2003   4F8 - Image Processing - Solutions   1

IIB CRIBS
4F8 2003

(#1) (a) During any filtering operation it is clear that the phase of an image may be altered. If a filter frequency response is zero-phase then there is no phase distortion/alteration - this requires the frequency response to be purely real, ie

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

which leads to the following relation between the impulse response coefficients

$$h(-n_1, -n_2) = h(n_1, n_2) \qquad (\text{for } h(n_1, n_2) \text{ real})$$

Unlike eg audio signals (1D), image ~~pe~~ perception is very much concerned with lines and edges - if a filter phase response is non-linear then the various frequency components that contribute to an edge will be phase shifted and dispersion of the edge will result. Thus we mostly attempt to construct image filters with this zero-phase property.

Ideal rectangular bandpass filter clearly has frequency response given by ~~$H(\omega_1, \omega_2) =$~~

$$H(\omega_1, \omega_2) = H_u - H_l$$

where

$$H_u(\omega_1,\omega_2) = \begin{cases} 1 & |\omega_1| < \Omega_{u1} \text{ and } |\omega_2| < \Omega_{u2} \\ 0 & \text{otherwise} \end{cases}$$

$$H_l(\omega_1,\omega_2) = \begin{cases} 1 & |\omega_1| < \Omega_{L1} \text{ and } |\omega_2| < \Omega_{L2} \\ 0 & \text{otherwise} \end{cases}$$

The filter impulse response is given by

$$h(n_1,n_2) = \frac{\Delta_1\Delta_2}{(2\pi)^2}\int_{-\pi/\Delta_2}^{\pi/\Delta_1}\int_{-\pi/\Delta_1}^{\pi/\Delta_1} H(\omega_1,\omega_2)e^{j(\omega_1 n_1\Delta_1+\omega_2 n_2\Delta_2)}\,d\omega_1 d\omega_2$$

$$= \frac{\Delta_1\Delta_2}{(2\pi)^2}\left\{\int_{-\Omega_{u2}}^{\Omega_{u2}}\int_{-\Omega_{u1}}^{\Omega_{u1}} e^{j(\omega_1 n_1\Delta_1+\omega_2 n_2\Delta_2)}\,d\omega_1 d\omega_2 - \int_{-\Omega_{L2}}^{\Omega_{L2}}\int_{-\Omega_{L1}}^{\Omega_{L1}} e^{j(\omega_1 n_1\Delta_1+\omega_2 n_2\Delta_2)}\,d\omega_1 d\omega_2\right\}$$

$$= \frac{\Delta_1\Delta_2}{(2\pi)^2}\left\{\left[\frac{e^{j\omega_1 n_1\Delta_1}}{j\cancel{\omega_1}n_1\Delta_1}\right]_{-\Omega_{u1}}^{\Omega_{u1}}\left[\frac{e^{j\omega_2 n_2\Delta_2}}{j\cancel{\omega_2}n_2\Delta_2}\right]_{-\Omega_{u2}}^{\Omega_{u2}} - \left[\frac{e^{j\omega_1 n_1\Delta_1}}{j\cancel{\omega_1}n_1\Delta_1}\right]_{-\Omega_{L1}}^{\Omega_{L1}}\left[\frac{e^{j\omega_2 n_2\Delta_2}}{j\cancel{\omega_2}n_2\Delta_2}\right]_{-\Omega_{L2}}^{\Omega_{L2}}\right.$$

$$= \frac{\Delta_1\Delta_2}{(2\pi)^2}\left\{\frac{2\sin\Omega_{u1}n_1\Delta_1}{n_1\Delta_1}\cdot\frac{2\sin\Omega_{u2}n_2\Delta_2}{n_2\Delta_2} - \frac{2\sin\Omega_{L1}n_1\Delta_1}{n_1\Delta_1}\cdot\frac{2\sin\Omega_{L2}n_2}{n_2\Delta_2}\right.$$

$$= \cancel{\frac{\Delta_1\Delta_2}{\pi^2}\Big\{\Omega_{u1}\Omega_{u2}\Big\{\mathrm{sinc}(n_1\Delta_1\Omega_{u1})\,\mathrm{sinc}(n_2\Delta_2\Omega_{u2})}$$

$$- \cancel{\mathrm{sinc}}$$

$$= \frac{\Delta_1\Delta_2}{\pi^2}\Big\{\Omega_{u1}\Omega_{u2}\,\mathrm{sinc}\,(n_1\Delta_1\Omega_{u1})\,\mathrm{sinc}\,(n_2\Delta_2\Omega_{u2})$$

