Formati grafici e multimediali

Davide Rossi

Part I

Colours and Colour Systems

Davide Rossi 2003

Colours and Colour Systems

The human eye can percept light frequencies in the range 380-770 nanometers and can distinguish about 10000 different colours simultaneously.

The colour the eye is more sensible to is the green, followed by the red and the blue.

In computer graphics we typically use a trichromatic colorimetric system.Depending on the device used these systems can be separated in two categories:

- **Additive:**
  colors are added to black to create new colors; the more color is added, the more the resulting color tends towards white.
  CRTs are additive.
- **Subtractive**
  colours are subtracted from white to create new colours; the more colour is added, the more the resulting colour tends towards black.
  Printers are subtractive.

Colour Spaces

RGB Red-Green-Blue is an additive color system. In a [0,1] color intensity range (0,0,0) is black, (1,1,1) is white.

CMY Cyan-Magenta-Yellow is a subtractive color system. (0,0,0) is white, (1,1,1) is black.

HSV Hue-Saturation-Value is an encoding of RGB.

YUV Luminance-Chrominance. Is a linear encoding of RGB used in television transmission. Y contains Luminance (brightness) information; U and V are colour information. (Similar colour spaces are YCrCb and YPbPr0).

Displays and Colours

In a computer display the images are rendered as a grid of dots called pixels.

The pixel grid is stored in an ad-hoc memory of the Video Adapter usually referred to as Video RAM or Video Memory.

Depending on the number of colours associated to each pixel, the amount of memory needed to contain the display data can be very different.If our display can only contain black and while pixels we can encode the video memory in such a way each byte represents 8 pixels. Thus a 1024x768 grid can be stored in 98304 bytes. If the display can show 16777216 simultaneous colours we need three bytes per pixel for a total amount of 2359296 bytes (i.e. 24 times more than the black and white case).

Usually, if the display adapter maps directly the video memory to RGB components, the memory can be arranged in such a way each pixels is encoded in two or three bytes (5-5-5, 5-6-5, 8-8-8 bits format) often referred to as hi-colour and true-colour modes, respectively.

Palettes

Mostly because of physical limitations of the output devices the number of colours that can be used simultaneously can be limited.

Suppose we have a video adapter that uses the RBG colour space and is able to handle 256 levels of intensity range for each primary colour.

This video adapter has a grid of 1024 * 768 pixels but only 1MByte of video memory; using three bytes per pixel is then impossible since we would need more than 2MByte. To solve this problem the device uses a colour palette to store 256 different colours encoded using three bytes each and uses each byte in the video memory as an index to select the colour from the palette. This way only 787200 bytes of memory are needed but only 256 colours can be displayed simultaneously.

Bitmaps, Vectors & Metafiles

Depending on the use they are created for, the input devices they are generated by (digital cameras, scanners, etc), the output devices they are destined to (displays, printers, VCRs, plotters, etc), whether they are animated or not, images can be encoded using:

- Bitmap
- Vector
- Metafile
- Scene
- Animation
- Multimedia formats.

Still Images: Vectors

Vector images are built from mathematical descriptions of one or more *image elements*. Usually not just simple vectors are used in the encoding of vector images but also curves, arcs and splines.

Using these simple components we can define complex geometrical shapes such as circles, rectangles, cubes and polyhedrons.

Vector images are then encoded using sequences of basic shapes and lines with their parameters (starting point, length, etc).

Vector images are useful to encode drawings, computer-generated images and,in general, each image that can easily be decomposed in simple geometrical shapes.

Editing Vector Images

Vector images can be edited by adding/removing shapes and by changing shapes parameters by applying *transformations* (such as scale, translation, etc).

It is important to remark that by applying transformations no information is lost: in fact we can always apply new transformations to restore the previous state of the image.

Pros and Cons of Vector Formats

Advantages:

- Vector data can be easily scaled in order to accommodate the resolution of the output device.
- Vector Image files are often text files and can be easily edited.
- It is easy to convert a Vector Image to a Bitmap Image.
- Translate well to plotters.

Drawbacks:

- Vector cannot easily be used to encode extremely complex images (such as photographic images) where the contents vary on a dot-by-dot basis (but: see fractal image compression)
- The rendering of a Vector Image may vary depending on the application used to display the image
- The rendering of an image may be slow (each element must be drawn individually and in sequence)

Still Images: Bitmaps

Bitmap images are generated by scanners, digital cameras (and few other devices) and are the “natural” formats for displays and printers.

Bitmap images are built by a grid of colours.

In a display the image is grid of pixels, in a printer is a grid of dots.

Depending on the capability of the device the pixels/dots can have from two to millions of colours.

Editing Bitmaps Images

Bitmap images can easily be edited using interactive or batch programs.

