Version JL/2

EGT3
ENGINEERING TRIPOS PART IIB: SOLUTIONS

Tuesday 2 May 2023 14:00 to 15:40

Module 4F8

IMAGE PROCESSING AND IMAGE CODING

*Answer not more than **three** questions.*

All questions carry the same number of marks.

*The **approximate** percentage of marks allocated to each part of a question is indicated in the right margin.*

Answers to questions in each section should be tied together and handed in separately.

*Write your candidate number **not** your name on the cover sheet.*

STATIONERY REQUIREMENTS
Single-sided script paper

SPECIAL REQUIREMENTS TO BE SUPPLIED FOR THIS EXAM
CUED approved calculator allowed
Engineering Data Book

10 minutes reading time is allowed for this paper at the start of the exam.

You may not start to read the questions printed on the subsequent pages of this question paper until instructed to do so.

You may not remove any stationery from the Examination Room.

1 (a) Windowing methods:

(i) The two most popular methods of forming 2D windows from 1D windows are

A. Taking the product of 1D windows:

$$w(u_1, u_2) = w_1(u_1)\, w_2(u_2)$$

B. Rotating a 1D window:

$$w(u_1, u_2) = w_1(u)\big|_{u=\sqrt{(u_1^2+u_2^2)}}$$

Taking the inverse FT of the ideal frequency response will give an impulse response which does not have finite support – to remedy this we multiply by a *window function* which forces the impulse response coefficients to zero for (n_1, n_2) outside R_h, the desired support region. The actual filter frequency response $H(\omega_1, \omega_2)$ is then given by the **convolution** of the desired frequency response $H_d(\omega_1, \omega_2)$ with the window function spectrum $W(\omega_1, \omega_2)$.

This is exactly as we should expect since we multiply in the spatial domain and must therefore convolve in the frequency domain.

Thus the effect of the window is to smooth H_d. [15%]

(ii) This is a product window. The $\cos^2$ compared to the cos is shown in figure **??** (left) for $U_1 = \pi/2$. The window is then (again for $U_1 = U_2 = \pi/2$) shown in figure **??** (right). Anything which approximates this will be OK. [10%]

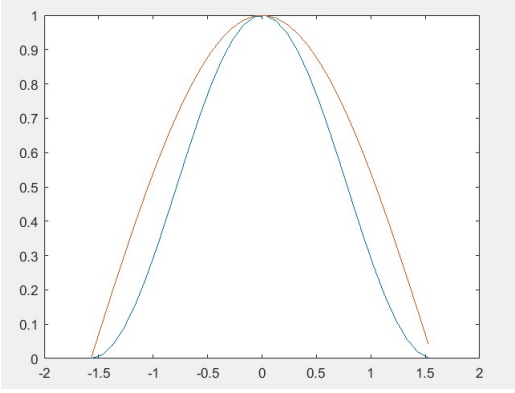


Fig. 1

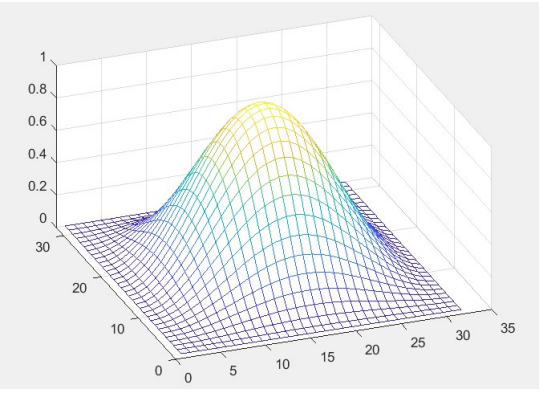


Fig. 2

(iii) We can see that this is a sinc function with its zeros (which delineate the sidelobes) at $U\omega = \pm n\pi$, $n = 1, 2, \ldots$ – however, the denominator ensures that the amplitude falls off quickly as ω increases. For $U = 1$ the Fourier transform (right) is sketched in figure **??**, with the numerator and demominator also sketched (left). [20%]

 (cont.

