IMAGES – COMPRESSION AND DITHERING

TOPICS TO BE COVERED

Introduction

- Image Types
- Image Representation
- Applications where images are used

Need for Compression and Standards

Overview of Image Compression Algorithms and Formats

JPEG

- Needs and Requirements
- Compression Algorithms for supported modes
- Performance Issues

JPEG 2000

- Wavelet Compression

Drawbacks of Image Compression - Image Dithering

TYPES OF IMAGES

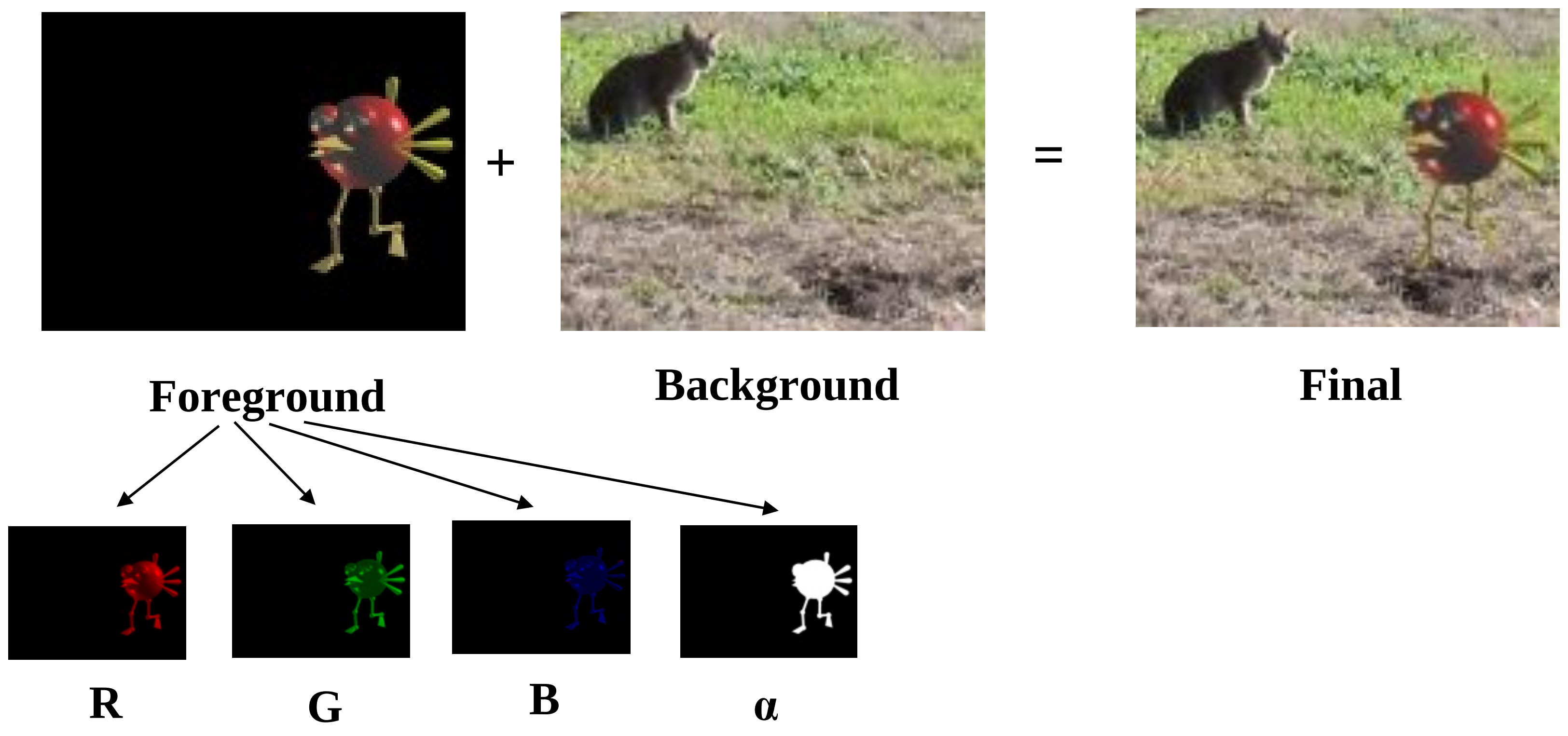
We consider here *still images*, for example:

- Photographs (color or grayscale)
- Fax (bi-level and multilevel)
- Documents (text, handwriting, graphics and photographs)
- Mosaics
- Stereo Images

Representation of Images –

Consists of components
Gray Images – single component
Color Images – r, g, b components
Sometimes images also have an alpha component.
Each component is represented as a number of bits per pixel (application dependent).

EXAMPLE OF ALPHA CHANNEL



Final[i][j] = Foreground[i][j]* α [i][j] + Background[i][j]*(1-α[i][j])

APPLICATIONS

Application examples - variety of low/medium resolutions (usually 8 bits per component or less)

- **Information – news,**
- **Entertainment –** *Slide-show o***n the Internet, games**
- **Commerce – ecommerce (catalogs, home shopping), real-estate viewing**

Application examples - high resolution (more than 8 bits per component)

- **Medical applications - (X-rays, CAT scans, etc.)**
- **Film Applications**
- **Remote Sensing.**

NEED FOR COMPRESSION

Multimedia Image Data	Size Of Image	Bits/Pixel or Bits/Sample	Uncompressed Size (B for bytes)	Band Width (b for bits)	Transmission Time	
					56K Modem	780Kb DSL
Grayscale Image	512 x 512	8 bpp	262 KB	2.1 Mb/image	42 seconds	3 seconds
Color Image	512 x 512	24 bpp	786 KB	6.29 Mb/image	110 seconds	7.9 seconds
Medical Image	2048 x 1680	12 bpp	5.16 MB	41.3 Mb/image	12 min	51.4 seconds
SHD Image	2048 x 2048	24 bpp	12.58 MB	100 Mb/image	29 min	2 min

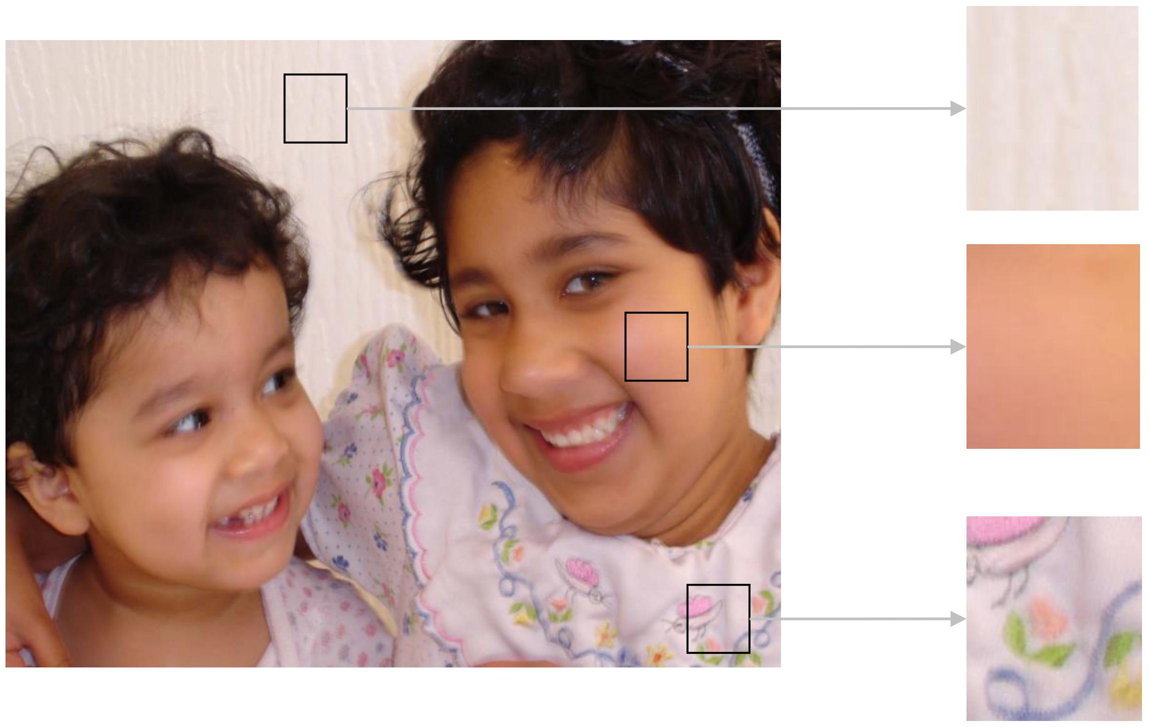
SOME COMPRESSION FORMATS

There are many different uncompressed and compressed formats used in the industry! Some of commonly used compressed formats are

- RIFF – Resource Interchange File Format
- GIF – Graphics Interchange File Format
- PNG – Portable Network Graphics
- JPEG – Joint Photographic Expert Groups
- JPEG 2000
- MPEG4-VTC – Visual Texture Coding

IMAGE REDUNDANCY

- Spatial Redundancy
- Spectral Redundancy



TYPES OF IMAGE COMPRESSION

Makes use of Lossless and Lossy Schemes or a combination of both (Hybrid)

Lossless Schemes

Already discussed in Information Theory Lecture

Lossy Schemes

Frequency Domain Based

- **Discrete Fourier**
- **Discrete Cosine**
- **Wavelet Transforms**

Fractal Based Compression

http://pi4.informatik.uni-mannheim.de/pi4.data/content/animations

JPEG BACKGROUND – REQUIREMENTS AND SELECTION PROCESS

Be at or near the state of art with regards to compression rate and accompanying fidelity

Make no assumptions about the type of image – should be applicable to practically any kind of continuous-tone digital source image.

Have a tractable computational complexity

Support flexibility by allowing the following modes of operation

- **Lossless and Lossy Encoding**
- **Sequential Encoding**
- **Progressive Encoding**
- **Hierarchical Encoding**

JPEG IMAGE CODING

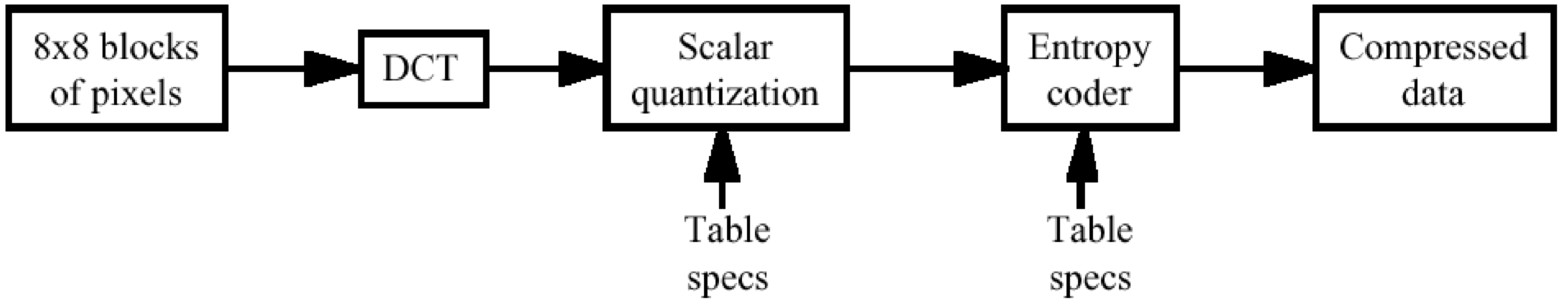
JPEG is the standard algorithm for compression of still images

