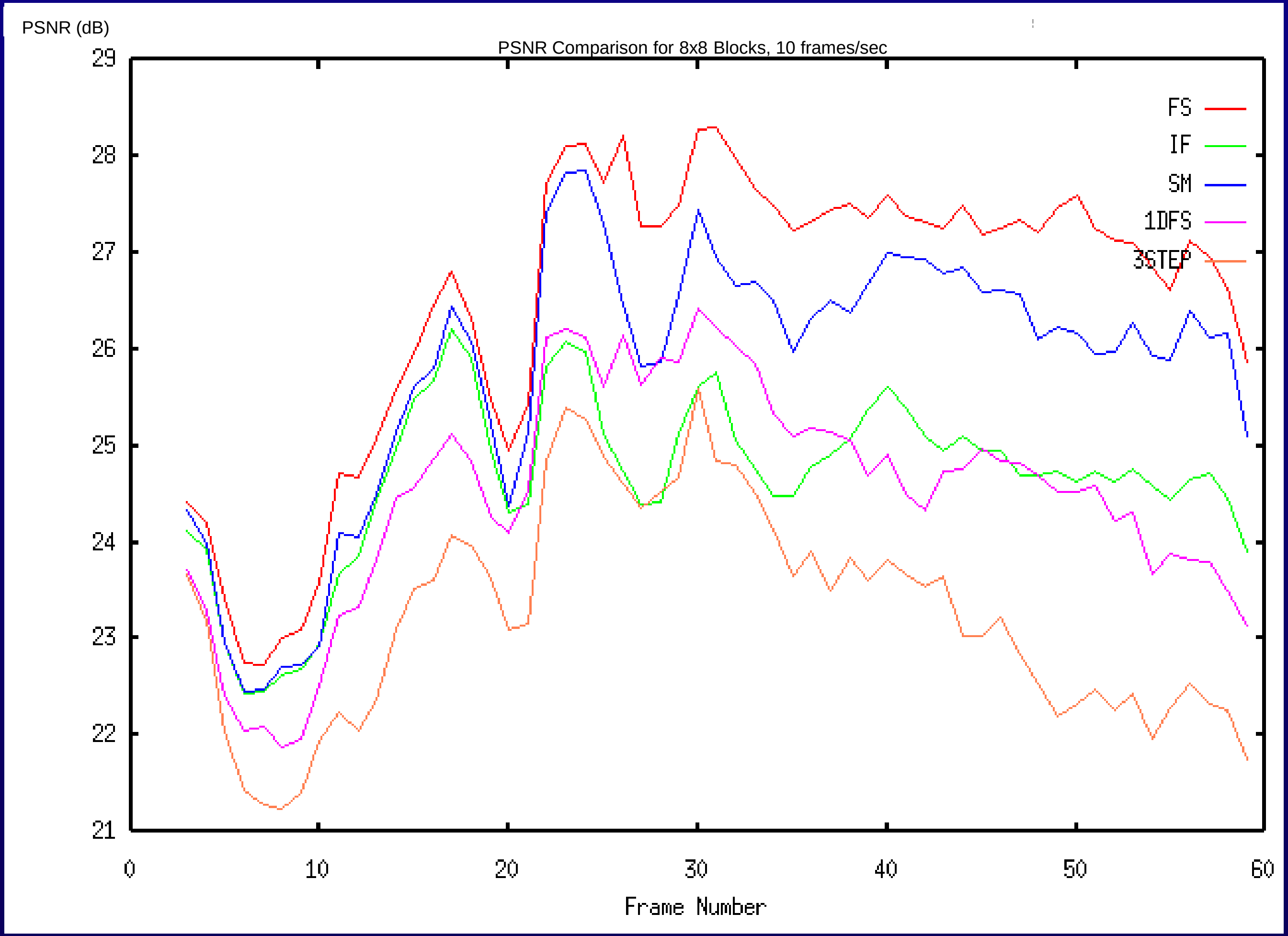
minimum gray-level transition across four boundaries