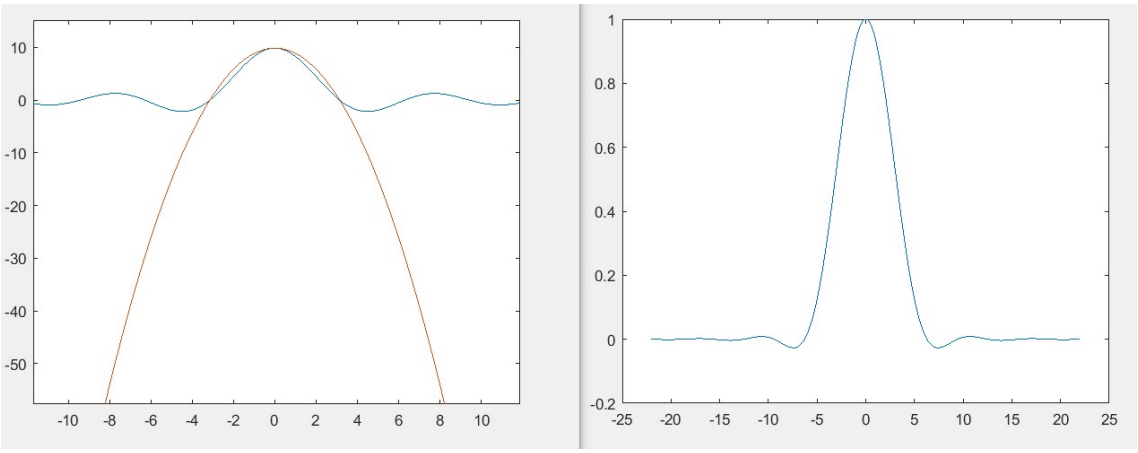


Fig. 3

We can see from the figure that there are no significant sidelobes after the first, and hence there will be minimal disturbance from sidelobe artefacts. however, the width of the main lobe is reasonably wide, but better than that produced by a rectangular window and a raised cosine window.

(b) Image deconvolution:

(i) Take the Fourier transform of each side of the convolution equation, disregarding the noise (assuming it is negligible) to give:

$$Y(\omega_1, \omega_2) = H(\omega_1, \omega_2)X(\omega_1, \omega_2)$$

$$\therefore X(\omega_1, \omega_2) = \frac{Y(\omega_1, \omega_2)}{H(\omega_1, \omega_2)}$$

Now, if $H(\omega_1, \omega_2)$ has zeros, then the inverse filter, $1/H$, will have infinite gain. i.e. if $1/H$ is very large (or indeed infinite), small noise in the regions of the frequency plane where these large values of $1/H$ occur can be hugely amplified. To counter this we can threshold the frequency response, leading to the so-called, pseudo-inverse or generalised inverse filter $H_g(\omega_1, \omega_2)$ given by

$$H_g(\omega_1, \omega_2) = \begin{cases} \frac{1}{H(\omega_1,\omega_2)} & \frac{1}{|H(\omega_1,\omega_2|} < \gamma \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

or

$$H_g(\omega_1, \omega_2) = \begin{cases} \frac{1}{H(\omega_1,\omega_2)} & \frac{1}{|H(\omega_1,\omega_2|} < \gamma \\ \gamma\frac{|H(\omega_1,\omega_2|}{H(\omega_1,\omega_2)} & \text{otherwise} \end{cases} \tag{2}$$

Clearly for $\frac{1}{|H(\omega_1,\omega_2|} \geq \gamma$ in equation **??**, the modulus of the filter is set as γ, whereas in previous equation it is set as 0.
Although the *generalised inverse filter* may perform reasonably well on *noiseless images*, the performance is unsatisfactory with even mildly noisy images due to the still significant noise gain at frequencies where $H(\omega_1, \omega_2)$ is relatively small. [15%]

(ii) As in the previous part, $H(\omega)$ is the FT of the psf, P_{xx} is the power spectrum (FT of the autocorrelation) of the original image. P_{dd} is the power spectrum of the noise. If P_{dd} is not known, we might reasonably estimate it by isolating a portion of the image that we believe contains only noise (recall our model is that the noise is additive) and forming P_{dd} from that region – we then make the assumption (not necessarily correct) that this then holds for the whole of the image.
Estimating the power spectrum of the original signal is not such a simple matter; one method would be as follows: assume we have estimated P_{dd}.

$$y(\mathbf{n}) = Lx(\mathbf{n}) + d(\mathbf{n})$$

then taking FTs gives

$$Y(\omega) = H(\omega)X(\omega) + D(\omega)$$

where $H = FT(L)$. Thus, if we have an estimate of D, we can estimate $Y - D$ to be approximately the FT of the signal. Or, if our blurring function is known we can obtain a better estimate of X by pseudo-inverse filtering, i.e.

$$X(\omega) = \frac{Y(\omega) - D(\omega)}{H_g(\omega)}$$

where H_g is the pseudo-inverse filter formed from H. [15%]

(iii) The matrix form of the Wiener filter is given by:

$$W = (C^{-1} + L^T N^{-1} L)^{-1} L^T N^{-1}$$

Where $C = E(\mathbf{x}\mathbf{x}^{\mathrm{T}})$ (corresponding to P_{xx}), $N = E(\mathbf{d}\mathbf{d}^{\mathrm{T}})$ (corresponding to P_{dd}) and L is of course the distortion (corresponding to H). [20%]

(iv) Notice how, in our expression for W, if the C^{-1} were not present in W we would just have $W = L^{-1}$ [since with no C^{-1} we can write $W = (L^T N^{-1} L)^{-1}(L^T N^{-1} L)L^{-1} = L^{-1}$]
We say we have *regularized* the inverse; effectively we have added on another term in order to avoid singularities. [5%]

2 (a) Sampling on grids:

