SERIE B — INFORMATIK

Digital Image Compression – A Brief Overview

Yachin B. Pnueli*

B 94–14
June 1994

Abstract

We give a brief survey of the framework, relevant issues and existing methods in the field of digital image compression. Accept for section 3 where additional references are given, we basically follow the exposition of the field as presented in

Digital Pictures - Representation and Compression
Arun. N. Netravali and Barry G. Haskell
AT&T Bell Laboratories Murray Hill, New Jersey
Plenum Press, London and New York 1988.

All figures that contain page numberings are taken from the above.

*Institut für Informatik, Fachbereich Mathematik und Informatik, Freie Universität Berlin, Takustr. 9, D-14195 Berlin, Germany, e-mail: yachin@inf.fu-berlin.de. Since Fall 1993 Minerva Fellow.

Contents

1 Introduction - Images and Digital Images

We define the concepts of an analog image, a digital image and a movie. We discuss some issue related to A/D, D/A conversion. Finally we concentrate on two special cases: raster scan images and black/white images.

1.1 Analog Images

For our purposes, we define:

- A *monochromatic image* is a function $I_{mono}(x, y) : \mathbb{R}^2 \to [0, 1]$.
- A *color image* is a function $I_{color}(x, y) : \mathbb{R}^2 \to [0, 1]^3$.
- A *movie* or a sequence of images is obtained by adding a third parameter which represents time -$I_{mono}(x, y, t) : \mathbb{R}^3 \to [0, 1]$.

By restricting ourselves to 2-D functions we exclude from our discussion certain classes of "objects" which do contain elements of 3-D structure and which might intuitively be thought of as images, for example:

- A representation of a 3-D scene in a computer (such as the output of a solid modeler in a CAD system).
- A "scene" as viewed via an optical system when we have control over the focus of the lenses and hence can obtain 3-D information.
- Any "real-world scene" as viewed by most of us - as our binocular vision gives much 3-D information on the scene.

This definition is, however, not too restrictive as most images we would like to transmit/compress fall naturally within it. It includes photos, images as scanned by video and other cameras, outputs of scanners and computer generated images (after passing the "ray tracing stage"). Note that our images can include 3-D information (such as shading) but only implicitly via the 2-D intensity values.

A further demand which we impose and which is "natural" is that an image has finite support (it is non zero only within a certain region in the plain). Usually this support is a rectangle and then the ratio between its length and its width is called the *aspect ratio* of the image. Television images have an aspect ratio of $3 : 4$ and devices called *high definition television* have an aspect ration of $3 : 5$.

A monochrome image is formed by capturing the intensity of light at every point in some sensing device and hence, we can think of a value 0 as black (no light) and 1 as white (full intensity). Any value in between represents some grey level. Intuitively a color image contains at each point also information about the waveform of the light. One might rightfully suspect that a range of $[0, 1]^3$ is insufficient to represent this waveform information in full. It turns out, however, to be sufficient for representing our color sensation in full. This is due to the way our visual system perceives color.

In our visual system we have essentially 3 independent channels which sense and transmit visual information to the brain (4 if one counts the monochromatic night-vision system). The color sensation we perceive comes from the respective intensities of the pulses along these three channels. Hence the range of $[0,1]^3$ does not model the waveform of the light but rather our trichromatic response to it. As the three channels of the visual system have pick response at wavelengths $\sim 445, \sim 535, \sim 570$ nanometers and hence are known as the blue, green and red channels respectively, it is common to use the range $[0,1]^3$ to represent the so called *blue, green* and *red* components of the image. There are other tri-valued ways to represent color and essentially one can transform any such representation to another and hence any such representation can be used. (We'll say more on the subject of color when we discuss the human visual system).

1.2 Digital Images and A/D, D/A conversions

A *monochromatic digital image* is defined as a function $I_{mono}(x,y) : \mathbb{N}^2 \to \{i/2^n\}$ where n is some fixed integer representing the number of bits per pixel and i runs from 0 to 2^n. A *color digital image* is respectively a function $I_{mono}(x,y) : \mathbb{N}^2 \to \{i/2^n\}^3$. As our functions have finite (rectangular) support, a digital image can be represented as a matrix where each entry is an integer (a vector of 3 integers - for color).

When we discuss digital compression schemes we usually assume a set-up as shown in figure 1.1a and hence usually when we speak of images we shall assume we are speaking about digital images.

However in many cases, especially when the image is for human consumption, the more realistic set-up is as shown in fig 1.1b. Hence we shall now discuss in some detail the important issue of converting analog images to digital and vice-versa. Basically this conversion can be though of as involving two distinctive processes

- Digitization - the process of converting a function $\mathbb{R}^2 \to [0,1]$ to a function $\mathbb{N}^2 \to [0,1]$.
- Quantization - the process of converting a function $\mathbb{N}^2 \to [0,1]$ to a function $\mathbb{N}^2 \to \{i/2^n\}$.

1.2.1 An Intuitive Treatment of Digitization and Analogization

To convert an analog image to a digitized image we do a process of sampling - that is we impose a grid G over the support of $I_{analog}(x,y)$ and set

$$\forall i,j \in G : I_{digitized}(i,j) = I_{analog}(x=i, y=j)$$

Typically the grid we use is rectangular or shifted rectangular (note that some shifted rectangular grids can be regarded as rectangular if we rotate our axis by 45^o). In some particular cases other grids are used such as, hexagonal grids and aperiodic grids which models the density of sensors in the human retina (fig. 1.2).

Prior to sampling an image one must decide on the resolution of the grid (the distance between adjacent grid points). Clearly if the resolution is too small we loose information about rapid changes in image intensity and if the resolution is too large we pay a penalty in the size of the digitized image. For different tasks different specific compromises might be chosen as a tradeoff between the desired fidelity and the quantity of information we can handle. There are however generally accepted standards for "general purpose viewing" of "regular images".

Many times it is desirable that the digitized image be a square matrix. In such a case the aspect ratio of the original image is also the aspect ratio of each pixel in the digitized image.

Consider now an image sampled at a given resolution. If the image contains changes that are too rapid relative to this resolution the digitized image might not be a good representation of the original (see fig. 1.3).

This problem is known as *pre-aliasing* and can be solved by taking as the sampled value at i, j, instead of the value $I(x, y)$, some weighted average of the values in the neighborhood of x, y. Mathematically this is described by a convolution:

$$\forall i, j \in \mathbb{N} : I_{dig}(i, j) = \int_{i-\delta}^{i+\delta} \int_{j-\delta}^{j+\delta} I_{anal}(x, y) w_{pre}(i - x, j - y) dx dy$$

with the kernel, w_{pre}, being an averaging function which peaks at $(0, 0)$ and integrates over its entire support to 1. A typical example is the Gaussian filter $Kexp(-ax^2 - by^2)$ cut off to the region $[-\delta, \delta]$. Note that in real situations it might be difficult to perform pre-filtering with an exact given function, however, usually we have some freedom in selecting the averaging function so that by defocusing the optics or similar such procedures good results can be obtained.

What happens at the other end, when we wish to display the digitized image on an analog device? This question can be rephrased as "what values to give the analog image in those points which are not grid points?" A trivial solution is to set each value to that of the nearest grid point. The resulting image can be acceptable and even good, however only when the resolution of the digital image was high, relative to the resolving power of the human visual system. At other times artificial patterns not present in the original image appear in the displayed image. This effect is known as *post-aliasing* and can be corrected via interpolation methods:

$$I_{display}(x, y) = \sum_{\forall i,j} I_{dig}(i, j) w_{post}(x - i, y - j) \tag{1}$$

Here w_{post} is an interpolation matrix (linear interpolation being the simplest). (Fig. 1.4).

1.2.2 Formal Treatment of Digitization and Analogization

For a more formal treatment of the issue of digitization, we turn to an analysis of images in the Fourier domain.

It is well known that if a 1-D signal is band-limited (there is a cut-off frequency above which the Fourier transform of the function is always zero), it can be sampled without any loss of information, if the sampling rate is above the, so called, Nyquist rate, which is twice the cut-off frequency. This theory generalizes to any dimension and hence applies to images and movies as well. Note, however, that we might need to use different sampling rates for the x, y and t coordinates (when it is advantageous to sample x and y at the same rate we use the maximum of the separate rates).

The main advantage of this theory is that for band-limited images it gives a good criteria of what is the *proper* sampling rate (and hence the proper resolution) for general applications. There is however the problem that our images are assumed finite and hence are never band-limited. This "technical" problem is circumvented by assuming that our image is infinite but periodic - $\forall x, y : I(x, y) = I(x+n, y+m)$ with n, m the dimensions of the actual finite image. This sometimes causes sharp discontinuities at the "connection points" between successive periods. Such discontinuities give rise to high frequency components which cannot be filtered out without noticeably degrading the image, but have little to do with the actual information content of the image. This type of problem is known as the *Gibbs phenomenon* and is sometimes eased by using the cosine transform instead of the Fourier transform. (The idea is that instead of replicating the image, we first reflect it and then replicate the pair of image+reflection thus giving rise to a continuous image).

An interesting side issue is that when images are periodic they can be represented by a Fourier series (a discrete set of coefficients in the Fourier domain). If the image is band-limited as well, this set of coefficients is finite and determined by the Nyquist rate.

$$I(x, y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} c_{ij} \frac{1}{\sqrt{nm}} exp 2\pi \sqrt{-1} \left[\frac{ix}{m} + \frac{jy}{n} \right]$$

with

$$c_{i,j} = \int_0^m \int_0^n I(x, y) \left(\frac{1}{\sqrt{nm}} exp 2\pi \sqrt{-1} \left[\frac{ix}{m} + \frac{jy}{n} \right] \right)^* dxdy$$

with * denoting the complex conjugate. We can now think of the restoration problem (given a digital image find the analog which gave rise to it) as no more than the problem of finding the $n \cdot m$ coefficients c_{ij}. In principle this can be done if we are given any $m \cdot n$ values of the image *regardless of their locations*, provided they give rise to linearly independent equations.

There are however two important advantages to using the regular rectangular grid. The first is that our sensitivity to small perturbations of the sampled values (via sampling noise or quantization) reduces significantly, and the second is that for the regular rectangular grid we can restore the image without the need to solve sets of linear equations:

It is well known that the inverse transform of the step function is the *sinc* ($sinc(x) = sin(x)/x$). Hence we can restore an image by simply convolving the digitized image with the appropriate *sinc* function (eq. 1 with $w_{post} = sinc(x/a)$).

How does the above theory relate to real world situations? Are real images band-limited? Relatively simple arguments can be used to show that real images are seldom band-limited. The theory still remains useful, however, as our visual system is band-limited. Tests have shown that the human visual system can be approximated to a large degree of accuracy as a band pass filter (more on this in section 3). Specifically, spatially 14 cycles/degree and temporally 10-15 Hrz. are representative threshold values above which we do not detect changes, (lighting conditions etc. may effect the results).

It therefore follows that images as perceived by humans contain only frequencies up to a given cut-off frequency.

$$I_{perceived}(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_{anal}(s,t)w_{pre}(x-s,y-t)dsdt$$

with w_{pre} being the appropriate *sinc* function. In general applications, when the image is for human consumption, or is used in some task that replaces the human visual system, we are usually content with representing $I_{perceived}$ and hence the sampling theorem can be used and gives us an indication of the necessary resolution. However that to get correct results we must sample $I_{perceived}$ and not I_{anal}:

When we sample a function in spatial space this results in replications of its image in the frequency space (fig 1.5a). If this is done for a function which is not band-limited (or for a band-limited function but at an insufficient rate) the result of the replications is an overlapping of the replicas in the Fourier domain. (fig 1.5b). This is the *pre-aliasing* effect we identified previously and the solution is as before (accept that we now use $sinc(x/a)$ - the Fourier inverse of the step function as the filter). For a general (non band-limited) image this has the effect of transforming I_{anal} to $I_{perceived}$ (or the best perceived approximation under the given cut-off frequency as determined by the resolution).

When we restore an image, again it is possible that we use an incorrect filter or a filter of incorrect width, with the result that we get an incorrect image (fig 1.5c) this is the afore mentioned *post aliasing* effect.

We note that in practice much of the above gets approximated - we do not use the infinite support of the *sinc* but only part of it, and sometimes we prefer other functions as our filters. The theory does give us, however, an indication of the errors we introduce by the approximation.

1.2.3 Quantization

As with digitization, the first issue of quantization is to decide on the resolution - the number of bits used to represent a value. Typical choices are 6,7, and 8 bits. 6 or 7 bits are sufficient when images are noisy or for some specific tasks. 8 bits are usually sufficient for any general application. Of course in specific systems we might be constrained or privileged to use less or more.

Given that we are going to represent a value by n bits, deciding on a quantization scheme is then no more that partitioning the real segment $[0,1]$ into 2^n sub-segments where a value in each range is transformed to the labeling of its segment.