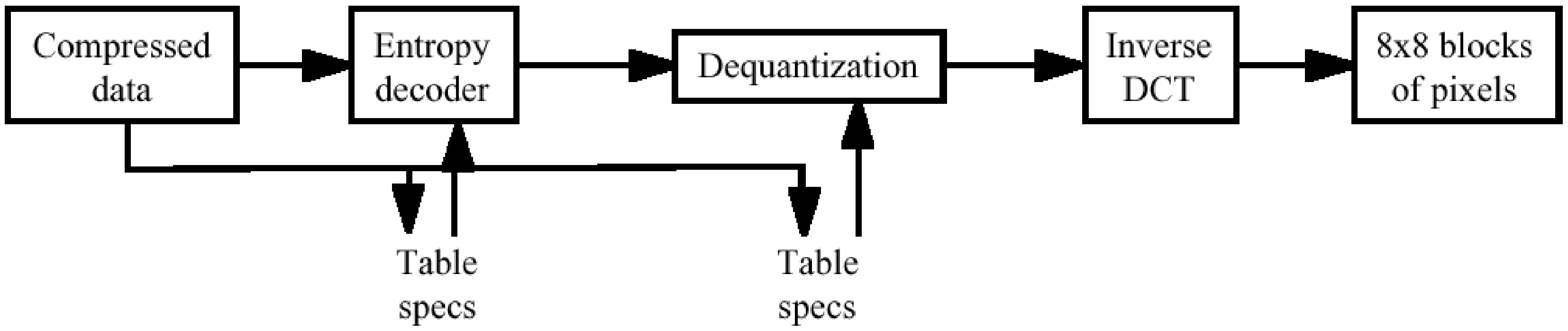
- **Started in June 1987**
- **Finalized and Accepted in 1991**

It is of reasonably low computational complexity, is capable of producing compressed images of high quality, and can provide both *lossless* **and** *lossy* **compression of arbitrarily sized grayscale and color images**

JPEG LOSSY CODEC SCHEME

ENCODER





DECODER

JPEG COMPRESSION ALGORITHMIC OVERVIEW

The RGB color vectors of the input image are converted into YCrCb values and each channel is encoded independently

Each channel image is converted into a series of 8-by-8 pixel blocks, which are then processed in a raster scan sequence from left to right and from top to bottom

Each 8x8 block of pixels is spectrally analyzed using DCT (transform coding) and coefficients are quantized

After DCT coding and quantization, coefficients are entropy coded

DCT FORMULA IN 2D

Discrete Cosine Transform

$$F(u,v)=\left(\frac{1}{4}C(u)C(v)\right)\left[\sum_{x=0}^{x=7}\sum_{y=0}^{y=7} f(x,y)\times\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right]$$

Inverse Discrete Cosine Transform

$$f(x,y)=\frac{1}{4}\left[\sum_{u=0}^{x=7}\sum_{v=0}^{y=7} C(u)C(v)\times F(u,v)\times\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right]$$

$where: C(u), C(v) = {}^{1}\!/\!{\sqrt{2}}$, *for u, v =0*

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

WHAT DO THE DCT COEFFICIENTS REPRESENT

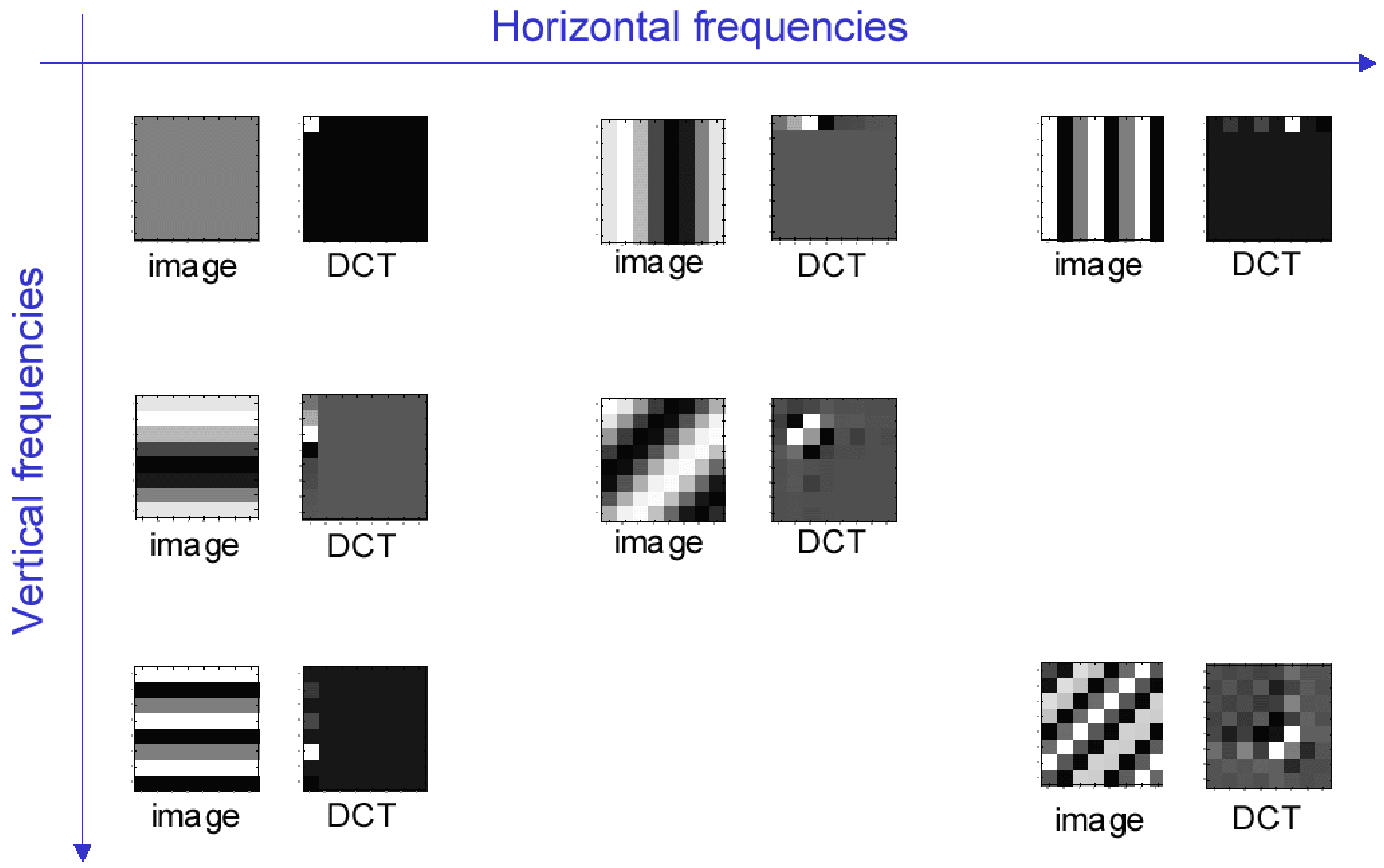
The DCT coefficients represent the *spatial frequency content* within a 8x8 image block

The (0, 0) coefficient contains is called *DC coefficient*, and is equal to a measure of the *average* of the 64 image values in the block

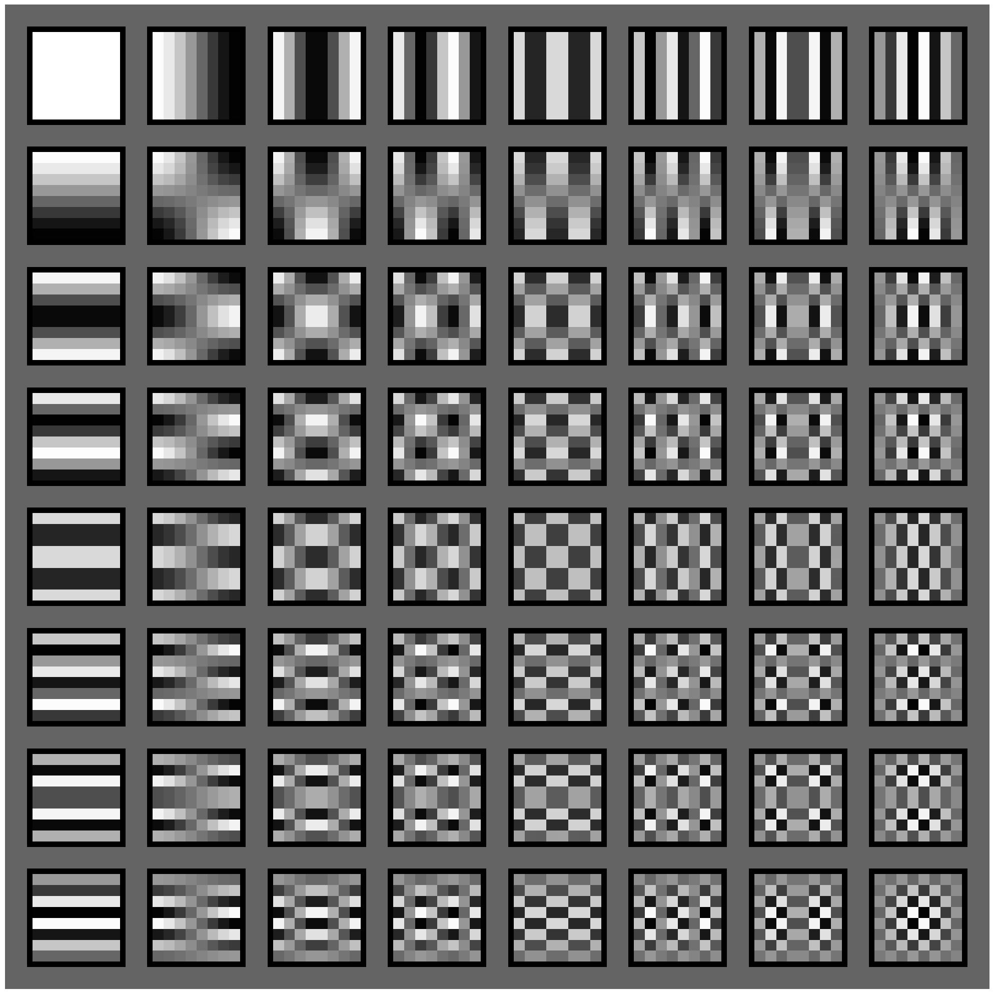
As we move to the right of the block, the coefficients represent the energy in higher *horizontal frequencies*

As we move down the block, the coefficients represent the energy in higher *vertical frequencies*

EXAMPLES OF 8X8 DCTS



EXAMPLE OF 8X8 DCT BASIS FUNCTIONS



DCT COEFFICIENT QUANTIZATION

Each DCT coefficient is quantized independently using a uniform quantizer

The number of quantization intervals per each DCT channel is chosen independently for each coefficient

After quantization, the DC coefficient of a block is encoded as the difference from the DC term of the previous block in the processing order.

This is because there is usually strong correlation between the DC coefficients of adjacent blocks

RULES FOR DCT COEFFICIENT QUANTIZATION

Low-frequency coefficients usually have more energy than high-frequency coefficients - therefore require more quantization intervals

The Human Visual System is more sensitive to low frequencies. Hence, low-frequency coefficients require more quantization intervals

The Human Visual System is more sensitive to luminance channels than to chrominance channels. Hence, luminance channels require more quantization intervals than chrominance channels

"Note: the quantization table is not standardized!"