(i) The Fourier transform of the sampled signal is given by:

$$G_s(\omega_1, \omega_2) = \frac{1}{\Delta_1 \Delta_2} \sum_{p_1=-\infty}^{\infty} \sum_{p_2=-\infty}^{\infty} G(\omega_1 - p_1\Omega_1, \omega_2 - p_2\Omega_2)$$

It can therefore be seen that the Fourier transform or spectrum of the sampled 2D signal is the periodic repetition of the spectrum of the unsampled 2D signal – precisely analogous to *aliasing* in the 1D case. It is therefore clear that for a bandlimited 2D signal, we must sample at more than twice the largest frequencies in the signal to keep these copies of the FT separate. Hence

$$\frac{2\pi}{\Delta_1} > 2\Omega_{B1} \quad \frac{2\pi}{\Delta_2} > 2\Omega_{B2}$$

These are the Nyquist frequencies, and if we sample below these we observe aliasing artefacts. [15%]

(ii) The signal $g(u_1, u_2)$ is sampled on the non-standard hexagonal lattice shown in figure **??**, to produce a sampled signal, $g_s(u_1, u_2)$.

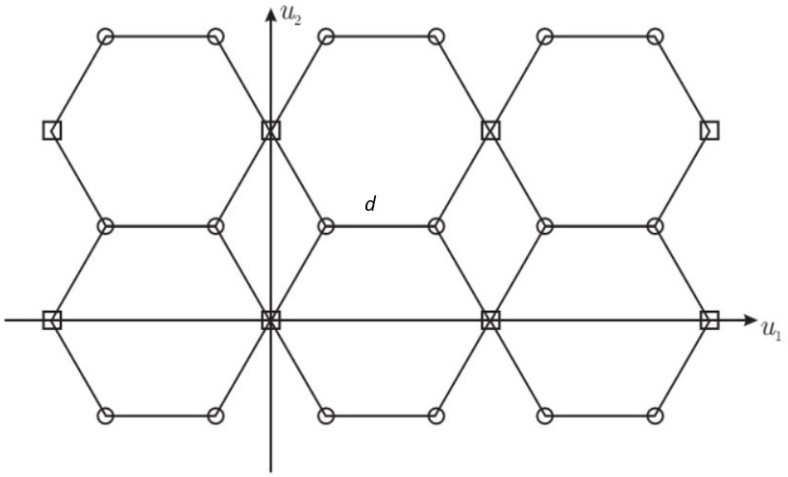


Fig. 4

Consider this array as the sum of two sampling functions, s_1 for the □ array, and s_2 for the ⊙ array.

By inspection, the sampling function, s_1, for the □ array is given by

$$s_1(u_1, u_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \delta(u_1 - n_1\Delta_1, u_2 - n_2\Delta_2)$$

where $\Delta_1 = 2d$ and $\Delta_2 = \sqrt{3}d$.
Similarly, the sampling function, s_2, for the $\odot$ array can be seen to be

$$s_2(u_1, u_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \delta(u_1 - [n_1 + \tfrac{1}{2}]\tilde{\Delta}_1, u_2 - [n_2 + \tfrac{1}{2}]\tilde{\Delta}_2)$$

where $\tilde{\Delta}_1 = d$ and $\tilde{\Delta}_2 = \sqrt{3}d$.
We can now use our result in the previous part to produce the FT of a signal sampled on this grid as the sum of the FT on a rectangular grid and the FT on a shifted rectangular grid:

$$\begin{aligned} G_s(\omega_1, \omega_2) \;=\; & \frac{1}{\Delta_1\,\Delta_2} \sum_{p_1=-\infty}^{\infty} \sum_{p_2=-\infty}^{\infty} G(\omega_1 - p_1\Omega_1, \omega_2 - p_2\Omega_2) + \\ & \frac{1}{\tilde{\Delta}_1\,\tilde{\Delta}_2} \sum_{p_1=-\infty}^{\infty} \sum_{p_2=-\infty}^{\infty} G\left(\omega_1 - p_1\Omega_1', \omega_2 - p_2\Omega_2'\right) \mathrm{e}^{-j(p_1\Omega_1'\mu_1 + p_2\Omega_2'\mu_2)} \end{aligned} \tag{3}$$

Where $\Omega_1 = \frac{2\pi}{\Delta_1}$ and $\Omega_2 = \frac{2\pi}{\Delta_2}$ and $\Omega_1' = \frac{2\pi}{\tilde{\Delta}_1}$ and $\Omega_2' = \frac{2\pi}{\tilde{\Delta}_2}$. The shift parameters are $\mu_1 = d/2$ and $\mu_2 = 1\sqrt{3}d/2$. So that the exponential term becomes $\mathrm{e}^{-j(p_1\pi + p_2\pi)}$.
Combining these (noting that the shifted FT can be written in terms of even and odd p_1) we have

$$G_s(\omega_1, \omega_2) = \frac{1}{2\sqrt{3}d^2} \sum_{p_1=-\infty}^{\infty} \sum_{p_2=-\infty}^{\infty} G\left(\omega_1 - p_1\frac{\pi}{d}, \omega_2 - p_2\frac{2\pi}{\sqrt{3}d}\right) [1+2\beta\mathrm{e}^{-j(p_1/2+p_2)\pi}]$$

where $\beta = 0$ if p_1 is odd and $\beta = 1$ if p_1 is even. [30%]