The most natural and common such partition is the uniform one, where the segments are chosen to be of equal length. It is also the optimal, if we have no a-priori knowledge on the distribution or inportance of grey values. There are however some cases where other quantization schemes are beneficial. When we know that most of the information in an image is in the high, low or mid-range, respectively, we might want to use a scheme which gives more resolution to the preferred range at the expense of others. This results in over or under exposure values for the darker or lighter regions. O'course in specific cases other even more complex partitions might be beneficial (see fig 1.6).

We note that while the issue of quantization is important, it rarely comes up in practice at the A/D or D/A stage, but rather only later when we consider compression schemes. We typically quantize an image to an 8-bit resolution via a uniform quantizer which results in no loss of significant information and then requantize if necessary as part of a compression scheme.

1.3 Raster Scan Devices

The most common way of obtaining images today is via devices which perform so called *raster scan*. These include television cameras, photocopiers etc. Because of its importance we shall describe it in some details.

A raster scan device transforms an analog image $I(x, y)$ into a one dimensional signal $I(t)$ (where t is time) as follows: The image is segmented into N_y adjacent strips (called lines) and is scanned line by line from left to right and top to bottom at a given rate. The resulting signal is a function of time. Typically it contains some "blank" segments between successive lines to allow the scan device to move to the beginning of the next line. Similarly when images change over time (Television) there are also gaps between successive frames to allow the device to return to the top left corner of the image.

To get a digital signal (or image) from this time dependent signal we perform digitization and quantization of the 1-D signal much as described above. There are however some points to notice:

Due to the nature of the process $I(t)$ usually contains sharp discontinuities in those point where one line ends and another begins. This causes problems at the edges of the image.

Due to the operation of the raster scan, pre-filtering is not symmetrical in the horizontal and vertical direction (it is much easier to perform in the horizontal direction). The result is often an insufficiently pre-filtered image. This can be corrected by pre-filtering before the raster scan (by the optics).

For the purpose of sampling, the rate is determined by the horizontal frequency cut-off of the image. (The vertical direction is in a sense already digitized by the raster scan process, hence the number of lines and their width - should depend on the cut-off frequency in the vertical direction).

When we want to restore such a signal - by feeding it into a raster display device. We have a similar asymmetry in post-filtering. While in the horizontal direction

post-filtering is easy, it is difficult in the vertical. Hence many times no vertical post-filtering is performed. This is at all acceptable because our eyes perform some form of post-filtering as we see the image, however sometimes this is insufficient and horizontal lines are visible on many CRT displays when viewed at close range. Sometimes the problem is solved or eased by wobbling the scanning beam as it passes along a line.

When a movie is raster scanned with a T.V. camera, we have an additional problem in that the scanning itself takes time. Unlike motion picture cameras where even when raster scanned, each frame is captured at once, it is possible that objects have moved during the time it takes the scanning device to complete a frame. This effect can be especially noticeable if the fast moving object is large (hence spreads on many lines). It is very difficult to perform proper post filtering over time (as the distance between "relevant regions" in $I(t)$ is large), hence usually no post filtering is performed. Again this can be acceptable, as our eyes perform some filtering over time. However this filtering is much stronger for higher spatial frequencies than for lower ones and this effect is often used to improve the perceived image by what is known as *2-to-1 interlace*:

The moving scene is scanned in a manner that first we raster scan all the even lines of a frame and then all the odd lines of a frame. When we display this signal (first showing the even lines and then the odd lines), we achieve an effect as if the lower spatial frequencies are displayed at twice the rate of the higher ones and hence reduce some of the problem. The price we pay is that now vertical lines appear more noticeable, and if objects are moving fast in the horizontal direction their edges appear jagged.

1.4 Black/White Images

A special class of images which deserves special attention in the class of black and white images (functions to the set $\{0, 1\}$). Such images arise naturally when one is concerned with printed documents, text, engineering drawings, etc. For our purpose here we note the following important factors:

1. While it would seem that B/W images should be trivially quantizable, in reality this is not so. Due to dirt, "paper noise" and scanning noise the "raw" image of a B/W document can contain shades of grey. It is of great advantage if these can be correctly eliminated, as besides increasing the legibility of the document they also ease the process of compression considerably. Hence there is a large variety of theories and ad-hoc methods to properly threshold such documents.

2. The fidelity criteria required for such images are sometimes higher or different than for images, especially considerations such as character legibility and reading fatigue become of utmost importance.

3. Due to the special structure of many such documents - text, line drawing, mosaics etc. They usually can benefit much from special purpose compression schemes as opposed to general images. Thus there are specific methods for text compression, line drawing compression etc.

4. This does not apply to B/W pictures (so called halftone images) where an illusion of grey level is produced by dithering black and white pixels at appropriate rates. While a halftoned image contains less information than a grey level image at similar resolution, its pixels are usually less correlated and hence is more difficult to compress.

2 Statistical Methods

We present the statistical approach to image compression. This approach assumes that the signal (or image) we wish to compress has some underlying probability distribution or statistics. Given the probability distribution we can either achieve optimal compression or show that no compression is possible. We distinguish between *lossless* methods where the uncompressed signal can be recovered exactly and *lossy* methods where information is lost. The compression schemes we cover fall under the broad categories of *Huffman coding, arithmetic coding, predictive coding* which are typically lossless and *transformation coding* which is typically lossy. Some of the methods used to build more sophisticated coding schemes apply to all the above basic methods. These we describe in detail only for Huffman coding.

Since many times even if a probability distribution for the signals to be encoded exists, it is not known a-priori, we also describe some approaches to obtaining or approximating it. Specifically we shall describe common models for images and *universal coding* and *adaptive-statistic measuring* which gather their statistics as they encode the image.

We note that much of what we say applies to signal compression in general (text compression, audio signal compression, etc.). In such cases we shall refer to the image as yet another 1-D signal (say obtained from the 2-D matrix via raster scan order). The main distinction between the 1-D case and images (and movies) stems from the fact that images (and movies) are "in reality" 2-D (3-D) entities and hence display certain 2-D (3-D) correlations which can be utilized to achieve better results.

We remark that the assumptions that a probability distribution exists and that it is "stationary enough" for us to "capture it", hold only as an approximation to real situations. Hence although the methods described are commonly used and achieve good results, the "real" situation is less "clean" and simple than our presentation implies.

2.1 The General Set-Up

Let $S(t)$ be a digital (1-D) signal. Without loss of generality we can view it as a sequence of binary bits or as a sequence of x-bit binary words. Typically it is

convenient to view it as a stream of 8-bit words (bytes), as this is a natural unit for computers, and as bytes can represent in images grey levels at given locations, and in text the ascii value of symbols. An *encoding* of $S(t)$ (denoted by $C(S(t))$) is a different sequence of bits which (hopefully) displays some desirable properties. Such desirable properties fall into two categories:

- *Compression* - making the total length of the signal smaller (or its bit-rate lower, if it is infinite).
- *Robustness* - making the signal immune to less than perfect communication channels. This can be either via *error detection* - being able to detect an error, or via *error correction* - being able to recover the original signal despite an error.

We note that these two aims are somewhat contradictory - as the first attempts to remove all redundancy from the signal and the second must insert some redundancy. However in practice, encodings to achieve both aims are applied to the same signal. Compression - at the image (or source) encoding stage and robustness - at the channel encoding stage (see fig.1.1). If both are done properly, the resulting signal is both significantly shorter and more immune to errors than the original. We shall mostly ignore the issues of error detection and correction (which are known as *coding theory*) and assume that our channels are perfect.

Given a source which generates signals composed of say 8-bit words (all 2^8 of which can appear), one way to compress a signal is to use variable length code words. The idea is to map the source words into code words from some set Λ such that the length of the words in Λ is not fixed.

For example, consider the set of 2-bit words $\{00, 01, 10, 11\}$. A possible signal over this set could be the sequence 00 01 11 11 11. Consider $\Lambda = \{00, 010, 011, 1\}$ and the map

$$00 \to 00, \quad 01 \to 010, \quad 10 \to 011, \quad 11 \to 1$$

Under this encoding, the above signal will be encoded to 00 010 1 1 1. Intuitively one can guess that this code would compress signals for sources where $Prob(11) >> Prob(10)$ and $Prob(11) >> Prob(01)$.

To make an encoding meaningful, we must be able to decode. This implies a restriction that Λ be *uniquely decipherable* - Λ must be such that no sequence of bits can be parsed into two different compositions of words over Λ.

This condition trivially holds if Λ is chosen to be $prefix$, that is no word is a prefix of another. It can be shown that for each uniquely decipherable code there is an equivalent (with equal word lengths) prefix code. Therefore, we can restrict out attention to prefix codes. We also note that such codes have the advantage that they can be decoded *on-line* with a memory of no more than the size of the largest word in Λ, whereas for non-prefix codes, in the worst case, decoding can start only after the entire message is seen.

Our aim is to construct codes (sets Λ and mappings from source words to Λ) such that the length of encoded signals would be minimal. As this cannot be achieved for every possible signal, we would like to achieve this in the expected sense:

Suppose we have a source which generates signals of size n over a set of words Σ (8-bit sequences in our example). An encoding associates with each such signal $\vec{\sigma} \in \Sigma^n$ a code sequence $\vec{\omega} \in \{0,1\}^*$ (or $\in \Lambda^n$ if we encode each source word into a code word). The length of the encoded signal is denoted by $|\vec{\omega}|$. Suppose that for this source, we can associate with each signal a probability $P(\vec{\sigma})$ indicating how often we can expect $\vec{\sigma}$ to be generated. Our aim is to find an encoding scheme C, which minimizes the expected length of the encoded signal.

$$\sum_{\vec{\sigma}} |C(\vec{\sigma})| P(\vec{\sigma})$$

If our probability function is simple, this expression can be simplified. For example, if words in the signal are random variables independent from each other, it is sufficient to minimize the average word length.

$$\sum_{\sigma} |C(\sigma)| P(\sigma)$$

Before we show how to construct codes, we turn to a more theoretical avenue and ask is there a bound on the expressions above? and what is it?

2.2 Information Theory

The aspect of information theory which concerns us is the investigation of the compression problem without recourse to specific methods. The usefulness of this approach is mainly in demonstrating that certain assumptions on the source and the compression method lead to certain absolute lower bounds. This is useful especially when we have codes which achieve such bounds, since then we know they are optimal. Information theory does not, however, specify how to actually construct codes achieving or approaching its bounds nor does it address the issue of the computational complexity of such codes (how to design codes where decoding and encoding can be done efficiently). By necessity, we only briefly review this very rich field.

The main quantity which information theory uses is the *measure of information in a signal* or the *source entropy*. Let $\vec{\sigma}$ be a random variable describing the output of the source. Then entropy is defined as

$$H(\vec{\sigma}) = -\sum_{\vec{\sigma}} Prob(\vec{\sigma}) \log_2 P(\vec{\sigma})$$

The main result of information theory is that no lossless encoding can reduce the expected output length below the source entropy.

$$\forall C: \quad H(\vec{\sigma}) \leq \sum_{\vec{\sigma}} |C(\vec{\sigma})| P(\vec{\sigma})$$

and that there are codes which approach this bound arbitrarily close. This result comes in many variations depending on the type of encoding we use and the probability function of the source. For example, if the source words are independent random variables we have:

$$H(\sigma) = -\sum_{\sigma} Prob(\sigma) \log_2 P(\sigma)$$

and

$$H(\sigma) \leq \sum_{\sigma} |C(\sigma)| P(\sigma)$$

This last result also holds for arbitrary sources when we use encoding schemes that statically map each source word to a code word.

An important observation is that the entropy function is maximized when all possible signals occur with equal probability. This has a number of interesting consequences:

- When all signals occur with equal probability, no compression is possible.
- For an optimal encoding of a signal, the "expected length" (length times probability) of all possible outputs is equal.
- The output of an optimal encoding scheme cannot be compressed further.

The above results have extensions to cases of compression with loss of information and to cases of noises channels. In these cases, to allow formal treatment, we introduce a distortion measure, $d(\vec{\sigma}, \vec{\sigma}')$, which reflects the cost of "seeing' $\vec{\sigma}'$ instead of $\vec{\sigma}$, and assume that the *average* distortion

$$\bar{d} = E(d(\vec{\sigma}, \vec{\sigma}'))$$

is a meaningful measure of performance. We then define a quantity which is called *mutual information*

$$I(\vec{\sigma}, \vec{\sigma}') = \sum_{\vec{\sigma}, \vec{\sigma}'} Prob(\vec{\sigma}) Prob(\vec{\sigma}'|\vec{\sigma}) log_2 \frac{Prob(\vec{\sigma}'|\vec{\sigma})}{Prob(\vec{\sigma})}$$

and a function called the *rate distortion function*

$$R(D) = 1/n \min_{P(\vec{\sigma}'|\vec{\sigma})} I(\vec{\sigma}, \vec{\sigma}')$$

Where D is a variable indicating the threshold of "accepted" average distortion, and the minimum is taken over all distributions $P(\vec{\sigma}'|\vec{\sigma})$ for which $\bar{d} \leq D$.