EXAMPLE

139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

Source image sample

235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3
-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2
-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1
-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3
-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3
1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0
-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8
-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4

DCT coefficients

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

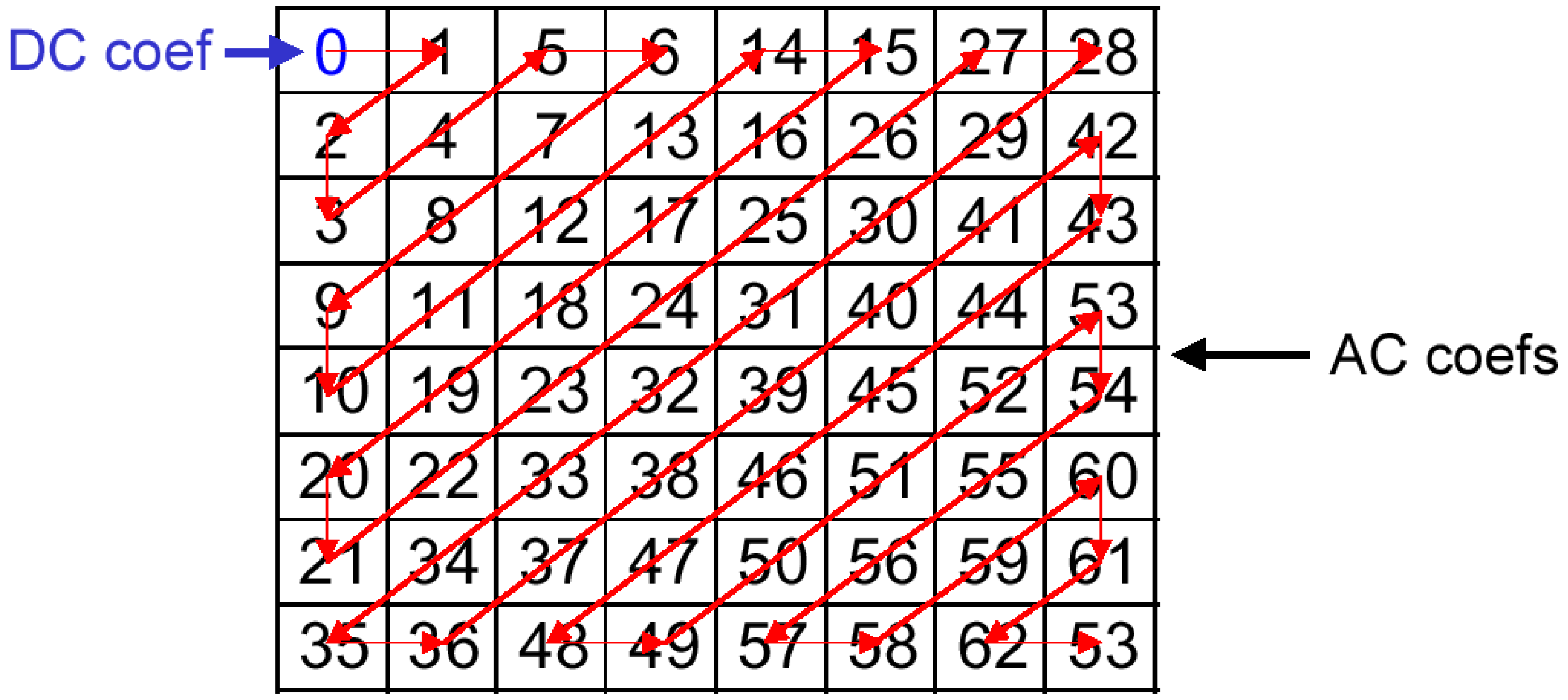
Quantization table

15	0	-1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantized coefficients

ENTROPY CODING OF DCT COEFS

The quantized DCT coefficients are first processed in *zigzag order*



0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	53

INTERMEDIATE REPRESENTATION

Zigzag ordering of coefficients places low frequency coefficients before high-frequency coefficients.

- **Low frequency coefficients are more likely to be non-zero**
- **High frequency coefficients are more likely to be null after quantization**

In this way, the sequence of quantized coefficients is such that there often are long sequences of null values

AC coefficients are represented in an *intermediate run-length representation, which* **encodes the runs of zero preceding any non-null coefficient**

INTERMEDIATE REPRESENTATION (2)

Each non-zero AC coefficient is represented in combination with the run-length of the null coefficients before it, using a pair of symbols:

- **Symbol-1 (RUNLENGTH,SIZE)**
- **Symbol-2 *(AMPLITUD*E)**

***RUNLENGTH* is the number of consecutive null coefficients before the non-zero symbol; *SIZE* is the size of the symbol used to encode the non-null symbol**

***AMPLITUDE* is the actual (encoded) value**

INTERMEDIATE REPRESENTATION (3)

RUNLENGTH **must be ≤ 15. To represent a run of > 15 zeros, there can be up to 3 consecutive (15,0) symbol-1 extensions**

If the last run of zeros contains the last DCT coefficient, then the special EOB (End-Of-Block) symbol-1 value (0,0) terminates the representation

DC coefficients are represented by only one pair:

- **Symbol-1 (*SIZ*E)**
- **Symbol-2 (*AMPLITUD*E)**

ENTROPY CODING

For both DC and AC coefficients, each symbol-1 is Huffman encoded (variable length prefix code)

- Huffman tables are not specified in the standard and must be input to the encoder

Each symbol-2 is encoded with a Variable Length Integer (VLI) code

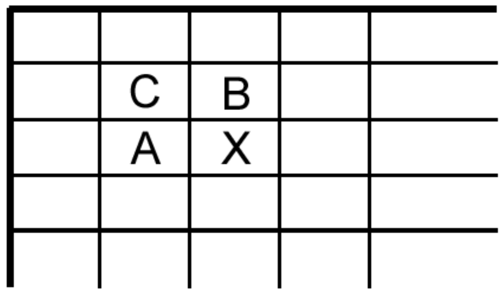
- It is different from a Huffman code because the symbol length is known in advance
- The coding table is hard-wired in the proposal

LOSSLESS JPEG MODE

It is wholly independent of DCT processing

Each sample value is *predicted* **from a combination of nearby sample values, and the prediction residual is entropy coded (Huffman)**

Prediction effectively reduces the entropy of the residual, so that small average symbol length can be achieved



X can be predicted by a combination of A,B,C

PROGRESSIVE ENCODING MODES

Layering or progressive encoding: an image can be transmitted at a low rate (and a low quality) and then progressively improved by subsequent transmission

Convenient for browsing applications, where a low-quality or low-resolution image is adequate browsing

Three forms of JPEG progressive encoding:

- **Spectral selection**
- **Successive bit approximation (SNR scalability)**
- **Hierarchical (pyramidal) mode**

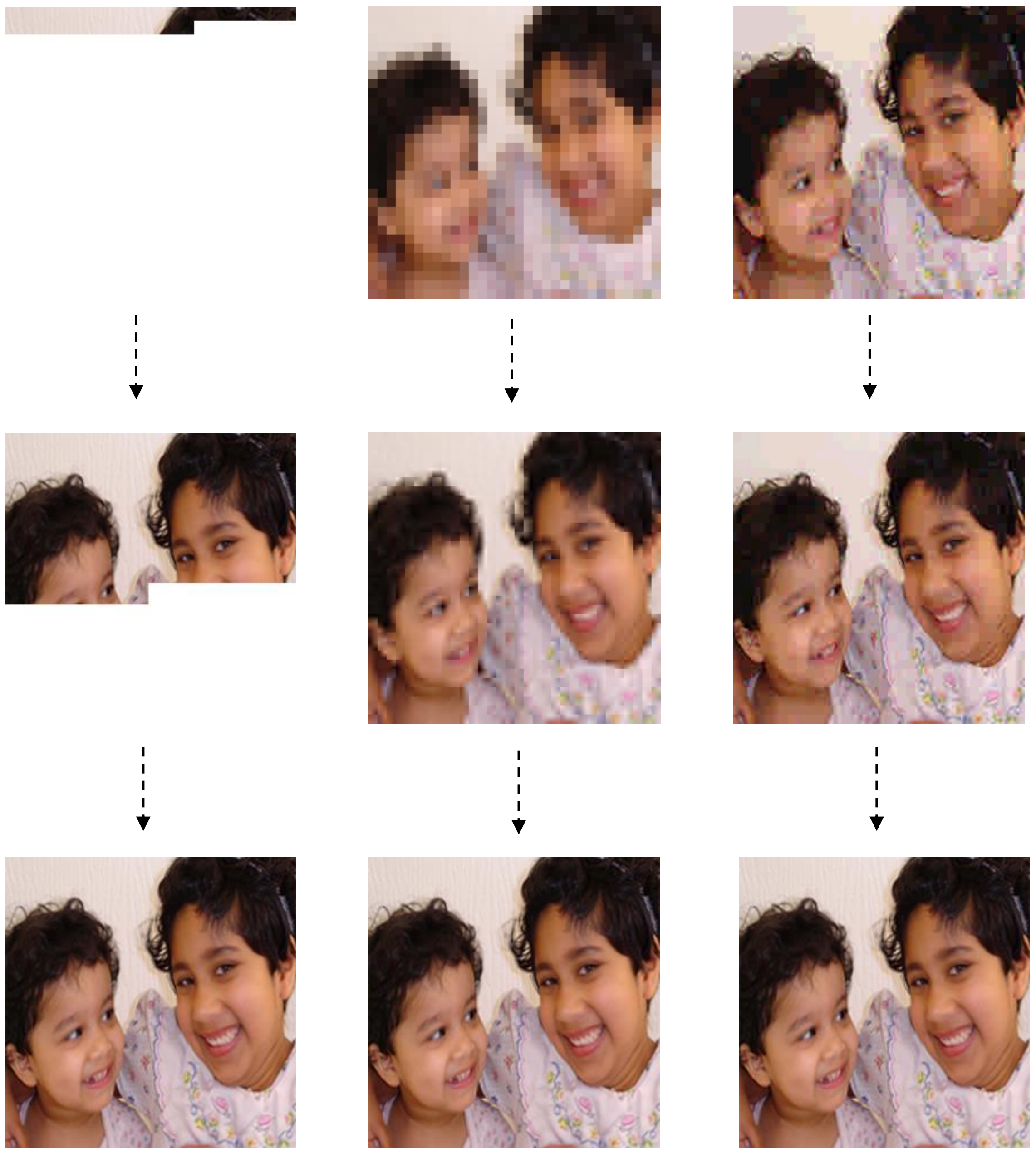
PROGRESSIVE ENCODING (2)

Spectral selection

- Initial transmission sends low-frequency DCT coefficients, followed progressively by the higher frequency coefficients, according to the zig-zag scan
- Simple to implement but reconstructed images from the early scans are blurred

Successive approximation (SNR scalability)

- Only the most significant bits of each coefficient are sent in the first scan, followed by the next most significant bits and so on until all bits have been transmitted