(b) Amplitude and Phase in image Fourier transforms:

(i) Perception of images is very much concerned with lines and edges. It can be shown that if we discard the amplitude information present in the 2D FT of an image, we can still reconstruct a recognisable image due to the fact that edge information is retained in the phases of the FT.
If a filter phase response is non-linear, then the various frequency components which contribute to an edge in an image will be phase-shifted with respect to each other in

such a way that they no longer add up to produce a sharp edge – i.e. dispersion takes place. It is often simplest to enforce the *zero-phase* condition, i.e. insisting that the frequency response is purely real, so that

$$H(\omega_1, \omega_2) = H^*(\omega_1, \omega_2)$$

Thus, ensuring that our filters are zero-phase will ensure that we preserve edges – crucial for image recognition. [10%]

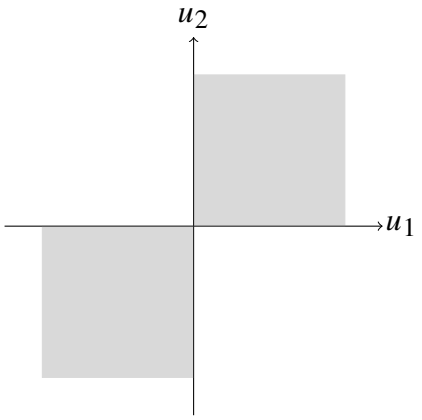


Fig. 5

(ii) ie g in the 4 quadrants of the image plane we see that it takes the following form:

$$g(u_1, u_2) = \begin{cases} +1 & \text{if } u_1 < 0 \text{ and } u_2 < 0 \\ +1 & \text{if } u_1 \geq 0 \text{ and } u_2 \geq 0 \\ -1 & \text{if } u_1 < 0 \text{ and } u_2 \geq 0 \\ -1 & \text{if } u_1 \geq 0 \text{ and } u_2 < 0 \end{cases}$$

Thus we have a 'checkerboard' corner, which displays edges along the u_1 and u_2 axes.
Now form the 2D Fourier transform of g: using the result given, we see that this is trivially:

$$G(\omega_1, \omega_2) = \left(\frac{2}{j\omega_1}\right)\left(\frac{2}{j\omega_2}\right) = \frac{-4}{\omega_1\omega_2}$$

[15%]

(iii) We can see that figure 1 is just a rotated version of the corner pattern above, so that

$$g_r(u_1, u_2) = g(u'_1, u'_2)$$

where u'_1 and u'_2 are the coordinates formed by rotating (u_1, u_2) anticlockwise through 90 degrees, ie $\mathbf{u}' = R\mathbf{u}$, where R is the matrix which performs this rotation. Therefore we know that the FT of g_r will be given by

$$G_r(\omega_1, \omega_2) = \left(\frac{2}{j\omega'_1}\right)\left(\frac{2}{j\omega'_2}\right) = \frac{-4}{\omega'_1\omega'_2}$$

where $\omega'_1 = \frac{\omega_1+\omega_2}{\sqrt{2}}$ and $\omega'_2 = \frac{-\omega_1+\omega_2}{\sqrt{2}}$. [20%]

(iv) From G_r we can form a new function G_R^{phase} where we set all the amplitudes to unity, thus preserving only the phase information. From the form of G_r we can see that

$$G_r^{phase}(\omega_1, \omega_2) = \begin{cases} \exp(j\pi) & \text{if } \omega'_1 < 0 \text{ and } \omega'_2 < 0 \\ \exp(j\pi) & \text{if } \omega'_1 \geq 0 \text{ and } \omega'_2 \geq 0 \\ \exp(j0) & \text{if } \omega'_1 < 0 \text{ and } \omega'_2 \geq 0 \\ \exp(j0) & \text{if } \omega'_1 \geq 0 \text{ and } \omega'_2 < 0 \end{cases}$$

G_r^{phase} has essentially the same form as g_r, so when we take the IFT we will find:

$$g_{phase} \propto \frac{1}{u'_1 u'_2}$$

where $u'_1 = \frac{u_1+u_2}{\sqrt{2}}$ and $u'_2 = \frac{-u_1+u_2}{\sqrt{2}}$.
As u'_1 or u'_2 tend to zero, this function spikes, so we are picking out the rotated axes, which are indeed the edges in our image. [10%]

3 (a) We first transform the columns of X by multiplying on the left by T and then the rows of the resulting matrix by multiplying the the right by T^{T}, ie

$$Y = TXT^{\mathrm{T}} = \begin{bmatrix} a+b+c+d & (a+c)-(b+d) \\ (a-c)+(b-d) & (a-c)-(b-d) \end{bmatrix}$$

These elements correspond to the following *filtering* processes:

Top L:$a+b+c+d$ = 4-point average or 2D lowpass (Lo-Lo) filter.