Predictive ME Using Side Matching

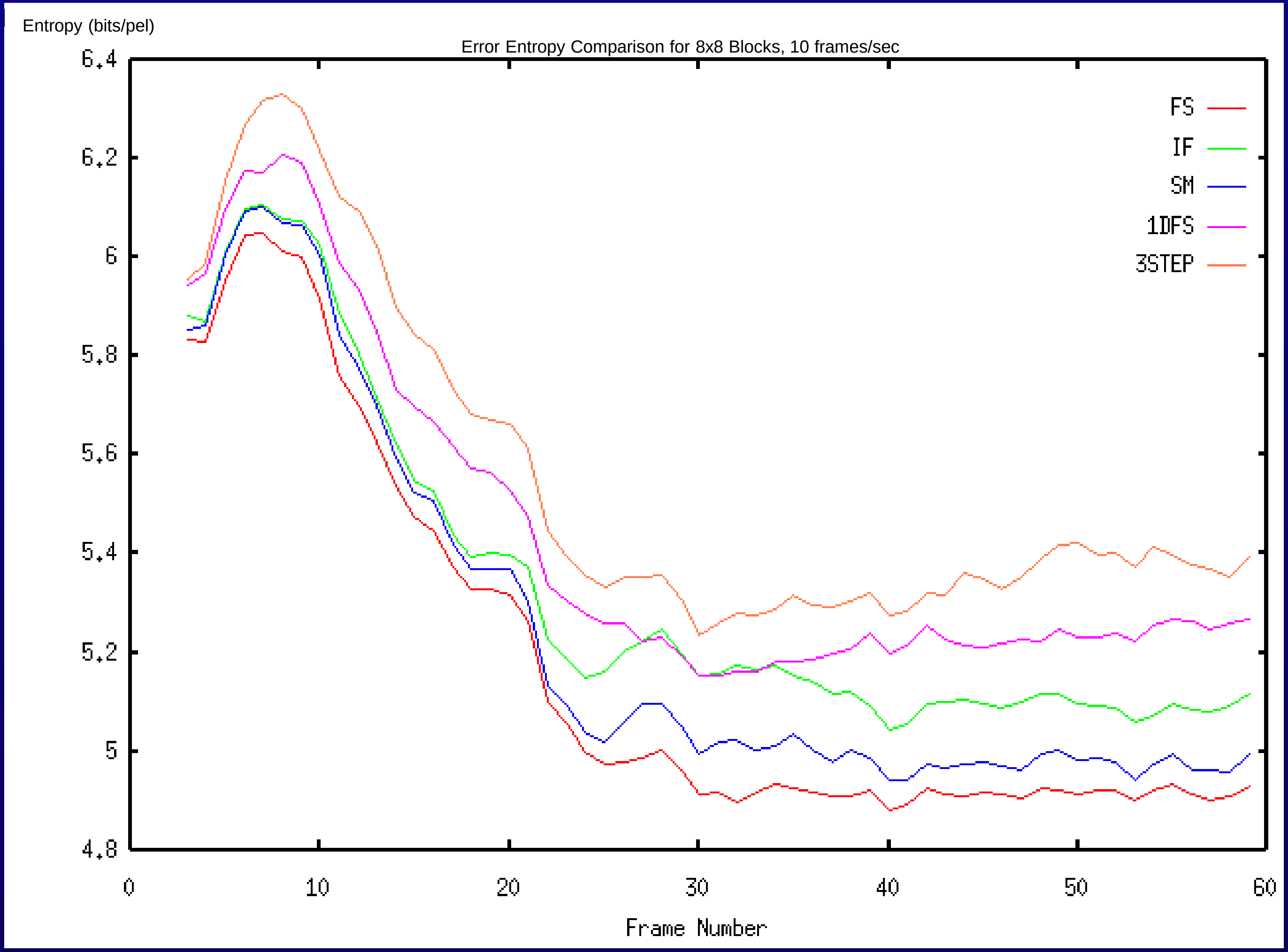
Algorithms	30 frames/sec	15 frames/sec	10 frames/sec
Full Search	27.33	26.90	26.49
Inter-Frame	25.73	25.23	24.66
Inter-Block	26.09	25.52	25.22
Median	25.57	24.93	24.23
Boundary Match	26.40	26.03	25.68
Side Match	26.52	26.17	25.77
IDFS	26.23	25.09	24.49
3-Step	24.94	23.06	23.26