$$- \Omega_{L1}\Omega_{L2}\,\mathrm{sinc}\,(n_1\Delta_1\Omega_{L1})\,\mathrm{sinc}\,(n_2\Delta_2\Omega_{L2}$$

$$h(n_1\Delta_1, n_2\Delta_2) = \frac{\Delta_1\Delta_2}{\pi^2}\Big\{\Omega_{u1}\Omega_{u2}\,\mathrm{sinc}\,\tilde{\Omega}_{u1}\,\mathrm{sinc}\,\tilde{\Omega}_{u2} - \Omega_{L1}\Omega_{L2}\,\mathrm{sinc}\,\tilde{\Omega}_{L1}\,\mathrm{sinc}\,\tilde{\Omega}_{L2}$$

---

where $\tilde{\Omega}_{u1} = n_1\Delta_1\Omega_{u1}$, $\tilde{\Omega}_{u2} = n_2\Delta_2\Omega_{u2}$ etc...

The idea behind the windowing method is to multiply the infinite support filter by a window function $\omega(n_1, n_2)$ which forces the impulse response coefficients $h(n_1,n_2)$ to zero for $n_1, n_2 \notin R_h$ where R_h is the desired support region.

$$h_w(n_1,n_2) = h(n_1,n_2)\,\omega(n_1,n_2).$$

If $h(n_1,n_2)$ and $\omega(n_1,n_2)$ satisfy the zero-phase requirement then so does $h_w(n_1,n_2)$.

Since multiplication in the spatial domain $\Rightarrow$ convolution in the frequency domain, we see that the filter frequency response, $H_w(\omega_1,\omega_2)$ is given by

$$H_w(\omega_1,\omega_2) = H(\omega_1,\omega_2) \underset{\substack{\uparrow \\ \text{convolution}}}{*} \underbrace{W(\omega_1,\omega_2)}_{\text{window spectrum}}.$$

⇒ the effect will generally be to smooth $H(\omega_1,\omega_2)$.

2 methods of obtaining 2D windows are

i) product of 2 1D windows: $\omega(n_1,n_2) = \omega_1(n_1)\,\omega_2(n_2)$

ii) rotation of a 1D window: ~~$\omega(n_1,n_2) =$~~ .

$$\omega(u_1,u_2) = \omega_1(u)\Big|_{u=\sqrt{u_1^2+u_2^2}} \qquad (\omega_1(u) \text{ continuous})$$

then sample $\omega(n_1,n_2) = \omega(u_1,u_2)\Big|_{u_1=n_1\Delta_1,\; u_2=n_2\Delta_2}$

Some possible suitable window functions would be

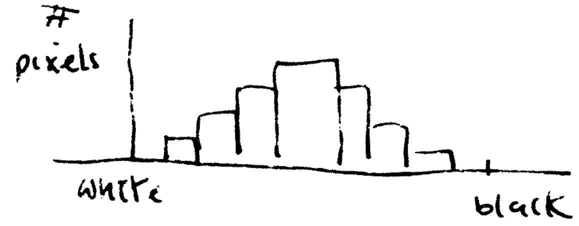
a) rectangular : $\omega(u_1,u_2) = \begin{cases} 1 & |u_1|<U_1,\ |u_2|<U_2 \\ 0 & \text{otherwise} \end{cases}$

b) Hamming : $\omega(u_1, \quad) = \begin{cases} 0.54 + 0.46\cos\frac{\pi u_1}{U_1} & \text{if } |u_1|<U_1 \\ 0 & \text{otherwise} \end{cases}$

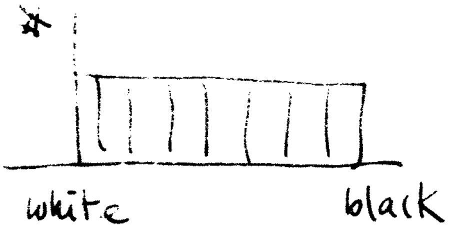
(similarly for rotated windows).

b) Often, an image will make poor use of the available grey levels, resulting from poor illumination of the scene or perhaps imaging non-linearities. The utilization of grey levels can be examined by plotting frequency of occurence vs grey level

eg



Here most pixels are concentrated around mid-grey. Suppose we apply a transformation to the image pixels such that the probability of occurence of any grey level is constant ⇒ a constant amplitude histogram:



This process is known as Histogram Equalisation and the required transformation is

$$y = g(x) = \int_0^x L\, p_x(u)\, du$$

where the output range is 0 to L and ~~the~~ $p_x(u)$ is the pdf of the pixels in the original image