The *data processing theorem* then states that $R(D)$ bounds the required channel capacity (bits per source word) necessary to achieve a compression with $\bar{d} \leq D$. (Higher compression is impossible and there are coding schemes which approach this bound arbitrarily close).

Common exercise with this approach are done with the distortion measure, set to the mean square error (MSE) criteria. This gives tractable expressions which can be solved analytically. In real situations (especially where the end consumer is the human visual system) such a measure is an approximation at best as there is absolutely no experimental evidence for it. (A "max" norm and a "worst case" rather then average expected distortion better capture the human reaction to loss of information, but using these results in expressions that are difficult to analyze and hence give little insight).

Information theory further extends to continuous (analog) signals as well, with results of similar spirit. There, it is common, for reasons of mathematical tractability, to model the source as a stationary Gaussian process. While this model has absolutely no support in real situations, we remark that it is nevertheless useful as the rate distortion function in that case bounds the rate distortion function for any other distribution, and hence serves as a "worst case" analysis.

2.3 Huffman Codes and Their Variations

Assume a source producing words which are independent random variables all with the same known distribution. In this case optimal compression can be obtain by the well known *Huffman coding* method.

2.3.1 The basic method

To understand the method, observe that prefix codes with k words have a one to one mapping with binary trees containing exactly k leaves. An optimal code then corresponds to a minimum weight binary tree with k leaves. The algorithm is started by ordering the source words according to their probability. At each step, the two lowest-probability words are added to form a single new probability which is inserted in the order. After k steps we are left with one probability value (1.0) which is the root of the tree. To get the actual code-words and mapping, we unfold the process as shown in fig 2.1.

It is easy to show that the code thus produced (called *the Huffman code*) is optimal, in the sense that no prefix code exist which gives a lower value to the expected average word length.

$$\sum_{\sigma} |C(\sigma)|P(\sigma)$$

Also it is easy to show that if the probabilities are all rationals with denominator a power of 2 the Huffman code achieves the entropy of the source and is thus optimal in the absolute sense.

Why is this not so when the probabilities are arbitrary? The reason is that for codes which map a source word to a code word, we have an additional "requirement" that each source word is mapped to an integral number of bits.

2.3.2 Block coding

This restriction can be removed if we consider *Huffman block coding*. The idea is to assign different code words not to individual source words but rather to blocks of, say, k source words grouped together. For each of the 2^{8k} possible blocks we then assign a code word according to the Huffman algorithm. Note that we still retain the independence assumption between words that is

$$Prob(\vec{\sigma}) = Prob(\sigma_1)Prob(\sigma_2)\ldots Prob(\sigma_k)$$

It can be shown that as k - the block size, approaches infinity, the average coded signal length approaches the source entropy. Hence from the information theoretic point of view this procedure is optimal. However from the practical point of view, as k is increased the drawbacks of the approach increase until they outweigh the increasingly marginal gain. These drawbacks include:

- Increasingly longer code words.
- Exponentially increasing size of the code (i.e. the mapping "table").
- Increasing lag between input and output (as k words have to be seen before the first output bit can be produced).

In practice block coding is used but with low values of k and hence with less than optimal compression rates.

2.3.3 Higher order distributions and conditional coding

Sometimes we can obtain higher order probability distributions for our source. That is, we know not just the probability of individual source words, but rather the probability of sequences of m words. As this gives us better, (i.e. lower), estimations of the source entropy, such higher order probability distributions can be utilized to construct better codes.

A natural way to do this is via the block coding method described above, however as noted this method has serious drawbacks. A better method often used is called *conditional Huffman coding*. The idea is that having sent the coded symbols for $\sigma_1 \ldots \sigma_{m-1}$, the probability of the next symbol can be estimated both at the encoder and the decoder as $Prob(\sigma_m|\sigma_1 \ldots \sigma_{m-1})$. If we construct a special (unblocked) Huffman code for this case (that $\sigma_1 \ldots \sigma_{m-1}$ were the last symbols seen) we can get full benefit of our statistics while avoiding some of the problems of block coding.

The table size will still be large, as while each code is only of size 2^8 we must now maintain $2^{8(m-1)}$ such codes. However the code words can be kept shorter and the lag between input and output is at most a single source word.

We can further show that m–conditional coding makes better use of our statistics than m–block coding: When coding blocks of size m - the i'th block does not make use of the information provided by symbols from the $i-1$'th block, unlike the case

in conditional coding. So although block coding also in the limit approaches the entropy bound, conditional coding gives better results.

In practice conditional coding is usually preferred to block coding as the means of utilizing higher order statistics. However sometimes hybrid - *conditional block codings* are used as well.

A remaining problem is the large code (table) size. Two approaches can be taken to ease this problem. The first is to ignore "too higher" order probabilities. That is to assume each symbol depends only on a small number of predecessors, (in the limit a single predecessor). While this approach might seem unwise, it is often quite consistent with the behavior of real data (more on this is section 2.7 where we discuss statistical models of images).

The second approach is through what is known as *state space conditional coding*. The idea is to divide the set of $2^{8(m-1)}$ possible predecessors into J groups such that the conditional probability of the next symbol is roughly the same for all members of a group. Then one needs only J specific codes as opposed to $2^{8(m-1)}$ codes. Again this approach is useful since real data displays behavior consistent with the underlying assumption that many sequences of predecessors have roughly the same conditional probabilities.

We terminate our discussion of Huffman codes with two practical remarks: The first is that sometimes we would like the code to contain only "short" words. In this case sacrifice optimal mapping by giving "lower probability" source-words "shorter" code words. The second is that when we decode (Huffman codes or any other variable length code), an error in even one bit of input to the decoder can result in a significant error in the resulting output image. To reduce this danger often *synchronization* words are inserted into the code.

2.4 Arithmetic coding

A newer method than Huffman coding, is *arithmetic coding*. The main objective of this method is to by-pass the problems associated with Huffman block coding while retaining the advantages of that approach. As a general purpose method arithmetic coding is lately gaining in acceptance and is already on par with Huffman coding.

The idea is to view signals as subsegments of the real segment $[0,1)$. Suppose, for example, that a source produces 2-bit words according to the distribution:

$$Prob(00) = 0.4 \quad Prob(01) = 0.3 \quad Prob(10) = 0.2 \quad Prob(11) = 0.1$$

We can divide the real segment $[0,1)$ into four subsegments according to these probabilities, such that all signals starting with the word 00 correspond to subsegments of $[0, 0.4)$, all signals starting with 01 correspond to subsegments of $[0.4, 0.7)$ etc. Applying this idea recursively, we get that each finite sequence of words corresponds to a subsegment of $[0,1)$. For example, the sequence 001101 corresponds to the segment $[0.376, 0.388)$. If our source is such that no signal is the prefix of another (signals are either infinite, all of fixed length or terminate with a unique "end of

signal" symbol), than this correspondence is unique and can be used as a basis for encoding.

The encoding proceeds as follows: as the encoder sees symbols of the message it computes the "current" lower and upper bounds of the segment. As soon as both lower and upper bounds have the same significant bit, it is transmitted and the segment is scaled by 2. This procedure is repeated until the message terminates and then a special signal plus sufficient bits from either lower or upper bound are sent to allow unique decoding. Decoding proceeds in the opposite direction - as bits are received, lower and upper bounds are updated until a unique source symbol corresponds to the entire range. It is then produced and the segment is scaled. The operation proceeds until the end of transmission signal is encountered (or sufficient bits are read). Details of this method (together with a "C" implementation) can be found among other places in

I.H.Witten, R.M.Neal and J.G.Cleary - "Arithmetic coding for data compression" *Communications of the ACM*, 30(6):520-540, 1987.

As with Huffman coding, we can utilize probability distributions of higher order via *conditional arithmetic coding*, that is at each step the probability (and hence the subsegment) assigned to each symbol depends on those symbols already transmitted.

It can be shown that arithmetic coding has the following advantages:

- It approaches (without blocking) the source entropy as close as the statistics available allow.
- The delay between input and output is small (except for extremely unlucky cases).
- There is no need to maintain a large table of code words.

We should note, however, that unlike Huffman coding where all the "thinking" is done "off-line" and given the code-table, encoding can be done with very simple hardware, arithmetic coding requires the ability to do arithmetic, hence it is used only in applications where such processing is available.

2.5 Predictive Coding

Predictive coding is not a method of coding, but rather a method to enhance other coding schemes. We list it separately because of its importance and wide use.

Suppose that we have a model of the behavior of the source, (typically, represented in statistical terms, but any algorithmic model will do). Suppose now that we have already transmitted some words $\sigma_1 \ldots \sigma_k$. As our model is algorithmic, we can generate $\tilde{\sigma}_{k+1}$ - a guess or prediction of σ_{k+1}. If our model is good, then most of the time the difference $\Delta\sigma = \tilde{\sigma}_{k+1} - \sigma_{k+1}$ will be small. Since $\tilde{\sigma}_{k+1}$ can be produced both at the the encoder and the decoder, predictive coding suggests to transmit only an encoding of $\Delta\sigma$. (see fig. 2.3)

What do we gain by this? From an information theoretic point of view - nothing, as no information is gained or lost via this scheme. From the practical side we usually gain much, for the following reasons:

1. The statistical dependency between successive differentials is usually much lower than between successive actual values (in optimal predictors the successive differentials are statistically independent). This means that for state space conditional coding we can do with a small set of states, and in some cases even a single state (i.e. no conditional coding) can be used without degrading performance.

2. The differentials themselves are usually small and only very rarely large. Hence the entropy of $\Delta\sigma$ is low and we can make the most benefit from a variable length encoding.

O'course one could argue that the second benefit could be reaped directly via better variable encoding, however in practice we never have full information about the probability of the source, and often the statistical behavior of $\Delta\sigma$ is much more explicit than that of the absolute signal (see section 2.7).

Examples of predictors based on a statistical model of the source are:

- *Maximum likelihood* - Choosing for pixel i the value x which maximized the expression $P(x|x_1 \ldots x_{i-1})$. Where $x_1 \ldots x_{i-1}$ are the values of all previously transmitted pixels.

- *Conditional mean* or *minimal mean square error*

$$x = \sum_{\forall x} xP(x|x_1 \ldots x_{i-1})$$

- *Linear predictors* - $x = \sum_{j=1}^{i-1} \alpha_j x_j$ where the α's are chosen according to the criteria of the model.

2.6 Transform Coding

Another approach which is not a method to itself, but rather a method to enhance other coding schemes is transform coding. Assume we have some reversible transformation T such that

$$T^{-1}(T(S(t))) = S(t)$$

Given a signal $S(t)$ we can send $T(S(t))$ (actually $C(T(S(t)))$ instead. As with predictive coding, from the information theoretic point of view nothing is gained or lost by transform coding, however in practice three advantages are obtained:

1. A transform can decorrelate the data, hence there is no need for block coding or predictors with "long memory".

2. A transform can "concentrate the energy" of an image, that is by transmitting only the first k coefficients of the transform we still obtain a good approximation of the image in the mean square error sense.

3. A transform can "model" the human visual system, in a sense that we are more sensitive to some of its coefficients than to others. In this case we can ignore some of the coefficients and loose only information that for us is redundant.

The first two advantages are best obtained by the Karhunen–Loeve transform and the last by the Fourier transform. Before we detail specific transforms, we note some general aspects which are desirable.

- When using transforms the explicit 2-D (3-D) representation is used (as it makes explicit the correlations in the image.

- For reasons of mathematical tractability, up to date, only linear transforms have been used (i.e. $T(S) = \mathbf{T}S$ with $\mathbf{T}$ some matrix), and in particular orthonormal or unitary transforms (where $\mathbf{T}^{-1} = \mathbf{T}'$).

- Normal matrix multiplication requires somewhere between $O(n^3)$ to $O(n^{2.5})$ operations depending on the type of algorithm used. This complexity is often too high for practical purposes, hence *fast* transforms are preferred. *Fast transforms* such as the *fft - fast Fourier transform* can be typically performed in $O(n^2 \log(n))$ operations. Such algorithms are possible if the transform matrix is "sparse" in some sense.

- To do a transformation, one must have access to the entire image before processing can begin. If the implied storage requirements and time lags between input and output cannot be met, a compromise can be obtained by breaking the image into smaller "blocks", each of which is transformed separately.

2.6.1 Karhunen Loeve (Hoteling) Transform

Assume that for a given source of images, we have up to second order statistics - that is the mean values and autocorrelation matrices. Let C be the "normalized to zero mean" autocorrelation matrix. We can then use the eigenvectors of $\mathbf{C}$ (ordered by their respective eigenvalues) as the basis for a transform. Such a transform is known as the *Karhunen Loeve* (KL) or *Hoteling* Transform. It can be shown that this transform is optimal in the following sense:

- The coefficients of the transformed signal are uncorrelated.

- The transform achieve maximal concentration of the energy of the image: For every number k, restoring the image from the first (of largest eigenvectors) k coefficients of this transform is a better approximation of the image in the mean square error sense, than the restoration using any k coefficients of any other transform.