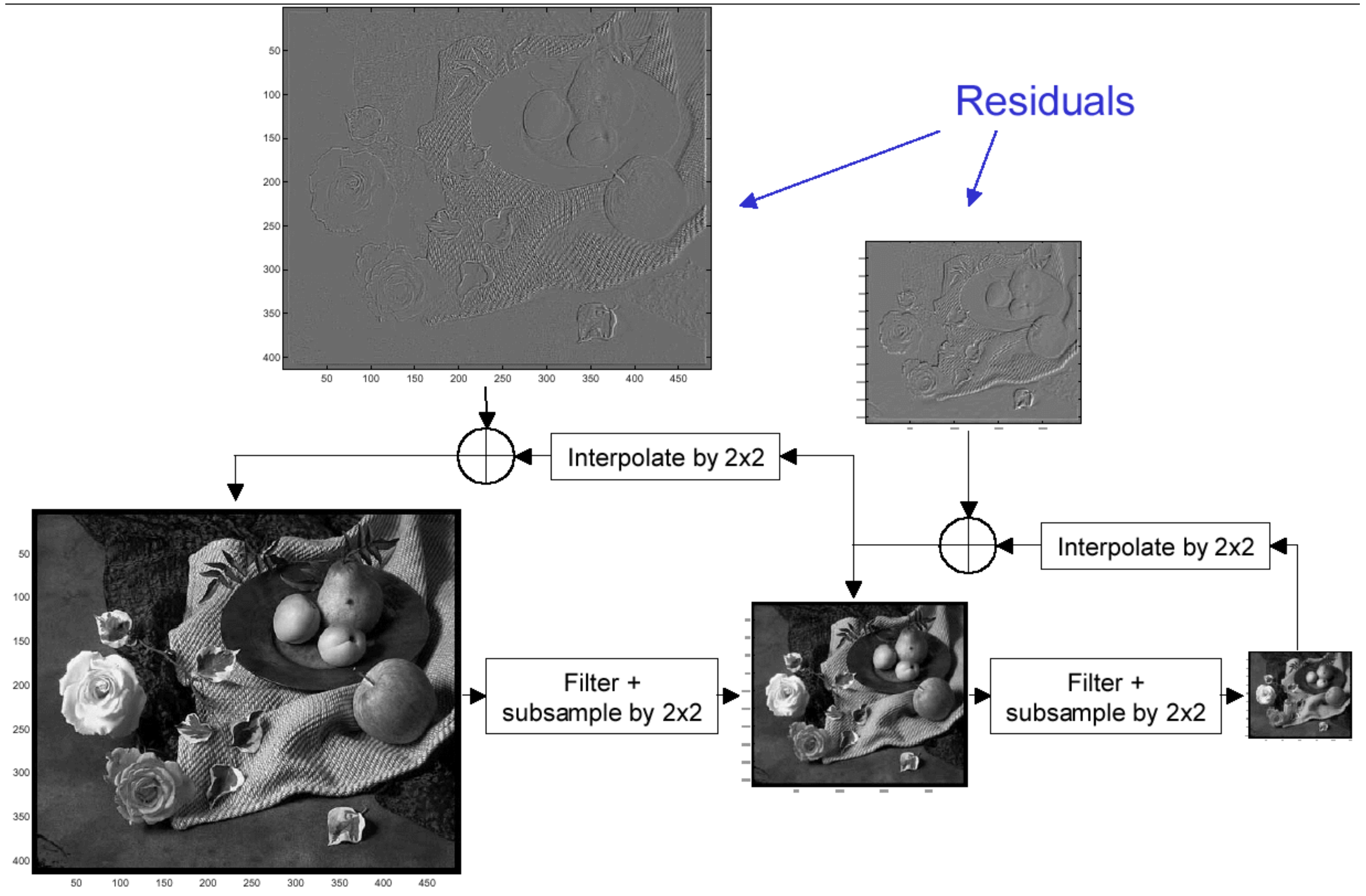


PROGRESSIVE ENCODING (3)

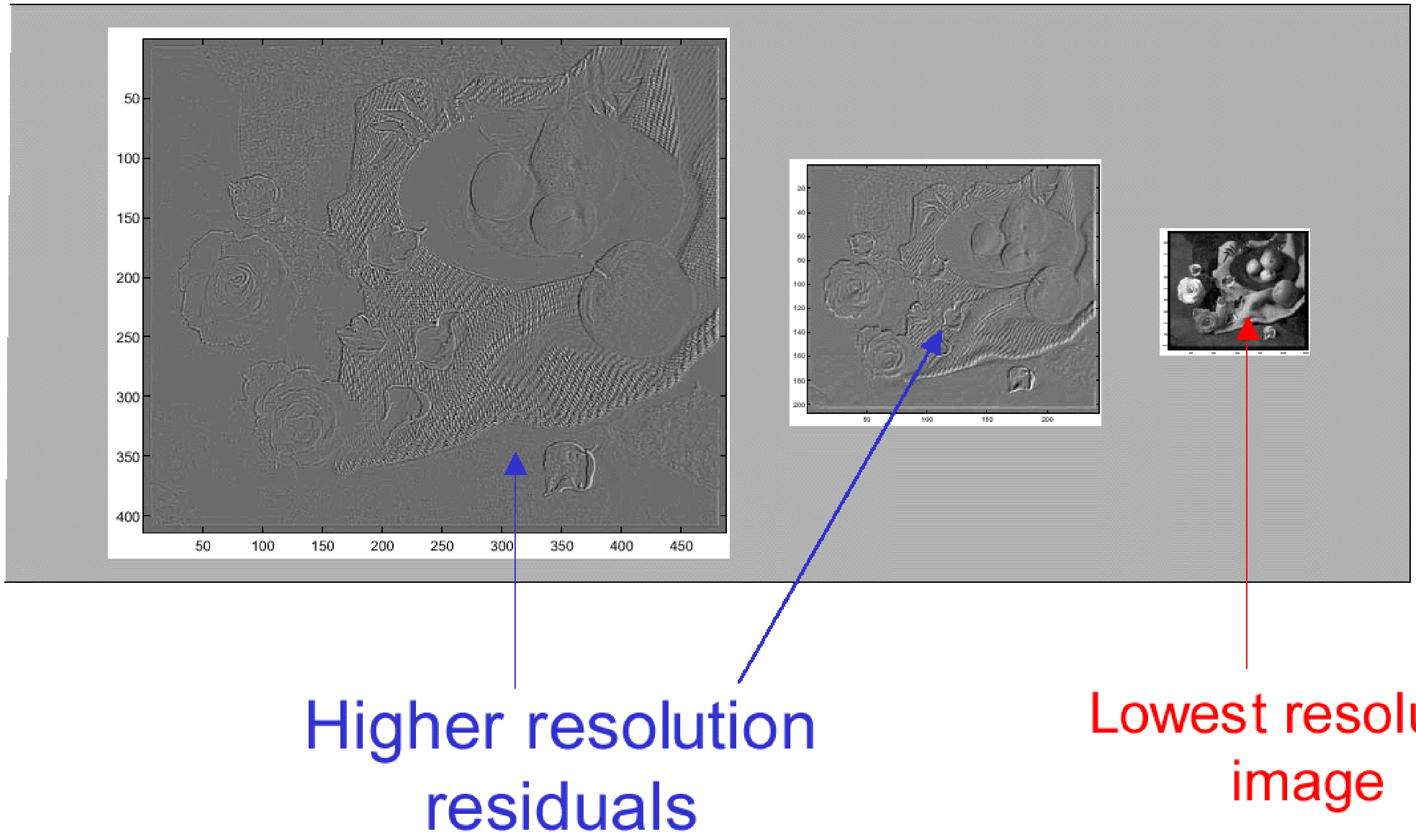
Hierarchical (pyramidal) mode

- Image can be sent in one of several *resolution modes* to accommodate different kinds of displays
- Different resolution modes achieved by *filtering and down sampling* the image in multiples of two in each direction. The resulting image is interpolated and subtracted from the next level to create a *residual*
- The different resolution modes are transmitted together with the residuals

PYRAMID DECOMPOSITION



PYRAMID DECOMPOSITION(2)



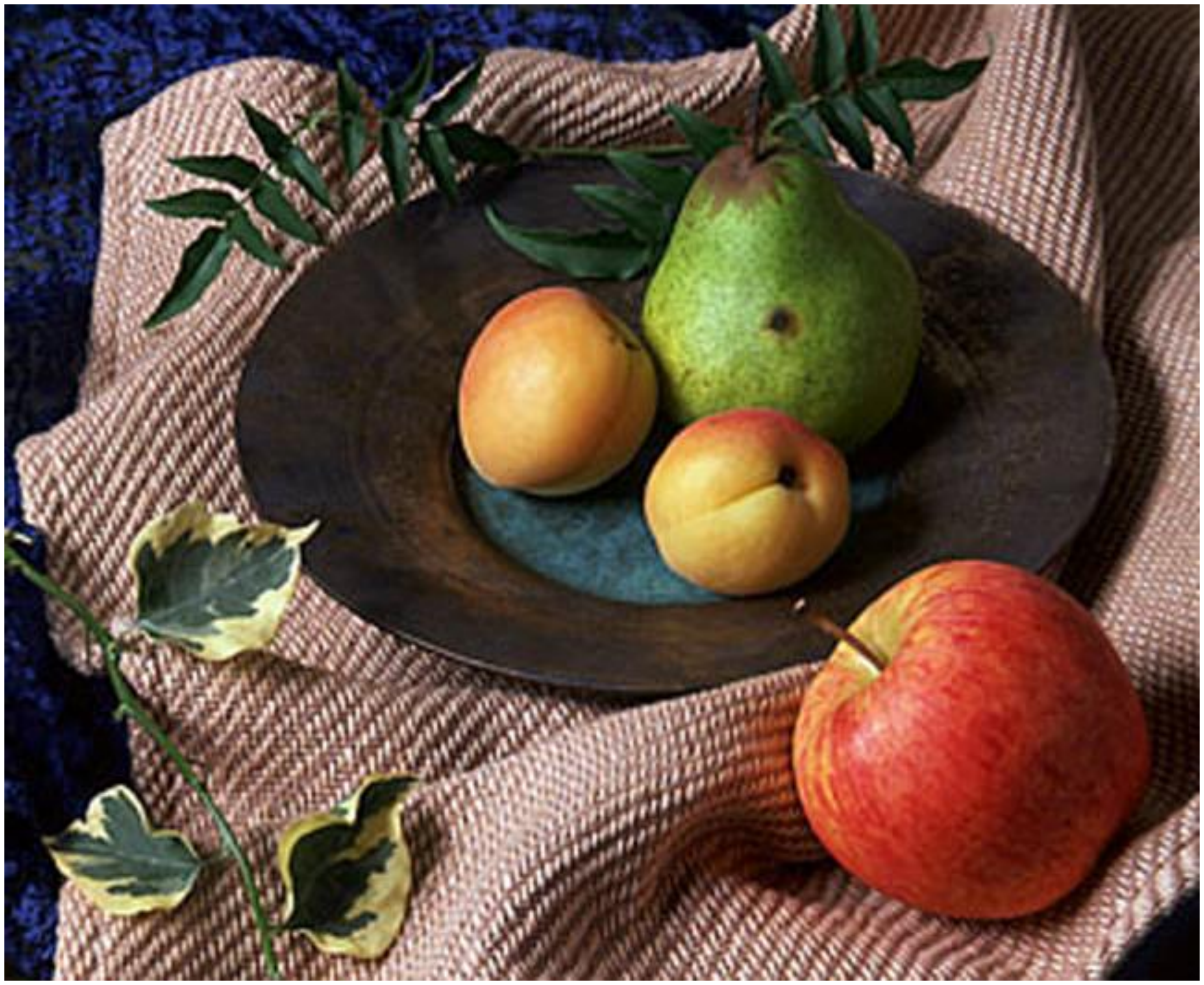
JPEG PERFORMANCES

Assume the pixels of an arbitrary color image are digitized to 8 bits for luminance and chrominance channels with 4:2:2 color sampling scheme.