Top R:$(a+c)-(b+d) \equiv (a-b)+(c-d)$ = Average horizontal gradient or horizontal highpass and vertical lowpass (Hi-Lo) filter.

Lower L:$(a+b)-(c+d) \equiv (a-c)+(b-d)$ = Average vertical gradient or horizontal lowpass and vertical highpass (Lo-Hi) filter.

Lower R:$(a-b)-(c-d) \equiv (a-c)-(b-d)$ = Diagonal curvature or 2D highpass (Hi-Hi) filter.

[15%]

(b) From the form of T we can see that $T = T^{\mathrm{T}} = T^{-1}$, it is thus an *orthonormal transformation* such that $T^{\mathrm{T}}T = I$, where I is the identity matrix. We see this using

$$\text{(Energy of } \mathbf{y}) = \mathbf{y}^T\,\mathbf{y} = \mathbf{x}^T\,\mathbf{T}^T\,\mathbf{T}\,\mathbf{x} = \mathbf{x}^T\,\mathbf{I}\,\mathbf{x} = \mathbf{x}^T\,\mathbf{x} = \text{(Energy of } \mathbf{x})$$

To apply the first level Haar transform to an $N \times N$ image (assume N eve), we split the image into 2×2 blocks and apply T to each block – then we rearrange the output so that there are 4 blocks of $N/2 \times N/2$ elements. The top LH block contains all the Lo-Lo elements of each of the Haar transform of the 2×2 blocks; the top RH block contains all the Hi-Lo elements; the bottom LH block is the collection of Lo-Hi elements and the bottom RH block collects together all of the Hi-Hi elements.

To then apply a level-2 Haar transform, we take the top LH block, which contains the lowpass components (and therefore contains the majority of the energy of the image) and make a second application of the Haar transform. This process will result in 4 blocks of $N/4 \times N/4$ and 3 blocks of $N/2 \times N/2$.

If we apply the first level of the Haar transform, we obtain 4 Lo-Lo elements of each of the 4 2×2 blocks – this lowpass block is given by :

$$T1 = \begin{bmatrix} z_{11}+z_{12}+z_{21}+z_{22} & z_{13}+z_{14}+z_{23}+z_{24} \\ z_{31}+z_{32}+z_{41}+z_{42} & z_{33}+z_{34}+z_{43}+z_{44} \end{bmatrix}$$

Now the Haar transform is again applied to $T1$, and we know from above that the Hi-Lo component is given by $(T1_{11} + T1_{21}) - (T1_{12} + T1_{22})$. In terms of the zs this becomes:

$$\left(\sum_{i,j=1}^{i,j=2} z_{ij} + \sum_{i=3}^{i=4}\sum_{j=1}^{j=2} z_{ij}\right) - \left(\sum_{i,j=3}^{i,j=4} z_{ij} + \sum_{j=3}^{j=4}\sum_{i=1}^{i=2} z_{ij}\right)$$

[25%]

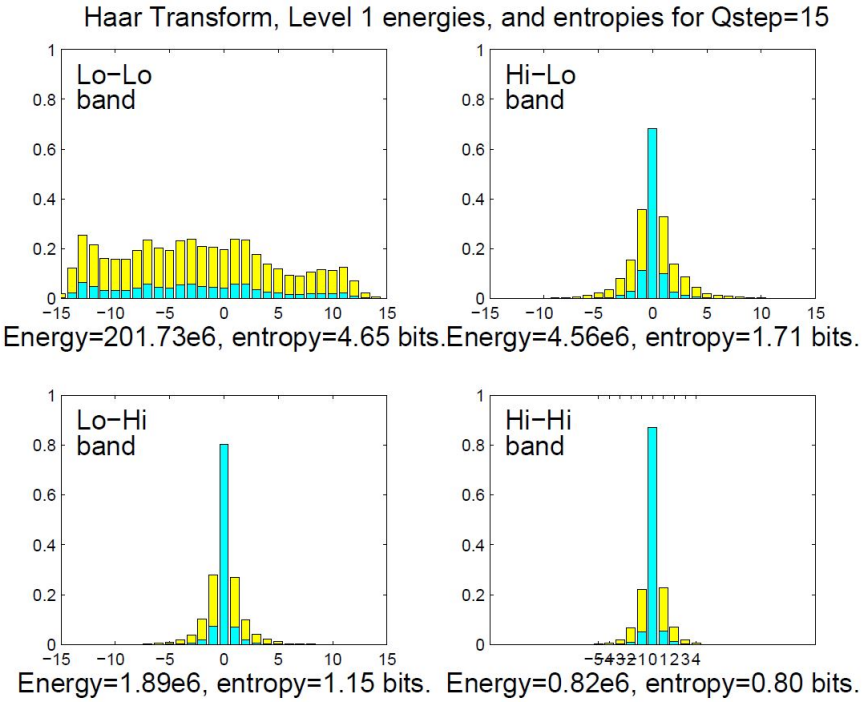
(c) Entropy of source information for an image X quantised to M levels, is H_X and is defined as:

$$H_X = \sum_{i=0}^{M-1} p_i \log_2\left(\frac{1}{p_i}\right) = -\sum_{i=0}^{M-1} p_i \log_2(p_i)$$

where p_i, $i = 0$ to $M-1$, is the probability of the i^{th} quantiser level being used (often obtained from a histogram of the pixel intensities).

This is useful as it gives the minimum number of bits per pixel needed to represent the quantised data for the image/subimage, to a given accuracy, assuming that we use an *ideal entropy code*.

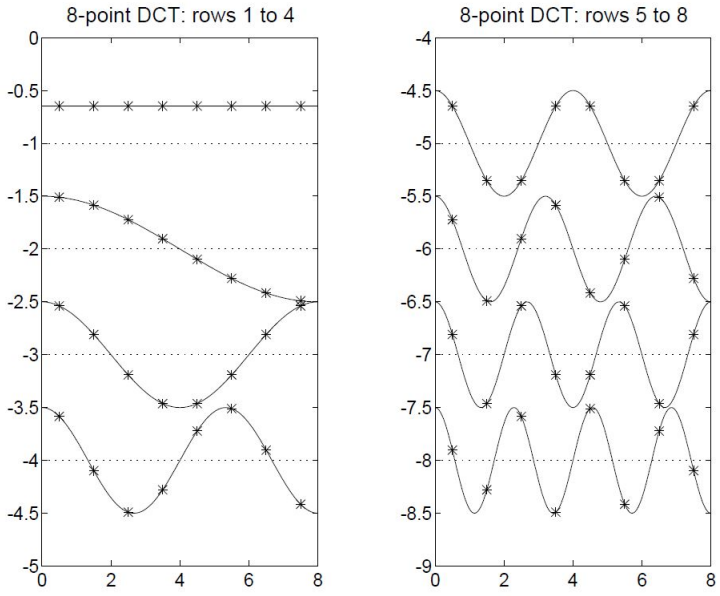
Figure **??** shows a typical distribution for both p_i (probability=yellow) and h_i (entropy= blue) for a natural image when the level 1 Haar transform is applied.



 (cont.

[20%]

(d) T is now the 8x8 DCT – if we plug in the values we obtain the following shapes for the rows of T



Note that rows 1,3,5,7 (odd rows) are all symmetric about the centre (between element 4 an 5), while rows 2,4,6,8 are all antisymmetric about the centre. This fact enables us speed up the multiplication $\mathbf{y} = T\mathbf{x}$.

First we form: two new 4-element vectors, **u**, **v**, made up from the elements of the vector **x**:

$$u(i) = x(i) + x(9-i) \quad \text{and} \quad v(i) = x(i) - x(9-i) \quad \text{for } i = 1 \rightarrow 4$$

and then form the odd and even terms in **y** from two 4×4 transforms:

$$\begin{bmatrix} y(1) \\ y(3) \\ y(5) \\ y(7) \end{bmatrix} = T_{\text{left,odd}}\,\mathbf{u} \quad \text{and} \quad \begin{bmatrix} y(2) \\ y(4) \\ y(6) \\ y(8) \end{bmatrix} = T_{\text{left,even}}\,\mathbf{v}$$

where $T_{\text{left,odd}}$ and $T_{\text{left,even}}$ are the 4×4 matrices formed by the left halves of the odd and even rows of T.

This reduces the computation to 8 add/subtract operations to form the u_i, v_i and $2 \times [4^2]$ mults and $2 \times [4 \times 3]$ adds – giving 32 mults and 32 adds, almost halving the total computation load.

Since $T_{\text{left,odd}}$ has the same symmetries as T (ie is a 4-point DCT matrix) we can apply this technique again and save even more computation. [25%]

(e) The 16×16 DCT is marginally better, in terms of entropy, than the 8×8 DCT. However, subjectively this is not the case since quantisation artefacts become more visible as the block size increases.

In practice, for a wide range of images and viewing conditions, 8×8 *has been found to be a good DCT block size* and is specified in many current coding standards (JPEG).

But, if we consider going to higher levels of the DCT, ie taking the top left subimage of the DCT and applying a further DCT, then we can obtain improved compression without loss of quality. For this reason, more recent standards (JPXR) use 2 levels of a 4×4 DCT. Of course, this means a slightly more complex pipeline. [15%]

4 (a) (i) See Figure **??**, where the left hand side is the Analysis filter bank and the right hand side is the Reconstruction filter bank.