When we apply this to a digital image the transformation becomes

$$y_k = \sum_{i=0}^{k} \frac{N_i}{NM} L$$

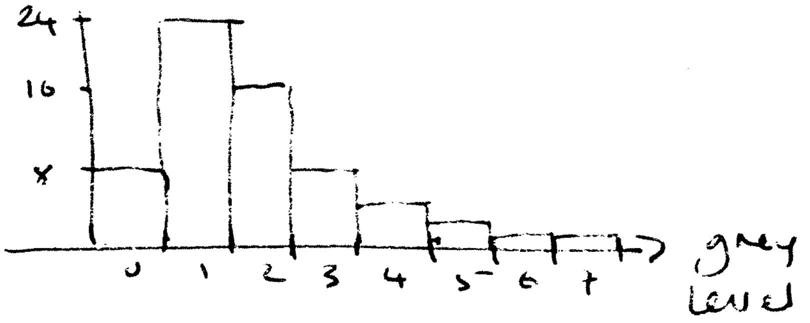
where $N \times M$ is dimension of image, N_i is # of occurences in bin i (x_i to $x_i + \Delta x_i$) and y_k is the transformed value of the kth level.

Because histogram equalisation improves contrast, the resulting image can often be significantly improved over the original in terms of intelligibility. However, sometimes the process can also make the image appear somewhat unnatural.

For the given 8×8 image we have 8 levels

k	#pixels	p_k
0	8	8/64
1	24	24/64
2	16	10/64
3	8	8/64
4	4	4/64
5	2	2/64
6	1	1/64
7	1	1/64
	64 total ✓	

⇒ histogram looks like



In this case the range we map to is 0 to $L=7$, $N\times M=64$ so that the transformation becomes

$$y_k = \sum_{i=1}^{k} 7\,\frac{N_i}{64} = \frac{7}{64}\sum_{i=1}^{k} N_i$$

$$\therefore\ y_0 = \frac{7}{64}\times 8 = 7/8 = \frac{56}{64}$$

$$y_1 = \cancel{\frac{7}{8} + \frac{7}{7}}\ \frac{7}{64}\{8+24\} = \cancel{12}\ \frac{7\times 32}{64}$$

$$y_2 = \frac{7}{64}\{32+16\} = \frac{7\times 48}{64}$$

$$y_3 = \frac{7}{64}\{48+8\} = \frac{7\times 56}{64}$$

$$y_4 = \frac{7}{64}(56+4\} = \frac{7\times 60}{64}$$

$$y_5 = \frac{7}{64}(60+2) = \frac{7\times 62}{64}$$

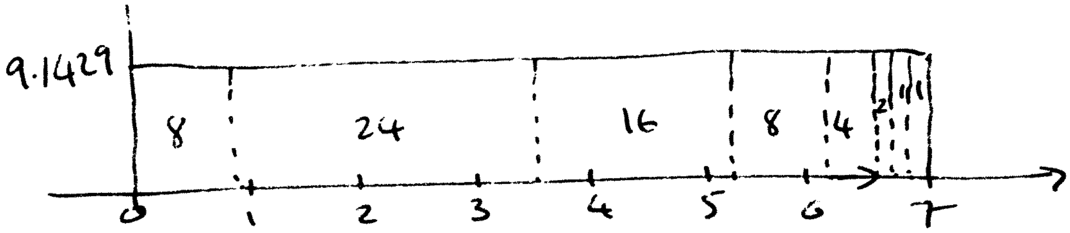
$$y_6 = \frac{7}{64}(62+1) = \frac{7\times 63}{64}$$

$$y_7 = \frac{7}{64}(63+1) = \frac{7\times 64}{64} =$$

$\Rightarrow$ our intervals become

$$\left\{ 0 \rightarrow 0.8750 \rightarrow 3.5 \rightarrow 5.25 \rightarrow 6.125 \rightarrow 6.5625 \rightarrow 6.7813 \rightarrow 6.8906 \rightarrow 7 \right\}$$

Sothat the 'theoretical' histogram would look like



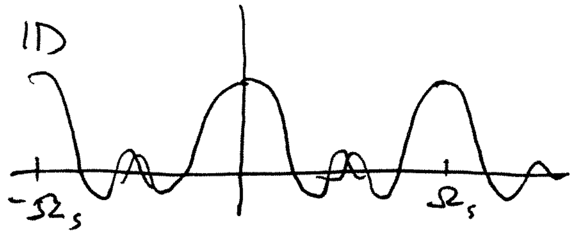
Note we should accept answers with different L's, ie L=8 etc..