PSNR Comparison



block size 8x8, 10 frames/sec, Table Tennis Sequence

Prediction Error Entropy Comparison

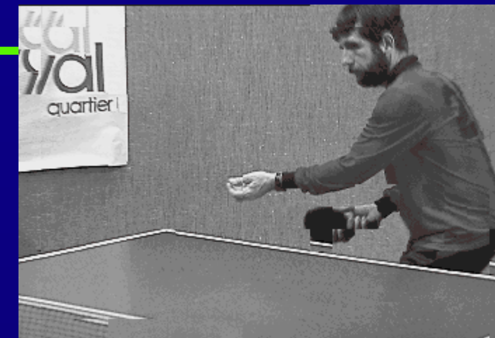


block size 8x8, 10 frames/sec, Table Tennis Sequence

Subjective Quality

16x16 blocks, 15 frames/sec,
Table Tennis Sequence

Original

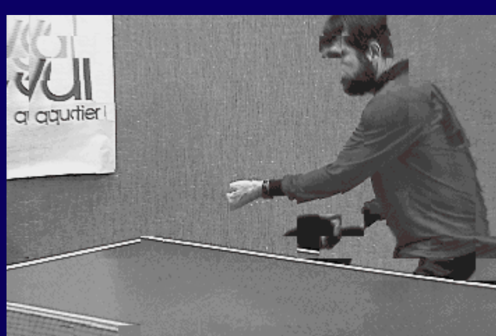


2DFS

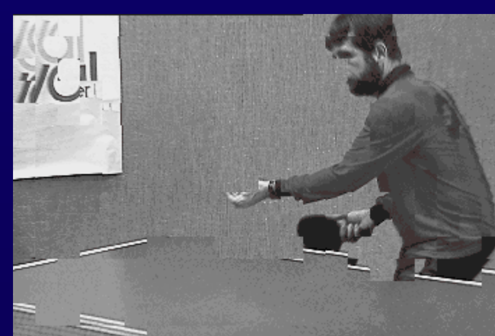
Inter-Frame

Inter-Block

Boundary Match



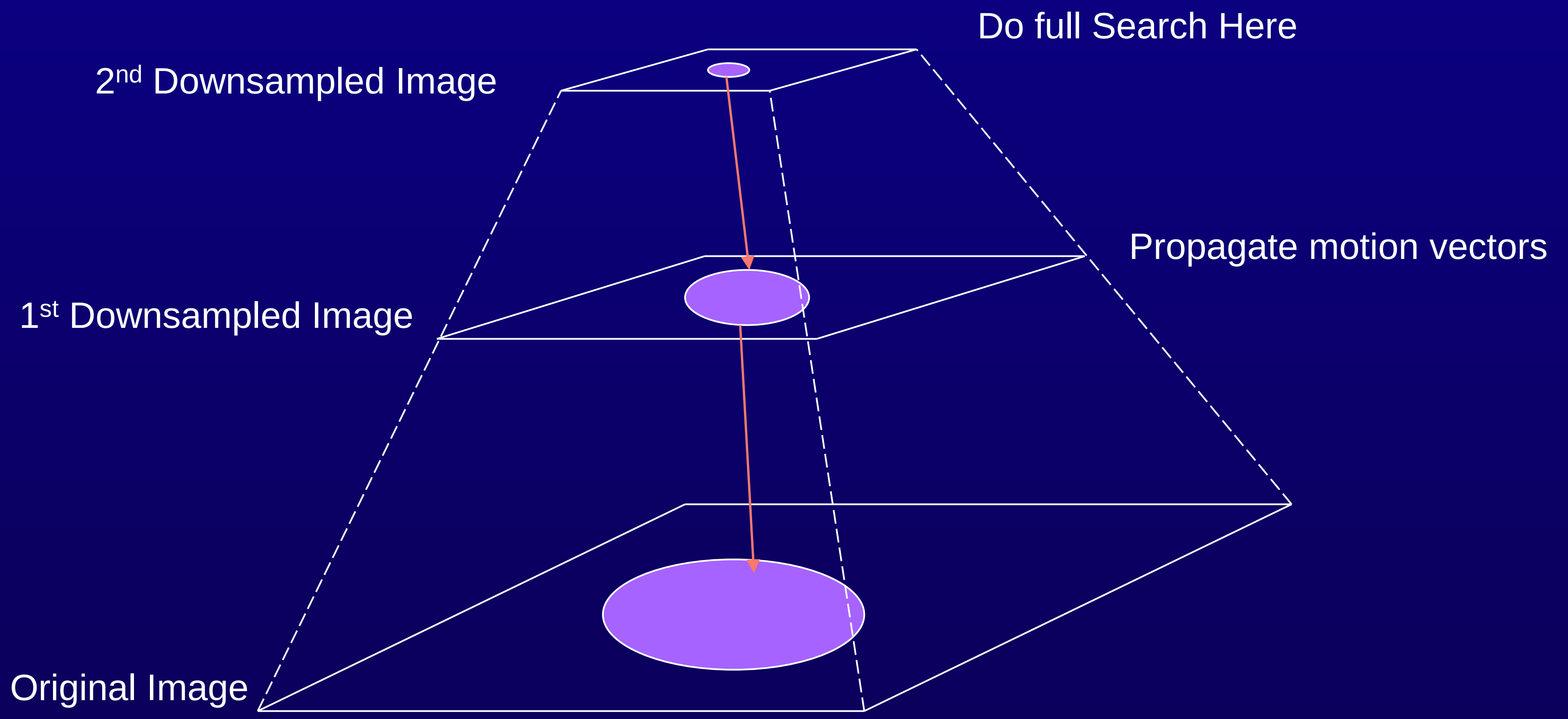
1DFS



3Step

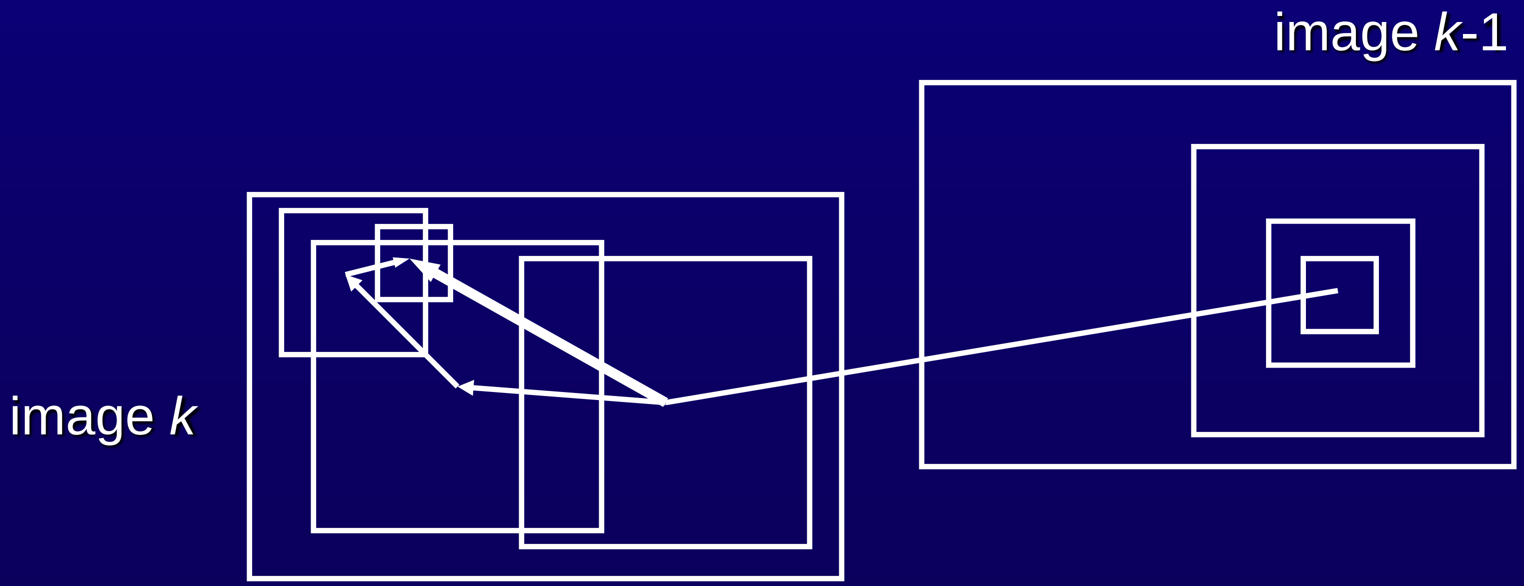
Side Match

Hierarchical Search Strategies



Hierarchical Block Matching

applicatios	desired properties
MCP (Motion Compensated Prediction)	vectors for minimum prediction error low transmission overhead for vectors
MCI (Motion Compensated Interpolation)	accurate estimation of true vectors high resolution of the vector field



- start with a subsampled low-pass filtered large block
- provides more accurate motion vector field with high resolution

Half-Pixel Motion Estimation

2 stage：先做到integer-grid，再到1/2 ¼ 1/8-pixel