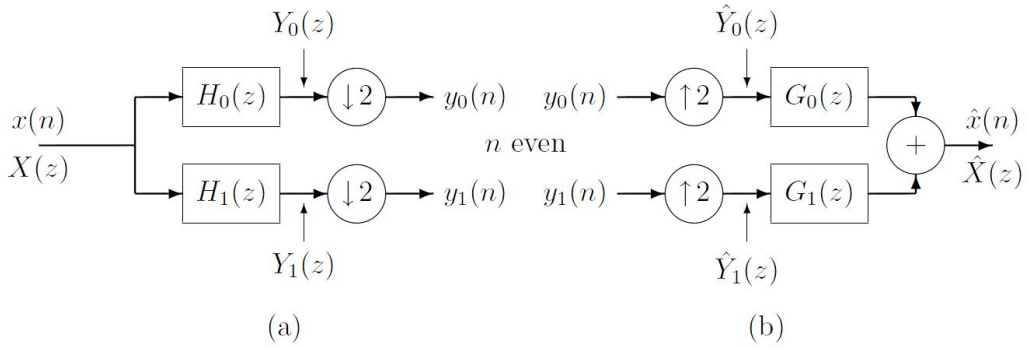


Fig. 6

$H_0(z)$ and $G_0(z)$ are lowpass filters and H_1 and G_1 are highpass filters.
Perfect Reconstruction means that we require that the reconstructed signal $\hat{X}(z)$ is identical to the input signal $X(z)$, which is of course important so that no distortion or loss takes place. [20%]

(ii) Consider the data samples y_n with z-transform

$$Y(z) = \sum_{n=-\infty}^{\infty} y_n z^{-n}$$

If y_n is downsampled by 2 and then upsampled by 2 to give $\hat{y}_n$, and if $\hat{Y}$ is the z-transform of $\{\hat{y}_n\}$, we have

$$\hat{y}_n = y_n \quad \text{for } n \text{ even} \qquad \hat{y}_n = 0 \quad \text{for } n \text{ odd}$$

Taking the z-transform of $\hat{y}_n$ then gives

$$\begin{aligned}
\hat{Y}(z) &= \sum_{\text{even } n} y_n z^{-n} \\
&= \sum_{\text{all } n} \frac{1}{2}\left[y_n z^{-n} + y_n(-z)^{-n}\right] \\
&= \frac{1}{2}\sum_n y_n z^{-n} + \frac{1}{2}\sum_n y_n(-z)^{-n} \\
&= \frac{1}{2}Y(z) + \frac{1}{2}Y(-z) \\
&= \frac{1}{2}\left[Y(z) + Y(-z)\right]
\end{aligned}$$

[15%]

(iii) For the *downsample-filter-upsample* case, first downsample and convolve with h – let $f(n) = x(2n)$:

$$\sum_i f(n-i)\ h(i) = \sum_i x(2(n-i))\ h(i) = \sum_i x(2n-2i)\ h(i)$$

Now upsample (fill in with zeros):

$$\begin{aligned} \hat{y}(n) &= \sum_i x(n-2i)\ h(i) \quad \text{for } n \text{ even} \\ &= 0 \qquad\qquad\qquad\quad \text{for } n \text{ odd} \end{aligned}$$

Take z-transforms:

$$\hat{Y}(z) = \sum_n \hat{y}(n)\ z^{-n} = \sum_{\text{even } n} \sum_i x(n-2i)\ h(i)\ z^{-n}$$

Reverse the order of summation and let $m = n - 2i$:

$$\begin{aligned} \Longrightarrow\ \hat{Y}(z) &= \sum_i h(i) \left(\sum_{\text{even } m} x(m)\ z^{-m}\ z^{-2i} \right) \qquad (4) \\ &= \left(\sum_i h(i)\ z^{-2i} \right) \left(\sum_{\text{even } m} x(m)\ z^{-m} \right) \end{aligned}$$

$$\begin{aligned} \hat{Y}(z) &= H(z^2)\ \frac{1}{2}[X(z) + X(-z)] \\ &= \frac{1}{2}[H(z^2)\ X(z) + H((-z)^2)\ X(-z)] \\ &= \frac{1}{2}[Y(z) + Y(-z)] \quad \text{where} \quad Y(z) = H(z^2)\ X(z) \end{aligned}$$

This describes the operations of *filter-downsample-upsample*.