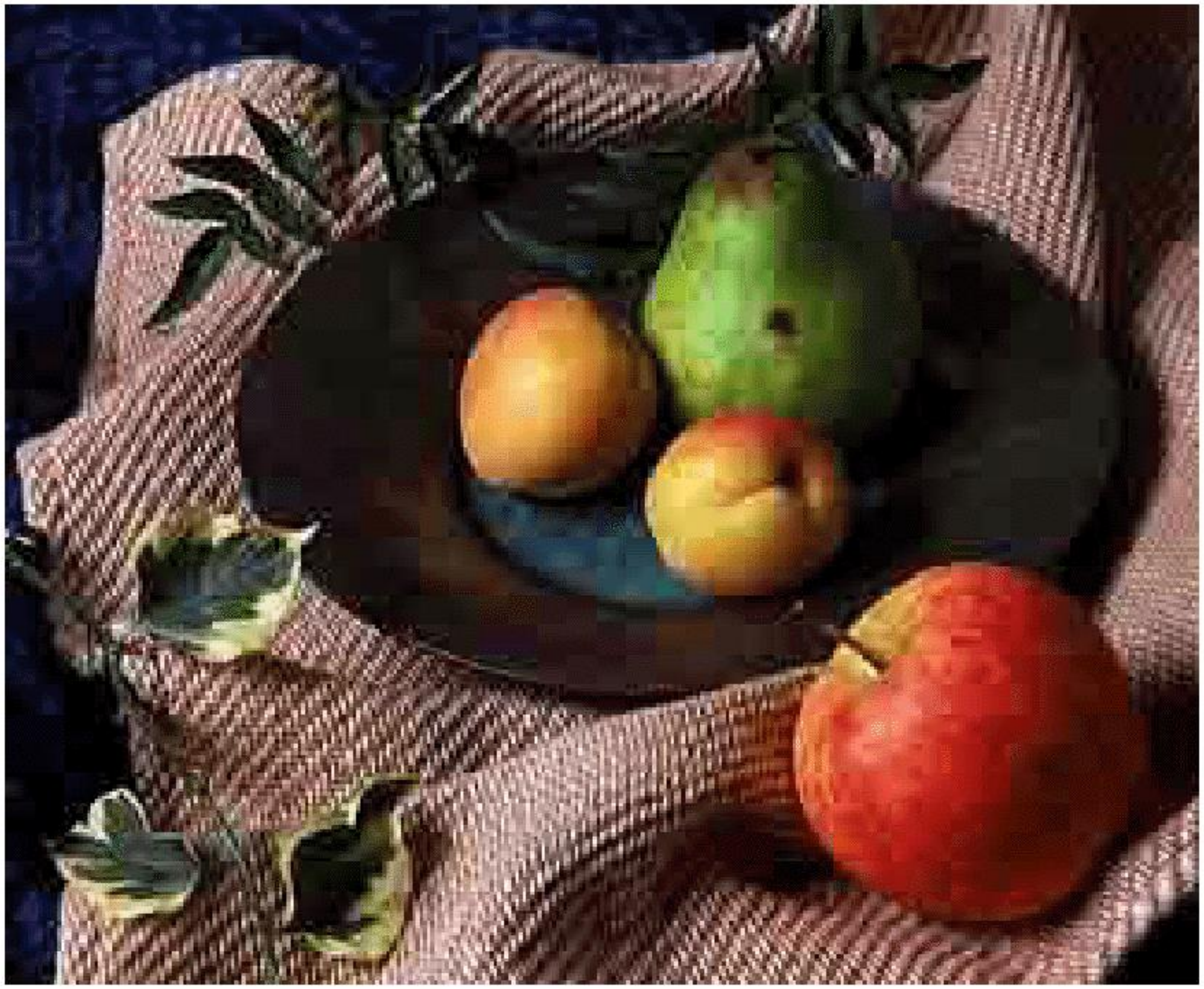
Then effectively the source image has 16 bits/pixel

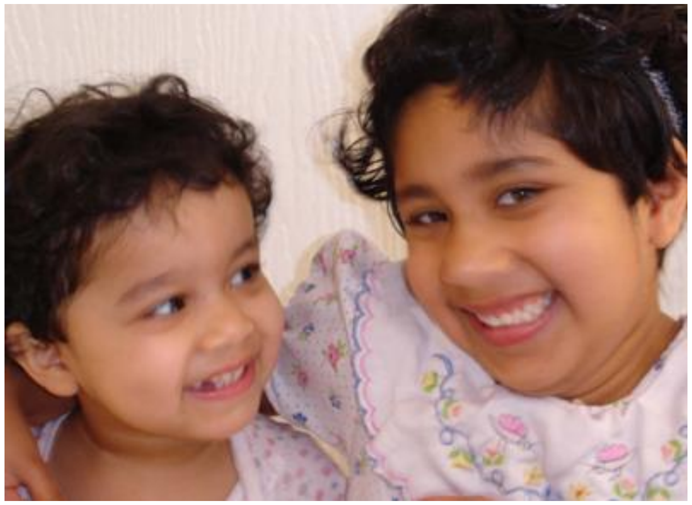
Bits/Pixel	***Quality***	***Compression Ratio***
0.25	*Fair*	*64:1*
0.5	*Good*	*32:1*
0.75	*Very Good*	*21:1*
1.5	*Excellent*	*10:1*
≥2	*Indistinguishable*	*8:1*

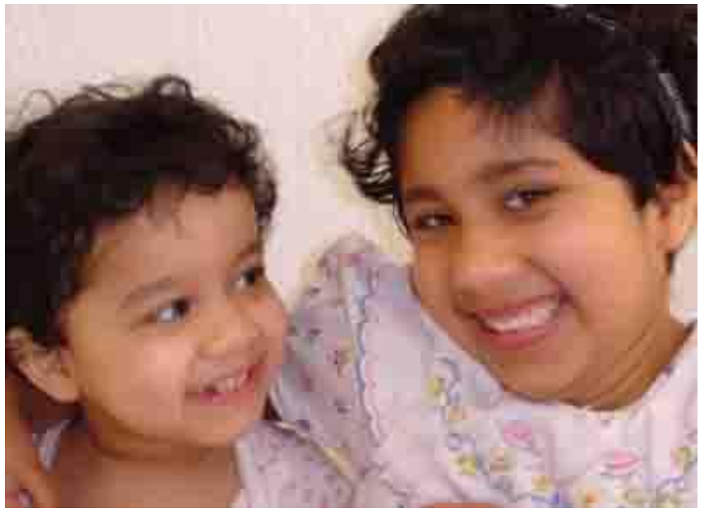
ARTIFACTS OF JPEG – ORIGINAL (380KB)

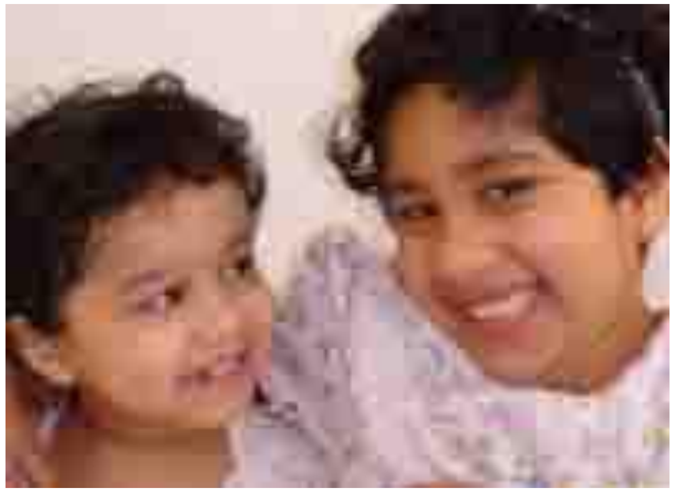


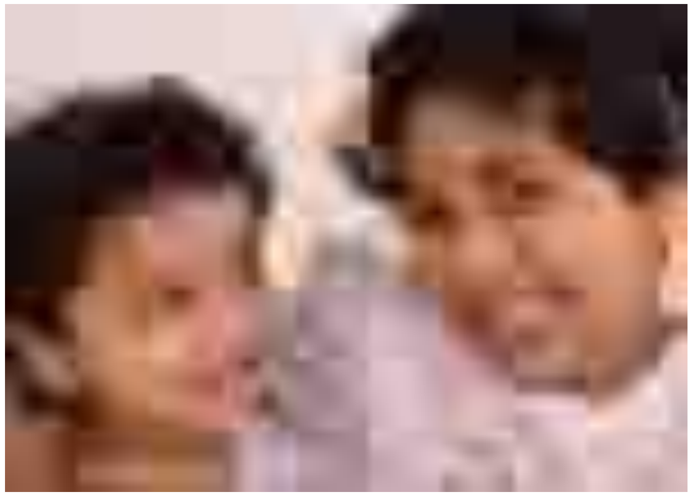
ARTIFACTS OF JPEG – COMPRESSED (40KB)