We can apply them filters, modify colours, edit small parts.

Usual operations include:

- Blur and Sharpen.
- Colour correction.
- Brightness/Contrast adjustment.
- Touch up.

The drawback is that they don't scale well. If we shrink a bitmap image and then we enlarge it back to its original size, information is lost!

Pros and Cons of Bitmap Formats

Advantages:

- Easily encoded in array of bytes.
- Are produced by many input devices.
- Easy to edit.
- Translate well to grid output devices such as CRTs and printers.

Drawbacks:

- Large.
- They do not scale well (it is easy to lose information).

Bitmap vs. Vectors

Converting images from one format to the other is troublesome and, also if the operation is achieved with success, further issues must be considered.

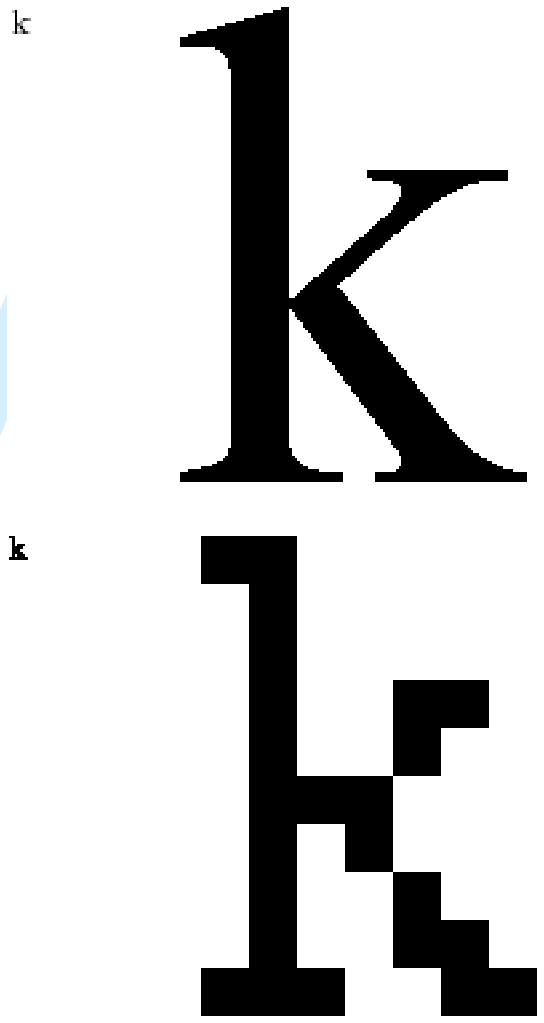
Vectors to Bitmap

- The operation is quite easy: the application has simply to render the vector image.

Bitmap to Vectors

- The operation is troublesome: complex math algorithms come into play and, for complex images, they often fail!
  The resulting image can be much bigger (as in the case of photographic images) and the rendering can take lot of time.

Bitmap & Vector Characters



Part II

Data Compression

Data Compression

As stated before one of the drawbacks of the bitmap formats is that they need lots of memory to encode an image. This affects mostly the file size of a bitmap image and the time needed to transmit the image over a network.

A wide variety of data compression algorithm have been applied to bitmap images in order to reduce the resulting file size.

While conceptually every data compression algorithm may be used to compress a bitmap image we will see that some algorithm results more effective than others on image data.

Compression Terminology

Lossless/Lossy

- The first distinction we have to make about compression methods is whether they allow or not perfect data restoring (we say they are,respectively, *lossless* or *lossy*)

Raw and Compressed Data

- We use these terms to refer to the original image data and to the compressed image data

Compression Ratio

- The ratio of raw data to compressed data

Symmetrical and Asymmetrical Compression

- When a compression algorithm uses roughly the same amount of work to archive both compression and decompression is said to be *symmetrical*

Common Bitmap Compression Methods

Lossless methods:

- Pixel Packing
- Run-Length Encoding (RLE)
- Lempel-Ziv(-Welch) Compression (Dictionary-based Compression)
- Arithmetic Encoding, Huffman Encoding (Entropy Encoders)

Lossy methods:

- DCT Compression (JPEG)
- Wavelet compression
- Fractal Compression

Pixel Packing

Pixel Packing is not a compression method per se: it is simply a convenient way to store the colour data in a byte array. Suppose you have a palette-based,four colour image. We can use one byte for each pixel but we could also encode the colour information so that each byte is used to store four pixels by splitting the byte in four couples of bits.

Run-Length Encoding (RLE)

RLE is mostly useful when we have to deal with palette-based images that contain large sequences of equal colours.

The idea in RLE is in fact to encode long sequences of the same value with the shortest possible encoding.

A possible RLE encoding is the following:

each sequence in the file is a control number followed by a variable number of bytes.