In general this approach can be computationally intensive, and it might be difficult to "collect" the required statistics. However several simplifications are possible - usually we assume that the autocorrelation matrix is stationary. Sometimes the image is broken into blocks (say of 16×16 pixels and the statistics are collected on these blocks). At other times a simple statistical model is assumed, in which case the transform has a "fast" version.

2.6.2 Fourier and related transforms

Other commonly used transforms are all "variations' of the well known *discrete Fourier transform*. This transform is described by a matrix such that $t_{i,j}$ the element in position i, j is given by

$$t_{i,j} = \frac{1}{\sqrt{n}} exp\left[-\frac{2\pi}{n}\sqrt{-1}(i-1)(j-1)\right]$$

As mentioned before the Fourier transform suffers from the so called *Gibbs phenomena* and hence is often replaced by the *discrete cosine transform*

$$t_{i,j} = \frac{1}{\sqrt{2n}} cos\left[\frac{\pi}{n}(i-1/2)(j-1)\right]$$

which has the added advantage that all coefficients are real hence the fast cosine algorithm is faster than the fft.

Sometimes the *Walsh–Hadamard* transform is used. The matrix of size $n \times n$ for this transform is defined recursively as:

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \mathbf{H}_{2n} = \begin{bmatrix} \mathbf{H}_n & \mathbf{H}_n \\ \mathbf{H}_n & -\mathbf{H}_n \end{bmatrix}$$

The advantage of this transform is that no multiplication operations are necessary. While today such savings decrease in importance sometimes they are still significant.

2.7 Obtaining the statistics and Universal Coding

A common requirement of all the methods we surveyed so far is that we have sufficient information on the statistical behavior of the images we transmit. Naively this is a simple task - one observes the behavior of the source over time or that of simpler sources, and uses the results to produce statistics - probability of grey levels, the autocorrelation matrix etc.

However if one wants good and reliable statistics this problem is extremely difficult if not impossible, since in some sense any source is unique so what we know about one source may not apply to another seemingly similar source, and that sources are rarely completely stationary (that is behavior changes over time/space) and hence it is difficult to estimate behavior on passed events.

However as approximations, the assumption that our sources are stationary and similar to others often hold to various extents, and then the problem of obtaining

statistics becomes a tradeoff between how much information we deduce from past events and how wide is the field of applicability of this information. As an example consider English text. If we consider only the so called *first order distribution* (the distribution of individual letters). Many sufficiently long English texts could be used to give a good approximation of the distribution for any other English text. On the other hand when we go to higher (say 5 or 6'th order distributions (the probability of sequences of 5-6 letters), we get much more accurate information, but the applicability of the results of a specific text is limited only to other texts of the "same kind" (even technical writings in related but different fields already display different 5-6'th order distributions).

We distinguish between *off-line* methods where statistics are obtained before encoding is started and on-line methods where encoding and statistics collection are done simultaneously.

2.8 Off-line Methods

A naive approach to obtaining statistics on images is to start with the assumption that images almost always display "something" i.e. 2-D representation of real world objects. Regretfully this assumption does not provide us with any concrete statistics, but it does give us some basic rules of thumb which are quit useful: As objects are continuous, pixels which are close to each other spatially, belong with a high probability to the same object and hence with high probability display the same *texture*. If this texture is simple, as is usually the case, this statement translates to the statement that close pixels will display close grey values. If textures are not simple than we still have hopes of determining the "period" of the texture and thus get good approximations. As the distance between pixels increases the correlation is expected to drop sharply. For example, assuming that the probability of a pixel being and *edge* (on a boundary between objects) is p then the probability that pixels of distance Δx are uncorrelated is $1-(1-p)^{\Delta x}$.

Under the assumption that neighboring pixels belong to the same object, we can develop more sophisticated estimators of a pixel's value as a function of its "past" neighbors. Typically only close neighbors are considered (2, 4 and at most 12), and common models include linear approximations and various regression type methods. A point to mention is that often the correlation displayed in the vertical direction is higher than in the horizontal. This is usually due to the aspect ratio of the pixels in most images which is 3:4 hence the real distance between neighbor is less in the vertical direction.

One such model is the Laplacian model

$$p(x) = \frac{\lambda}{2} e^{-\lambda|x|}$$

where x represents the difference of the grey level of a pixel from its predecessor in the 1-D case or from the average of its neighbors in the 2-D case (typically only 2 or 4 neighbors are taken). This model is in good agreement with many experimental

results, and hence is often used for predictive coding (or predictive + transform coding).

This model is also often used for predictive coding of transform coefficients (if we divide an image into 16×16 blocks, and perform a transform on each block, typically the distribution of the i'th–coefficient in the various blocks will have a Laplacian distribution.

Also it was found that a Laplacian distribution is a good approximation of the stationary autocorrelation matrix of images

$$R(x, y, x', y') \propto e^{-\lambda_1 |x-x'| - \lambda_2 |y-y'|}$$

Hence this model (which is also called a *wide sense Markov process* is often used for the Karumem–Loeve transform, with the additional advantage that in this case the KL transform has a *fast* version. However it was found that under this model (and in general any model where the correlation between pixels decreases quickly with the distance) other commonly used transforms and specifically the cosine transform are very good approximations of the KL transform.

For the purpose of predictive coding, note that this model has a free variable in it. To use it as a predictor, the encoder and decoder must first agree on its value. This can be done either by setting it statically, or as is more common, by sending it as part of the transmission. In this case the first step in encoding would be to estimate the λ which best approximates the data. Then send this λ and then proceed with usual predictive coding. If the estimation power of the model is good, the added cost of sending λ is more than offset by the saving in the image itself. (Note that an adaptive version can be used where λ is updated every so many pixels).

A special case of "parameter" predictive coding is in movies or video conferencing compression schemes: In movies, we can usually assume that much of the change between the two consecutive frames comes from the motion of the camera, hence we can attempt to match the two frames in terms of this motion (usually this involves 2 translation parameters, a rotation parameter and a zoom parameter). These parameters are sent prior to the image, and then the differences between the actual frame and the camera-motion compensation of its predecessor is sent (it is possible that a special parameter be used to indicate a total change - such as when a scene is "cut" and another is started to avoid the sending of an execesive number of now useless parameters). Also note that for estimation the value of a pixel in the next frame, the pixels corresponding to its neighborhood in the previous frame should be used (to compensate for sub-pixel differences and small motions.

In video conferencing a similar approach is currently being tested. There it is not the camera that moves, but rather the frame is segmented to "person" and "background" and the motion of the person relative to the background is estimated. (in sophisticated such schemes, it is possible that the entire background is stored in the predictor even though it is never completely seen).

2.9 On-line methods, universal coding

The most well known method of "statistically' encoding a signal of unknown statistic is the *Lempel Ziv Welch* universal coding scheme. This scheme keeps a table of say 2^J entries. Initially 2^k of these entries are set to all possible strings of length k and the rest are empty. The encoder reads bits from the input until it reads a bit s_{m+1} such that the sequences$_0, \ldots s_m, s_{m+1}$ is not in the table but the subsequence $s_0, \ldots s_m$ is. In this case, the position of the entry for $s_0, \ldots s_m$ is transmitted and if the table is not empty $s_0, \ldots s_m, s_{m+1}$ is inserted as a new entry. It can be shown that if the signal is stationary, by making the table large enough optimal compression eventually results.

There is a large number of variations on this approach. The point we would like to mention here is the problem of having the table "clogged" with entries that are irrelevant or suboptimal. This can happen either if the table is too small or if the source is not truly stationary but rather slowly changing with time (or space) - as is often the case. When this occurs the proper thing to do is to discard either some (the least used) or all of the entries in the table and start afresh. An indication of such a situation could be that the "average compression" drops below some acceptable level.

A simple version of this "adaptive' table resetting is a method of obtaining statistics which we shall call *adaptive statistic measuring*. In this case a "window" of say k symbols is used as the "probability space" and for the i'th symbol we estimate any statistics we believe significant, according to the previous k symbols we saw. In the simplest case, which often gives very good results, we approximate the probability distribution of the i'th letter via the frequency of letters in the last k symbols. Such methods have the advantage that they are simple and quickly adapt to change. They are better than universal coding methods if we believe our source to be either of a simple statistical behavior or to be changing over time.

3 The Human Visual System

It seems almost obvious that any image compression scheme where the "end-client" is human, could benefit from an understanding of the human visual system (HVS), this either via more accurate distortion measures or by allowing us to ignore "components" of the image which are redundant or unimportant to the HVS. Regretfully, our current understanding of this complicated "machinery", which we all possess, is such that for the most part, it is difficult to apply it directly to image compression. Hence in this brief exposition we relate only rarely to image compression. Still even this limited knowledge is useful as it improves our intuition about the HVS and guards us against more obvious mistakes.

Our presentation starts with physiological evidence. Then we describe some psychophysical results and then we devote a section to color perception. We end with a description of current mathematical models used for image distortion. In this section we rely (apart from Natraveli's book) on

Sensation & Perception
Stanly Coren and Lawrence M. Ward
University of British Columbia
Harcourt Brace Jovanovich, Publishers 1989.

and

Vision in Man and Machine
Martin D. Levine
McGill University
McGraw Hill 1985.

We assume some basic familiarity with the subject matter and omit details where possible.

3.1 Results Obtained for Physiology

3.1.1 A single nerve cell (neuron)

We start with the behavior of a single nerve cell or *neuron*. This is important as neurons are the building blocks of all more complicated nerve structures, and as these more complicated structures often display properties already present in the single neuron.

A neuron is composed of *dendrites*, a *body* and an *axon*. In general dendrites receive input and the axon produces the output (however there are exceptions). The connection between two neurons is called a *synapse*. Intuitively one can think of transmission between cells as chemical while transmission within a cell is either electro-chemical or electrical in nature.

A live neuron maintains a constant voltage difference (called *resting potential)* of apprx. -70mv between its inside and outside. External stimuli (from sensor-cells or other neurons) can cause this potential to change. Such a change is called *hyperpolarization* if the inside of the cell becomes more negative or *depolarization* if the inside becomes less negative or even positive. Hyperpolarization is always *graded* - that is the greater the excitory stimulus the greater the hyperpolarization. Up to a certain threshold level of stimulus depolarization is also *graded*. Beyond this it becomes a *spike* or an *action potential* - that is any stimulus beyond a given threshold will produce the same dramatic depolarization effect of apprx. +40mv. After such a spike is produced the neuron hyperpolarizes back to -70mv. and needs some time (a few msec.) before it can transmit another spike (this is known as the *refractory period*).

When stimuli are higher than the threshold for the action potential, the fact that a stimulus is stronger manifests itself in the *frequency* of the action potentials (spikes). Thus a stronger stimulus leads to more frequent action potentials (up to a limit of 1000 spikes/sec as determined by the refractory period). However most cells display "fatigue" effects in that the level of stimulus necessary to create an action potential (or maintain a given frequency of action potentials) increases with

the time the neuron has been active. It should be mentioned that even without stimuli most neurons shoot action potentials at some "rest frequency".

For neurons which serve as communication systems (are far from sensors or effectors) action potentials are the main means of operation. For such cells transmission along the axon is electrical and can reach peak speeds of 3×10^8 m/sec. (As other components of the cell are slower the average speed through such a cell is up to 120m/sec).

When an action potential reaches a synapse it causes the release of a transmitter substance which can be detected by a neighboring cell. such substances can be excitory or inhibitory depending on the type of synapse. Over short periods of time one can think of the cell a performing some integration over time of the (positive and negative) stimuli it receives.

From a computational point of view we can summarize the above as follows:

- Neurons code stimuli either as "amplitude" or "frequency".
- In both "codings" a neuron can have "positive" and "negative" values.
- The effect of a spike from one neuron on its neighbor (via a synapse) is either excitory or inhibitory.
- Using synapse effects and the time integration of inputs, neurons can behave as *logical gates* (though these *gates* are boolean or tri-state only in an approximated sense).
- Neurons are more tuned to change than to absolute levels of stimuli. (Over extended active periods higher levels of stimulus are necessary to maintain constant levels of activity). There are exceptions to this rule.

3.1.2 The Eye

The structure of the eye is given in figure 1. Its optical part is composed of the *cornea*, the *pupil*, the *lens*, the *vitreous humor* and finally the *retina*. The cornea serves as a first lens, the pupil as a shutter and the *lens* is a lens whose focal point can be modified via the *ciliary muscles*. The *vitreous humor* is there for mechanical reasons (to maintain the shape of the eye) but does interfere in the viewing process when it is not clear. Also with age it becomes yellow and hence color sensation is changed with age though this is usually unnoticed by the person himself. We mention that in signal processing theory the effect of the optics of the eye is modeled by low-pass filter.