Original - 24 bits per pixel

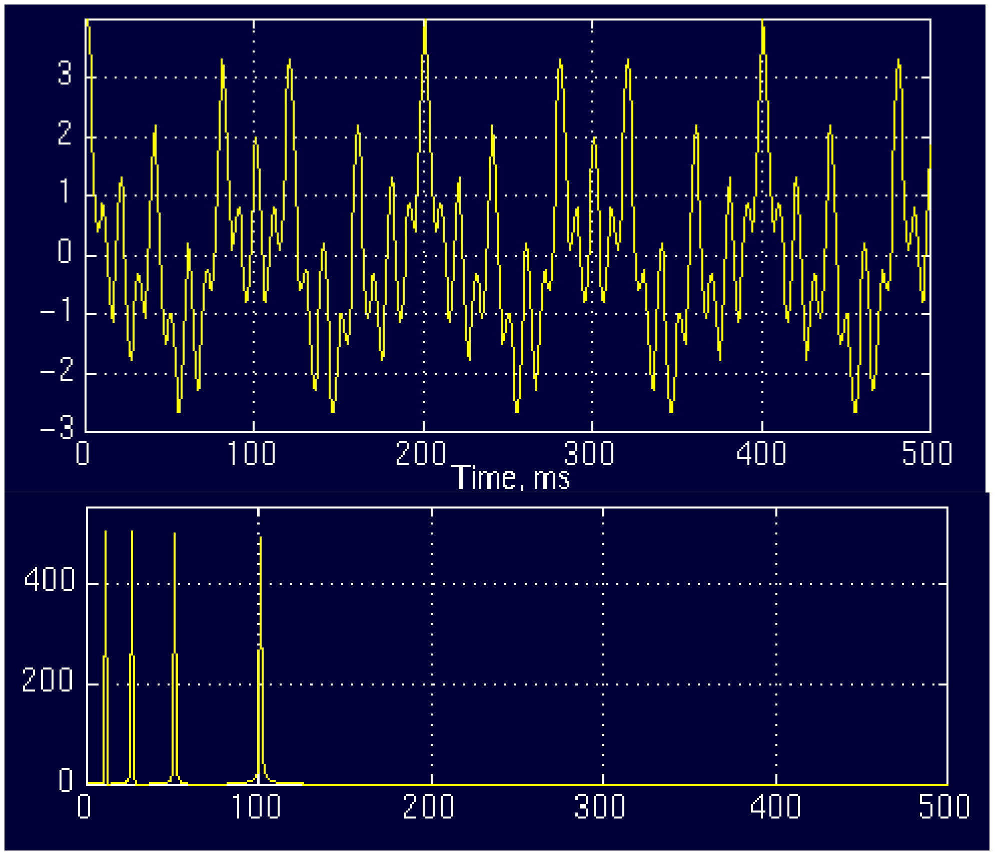

Compressed – 1.0 bits per pixel


Compressed – 0.5 bit per pixel


Compressed - 0.15 bits per pixel

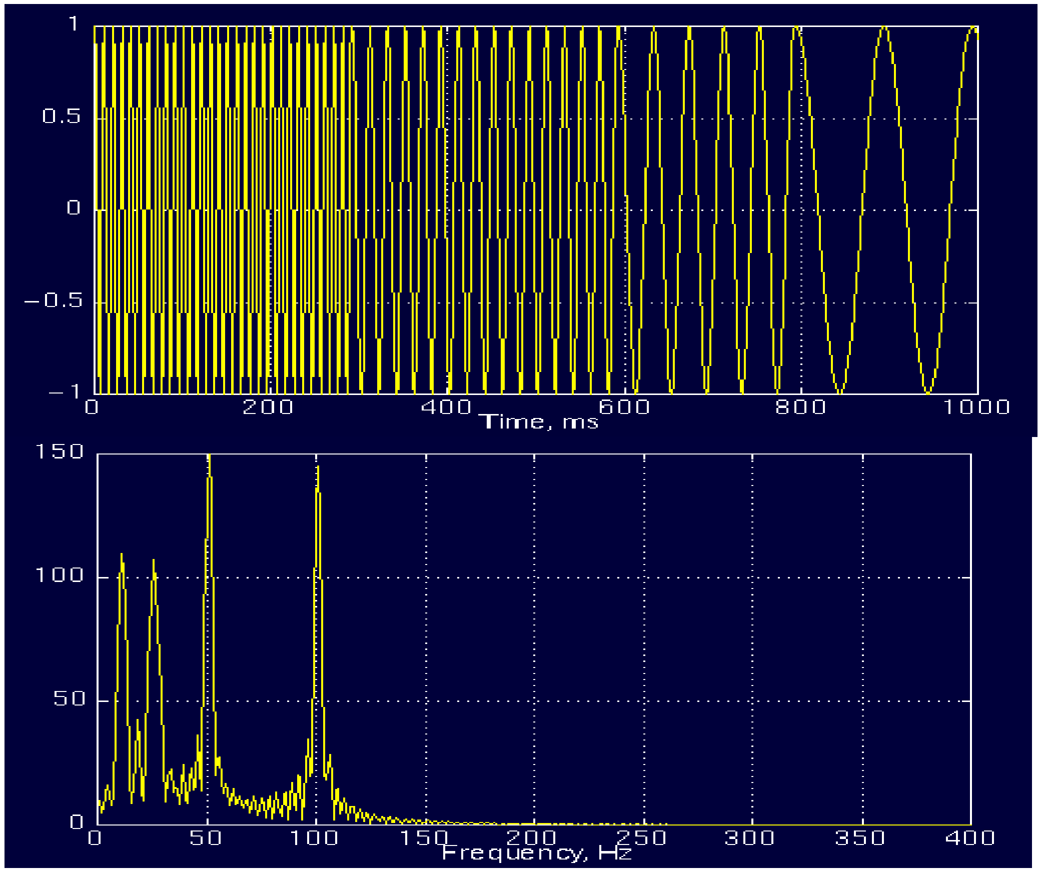
DRAWBACKS OF DCT – STATIONARY SIGNALS

The DCT Transform is good for stationary signals – frequencies are present all the time

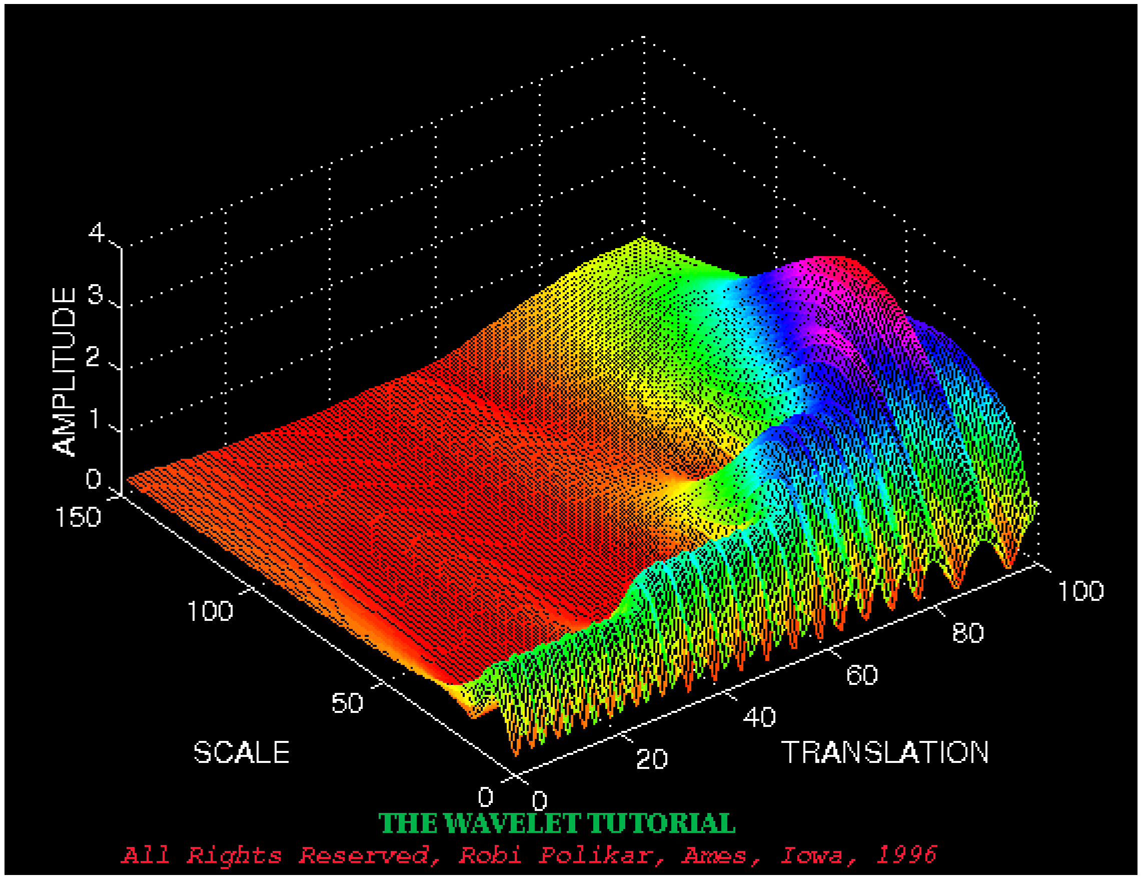


DRAWBACKS OF DCT – NON STATIONARY

Non stationary signals have frequencies that vary with time.



TIME FREQUENCY ANALYSIS



JPEG-2000

INTRODUCTION

Image compression should not only reduce *storage* and *bandwidth* requirements, but also allow extraction for *editing* and *processin*g – one of the primary motives of JPEG-2000: International Standard by 12/2000

JPEG-2000 has better compression performances than JPEG; more importantly, from a single bit-stream, JPEG-2000 allows extraction of

- **Different resolution**
- **Different pixel fidelities**
- **Different regions of interest**
- **Different color component**

JPEG-2000 FEATURES

State-of-the-art low bit-rate compression

Progressive transmission by quality, resolution, component, or spatial locality

Lossy and Lossless compression (with Lossless decompression available naturally through all types of progression)

Random (spatial) access to the bit stream

Pan and zoom (with decompression of only a subset of the compressed data)

Compressed domain processing (e.g., rotation and cropping)

Region of interest coding by progression

JPEG-2000 COMPRESSION STEPS

An image is first divided into *tiles*

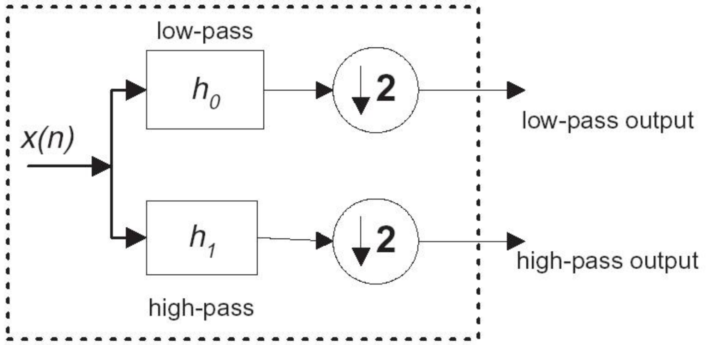
Each tile is subband-transformed and quantized

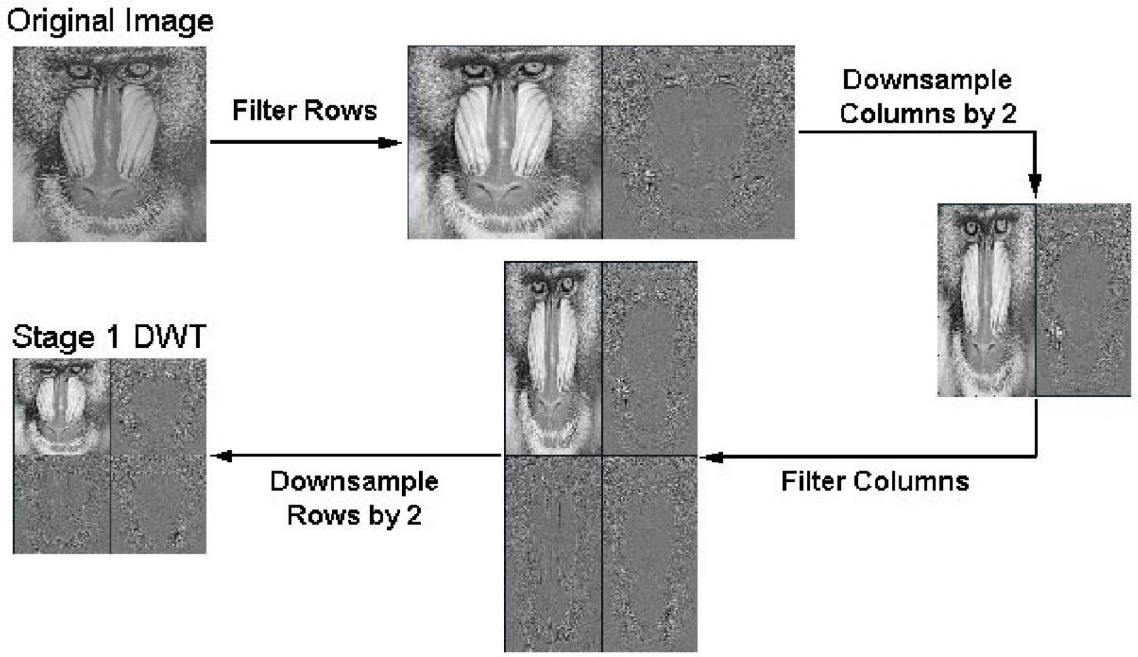
Each subband of a tile is divided into *packet partitions*; each packet partition is divided into *code-blocks*

Each code-block is entropy-coded independently (using arithmetic coding)

The image is encoded "naturally" progressively: if we decode all of the bit-stream, we obtain the Lossless version of the image