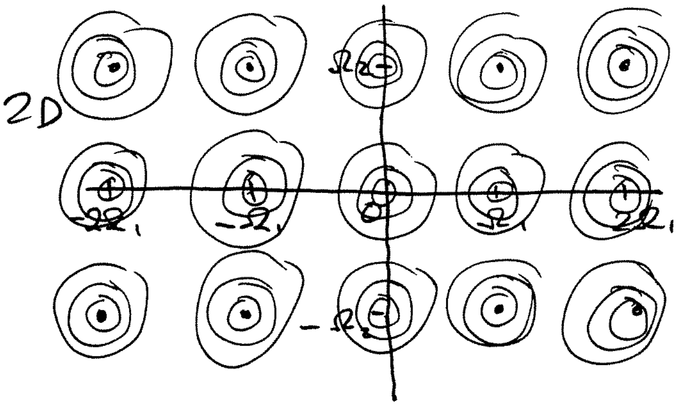
2003 **4F8 Image Processing** Answers

(2) If an image is sampled at spacings Δ_1, Δ_2 in the u_1, u_2 directions, then the FT of the sampled image is

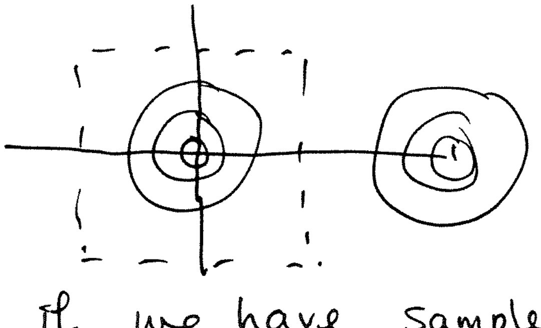
$$G_s(\omega_1,\omega_2) = \frac{1}{\Delta_1\Delta_2}\sum_{p_1=-\infty}^{\infty}\sum_{p_2=-\infty}^{\infty} G(\omega_1 - p_1\Omega_1, \omega_2 - p_2\Omega_2)$$

where G is the FT of the original image and $\Omega_i = 2\pi/\Delta_i$. Thus, the spectrum of the sampled image is the original repeated at every interval of the sampling frequencies:





We see that we require $\Omega_i > 2\Omega_{Bi}$, where Ω_{B1} and Ω_{B2} are the highest frequencies (ω_1, ω_2 directions) in the signal — assuming it is bandlimited. ~~$2B_1$~~ $2\Omega_{B1}$ and $2\Omega_{B2}$ are known as the 2D Nyquist frequencies. If we sample at rates greater than the Nyquist frequencies we see that we can low pass filter our sampled spectrum and recover the original spectrum exactly.



This is not the case if we have sampled at rates lower than the Nyquist rates – in this case we have overlapping of the repeated spectra, producing aliasing.

$$f(x,y) = 2\cos(3x+4y)$$

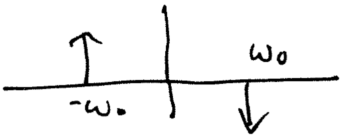
$$= 2\cos 3x \cos 4y - 2\sin 3x \sin 4y$$

$$F(\omega_1,\omega_2) = \int_{-\infty}^{\infty}\int g(x,y)\, e^{-j(\omega_1 x+\omega_2 y)}\, dx\, dy$$

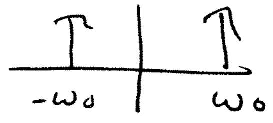
$$= 2\int_{-\infty}^{\infty} \cos 3x\, e^{-i\omega_1 x}\, dx \int_{-\infty}^{\infty} \cos 4y\, e^{-i\omega_2 y}\, dy$$

$$- 2\int_{-\infty}^{\infty} \sin 3x\, e^{-i\omega_1 x}\, dx \int_{-\infty}^{\infty} \sin 4y\, e^{-i\omega_2 y}\, dy$$

We know that $\displaystyle\int_{-\infty}^{\infty} \sin\omega_0 t\, e^{-i\omega t}\, dt = i\pi\left(\delta(\omega+\omega_0)-\delta(\omega-\omega_0)\right)$



and $\displaystyle\int_{-\infty}^{\infty} \cos\omega_0 t\, e^{-i\omega t}\, dt = \pi\left(\delta(\omega+\omega_0)+\delta(\omega-\omega_0)\right)$