In the retina light signals are transformed into nerve signals. The structure of the retina is given in figure 2. It is composed of three layers of cells:

1. *photoreceptors* which transform the light signal to a nerve signal. There are two types of photoreceptors: *rods* and *cones*. Rods are responsible for night

(*scotopic*) vision and are of a single type. Cones are responsible for day (*photopic*) vision and come in three types (known as blue, green and red) which have peak sensitivity at different wavelengths.

2. The next level contains three types of cells: *bipolar cells*, *horizontal cells* and *amacrine cells*. These serve as mediaters. The main function of bipolar cells is to collect information from a number of receptors and compress it into a single signal. The function of the other types of cells here is not clear but probably has to do with *lateral inhibition* and similar effects (see ahead).

3. *Ganglion cells* these come in three types (called X, Y and W), each is connected to a few bipolar cells and performs some rudimentary computations. They transmit the visual information from the retina to the brain.

To give an impression of the quantities involved and the compression that takes place in the bipolar cells, there are apprx. 120 million rods and 5 million cones in the human eye, but only about 1 million axons in each optical nerve.

The distribution of photoreceptors and the compression of information via bipolar cells is not even throughout the retina. The distribution of rods and cones is given in figure 3. As can be seen, directly in front of the pupil (at visual angle 0) there is a special region called the *fovea*. In this region there are no rods and the concentration of cones is highest (actually only red and green cones). Furthermore the structure of the retina in this region is different (see figure 4): The cones are narrower and more elongated and the cells of the other levels are moved aside so that they do not interfere with the incoming light. furthermore in this region each photoreceptor is connected to a single bipolar cell.

Outside the fovea the concentration of cones quickly drops to a low constant. The concentration of rods peaks just outside the fovea and slowly drops to the periphery.

As can be expected it can be shown that our resolving power (as well as the ability to notice color) peaks in the fovea. We can therefore regard as a first (and rough) approximation the image on the fovea as the thing we look at and the rest of the retina as a mechanism to draw our attention to unexpected events. This naive model is however no more than an approximation as it is possible to attend to parts of the image which fall outside of the fovea.

On the other hand it is commonly believed that at illumination levels that allow cones to function, the scotopic system is indeed ignored (or serves only to notify us of unexpected events). It is only at low illumination levels that it "takes over".

The behavior of photoreceptors is simple in that the intensity of the response of a cell is directly related to the intensity of light that falls on it (plus fatigue effects as mentioned above). As we move to the ganglion cells, behavior becomes more complex. Both X and Y type ganglion cells have receptive fields which are known as *center-surround*, that is a field composed of two concentric circles such that light falling in the smaller circle cause one type of behavior and light falling on the difference between the circles causes the opposite behavior. These can be *on-off*, that is light on the center causes a more active response and on the surround a less

active response or *off-on*. The size of the receptive fields differs for different cells and is larger for Y cells. W cells display more complex behavior, some are center surround but others are sensitive to certain types of movements.

Later when we shall speak about the "independent channels model" we shall see that the difference in receptive field size, leads researchers to associate the X cells with one channel (or group of high-pass channels) and the Y cells with another (or group of low-pass channels). Such theorizations, while sometimes fruitful, can often be misleading as the difference in behavior is more complex - for example Y cells respond vigorously to change while X cells maintain relatively constant levels of response for a sustained signal.

3.1.3 Two paths to the brain

From the retina the nerve fibers collect (via the *blind spot* - so called because there are no receptors there) to form the optical nerve which transmits information to the brain. The first junction on the way is the *optic chiasm* where information from the two eyes meets and gets reshuffled so that the "left" channel has information about the left part of the visual field (and not what the left eye sees) and the right part about the right side of the visual field. From here the each path separates into two channels: one going to the *lateral geniculate nucleus* (LGN) and from there to areas 17,18,19 (or V1,V2,V3) in the brain, the other going to the *Tectum* and from there to the *pulvinar nucleus* and the *Lateral posterior nucleus* and then to areas 18,19 in the brain (see figures 5,6,8).

It is not clear what exactly happens in these two paths or at each junction along each of them. However based on experience with injuries it is believed that the LGN–area 17 system is involved in fine perception of patterns and color and the tecto–pulvinar system coordinates localization of objects in space, guidance of eye movements and gross pattern perception.

The structure of the visual path in the LGN was however studied, and it was found that cells there respond basically like the X or Y cells to which they are connected. There are however two points that should be mentioned: One is that the LGN is divided into six layer each of which contains an ordered *map* of the entire visual field (that is each point in our visual field corresponds to a point in such a map and proximity in the visual field implies physical proximity of the corresponding cells in the map). The second is that cells here display a high level of *rest–activity*. This can serve both as a means of maintaining continuity in what we see and as a means of allowing "negative" responses.

3.1.4 In the brain

The part of the HVS most studied in the brain is the primary visual cortex (area 17). This research started in the early 60's in the works of Hubel and Wisel and is still continuing to this day. In this area several types of interesting cells where found other than the center-surround type of lower areas. The first is, so called, *simple cells*

which respond to long bars of given width and orientation. Other cells, which where termed *complex cells*, respond to a long bar oriented in a specific way and moving in a specific direction. These cell are also to a large extent translation independent in the sense that a properly moving bar triggers a high response regardless of its position within the receptive field of the cell (which is typically larger than the receptive field of a simple cell). A third type of cells, with even more complex behavior, the so called *hypercomplex cells*, was also found. These respond only to bars of a given width, orientation and length.

The general topography of area 17 is such that detectors are ordered into three dimensional *hyper columns* according to 2-D location in the visual field and orientation (at roughly 10 deg. intervals) see figure 7.

In concurrence with computational models for machine vision, it was conjectured that ganglion cells serve as *point detectors*, simple cells as *edge detectors* or *curve detectors*, complex cells as *motion detectors* and hypercomplex cells where conjectured to be *corner detectors*. However such conjectures are always dangerous and fragile as for ganglion cells an alternative explanation is as band pass filters, and recently Dobbin, Zucker and Cynader showed that experimental data is compatible with hyper-complex cells serving as curvature detectors (an operator which they reasoned must be present to allow proficiency in certain detection tasks).

There where some investigations into higher levels of the visual path, but as yet there is no conclusive and undisputed results. There are indications that pattern recognition by sight takes place in the temporal cortex. These come from experiments either with injured patients, or from animals where part of these regions where removed. Recently more direct evidence was found by neuron mapping techniques, where some researchers reported cells which showed peak response for "hand" or "face" resembling patterns.

In parallel with such results, computational models where suggested demonstrating how ever more sophisticated pattern–matching–cells could be built upon less sophisticated detectors via the basic behavior of neurons and synapses. The trend seems to be that as we progress up the (human or computational) visual path, we find cells responsive to shapes with a increasingly higher structure and the response becomes independent of lower level parameters (such as translation, orientation and scaling). By the time computational and physiological models have progressed to "hand detecting" and "face detecting" cells, both enthusiasts and cynics alike proposed the existence of *the mythical grandmother cell* which peaks only when your grandmother enters your visual field. Needless to say that as yet no evidence for the existence or inexistence of such high level cells was found. (On the more philosophical level the argument on whether your grandmother invokes a specific cell or a specific pattern of activity is as yet unresolved).

3.2 Results Obtained from Psychophysics

A complimentary approach to physiological or nerve mapping techniques is psychophysics. The HVS is regarded as a black box whose behavior is to be inferred

via its response to specific tightly controlled situations. As in physiology - while the results of each experiment are clear cut, the conclusions drawn are often open to argument. Hence current theories are considered plausible when several different (psychophysical or physiological) experiments support the same conclusion.

A primary and relatively simple aim of psychophysics is to measure the fidelity of the HVS. Given some parameter (say intensity), we can conjure some tests to measure both absolute fidelity - what is the minimum intensity to elicit a response - and relative fidelity - what is the minimum difference between intensities which can be detected. For example, we can show the observer increasingly weaker stimulus until he no longer detects them, then show him increasingly stronger stimuli until he does detect etc. Similarly when we give him a "reference stimulus" we can perform discrimination tests. Before we describe some specific results we note some general properties.

The first is the apparent inconsistency of such tests. The intensity (or level of the test parameter) where the observer switches from detection to non-detection (and vice versa) fluctuates quit a bit. This is true even when observers are judged to be "honest". There is a variety of explanations for this effect (observer fatigue, expectations, observer noise etc.). On the practical side this means that we can never expect a rigorous threshold of fidelity - but rather only some distribution function (the probability that a user will detect a stimulus of intensity x is $p(x)$).

The second is that when observers are asked to categorize stimuli on a linear scale (but without giving then the ability to compare different stimuli to each other), the number of categories will be small: 2-3 bit of information = 4-8 categories. This has a parallel in our language where on any one parameter dimension we have only very few adjectives (say long, short, medium, very long, very short).

When the observer is allowed to compare between stimuli, the ability to distinguish differences is quit high. However here it was consistently found that the fidelity is not linear, but rather changes in a manner known as *Weber's law* saying that ΔI the minimal difference which is detected is proportional to I the absolute intensity of the stimuli:

$$\Delta I = KI$$

Where K is constant over a large range of I for a specific task. Moreover this relation is broken only on the ends of the detectable scale (when I is close to the minimal or maximal levels our system can handle). (For brightness $K = 0.079$).

This has led to *Fechner's law* conjecturing that the intensity of the sensation we feel is related to the intensity of the stimulus via a log rule

$$\textit{Intensity of what we feel} = W \log(\textit{Intensity of stimulus})$$

A different such *law* which is more consistent with direct questioner testing is *Steven's law* or the *Power law*

$$\textit{Intensity of what we feel} = W(\textit{Intensity of stimulus})^n$$

With n for brightness ranging from 0.5 to 0.33 depending on target size.

3.2.1 Intensity

When one measures the fidelity of the human visual system one finds that the minimal intensity that can be detected is quit low. In fact it is conjectured that it could be as low as a single photon being sufficient to elicit a reaction of a photoreceptor (it is confirmed that apprx. 6 light quanta are sufficient). To achieve such fidelity we must allow for a long period of adaptation to darkness (in the order of 20 minute to half an hour).

Also there are different values depending on the color of the object (some colors appear brighter than others for the same source intensity), on the size of the light source (when light reaches more photoreceptors the chance of detection by the person is increased) and on whether the light level and adaptation level is such that the photopic or scotopic mechanism is in operation (coming into darkness from a well lit room the minimal intensity to be detected slowly increase, until something like 10 minutes when the scotopic system "takes over" and then there is a dramatic improvement for some more time until the maximal fidelity of the scotopic system is approached.

As the maximal level of intensity we can observe is also quite high (say a natural scene in a sun lit day). We get that the dynamic range of the HVS is several orders of magnitude. From an engineering (or computational) point of view one realizes that Weber's law or the power law is a simple means of obtaining good fidelity over such a wide range. Also this response is consistent with the reflectance properties of real world objects where the light they reflect is a proportion of the light they receive (hence to distinguish between two objects under different lighting conditions a Weber's law type of response is sufficient). We mention that video cameras and VCR's also respond according to a power law (though o'course the power parameter is different).

Other than Weber's law, when asking people to judge relative intensities of patches of grey, two nonintuitive and contradictory phenomena manifest themselves:

The first is known as *brightness contrast* (see figure 9) and it causes an increase in the perceived contrast and the second is known as *brightness assimilation* (see fig. 10) and it causes nearby stimuli to appear of similar brightness.

A common explanation of brightness contrast, which also has the effect of *edge enhancement*, is through the *lateral inhibition* mechanism which creates the center-surround receptive fields of ganglion cells.

The explanation for brightness assimilation is through cognitive processes of attention. The reasoning here is that brightness contrast occurs when we are attending to an object and assimilation occurs otherwise. In the above assimilation example we attend to the lines and hence the grey background is assimilated. It is argued (and experimentally demonstrated) that when people attend to the grey background the effect is reversed. To my opinion this is not entirely satisfactory.

A third effect which is related to brightness is *brightness consistency* which will be described ahead together with other *consistency* effects.

3.2.2 Acuity

Acuity experiments attempt to measure the maximal resolving power of the HVS. Typical tests measure *recognition acuity* - that is the ability to distinguish different letters or other patterns from each other. It was found that for normal observers acuity is up to a visual angle of 1 minute of arc. This is consistent with what we know about the size and density of single receptive fields in the fovea. For other tasks such as *Vernier acuity* - testing whether a line is straight or broken it was found that acuity can be as high as 5 seconds of an arc (this is called *hyper acuity*. This is somewhat surprising as this value is about 25 times smaller that the receptive field of a single photoreceptor, however the phenomena can be explained using a six channel model (see ahead).

To measure the acuity in terms of frequencies a set of experiments was performed using synosoidal or block line gratings. The observer was asked to control the brightness until he just detects the gratings. The results are plotted in fig. 11 (full graph). These results lead to the common model of the HVS as a band pass filter whose frequency response is given by this graph. To make the model complete we note that by rotating the angle of the grating pattern we can measure acuity at different angles. It was found that acuity peaks in both horizontal and vertical direction and is minimal at the 45^o where it drops to apprx. 70%.