In the first line above,

$$\hat{Y}(z) = \frac{1}{2}[X(z) + X(-z)]\ H(z^2) = \hat{X}(z)\ H(z^2)$$

shows that the filter $H(z^2)$ may be placed after the down/up-sampler as in *downsample-upsample-filter*, which proves the second result.
While this moving of the downsampler/upsamplers to the end of the chain is not useful in practice, it is a very useful tool for analysis purposes. [25%]

(b) (i) The sketches of typical Y, U, V histograms are shown in figure **??**: [10%]

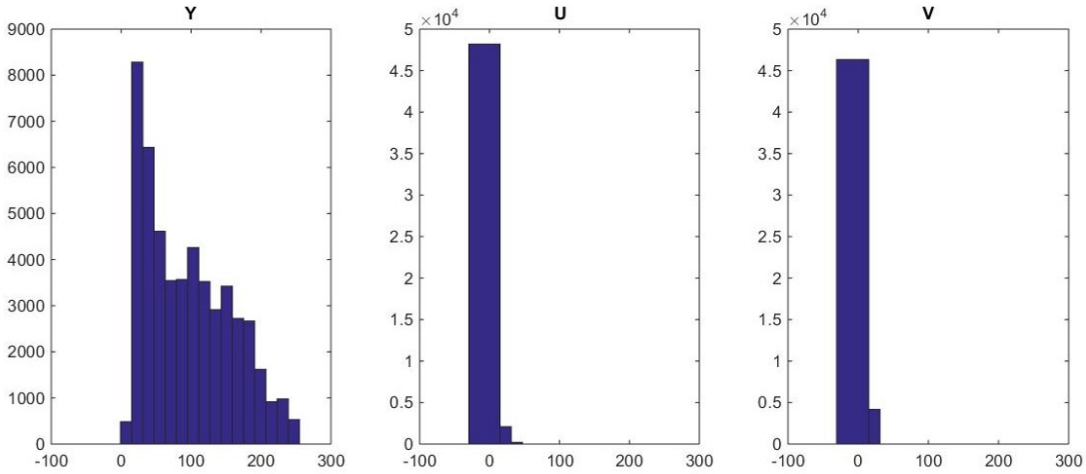


Fig. 7

(ii) The U and V components may be sampled at a lower rate than Y due to the *narrower bandwidth* as shown typically in Figure **??**. [5%]

(iii) The sketch below (figure **??**) illustrates the sensitivity of the human eye by plotting spatial frequency (freq of an alternating pattern of parallel stripes) against number of perceivable levels (a measure of human contrast sensitivity).

Look particularly at the sensitivity of the U (blue-yellow) and V (red-green) components: we see from Figure **??** that the maximum U sensitivity is about 1/6 that of the maximum Y sensitivity, and similarly the maximum V sensitivity is about 1/3 that of Y. Because of this low contrast sensitivity, we can quantise U and V more coarsely than Y. [10%]

(iv) $Y \implies 768 \times 1024 = 3 \times 2^{18}$ pixels.
$U, V \implies (768/2) \times (1024/2) = 3 \times 2^{16}$ pixels.
ie downsample U, V by a factor of 2 in each dimension.
Therefore the number of bits required is approximated by the $entropy \times No.pixels$.
For Y: $1.2 \times 3 \times 2^{18}$, for U, V: $3 \times 2 \times 0.5 \times \frac{1}{4} \times 2^{18}$.
Therefore total number of bits required is : $(1.2 + 0.25) \times 3 \times 2^{18} = 4.35 \times 2^{18} = 1.14 \times 10^6$. [15%]

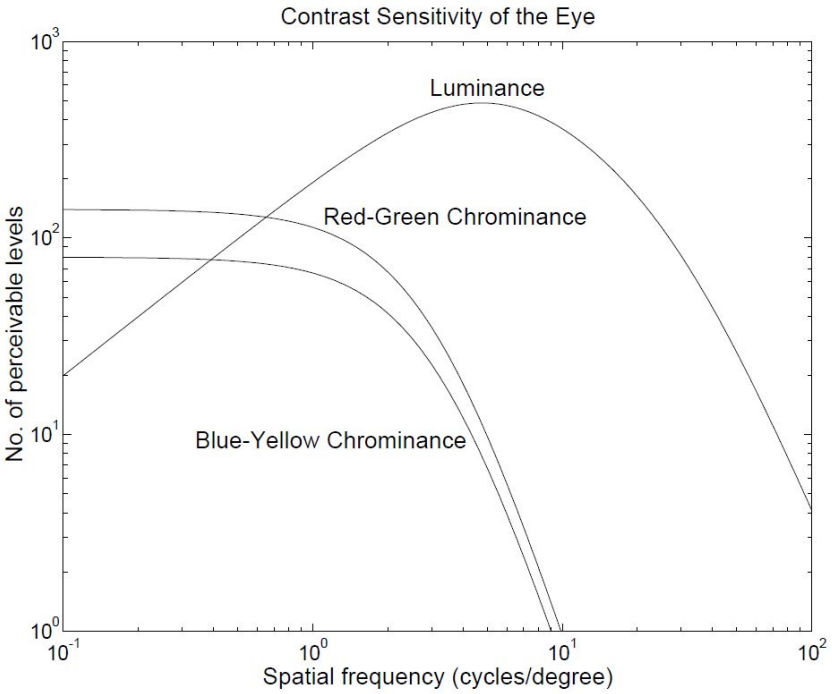


Fig. 8

END OF PAPER