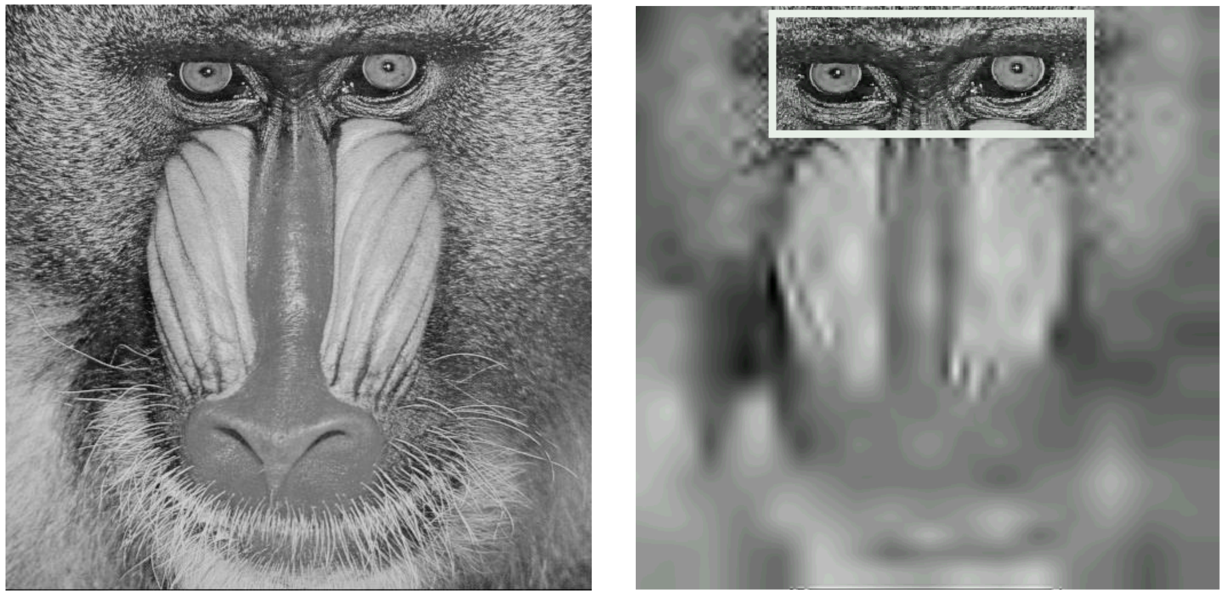
WAVELET ENCODING OF A SINGLE TILE





SPATIAL ACCESSIBILITY

A client may wish to obtain compressed data for only a particular portion of the image.



If Regions of Interest (ROI) are known in advance (at encoding time) JPEG-2000 can provide greater image quality to the foreground. (exploits image tiling)

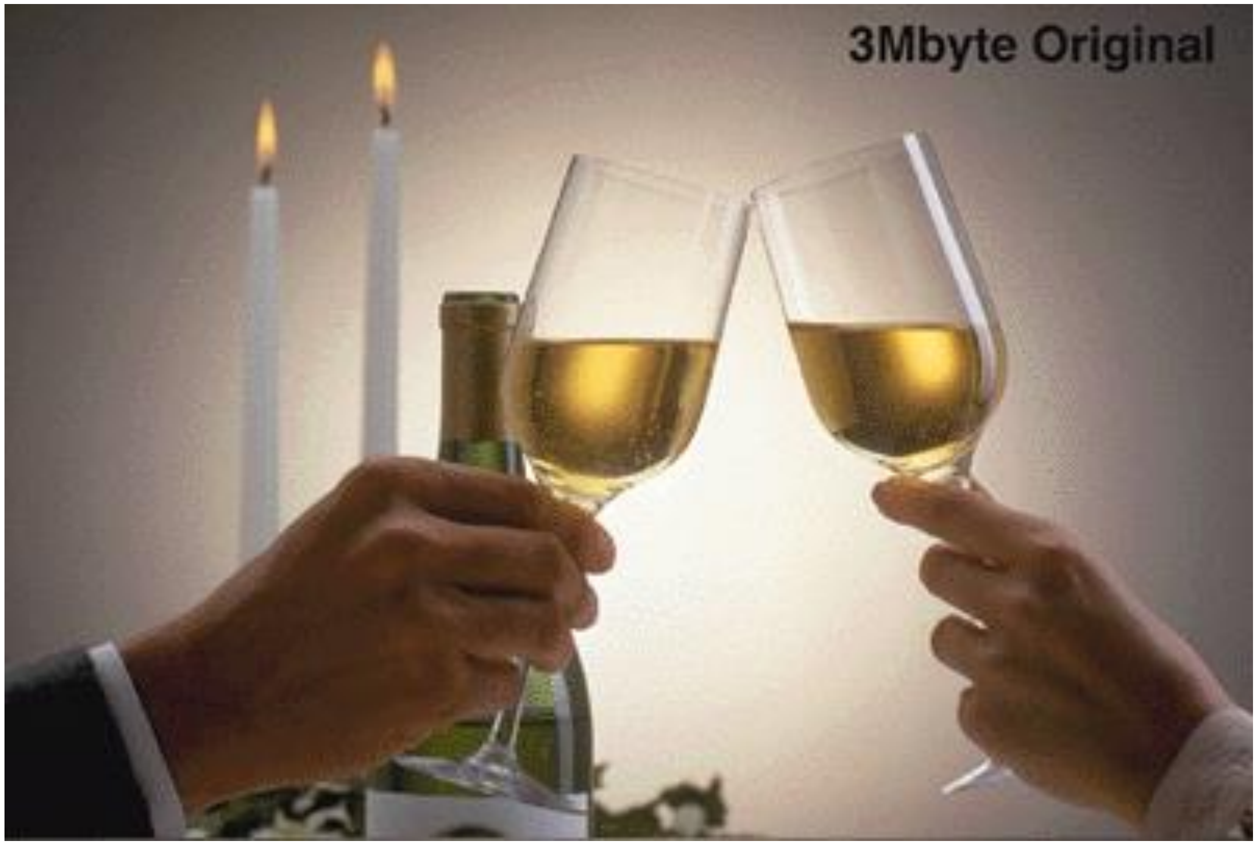
PERFORMANCE

Typically JPEG-2000 provides (with respect to JPEG) only a few dB improvements from 0.5 to 1.0 bits/pixel but substantial improvement below 0.25 bits/pixel and above 1.5 bits/pixel

JPEG progressive is not optimized (i.e. has worse performance than JPEG sequential) because the DCT coefficients stay the same but the entropy coding changes

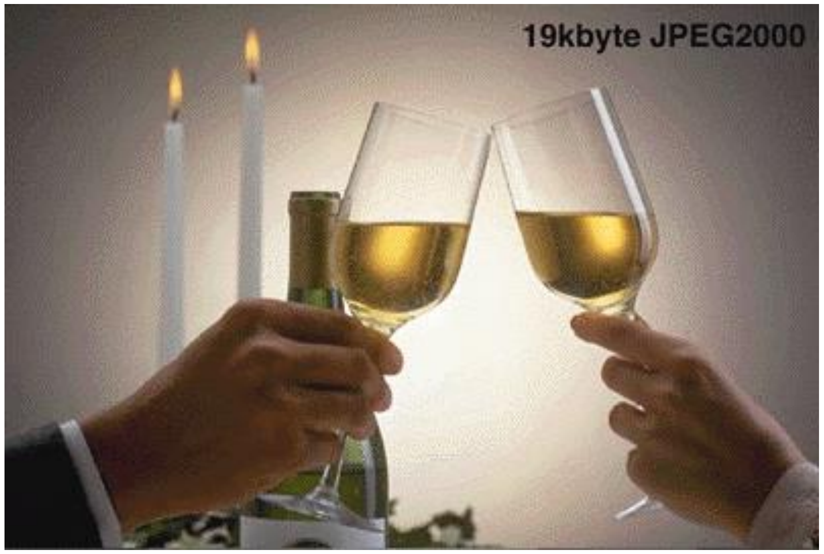
• With JPEG-2000 the progressive performance is almost identical to the single layer performance (because the encoded bits do not change)

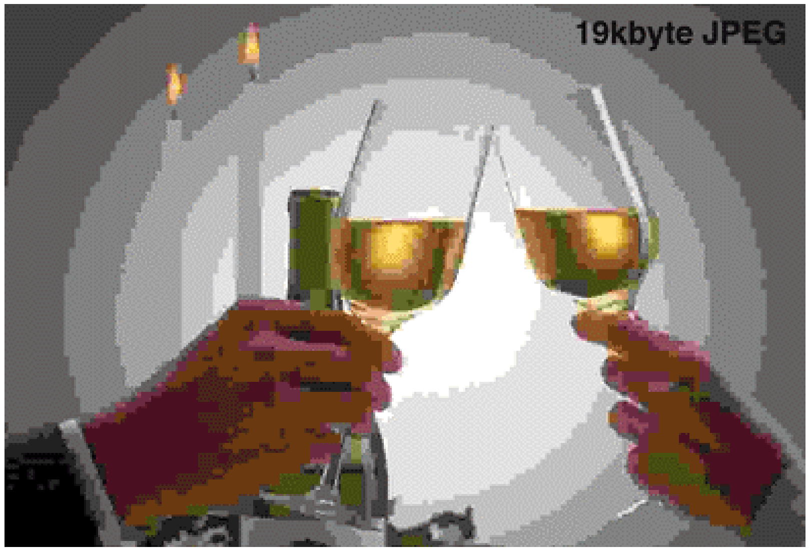
EXAMPLES – ORIGINAL



EXAMPLES

JPEG2000 **JPEG**





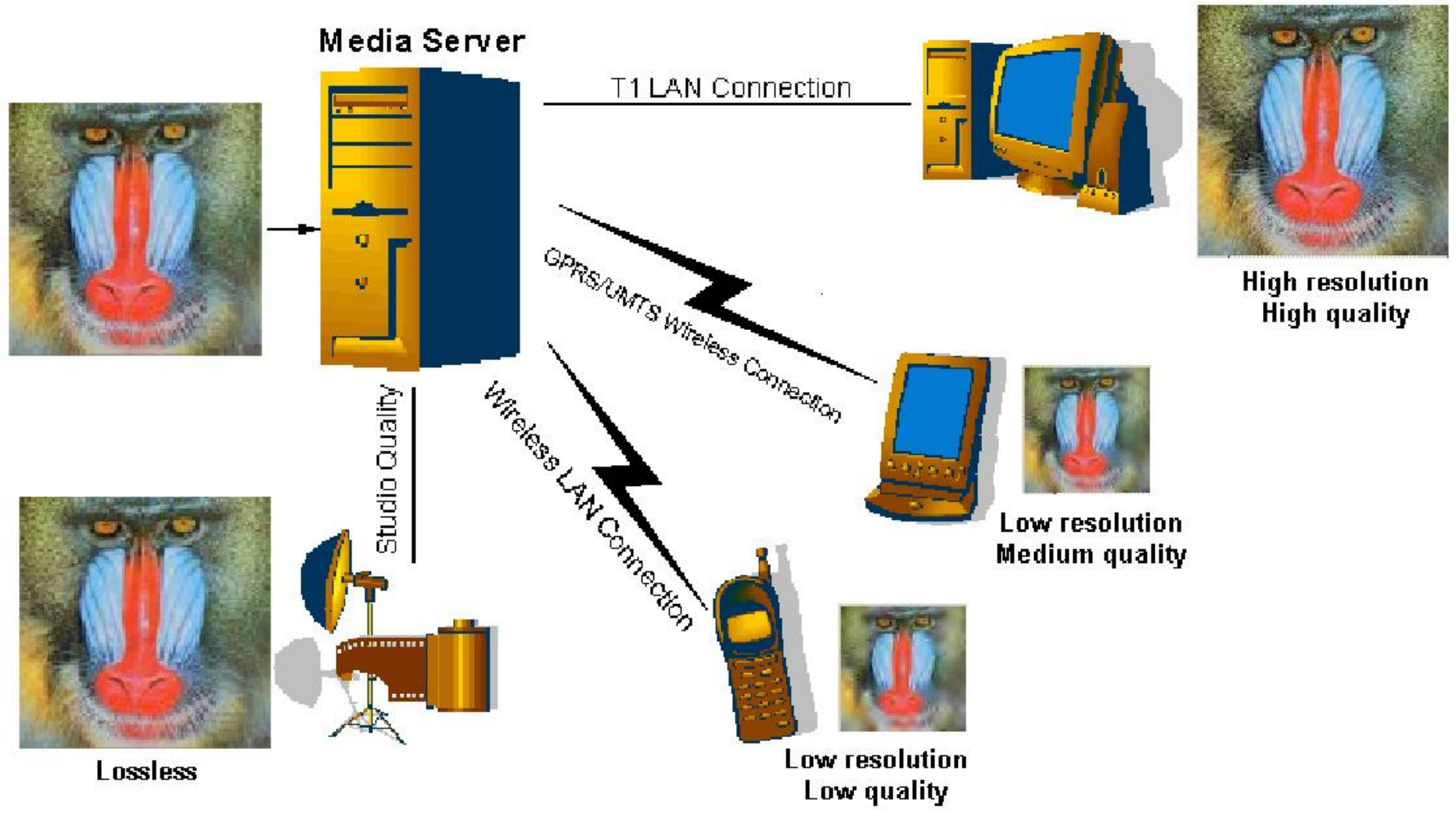
CONCLUSION

JPEG-2000 is unlikely to replace JPEG in low complexity applications at bit-rates in the range where JPEG performs well

However, for applications requiring either higher quality or lower bit rates, or any of the features provided, JPEG-2000 should be a welcome standard.