A more sophisticated variation of this model assumes that there are a few such independent channels each acting as a band-pass filter and our reaction to a scene is determined by the combination of their outputs which "make the most sense". Typically 4 to 6 channels are proposed. The advantage of the multi channel model over a single channel is that is allows to computationally explain a variety of phenomena - such as improved edge detection (the Canny model), hyper acuity and certain fatigue effect of the frequency response: By asking the observer to attend to a grating pattern of specific frequency we can fatigue a specific channel. When we perform the regular grating experiments under such conditions, typical curves look like the doted graph of fig. 11. It is also consistent with experiments where the patterns where different sinusoidals superimposed on each other.

We remark that the (single or multi-)channel model is very appealing as it makes many engineering decisions in image processing computationally (and even analytically) tractable. Also it gives us a natural transform coding compression technique. On the physiological side, there is sufficient evidence to support both models (viewing the X and Y ganglion cells as different channels) or at least to make it plausible (in area 17 cells sensitive to bars of different width where found and these two can be used as a basis for a band-pass Fourier analyzer).

For the temporal response of the HVS similar graphs exist, suggesting that the HVS can be modeled as a bandpass filter in time as well. the typical "cut-off" frequencies are apprx. 30 Hz. in time and 30 cycles/degree. In the temporal domain - typically a single "channel" is assumed.

3.2.3 Consistency

We end this subsection with a mention of a *higher level* set of effects knows as *consistency effects*. For a large number of parameters in the HVS (such as size, shape, brightness and color) it was found that the perceived stimuli is not the actual stimuli but rather the stimuli that "should be there". Taking brightness as an example a paper appears white and of the same "shade" of white under completely different lighting conditions. In fact a peace of coal in sunlight appears black even though it emits more light than a white piece of paper in a moderately dark room which appears white. Similarly the shape of an object or its size do not change as it rotates or moves away from us (the way its image on the retina changes), but rather remains constant.

When viewing "natural scenes" this mechanism performs so perfectly that we are rarely aware of it. It becomes apparent only in special constrained situations when it is manifested as optical illusions, or when one tries to make a computer vision program "understand" what appears obvious to our eyes.

It is not clear how this mechanism actually operates, however the heavy dependence on context suggests that it is either a "high level interpretation" or a feedback mechanism between lower levels which detect "features" and higher levels which make sense of them (as such it may involve multi-level feedback thus explaining the fact that some of the effects are more related to the meaning of the scene than others).

The usefulness of such a mechanism for an organism which tries to make sense of its environment is quite obvious. The effect on image compression is less clear as current schemes rarely understand the context of a scene in any detail.

3.3 Color

3.3.1 Physiology and Psychophysics

As already mentioned, our ability to detect color stems from the three different types of cones in our retina. Their sensitivity graphs (plus that of rods) is given in fig. 13. According to their peak sensitivities the three types of cones are known as *blue* (peak at 420nm), *green* (534) and *red* (564). Because our color perception stems from three types of cones it is called *tristimulus*. One might wonder whether the different profile of the rods should not cause us to have a four-stimulus color vision, however it turns out that the scotopic (or rods) vision is "turned off" when the light level allows for photopic (cone) vision. Hence we always receive either 3-stimulus color vision (photopic) or 1-stimulus black-white vision (scotopic). A point to mention about rods and color is that the rods are insensitive to those wave-lengths which we call red. This has the interesting effect that red objects in the dark appear black and that observing objects in red light does not reduce our adaptivity to darkness (when exiting a lit room into darkness one is much better adapted if the light was red).

As a result of our tri-stimulus light detection, it is not surprising that (almost) any color can be reproduced by mixing different quantities of some 3 primary colors. We shall say more on this latter, but for the moment we remark that the tri-stimulus approach does not explain all of our color sensation. It turns out that immediately after the receptors (probably in the ganglion level) the colors we see get encoded into three different channels which are knows as the *brightness*, *green-red* and *yellow-blue* channels. These channels are formed by connecting several receptors into one ganglion cell (in the same way we get the center-surround visual fields). The brightness is simply the addition of all three channels. The red-green is formed by inhibitory and excitory synapses to red and green receptors and a blue-yellow is formed by say excitory blue and inhibitory red and green. Fig. 14 gives a graph of the probability of a light having a given name as a function of the wavelength of that light (notice how the yellow fills the gap between the red and green).

Beyond the ganglion cells color information is coded in the LGN and primary visual cortex in a way that some of the *layers* (remember that each layer corresponds to a hyper-column generating a visual map) are tuned to color while others are color blind. This leads to the currently acceptable model that color information is transformed to the brain via separate channels. However an alternative approach that color is coded via the frequency of action potentials has some support in that black over white patterns of given frequencies can elicit color sensation. Still this phenomena might have other explanations as well.

A last point to mention about colors is that the perceived brightness is different for different colors - that is light sources of the same intensity (in energy units) but of different colors will be perceived as of different brightness by a human observer - the peak is around the yellow.

3.3.2 Representing Colors

In this subsection we cover some of the more common methods to represent colors. Suppose we are given a light source with a spectral energy distribution $S(\lambda)$. By the three-stimulus model, the response of the HVS to it should be:

$$R_s = \int_\lambda S(\lambda) r(\lambda) d\lambda$$

$$G_s = \int_\lambda S(\lambda) g(\lambda) d\lambda$$

$$B_s = \int_\lambda S(\lambda) b(\lambda) d\lambda$$

where r, g, b are the sensitivity functions of fig. 13. As a consequence two light sources $S_1(\lambda)$ and $S_2(\lambda)$ will appear the same provided:

$$R_s = \int_\lambda S_1(\lambda) r(\lambda) d\lambda = \int_\lambda S_2(\lambda) r(\lambda) d\lambda$$

$$G_s = \int_\lambda S_1(\lambda)g(\lambda)d\lambda = \int_\lambda S_2(\lambda)g(\lambda)d\lambda$$
$$B_s = \int_\lambda S_1(\lambda)b(\lambda)d\lambda = \int_\lambda S_2(\lambda)b(\lambda)d\lambda$$

This prediction is confirmed by psychophysical experiments and allows us to represent a color as a vector of three values.

A natural approach to representing colors is then to specify the "red", "green" and "blue" components. The way this is done is as follows: first three colors should be chosen as our standard primaries. Such a standard was established by the CIE to be $R_0 = 700nm$, $G_0 = 546.1$ and $B_0 = 435.8nm$ (chosen so that equal amounts of the primaries are needed to give equal energy white). Then we "map" the color space by performing the following experiment: The observer is given a "test color" in one display and an adjustable color in the second display, with the adjustable color formed by controlling the intensity of light coming from the R, G and B sources.

As the relative ratios between the R, G, B values are preserved at different intensities our model is confirmed and the values of the intensities can be used as the specification of the color. (Also the values given by different observers with "normal" color vision are within the variance of usual threshold-type experiments). It turns out that for most "test" colors it is possible to find a tri-stimulus combination which matches them (that is the observer sees the same color in both displays). However for some colors this is impossible, but then a match can still be obtained if we allow for negative combinations (experimentally this is possible by adding one of the primaries to the "test" color until a match is obtained). Thus all visible colors can be represented.

In some cases it is convenient to separate the intensity from the representation of color. Then we the have three coordinate system

$$I = R + G + B \quad r = R/I \quad g = G/I$$

where r and g are referred to as *chromaticity coordinates*. (A third chromatic coordinate can be defined as $b = B/I$ however it is redundant as $b = 1 - r - g$).

A question can now be asked "how do our results depend on the choice of primaries?" It turns out that the relationships between the different values we get for different sets of primaries can be computed by a matrix multiplication, where the matrix is simply the tri-stimulus values of one set of primaries using the other set of primaries a as basis. From this relation it is easy to see that given some arbitrary three primaries - all colors which lie in the 2-D triangle formed by these primaries will be positive, while colors which lie outside this triangle will have (some) negative coordinates.

To avoid the problem of negative coordinates, the CIE proposed three *imaginary* primaries X, Y, Z which lie outside any visible color, and hence using them any color has positive coordinates. The values where chosen such that:

$$\begin{pmatrix} R_0 \\ G_0 \\ B_0 \end{pmatrix} = M \equiv \begin{pmatrix} 0.490X + 0.117Y + 0.000Z \\ 0.310X + 0.813Y + 0.010Z \\ 0.200X + 0.010Y + 0.990Z \end{pmatrix}$$

Hence for some color C

$$(R_C G_C B_C)\, M = (X_C Y_C Z_C)$$

Again, one can normalize the intensity to obtain a 2-D plot of all colors in the x,y chromatic coordinates as shown in figure 15.

Looking at this diagram we note that for each color-point which lies on the boundary of the visible part of the vector field we can associate a wave-length. For most colors this is a positive wave-length, however for the purples and browns lying on the straight line at the bottom, these are negative wavelengths obtained by passing a line from the point through a standard center point (named *illuminate C*). As this center point is perceived as white, this gives us another form of color representation known as the *brightness, saturation, hue* system. Intensity (brightness) is as before. *Hue* (a synonym for color) is the positive or negative wavelength of the color. Saturation which describes how far the color is from white (grey) is obtained by drawing a line through the color point (say S) and the center and labeling the intersection of this line with the boundary as D. The ratio CS/CD is the saturation. Fig. 12 describes the sensitivity of the HVS to color in this coordinate system. Note that as a scene becomes too dark or too light (over/under exposure) colors appear less saturated.

Some practical notes on representation: for cameras and screens it is convenient to represent a color in "their" coordinates - that is using primaries that reflect the properties of the scanning or light emitting components. On the other hand for transmission it is usually preferable to use a system of intensity and two chromaticity coordinates. This is both for historical reasons (to be compatible with B/W systems) and for reasons of compression, as in this system the bandwidth necessary for the chromatic coordinates is typically much lower than for the intensity, a fact we intuitively know, as B/W systems can be used to represent most information in a scene.

A peculiarity of cameras in this scheme is that often the absorption or detection profiles of the three "primaries" in the camera do not correspond to any real or imaginary point on the color diagram (as there is no reason to assume they are related to our cone profiles). In such a case it will be impossible to produce all colors in a true form and some compromise must be made. The typical compromise is to calibrate a transition matrix such that "critical" colors (for example human skin) are rendered faithfully.

3.4 Models for the Purpose of Measuring Distortion

We end this section by briefly describing two common computational models of the HVS (for exact details see pg. 292-297 in Natraveli). They can be thought of as a computationally biased summary of known results about the HVS. However though they are based on incomplete knowledge available to date, they already emphasize the problematicity of modeling the HVS, in that they are gross simplifications of even

what we currently know, yet they are already too complex for many applications where the distortion function of the HVS is desired.

The first model is a space domain model (fig 16). In this model both original image and distorted image pass through a nonlinear filter incorporating Weber's law (either a log function or a cubic root function as outlined in section 3.2). Then the images are subtracted and passed through a band–pass filter emphasizing the mid-range thus modeling our (one channel) response. The output is weighted according to an "activity measure" which based on the local rate of change of the image (the HVS is edge enhancing and tuned to change). The result is then passed through some summation mechanism. Typically a MSE criteria (due to its tractability, but notice that here it makes more sense than when raw MSE comparison is used). Other criteria used are the maximum error or more often a mean-max error criteria (the image is divided into regions, for each we select a maximal error and then calculate the MSE over these maxima).

The second model is based on frequency domain analysis and the multi channel hypothesis (fig 17). Here the HVS is modeled as a bank of band pass filters. Typically 4 filters of bandwidth $(1-3),(2-6),(4-12)$ and $(8-32)$ (cyc/deg.) are used for each orientation. Orientations are disceretised to 10^o in the horizontal and vertical and slightly less in the 45^o direction where sensitivity is lower. Thus giving a total bank of some 50-100 channels. (when color or temporal response are needed - more appropriate channels are added).

Both original and distorted images are passed through the non-linear filter as before and then the difference between then is passed to the bank of filters. The error as detected by the k'th filter is computed via the functional

$$r_k = \int\int \left[\frac{w_k(x,y)\cdot v_k(x,y)}{m(x,y)}\right]^6 dxdy$$

Where v_k is the response of the (k'th) band-pass filter. w_k weights the sensitivity of different receptors to different parts of the visual field and $m(x,y)$ is a masking function similar to the "activity measure" of the one channel model. The raising to the 6'th power is a heuristic means of emphasizing the larger errors in the integration (thus allowing to take an average rather than a max).

Upon the values of each r_k a randomized threshold decision is made (by adding random noise and then thresholding). The results are then "OR"ed. If the output is 0 - than the error is considered as unnoticeable by the human observer, else it is considered noticeable. Ideally we would like our model also to predict "how noticeable the error is", however the values used in the calibrations of this computational multi–channel model are all derived from experiments in "just noticeable stimuli" detection, hence we cannot assess how well the model predicts the response to supera threshold errors.