If control number `n` is positive then the next `n` bytes are raw data; if `n` is negative then the next byte is repeated `-n` times in the raw data.

For example:

```
4536777764444570000011
```

becomes

```
4 4536 -4 7 1 6 -4 4 2 57 -4 0 -2 1
```

RLE is used in the TARGA file format and in Windows Bitmap (.bmp) file format.

Dictionary-based compression: LZ77

LZ77 (Abraham Lempel, Jakob Ziv - 1977) is a dictionary-based compression scheme and is the first of a set of similar data compressors often referred to as the LZ family.

The principle of encoding

The algorithm searches the window for the longest match with the beginning of the lookahead buffer and outputs a pointer to that match. Since it is possible that not even a one-character match can be found, the output cannot contain just pointers. LZ77 solves this problem this way: after each pointer it outputs the first character in the lookahead buffer after the match. If there is no match, it outputs a null-pointer and the character at the coding position.

The encoding algorithm

Set the coding position to the beginning of the input stream;

find the longest match in the window for the lookahead buffer;

output the pair (P,C) with the following meaning:

- P is the pointer to the match in the window;
- C is the first character in the lookahead buffer that didn't match;

if the lookahead buffer is not empty, move the coding position (and the window) L+1 characters forward and return to step 2.

LZ77 Example

Pos	1	2	3	4	5	6	7	8	9
Char	A	A	B	C	B	B	A	B	C

Step	Pos	**Match**	**Char**	**Output**
1	1	--	A	(0,0) A
2	2	A	B	(1,1) B
3	4	--	C	(0,0) C
4	5	B	B	(2,1) B
5	7	A B	C	(5,2) C

Entropy Encoding

The idea is to give a shorter codewords to more probable events. The data set could be compressed whenever some events are more probable then other. There are set of distinct events e_1, e_2, …, e_n which together are making probability distribution P (p_1, p_2, …, p_n). Shannon proved that the smallest possible expected numbers of bits needed to encode an event is entropy of P, H(P).

$$H(P) = - \sum_i p_i \ln(p_i)$$

Huffman Encoding

Huffman encoding is a well known encoding scheme based on statistical properties of the source data. Each code from the source is associated to a variable bit length code used in the output. Compression is achieved by associating shorter output codes to more frequent input codes.

The association between input and output codes can be pre defined or calculated at run time.

Huffman Algorithm

The algorithm for building Huffman code is based on the so-called "coding tree". The algorithm steps are:

Line up the symbols by falling probabilities

Link two symbols with least probabilities into one new symbol which probability is a sum of probabilities of two symbols

Go to step 2. until you generate a single symbol which probability is 1

Trace the coding tree from a root (the generated symbol with probability 1) to origin symbols, and assign to each lower branch 1, and to each upper branch 0

Arithmetic Coding

The Arithmetic coding with accurate probability of events gives an optimal compression.

Algorithm:

The coder starts with a “current interval” [H, L) set to [0,1).

For each events in the input filed/file, the coder performs an (a) and (b) step.

a. the coder subdivides current interval into subintervals, one for each event.
b. The size of a subinterval is proportion to to the probability that the event will be the next event in the field/file.the code selects the subinterval corresponding to the event that actually occurs next and makes it the new current interval

The output is the last current interval. Better to say, the output are so many bits that accurately distinguish the last current interval form all other final intervals.

Arithmetic Coding Example

Step	Events	Prob.
1	a,b	2/3,1/3
2	a,b	1/2,1/2
3	a,b	3/5,2/5

Input stream: b a b

[0,1)	[0, 2/3), [2/3, 1)	[2/3, 1)	[2/3, 5/6), [5/6, 1)	[2/3, 5/6)	[2/3, 23/30), [23/30, 5/6)	[23/30, 5/6)