The main reasons why it will be welcomed in the future

- Better qualitative performance at same bitrate with JPEG
- Random Access of Bitstream – supports features such as Region of Interest.
- Compressed Image Domain Manipulation
- Encode once – decode depending on platform


Media Server
T1 LAN Connection
GPRS/UMTS Wireless Connection
Wireless LAN Connection
Studio Quality
High resolution
High quality
Low resolution
Medium quality
Low resolution
Low quality
Lossless

DITHERING

THE PROBLEM

Suppose that an image uses *N* intensity levels (or uses N colors) to represent its content.

Suppose that the rendering device on which this is to be displayed can use only *n* colors, and

$n<N$

How can we reproduce the image in a satisfactory way?

Solution

- Quantization from *N* to *n* levels, but as *n* gets smaller, quantization starts showing artifacts
- What if $n=2$ – use Halftoning, Dithering, Error Diffusion

HALFTONING

Represent image by an array of circles with diameter proportional to *blackness* (1-Intensity).

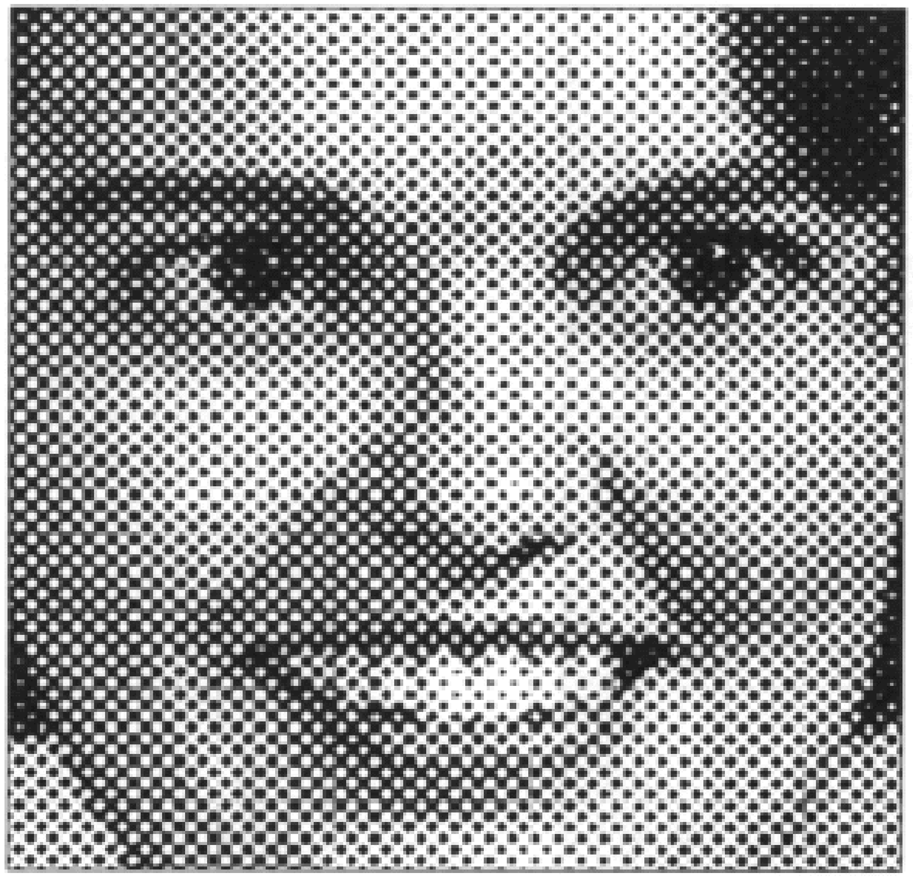
Human visual system performs *spatial integration* **(interpolation)**

> **If we view a very small area from a sufficiently large distance, our eyes average fine details and record only the overall intensity area**

Example of sizes and shapes used

- **Newspapers: 60-80 variable-sized and variable-shaped areas per inch.**
- **Magazine and books: 110-200 per inch**

HALFTONING - EXAMPLE



DITHERING

Dithering is a Generic Algorithm and can be formalized as follows:

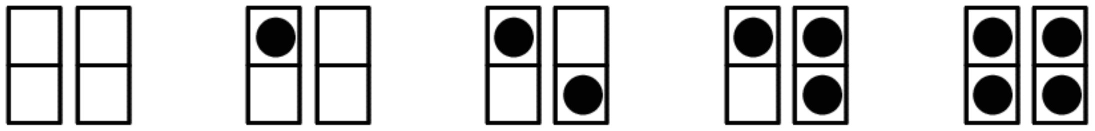
"Given an m x n **array** A **of pixels in grayscale, calculate an array** B **of the same size with zeros (white pixels) and ones (black pixels) such that for every pixel** $B[i\ ,\ j]$ **the average value of the pixel neighborhood will approximately equal the normalized value of** $A[i, j]$**. "**

Normally done by dividing the image into ***k*** **x** ***k*** **blocks. Each block can represent** k^2**+1 intensity levels.**

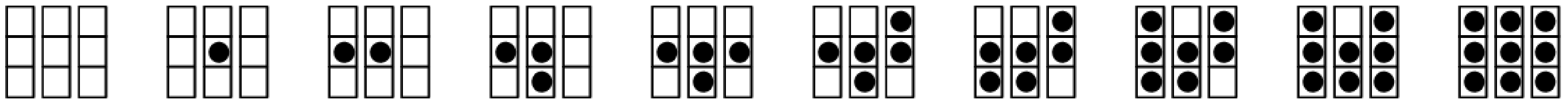
Problem: reduces image resolution to 1/*k* which is OK if display array is larger than the image array

DITHERING MATRICES

Example: k=2 $\Rightarrow$ D = $\begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix}$



Example: k=3 $\Rightarrow$ D = $\begin{bmatrix} 6 & 8 & 4 \\ 1 & 0 & 3 \\ 5 & 2 & 7 \end{bmatrix}$



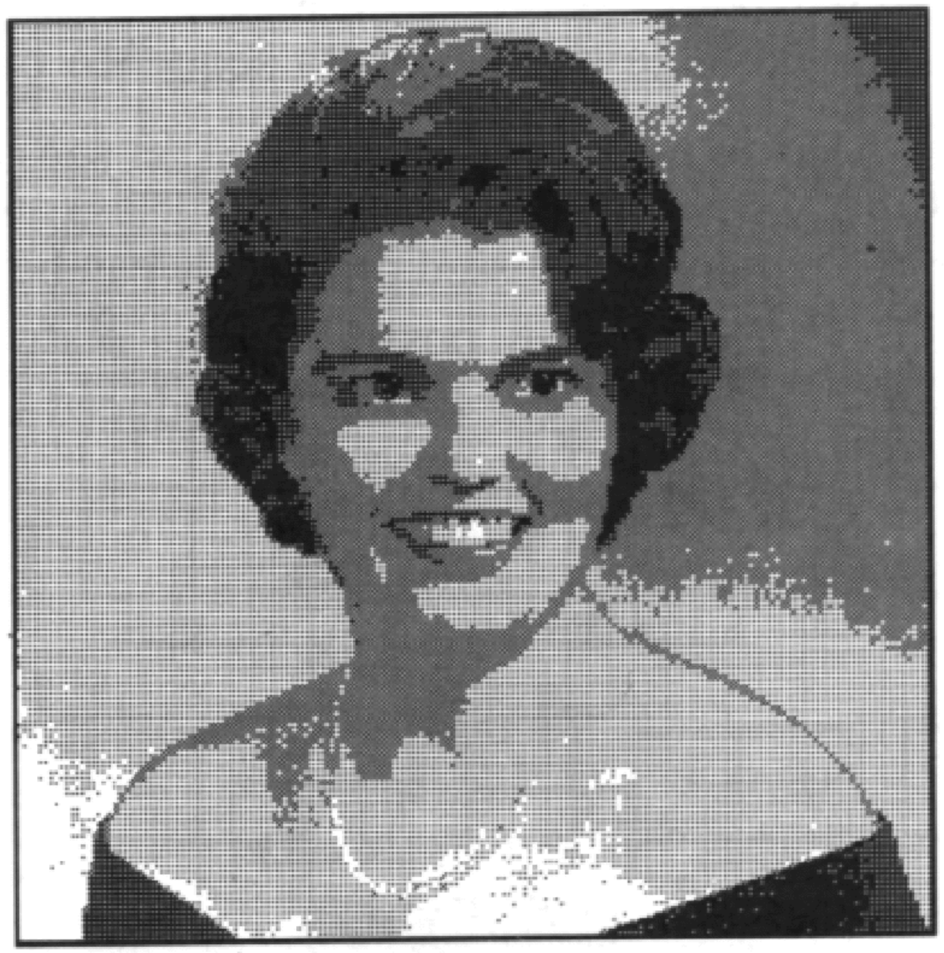
A DITHERING ALGORITHM

It is possible to keep the image resolution the same as the original image by ensuring that each image point *(x,y)* should control only the *(x,y)* display pixel

A possible technique:

- **Compute i = *x* mod k, j = *y* mod *k***
- **Intensify the pixel at point *(x,y)* only if *I(x,y)>D(i,j)*, where *D(i,j)* is the entry of D in the i-th row and j-th column**

DITHERING (EXAMPLE)



ERROR DIFFUSION

When trying to display an image having colors more in number than the display device (or printer device), a selection has to be made to approximate the value of the display color, which is done using

- Precomputed Lookup Tables (LUT)
- Dynamic Color Quantization

Either way the difference in the selector color value and the original value results in an error

Large color errors degrade the quality of the displayed image. If the error is diffused in the neighborhood, such effects are minimized. Error Diffusion Algorithms do this by distributing the color errors among all the pixels such that the total color error for the entire image is zero (or very close to zero).

ERROR DIFFUSION ALGORITHM

Let A be the original image and B be the final image

- Pick up the palette color that is nearest the original pixel's color. This palette color is stored in the destination bitmap B [i,j].
- Calculate the color error A [i,j] - B [i,j]for the pixel.
- Distribute this error to four of A [i,j]'s nearest neighbors that haven 't been scanned yet (the one on the right and the three ones centered below) according to a filter. Eg – Floyd-Steinberg

	X	7/16
3/16	5/16	1/16

ERROR DIFFUSION – EXAMPLE