4 Common Compression Methods and Quantization Issues

We cover in some detail a few aspects of common compression methods. As most of the methods were already presented from a theoretical viewpoint in section 2. We concentrate on the more practical side, and specifically on the issue of quantization.

4.1 PCM - Pulse Code Modulation

The most basic of image compression techniques is *Pulse Code Modulation* which, in principal, is no more than A/D conversion. The analog image is appropriately prefiltered and then sampled at its Nyquist rate. Each sampled value is quantized and transmitted.

There are two "free" parameters in this method - the sampling rate and the quantization scheme. When possible, the sampling rate is chosen to be above the cutoff frequency of our visual system, else the allowed rate of transmission determines the sampling rate. For quantization - a number of approaches are available:

The simplest is uniform quantization - dividing the dynamic range of the input (from black to white) into 2^n equal segments mapping each segment to the grey value which is the mid range of that segment. Under this scheme we typically need 7-8 bits per pixel for moving scenes and 6-7 bit for still images. The reason for the difference is that quantization noise is more noticeable when it is "moving" than when it is static.

Typically, quantization noise (using too few quantization levels) is structured, and thus manifests itself as *false contours* in the image. Up to a limit this problem can be eased by *dithering* - that is adding high-frequency noise to the image before quantization, thus breaking the attention grabbing low frequency structure of the quantization noise (compare figures 1a,1b).

A more sophisticated quantization scheme would be to apply Weber's law. As our absolute fidelity decreases with increased illumination, it makes sense to use courser quantization levels for higher illumination levels. This leads to quantizers based on the logarithmic or power law. However, much of the advantage of such quantizers is offset, in most systems, by the *gamma* function of the CRT display. The relationship between voltage and emmited intensity in CRT's is also related via a power law with $\gamma \approx 3$ thus the Weber's law effect is canceled, and for CRT'S the added complexity does not justify the marginal improvement over uniform quantizers.

To transmit color images, we have to select both a color representation scheme and a quantization scheme. When using an RGB system it is possible to utilize the sensitivity of the HVS to the different colors and use slightly lower bit rates for the red and blue channels, and also to use slightly less bits per pixel for these channels.

Better results are obtained by representation schemes which use luminance and two chromatic coordinates. Then, most of the information is carried by the luminance channel. The question then remains, how to best quantize the chromatic

channels. As in the monochrome case, it is possible to use psychophysical results as the basis for a quantizer optimal in perceived error rather then absolute error. However psychophysical results indicate that the fidelity of the two chromatic coordinates are never completely independent of each other (nor of the luminance). We therefore have to either use "worse case" fidelity (providing sufficient bits to allow for the fidelity of each coordinate under its most favorable conditions) or use more sophisticated quantization methods which quantizes all three coordinates simultaneously:

This is done either by mapping the tri-stimulus values into a space which approximates a *uniform chromaticity space* - where the distance between any two colors which are just noticeable is the same, and then a uniform quantizer is applied in this space, or else first a perceptually uniform space is chosen and then a uniform quantization of this space is inverse-mapped into the original domain. In this case the actual quantization is done via a look-up table. We note that by proper quantization, chromatic coordinates require 10-20% of the bits required for luminance.

A third approach often used with synthetic objects or computer displays is color maps. The idea is that while the variety of colors we can discern is rather large, in a given image or frame, we can often make due with a limited set of colors. The quantization scheme then includes an off-line selection of the palette of colors to be used, and quantization itself proceeds by mapping each colored pixel in the image to its nearest neighbor in the palette. A typical example is common "color" (actually *pseudo-color*) computer screens. There a full RGB range of $256 \times 256 \times 256$ levels is mapped to only 256 colors. The reason in this case is to reduce images to 8 bits/pixel which allows fast communication between screen and computer. The techniques of selecting the palette of colors (and of quickly quantizing arbitrary vectors to them) are described ahead in subsection 4.4.

4.2 Predictive Coding

Predictive coding (also called *DPCM*) predicts from previous transmitted values an approximate value for the next (current) pixel, and then transmits only the difference between the actual value and the prediction. In designing a predictive coding scheme we make three decisions:

1. Selecting a predictor.
2. Selecting a quantizer for the difference values.
3. Assigning code words to the possible difference values.

Unlike PCM where a reasonable assumption is that different values are equally likely , the whole point of predictive coding is that, hopefully, the differences we transmit will usually be small and only rarely large. Hence the simple uniform or uniform-HVS-response quantization in step 2 and the constant length coding in step 3 no longer make sense. Instead more sophisticated quantizers are used (see ahead) and variable coding (Huffman or arithmetic) is often the choice for step 3.

As to predictors, these can be model-based, linear (trading accuracy for simplicity) etc. We shall not give exact details, but will remark on the distinction between adaptive and non-adaptive predictors. A *non-adaptive* predictor is a scheme which remains constant throughout the image compression process, while an *adaptive* predictor uses *side information* to select from a number of algorithms (or parameters) it can use. For example consider the configuration

$$\begin{matrix} A & B & C \\ D & X & \end{matrix}$$

A non-adaptive linear predictor could be

$$X = 7/16D + 1/16C + 5/16B + 3/16A$$

An adaptive predictor could be

$$X = ARGMIN_{i=A,B,C,D}|X - i|$$

- the value of the neighbor closest in value to X. Since the information as to which pixel is closest, is not available at the decoder, it is called *side information* and should be transmitted too. Note that this adds a burden of 2 bits per pixel above the number of bits necessary to encode the difference between the chosen pixel and X. Still if our images always display a high directional correlation between pixels (yet in varying directions) this adaptive predictor scheme could be overall better than non-adaptive linear prediction.

A point to note about predictive coding schemes is that unlike PCM schemes where transmission error in one bit effects one pixel, here the effect of a single transmission error may spread over the rest of the image. The ways to guard against this are either via error correcting codes, or synchronization pixels (where once in n pixels a PCM value is sent) another possibility is via *leaky predictors* where the predictor includes a constant component independent of transmitted data, which "dampens" any error effects.

4.2.1 Delta Modulation

A special type of DPCM coding which will introduce us to some of the problems of DPCM quantization is *Delta Modulation (D-M)*. The unique feature of this scheme is that the difference between predicted and actual value is quantized to a single bit (two levels).

Typically the predictor for delta modulation is just the value of the previous pixel (though more sophisticated predictors can be used), thus D-M has the advantage of being a very simple scheme producing outputs which are easy to handle and process.

When non-adaptive quantization is used, we select a certain value d as the *step size*. Then whenever we receive a 1 we set $x_i = x_{i-1} + d$ and when we receive a 0 we set $x_i = x_{i-1} - d$. The problem is to select an optimal value of d:

If d is too small the method becomes more lossy as we cannot "track" sharp edges: our encoding of x_i can be at most $x_i = x_{i-1} + d$, hence at sharp edges it might be a few pixels before the encoded and actual x_i values coincide. This problem, known as *edge overloading*, is visually manifest as a smearing of the edges. On the other hand, if d is too large "tracking" of constant grey levels becomes difficult, and a problem known as *granularity noise* become visually noticeable.

There are a two ways to attack this dilemma: the first is higher sampling rates. then the inter-sample rate-of-change decreases, thus finer d can be safely used. The second, is adaptive quantization: for example we could decide that 3 consistent steps (111 or 000) signal edge overloading and hence the step-size is to be doubled, while a fluctuating sequence (01 or 10) signals granularity noise and the step size is to be halved. We note that while the increasing sampling rate is robust, it causes a decrease in compression rates. On the other hand adaptive quantization retains the compression rate, but can never be fully satisfactory, and in certain cases might even be *unstable* in *Control Theory* terminology. This problem can be eased via control theory means, where the D-M signal is viewed as tracking the original signal, but we note that as our adaptive quantization scheme becomes more sophisticated, we loose the basic appeal of delta modulation - its simplicity.

We end the discussion of Delta modulation with a remark that while it is rarely a good scheme for image compression, it is successfully applied to other problems such as speech compression.

4.3 Quantization for DPCM

Suppose we wish to use n levels to quantize the DPCM value of a pixel. The problem of selecting a quantizer is then to select n representative level values $\{l_i\}$ and $n-1$ threshold values $\{t_i\}$. Quantization of a value x then proceeds by finding an i such that $t_{i-1} < x \leq t_i$ ($t_0 = -\infty$ and $t_n = +\infty$) and assigning $\tilde{x} = l_i$. If there is an i such that $l_i = 0$ the quantizer is called *mid-tread* and if there is an i such that $t_i = 0$ it is called *mid-riser*.

The degradations associated with DPCM quantization (when the predictor is taken to be the value of the previous pixel) are of three types:

- If step sizes are too rough, we get *false contours* or *granular noise* (depending on whether the quantizer is mid-tread or mid-riser) in slow changing areas.
- If step size is too fine we get *edge overloading* in fast changing areas.
- When the rate of change across an edge is gradual, a too fine quantizer also results in *edge business* - an effect where the edge appears discontinuous or jittery.

Given a probability function $p(x)$ on the difference values and a distortion function $d(e)$ for quantization error e. The optimal quantizer problem can be phrased

as finding the sets $\{t_i\}, \{l_i\}$ which minimize the expression

$$\sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} d(x - l_i) p(x) dx \tag{2}$$

For the special case where $d(e)$ is the MSE criteria the solution becomes the well known *Lloyd-Max quantizer* which is the solution to the set of equations

$$t_i = (l_i + l_{i+1})/2$$

$$l_i = \frac{\int_{t_{i-1}}^{t_i} x p(x) dx}{\int_{t_{i-1}}^{t_i} p(x) dx}$$

Since these coupled equations are often analytically intractable, iterative schemes are used, where in one step the t_i's are computed from the l_i's and in the next the (new) l_i's are computed from the t_i's. (See also section 4.4).

For DPCM quantization (as in other cases) MSE is a bad criteria for HVS response and yields suboptimal quantizers. Better results are obtained by using actual psychophysical data. Two approaches are possible: Performing psychophysical tests to determine a more appropriate distortion function, and then optimizing eq. 2 for this function, or alternatively determining experimentally a graph of prediction error verses "luminance difference which is just detected". Such a graph is then used as a constraint (keeping quantization error to be sub-threshold) for an optimization of either the number of quantization levels or their entropy. (Experiments indicate that the resulting quantizers are usually very similar when either parameter is optimized). Note that optimizing for the minimal number of levels has a simple algorithmic procedure by emitting rays at 45^o which intersect the graph or the abscissa.

Typical values (for simple 2-D predictors) would be 13-21 levels, the lower value being for typical head and shoulders images and the latter for *resolutions charts* - which are heavily detailed images (and hence bad candidates for this type of predictor). Chromatic components usually require fewer levels (5-9).

Further improvement is obtained by using adaptive quantization schemes. The basic idea is that at different levels of "edginess" or "activity" in the image, different types of quantizers are beneficial. In constant or slowly changing areas, the dynamic range of the quantizer can be small, while information is mainly low frequency and hence the quantizer resolution should be high. At fast changing areas, the dynamic range should be high, but then information is mainly high frequency and hence the HVS resolution is poorer and a rougher quantizer will do.

Suppose we divide the pixels in the image into L groups according to the "level of activity" in them. We can then optimally quantize each group separately and obtain the above benefit. A suggested measure of the "level of activity" can be a *masking function* of the type used in models of the HVS as outlined in section 3.4. Let i, j be the discrete coordinates of the image grid. Let

$$\Delta_{i,j}^{H} = 2x_{i,j} - x_{i-1,j} - x_{i+1,j}$$

$$\Delta^V_{i,j} = 2x_{i,j} - x_{i,j-1} - x_{i,j+1}$$

Then

$$M(i,j) = \sum_{l,m=-1}^{+1} \alpha^{\sqrt{l^2+m^2}} \left(\left|\Delta^H_{i-l,j-m}\right| \; + \; \left|\Delta^V_{i-l,j-m}\right| \right)$$

with $\alpha = 0.35$ a psycophysically determined factor. For this measure a plot of subjective weighting of the distortion was obtained (fig. 2a). This plot is then broken into L segments, (according to the group characteristics), and for each a separate optimal quantizer is designed (because fidelity within each group is less variable even a uniform quantizer can be used). Note that the group each pixel belongs to is side-information that should be transmitted too, however using variable length coding (Huffman), this added burden is small (fig. 2b), and the total bit rate can be reduced by as much as 30%.

Two facts should be mentions with regards to such adaptive schemes: the first is that experimentally it was found that 4 is the optimal number of groups beyond which improvement is marginal. The second is that for chromatic coordinates the best measure for the "level of activity" is the masking function ($M(i,j)$ above) of the luminance channel, hence it is the one usually used.

4.4 Vector Quantization

Vector quantization is used either as a compression scheme in its own right, (by grouping the tri-stimulus values of one pixel or more into vectors and quantizing them) or as part of another scheme, where previous stages generate vectors rather than scalars which have to be quantized. The principals underling vector quantization are similar to those of scalar quantization and we basically try to minimize terms of the form of eq. 2, except that the t's now define regions in space and the x and l_i's are vectors.