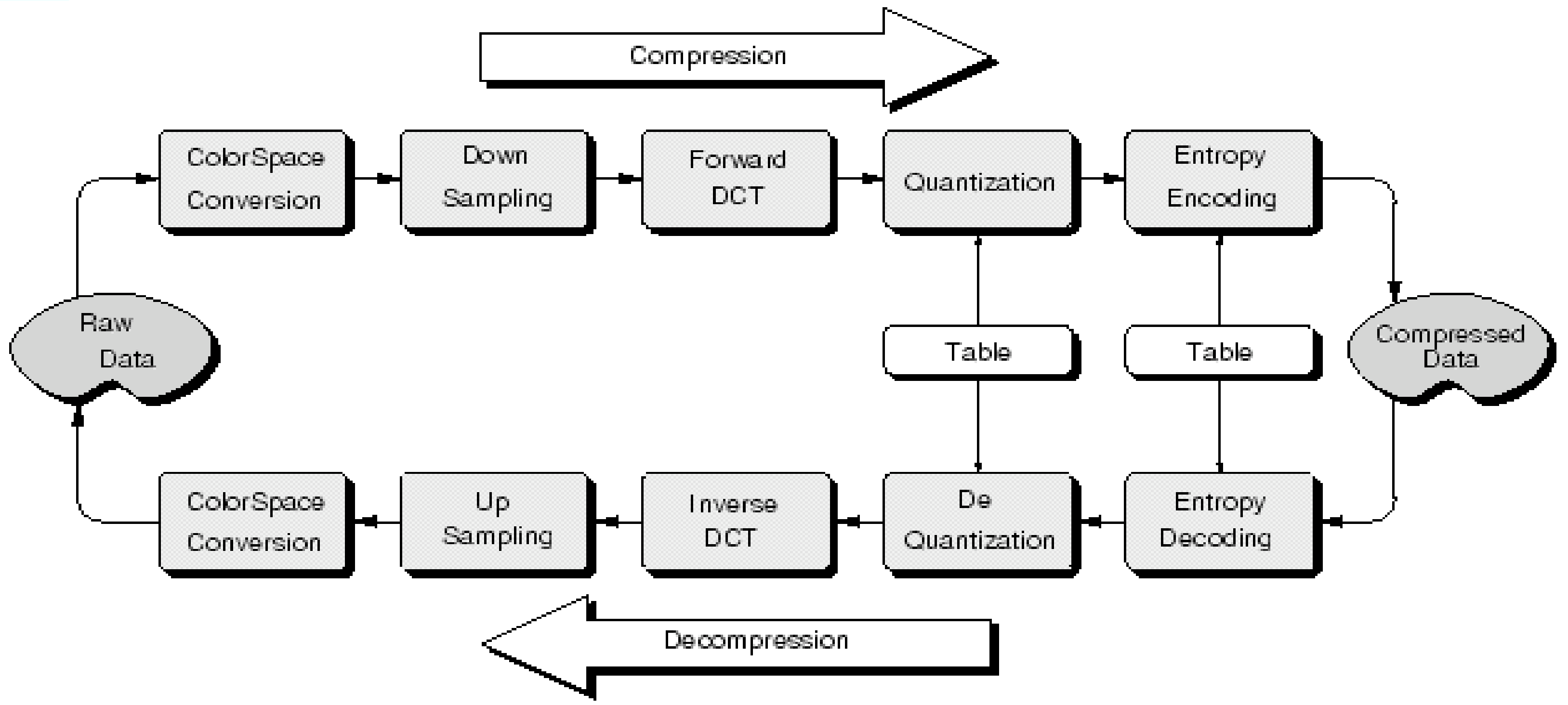
Result: 23/30

Baseline JPEG Compression

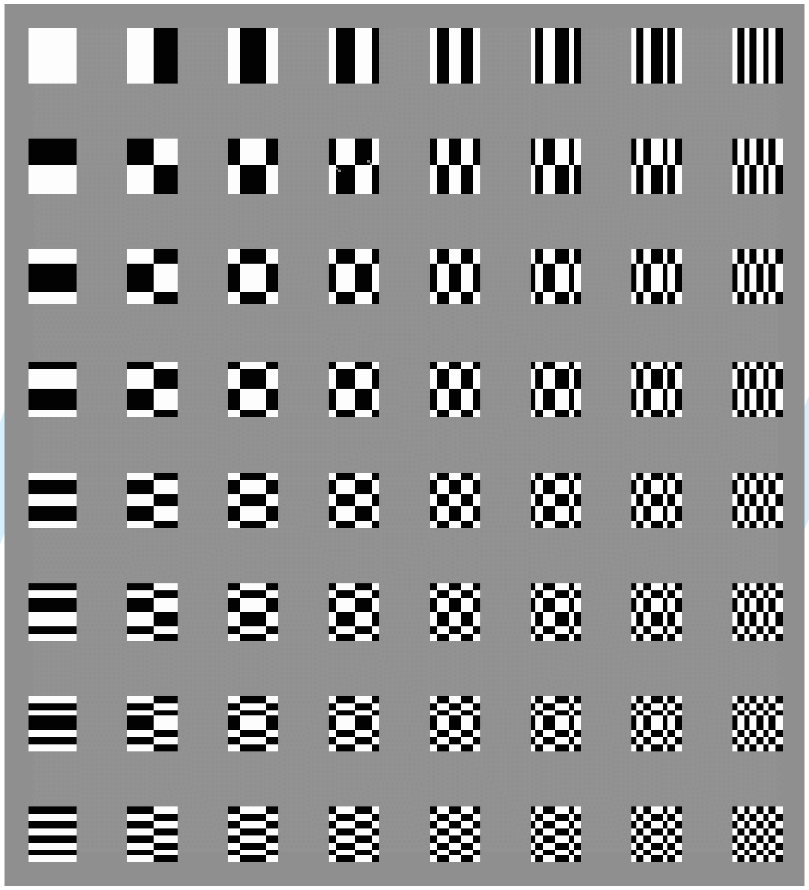
The *baseline* JPEG (Joint Photographic Experts Group) compression (from now on JPEG) is a lossy compression scheme based on colour space conversion and discrete cosine transform (DCT).