The major difference from the scalar case is that often $p(\vec{x})$ is unavailable or intractable. In that case we resort to the heuristic of a *training set* - that is we choose a set of vectors which seem typical in their distribution and from them we directly compute the quantizer without passing through an explicit distribution function.

To understand how this is done we observe that given the levels $\vec{l_i}$ it is relatively easy to classify a set of vectors $\{\vec{x}\}$ into groups such that

$$\sum_{i=1}^{n} d(\vec{x} - \vec{l}_{\vec{x}}) \tag{3}$$

is minimized. On the other hand, given that a set of vectors belongs to a specific group G, to find a vector $\vec{l_i}$ which minimizes the expression

$$\sum_{\vec{x} \in G} d(\vec{x} - \vec{l}_{\vec{x}}) \tag{4}$$

Is also computationally easy, as $d(\cdot)$ is usually differentiable or differentiable in parts. The algorithm, which is known as the *LBG* or *n–means*, algorithm is then an iteration of

- `Assign the elements of the training set into` n `groups.`
- `Find the best representative level for each group (as a weighted mean of its members).`

The algorithm is started with some initial set of $\{\vec{l}_i\}$ levels (say n vectors from the training set) and proceeds until changes are marginal.

It can be shown that this algorithm converges to a minimum, however this might not be a global one and results could be significantly effected by the chose of initial configuration. Also the algorithm suffers from a slow convergence rate.

A different approach known as the *nearest neighbor* approach attempts of alleviate these problems as follows:

- `Place each training set vector in a separate group.`
- `As long as there are more than` n `groups DO`
  - `Find the groups whose representative levels are closest.`
  - `Merge them into a single group.`
  - `Set the representative level to the weighted mean of all members.`

While this approach is fast and produces quite good results, we remark that it is best when applied as the initial configuration for the n–means algorithm which then converges faster and to even better quality results.

To improve the chance of obtaining the global minimum, it is possible to invoke the n–means algorithm a few times using different initial configurations. Clearly as we make more iterations the probability of a global minimum improves, but the computational penalty is high.

Once the n representative levels have been decided upon, actual quantization is done by replacing a vector $\vec{x}$ with the code for the level $\vec{l}_i$ which minimizes $d(\vec{x} - \vec{l}_i$. As such an *on-line* minimization might be time consuming, often a special data–structure called the *Voronoi diagram* of $\{\vec{l}_i\}$ is built (off-line). This data-structure allows a fast allocation of a member of $\{\vec{l}_i\}$ to an arbitrary vector $\vec{x}$.

We end this subsection with a remark that the problem of vector quantization is similar to the problem of *unsupervised learning* in pattern recognition and similar algorithms used in both cases. Borrowing from that literature we can also set (heuristic) criteria as to when we should add (split into) more groups or merge groups together.

4.5 Transform Coding

Transform coding is performed by dividing an image into blocks, performing on each block a transformation, and sending a quantized version of all (or some) of the transform coefficients. Other than the decorrelating effect of transform coding, discussed in section 2, compression in achieved by not sending some of the coefficients, or by quantizing them into rough levels. Once the specific transform to be used is chosen, there are three remaining types of *free* parameters: block size, which parameters are never sent and the quantization schemes for those quantizers that are sent.

Part of the compression in transform coding stems from the properties of the HVS - it being insensitive or less sensitive to some of the coefficients. For a specific transform scheme, these could be set once and for all. However much of the savings lie in the content of the specific image (or block) transmitted or in the interaction between it and the HVS - for example a block may be of low detail and then more coefficients can be ignored, or it may be of high detail and then HVS sensitivity decreases and rougher quantization can be used. The problem is how to obtain these benefits or "adapt" the method to the image/block transmitted, (indeed totally non-adaptive transform codings are rarely better than DPCM and do not justify the added complexity).

Three approaches are possible: *Apriori* - to select a set of "representative" images to be used as the means of optimizing the free parameters. *Fixed per image* - use all the blocks of an image as the statistical ensemble on which optimization proceeds, in this case the "results" have to be sent as *side information. Adaptive* - to modify the free parameters specifically for each block in the image according to its properties. O'course this entails more "side information". Alternatively, adaptivity can be somewhat compromised by predicting the properties of a given block on the basis of information already present at the decoder (such as previous blocks or those coefficients within the block already transmitted).

Selecting block size (N) is always done apriori. Besides the often dominating consideration of computational complexity and buffer sizes, the trade-off, is between on the one hand smaller block size, with a high chance of a block being either low detail, (most coefficients can be ignored) or high detail (rough quantization can be used), and on the other hand larger block size with the benefit of better decorrelation of coefficients and more concentration of energy in the lower coefficients.

Deciding which coefficients to ignore is either an apriori or a fixed per image decision. However it is possible for a specific block to ignore more coefficients by adaptively encoding this information. In this case we often benefit from the fact that the basis vectors of the transformation have a natural "frequency" ordering such that on real world images, if the coefficient of some basis vector can be ignored, all coefficients of vectors representing "higher frequencies" can be ignored too. In this case we can adaptively select the number of coefficients sent, simply by sending a special *end-of-block* code word indicating that no further coefficients are necessary.

In selecting the quantizers of the remaining coefficients all three methods can be used, but adaptive methods are by far superior. Before we dwell in details, we

note that the first coefficient of all transforms is the average luminance of the block. This coefficient which is termed the *DC component* of the block, should be treated differently from all other coefficients: While most coefficients can be modeled as Laplace distributions, the DC is either uniformly distributed or Gaussian distributed. Also the effects of quantization errors in the DC manifest themselves mainly as false contours along block boundaries, while most other coefficients (depending on block size) are manifest as either noise or loss of resolution along edges. The DC component is hence usually quantized via a uniform quantizer with a sufficient number of levels to make inter-block boundaries unnoticable.

For the other coefficients, quantization will be satisfactory if we have

$$|c_m - \tilde{c}_m| < T_m$$

Where T_m is the sensitivity of the HVS to the m'th frequency and $\tilde{c}_m$ is the quantized value. If block size is large, then the separate frequencies are not detected by the HVS independently, we must then group the coefficients into sets $\{S_i\}$ of similar spatial frequencies and require

$$\sum_{m \in S_i} (c_m - \tilde{c}_m)^2 < T_i \qquad \forall i$$

Where $T_m = T_i$ for all m in the frequency range of S_i. If we further want to take advantage of the fact that in more "active" areas HVS sensitivity to quantization error drops, we define for the entire block an activity measure A. This can be either as outlined in section 3.4 or using the (now available) frequency information

$$A = 1 + (normalizing\ factor) \sum_{m=2}^{N} c_m^2$$

Our requirement then becomes

$$\sum_{m \in S_i} \frac{(c_m - \tilde{c}_m)^2}{A \cdot T_m} < 1 \qquad \forall i$$

or

$$\sum_{m \in S_i} \frac{(c_m - \tilde{c}_m)^2}{A \cdot T_m} < (1 + \epsilon) \qquad \forall i$$

If we want to allow for the possibility of supra-threshold distortion. In designing a quantizer we can now either:
- for a fixed ϵ (distortion) minimize the average bit rate,
or
- for a fixed bit rate minimize ϵ.

Often the actual constraint is not the bit rate for a given coefficient, but rather the overall average bit rate per entire block. In this case the quantization problem

becomes a problem of *bit allocation* - that is how to allocate the available bits between the coefficients. For this purpose a simplified distortion measure is used:

$$D = \frac{1}{N}\sum_{m=1}^{N}\frac{E(c_m - \tilde{c}_m)^2}{A \cdot T_m} = \frac{1}{N}\sum_{m=1}^{N}\frac{\epsilon_m}{A \cdot T_m}$$

Given a statistical model for c_m and a quantization scheme (say Max–Lloyd with fixed word length or uniform with variable word length), the quantization MSE of c_m is a function of r_m - the bit rate for this coefficient. Thus the problems becomes: Minimize

$$D = \frac{1}{N}\sum_{m=1}^{N}\frac{\epsilon_m(r_m)}{A \cdot T_m}$$

Subject to

$$R = \frac{1}{N}\sum_{m=1}^{N} r_m$$

It can be shown that this problem has a solution of the form:

$$\frac{d\epsilon_m(r_m)}{dr_m}\frac{1}{A \cdot T_m} = -\theta \qquad \forall m$$

However note that in this solution it is mathematically possible that $r_m < 0$ for some m. As this is physically impossible, when this happens we set $r_m = 0$ (that is we do not transmit the m'th coefficient) and re-solve for the remaining coefficients.

For practical purposes such solutions were obtained for the case of Laplacian statistics (except for the DC component which is uniform) and for either Max–Lloyd with fixed word length or uniform quantization with variable word length.

A typical *Adaptive* approach is to classify each block according to its spatial activity into one of say L groups. For each group, because the level of activity is roughly the same, an optimal quantizer can be designed. The actual coding of the block then includes sending the class to which the block belongs, followed by the quantization (or ignoring) of the coefficients according to the optimal quantizer of that group.

A final issue to be considered for transform coding is that of *buffering*. When performing adaptive transform coding, most blocks are "short" (compressed well) however, once in a while, a "long" or a-typical block appears. This problem is solved via buffers which store information and transmit it at a constant rate. There could still, however, be a problem of possible buffer overflow. But for the penalty of added memory costs, it would seem that this can always be solved by using larger buffers. However, in applications which require *real time* such as video-conferencing, the time constraints set an upper limit on the possible buffer size. An alternative solution, which can then be used, is a scheme which detects the state of the buffer and allows higher distortion (and thus lower bit rates) when the buffer begins to fill and returns to lower distortion (and hence higher bit rates) as the buffer becomes empty.

4.6 Coding of B/W Graphics

Black and white graphics are a special family of images which can benefit from special types of compression schemes. In this section we briefly mention some of these. The first is *font* or *mosaic* encoding - many special purpose images (for example text) are compositions of only a small number of shapes. In this case these shapes can be made available both at the encoder and decoder and then the transmitted information can include only the "index" of each shape and its position. When the set is small but not available in the decoder, it can be sent as side information before transmission begins. For example TEX text is thus sent to laser printers - a font dictionary introduces all the symbols which appear in the text, and then the "pages" themselves are composed of locations and indexes of shapes.

We remark that while such schemes achieve superior compression, they are easy to implement only when the image is generated and not given as an image. Otherwise we need a preliminary *pattern recognition* step to identify the font or mosaic elements involved, a step which in current technology is often expensive and not completely reliable.

In more general cases, a commonly used method is *run length coding*. In this method the image is divided into lines according to raster scan and each line is encoded separately. Note that as each line is a sequence of '0's and '1's, it is sufficient to transmit the first bit and the locations of successive changes. Thus a sequence

$$000011111111100000000000001111111000000000011111111100000000000$$

would be encoded as

$$04, 9, 13, 7, 10, 9, 11$$

Which is then encoded for transmission via variable length codes. As real world graphics contain large areas of black or white, significant compression results. Note that most Black/White images can be characterized as black "elements" on white "background" (paper). Because of this the statistics typical of the "black" and "white" runs are different and hence two sets of Huffman codes should be used.

A disadvantage of *run length coding* is that is only utilizes the horizontal correlations. To take advantage of vertical correlations two methods are used - one is *predictive coding* and the other is *line to line run length coding*. In predictive coding the "current pixel" is predicted based on previously transmitted pixels. The difference (or error) signal in this case is also composed of 0's and 1's (but with lower entropy). This sequence is then run length coded.

In line to line run length coding the transmitted information includes the difference between a run in the current line and a corresponding run in the previous line. As it is possible that a run in this (or the previous) line has no corresponding line in the previous (or this line) two special messages called *MERGE* and *NEW_START* are also possible. These and the difference runs are then variable length coded.

Another possible method, which perform less well, is *block coding* where the

image is divided into blocks of size $n \times m$ which are then coded via a variable length code of all possible such blocks.

A variation of block coding is *quad trees*. In this method the image is divided into 4 squares. If a square happens to be entirely of the same color it is transmitted, else it is further divided into 4 recursively until only equi-color squares exist. Again the as real world graphics contain large areas of equi-color significant compression results. A major advantage of this method over other graphic encoding methods is that when only part of the information is transmitted the entire image is seen but at a poorer resolution (rather than only part of the image). This is useful for *browsing* and similar applications.

Another final method to mention, often used for edge maps, that is images which are all white except for one pixel thin black lines, is *chain coding*. In this method the position of one edge pixel is given and then at each step only one of 4 (or 8) directions are sent. These tell the decoder where to find the "next" edge pixel. Under the assumption that edge pixels are a small minority in the image significant compression results (as only 2-3 bits per pixel are necessary for an edge pixel and non for a white pixel). This method can also be used with arbitrary graphics by transforming the graphics into an edge map, where an *edge point* is a pixel in which a change of color occurs. Again significant compression results only if large regions have the same color (and hence edge pixels will be few).