JPEG works on true colour (24 bits per pixel) continuous-tone images and achieves easily compression ratio of 25:1 with no visible loss of quality.

JPEG Encoding Flow Chart



DCT Basis Functions



Wavelet Compression

Wavelet compression is similar (in principle) to JPEG compression. The main difference is the use of wavelet based techniques in place of DCT-IDCT transformations.

Wavelets are mathematical functions that cut up data into different frequency components. They have advantages over traditional Fourier and DCT methods in analizing signals that have discontinuities and spikes.

Comparative researches indicate that wavelet compression is slightly better than DCT-based JPEG but compression and decompression times are longer.

This compression technology is the compression technology used in the JPEG-2000 standard.

Fractal Compression

Fractal compression is a very complex (lossy) compression technique.

It is based on the transformation of a bitmap image to a vector-like mathematical representation using *iterated function systems* (e.g.fractals).

Fractal compression is asymmetrical as the compression step is very much slower than decompression (decompression is, in fact, just a rendering algorithm) but there is a lot of work going on to overcome this problem.

The advantages of fractal compression are the good compression ratio that can be achieved with little degradation of the image quality and the ability (just like with vector formats) to scale the image without losing information and adding noise.

The drawback is that not everyone agrees on the advantages.

Fractal Compression Algorithm

Graduate Student Algorithm:

Acquire a graduate student.

Give the student a picture.

And a room with a graphics workstation.

Lock the door.

Wait until the student has reverse engineered the picture.

Open the door.

Notes on using lossy compression

It should be noted that all the lossy compression schemes are always lossy: a decompressed image is never the same as the original one.

This means that re-compressing a JPEG compressed image results in added information lost so lossy compression is never a good choice for intermediate storage.

Part III

Still Graphics File Formats

The GIF 87a File Format

The GIF87a (Graphics Interchange Format) file format is useful for storing palette based images with a maximum of 256 colours.

The compression technique adopted by the GIF format is LZW so it is possible to achieve high compression ratios only with non-photographic images.

Within a single GIF file multiple images can be stored (with their own palettes called local colour tables).

Since LZW is a quite simple compression scheme it is quite easy to write a GIF decoder and this has lead to a wide adoption of this format among different applications

The GIF 87a File Format (2)

Images can be stored in a GIF file using the interleaving format: images line are not stored sequentially in a top-bottom order but using the following scheme:

```
0
3
2
3
1
3
2
3
```

The GIF 89a File Format

GIF 89a is an extension of the GIF 87a file format.

If GIF89a we have *Control Extension blocks* that can be used to render the multiple images in the same file in a multimedia presentation.

Control Extension blocks include Graphics Control Extension (how to display images),Plain Text Extension (text that have to be overlapped to the image),Comment Extension (human readable comments)and Application Extension (proprietary application information).

Since images could overlap during the rendering it is possible to define a palette index that is rendered as *transparent*.

The JFIF File Format

The JFIF (JPEG File Interchange Format) format is the standard file format adopted for JPEG compressed images.

A JFIF file is composed by segments identified by markers.

An optional segment in the file can contain a thumbnail of the image in uncompressed RGB format.

The JFIF format does not allow the storage of multiple images in the same file.

JFIF supports progressive JPEG encoded images: the decoder returns a set of images progressively closer to the original image.

The PNG File Format

The PNG format has been designed by the internet community to overcome patenting issues related to the use of LZW compression in GIF files.

PNG uses in fact a patented-free version of LZ encoding that archives higher compression ration than LZW.

Here is a (incomplete) list of improvements of PNG w.r.t. GIF:

- support for true-colour images
- support for alpha channels
- 16 bits for channel optional accuracy

Part IV

Animation

The MPEG Motion Image Compression

MPEG is a compression scheme for motion images and audio developed by the Motion Picture Expert Group committee. Its image compression scheme is based on the DCT and is quite similar to JPEG.

The main difference between JPEG and MPEG is the usage of motion-compensation techniques to archive higher compression ratios.

A MPEG (-1 or –2) video stream is a sequence of I (Intra), P (Predicted) and B (Bi-directional) frames.

The MPEG Motion Image Compression (2)

I-frames are encoded using only information from the original frame;their encoding scheme is very similar to that used in baseline JPEG.

P-frames contain motion-compensated information w.r.t. the previous I- or P-frame. The image in decomposed in macroblocks (16 by 16 pixels); each macroblock is enocoded either as new or as moved from a given position. Each moved macroblock has an associated 8x8 error block. The encoding of a moved macroblock is represented by a motion vector and the error block.

B-frames contain motion-compensated information w.r.t. the previous I- or P- frame and the next I- or P-frame.

The MPEG Motion Image Compression (3)

While MPEG techniques allows for a good compression ratio it turns out that to decode P-frames we have to store in memory a previously decoded I- or P-frame, and to decode B-frame we have also to decode frames that come later in the input stream.

A typical sequence of a MPEG stream looks like:

`IBBPBBPBBPBBIBBPBBPBBPBBI`

Typically I-frames recur every 12 frames in order to allow re-synchronization and to avoid error propagation.It should also be noted that the usage of B- and P-frames implies that MPEG is an asymmetrical compression scheme.

MPEG-4 Standard

MPEG-4 video adds:

Support for HBR and VLBR video

Use of Audio/visual objects (AVOs)

Motion prediction and compensation based on:

- Global motion compensation using 8 motion parameters that describe an affine transformation
- Global motion compensation based on the transmission of a static "sprite“
- Global motion compensation based on dynamic sprites

The AVI File Format

AVI (Audio Video Interleaved) is a general purpose file format introduced by Microsoft in the context of RIFF (Resource Interchange File Format).

AVI does not introduce new technologies, it simply defines a file format to store audio/video information that can be compressed using different codecs (e.g. the popular INDEO compression technology from Intel). INDEO is a video compression technology that uses a hierarchical image decomposition: the image is decomposed in smaller areas until the contents of each area can be considered as uniform. Motion compensation techniques among uniform areas are used to achieve higher compression ratios.

Part V

Audio

The MPEG Audio Compression

MPEG Layer1, Layer2, Layer3 and AAC are audio compression schemes based on psychoacoustic models.Technically speaking Layer1 and Layer2 are based on subband coding while Layer3 and AAC are based on hybrid (subband/transform) coding.The input signal is sampled at 32, 44.1 or 48 kHz.Typical bit rates for the 4 compression systems are:

Layer1	32-448 kbps
Layer2	32-384 kbps
Layer3	32-320 kbps
AAC	32-192 kbps

MPEG audio uses monophonic, dual-phonic, stereo and joint-stereo models, AAC adds surround support.

Psychoacoustics

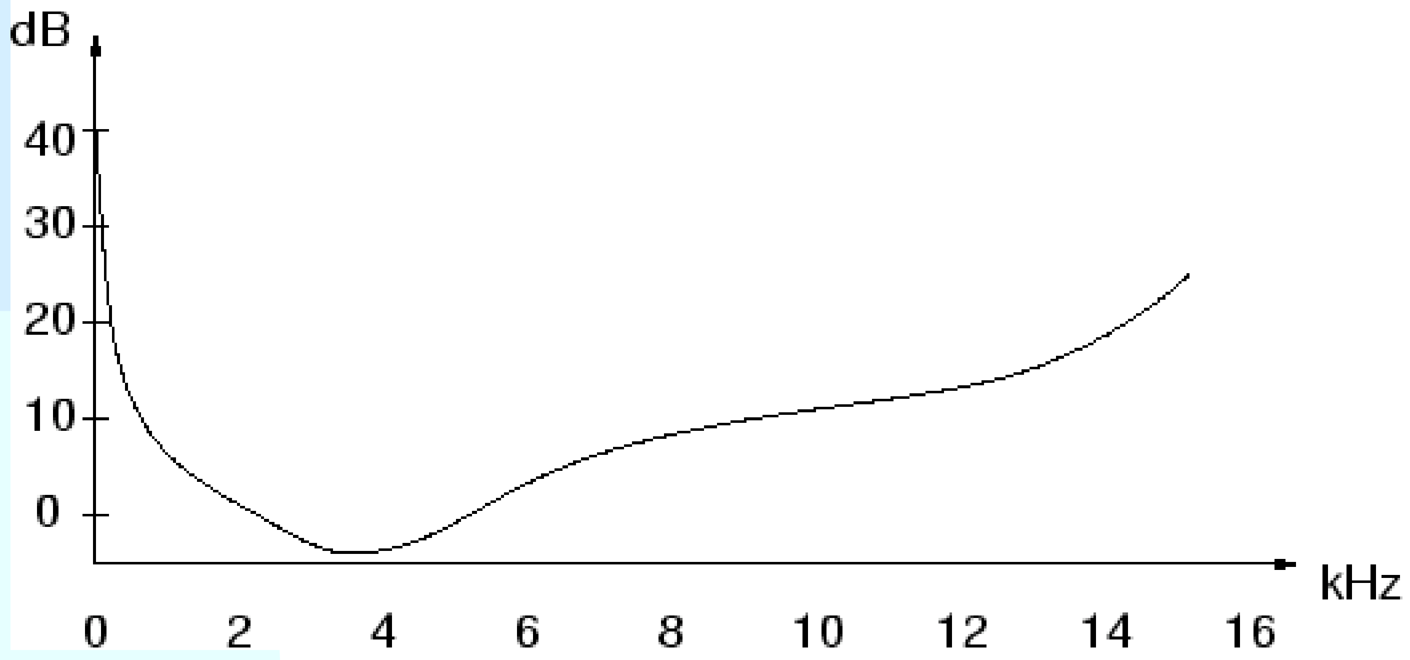
Psychoacoustic models used in MPEG audio compression are based on:

- ear sensitivity w.r.t. frequency
- simultaneous frequency masking
- temporal frequency masking.

Ear Sensitivity and Frequency

The human ear is not equally sensible to signals at different frequencies.

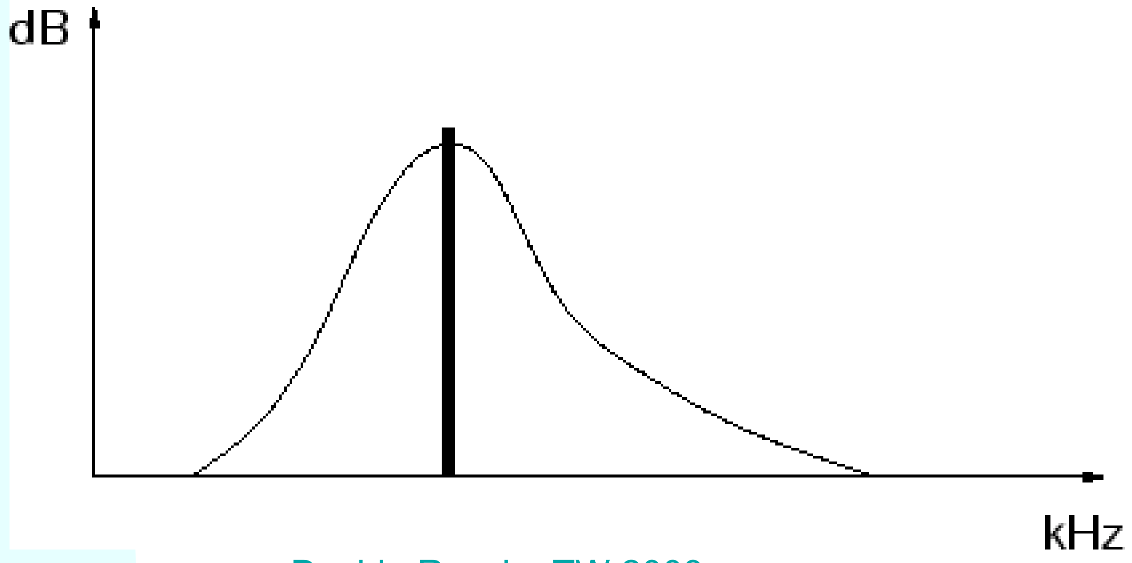
The diagram below plots the ear threshold in quiet.



Frequency Masking

High level tones at a given frequency mask lower tones at close frequencies.The masking band depends on the frequency of the masking signal.

The diagram below plots the masking for a tone at about 2 kHz.



Steps in MPEG Audio Compression

time-to-frequency transformation (uses a polyphase filter bank)

split tonal and non-tonal components

calculate mask spreading function

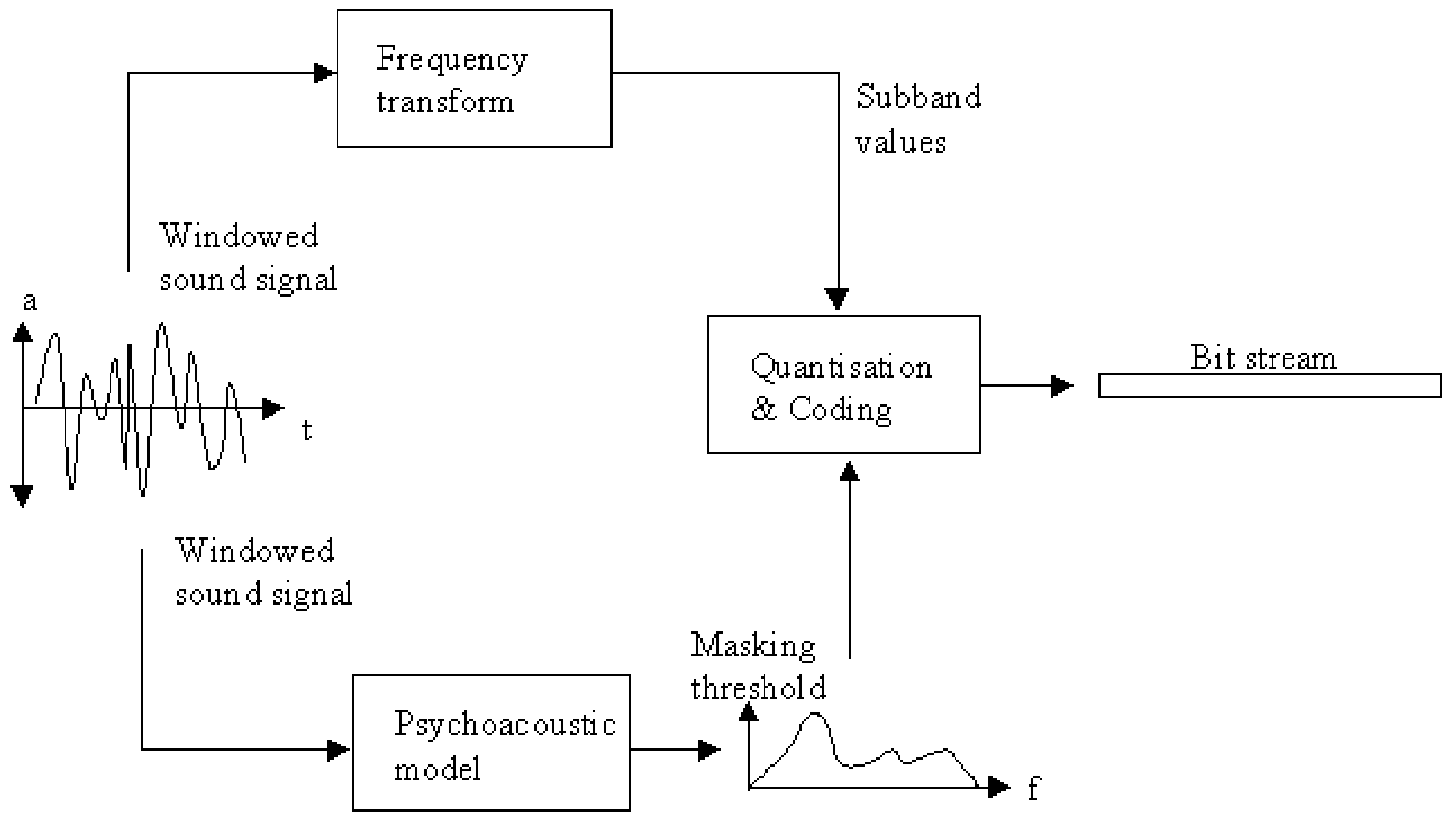
apply masking

calculate signal-to-noise ratio (SNR)

choose quantization

(layer 3 and AAC) use entropy encoder

Steps in MPEG Audio Compression



Usage of the Psychoacoustic model

The filter bank outputs 32, equally-spaced, signal bands (note: the bands are overlapping; they should map the critical bands but they don't).The output of the filter bank is used to compute the masking frequencies using the amplitude of the signal in each band and a spreading function.Signals in each band are then encoded using a quantization relative to the masking present for that band.

Block of 12 samples from each filter are analized at once in Layer1. Three12-samples blocks are analized for Layer2, Layer3 and AAC.The usage of three blocks at once allows for temporal masking to be taken into account; this also helps reducing data for level adjustment.

Main Enhancements in Layer3

Layer 3 MPEG audio encoding adds to Layer1 and 2 the following peculiarities:

- applies alias reduction
- applies non uniform quantization
- uses entropy encoding
- uses a bit reservoir

Main Enhancements in AAC

AAC MPEG audio encoding adds to Layer2 the following peculiarities:

- support for surround signals
- uses MDCT on filter bank's outputs
- uses Temporal Noise Shaping (TNS)
- uses backward adaptive prediction
- adds gain control and hybrid filter bank

Licensing

The myth: MPEG Audio is “freeware”.

The truth: you probably need a license!

SOFTWARE CODECS (mp3 licensing examples)

- Decoders. Freeware: OK; $0.75 per unit if sold or **$50,000** one-time paid-up
- Encoders. $2.5 (enc) $5 (codec) per unit or **$60,000** one-time paid-up

HARDWARE CODECS

- Decoders. $0.75 per unit
- Encoders. $2.5 (enc) $5 (codec) per unit

MUSIC DISTRIBUTION

- 2% of revenue (revenues > $100,000 year)

MINIMUM ROYALTIES

- $15000 (!!!)