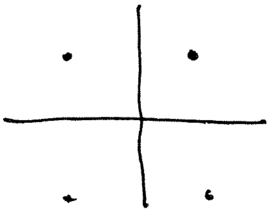


$$\Rightarrow F(\omega_1,\omega_2) = 2\pi^2\left[\delta(\omega_1+3)+\delta(\omega_1-3)\right]\left[\delta(\omega_2+4)+\delta(\omega_2-4)\right]$$

$$+ 2\pi^2\left[\delta(\omega_1+3)-\delta(\omega_1-3)\right]\left[\delta(\omega_2+4)-\delta(\omega_2-$$

$$\therefore \quad F(\omega_1,\omega_2) = 2 \times 2\pi^2 \left[\delta(\omega_1-3)\,\delta(\omega_2-4) + \delta(\omega_1+3)\,\delta(\omega_2+4) \right]$$

Since $F(\omega_1,\omega_2) = 4\pi^2 \left[\delta(\omega_1-3)\,\delta(\omega_2-4) + \delta(\omega_1+3)\,\delta(\omega_2+4) \right]$



we see that clearly

$$F(\omega_1,\omega_2) = 0 \text{ for } \begin{cases} |\omega_1| > 3 \\ |\omega_2| > 4 \end{cases}$$

So $\underline{\omega_{1c} = 3 \;,\; \omega_{2c} = 4}$ $\quad \therefore \omega_1^{nyq} = 6 \;,\; \omega_2^{nyq} = 8$

We sample the image at $\Delta x = 0.4\pi = \Delta y$; sampling frequencies are $\therefore \;\Omega_1 = \dfrac{2\pi}{\Delta x} \;,\; \Omega_2 = \dfrac{2\pi}{\Delta y}$

$\therefore \;\Omega_1 = \Omega_2 = \dfrac{2\pi}{0.4\pi} = 5$ which is less than the Nyquist rates ($\omega_1^{nyq} = 6, \;\; \omega_2^{nyq} = 8$)

We know that the FT of a sampled image, $g_s(x,y)$, is given by $G_s(\omega_1,\omega_2)$ where

$$G_s(\omega_1,\omega_2) = \frac{1}{\Delta_x \Delta_y} \sum_{p_1=-\infty}^{\infty} \sum_{p_2=-\infty}^{\infty} G\left(\omega_1 - p_1\Omega_1 \,,\, \omega_2 - p_2\Omega_2\right)$$

where $\Omega_i = \dfrac{2\pi}{\Delta i}$ and $G(\omega_1,\omega_2)$ is FT of $g(x,y)$

∴ In our case:

$$F_s(\omega_1,\omega_2) = \frac{4\pi^2}{(0.4\pi)^2}\sum_{p_1}\sum_{p_2} \delta(\omega_1-3-5p_1,\omega_2-4-5p_2$$

$$+\ \delta(\omega_1+3-5p_1,\omega_2+4-5p$$

$$\therefore F_s(\omega_1,\omega_2) = 25\sum\sum_{p_1,p_2=-\infty}^{\infty} \delta(\omega_1-3-5p_1,\omega_2-4-5p_2)$$

$$+\ \delta(\omega_1+3-5p_1,\omega_2+4-5p_2)$$

as given

We now low-pass filter the sampled image with

$$H(\omega_1,\omega_2) = \begin{cases} \frac{4\pi^2}{25} & |\omega_1|,|\omega_2| < 2.5 \\ 0 & \text{otherwise} \end{cases}$$

Let $\bar{F}(\omega_1,\omega_2) = H(\omega_1,\omega_2)F_s(\omega_1,\omega_2)$, then

$$\bar{F} = 4\pi^2\{\delta(\omega_1-2,\omega_2-1)+\delta(\omega_1+2,\omega_2+1)\}$$

since only $\{p_1=1, p_2=1\}$ and $\{p_1=-1, p_2=-1\}$ will pick out frequencies in the given range.

But we then know (from our previous result) that the IFT of this must be

$$\tilde{f}(x,y) = 2\cos(2x+y)$$

$\Rightarrow \underline{\alpha = 2 \quad \beta = 1}$

3. a) If $\underline{x}$ is an n-point column vector

$$\underline{y} = T\,\underline{x}$$ is the n-point DCT of $\underline{x}$

If X is an $n \times n$ matrix

$$Y_1 = T\,X$$ is a matrix whose columns are the n-pt DCTs of the columns of X.

Similarly $Y = Y_1 T^T$ is a matrix whose rows are the n-pt DCTs of the rows of Y_1. This is the 2-D DCT of X.

$$\therefore \quad Y = \underline{\underline{T X T^T}}$$

is the 2-D DCT of X.

Since T is orthonormal, $T^{-1} = T^T$

$$\therefore \quad T^T Y T = T^{-1} T X T^T T^{-T} = X$$

So to invert the 2-D DCT, we compute

$$X = \underline{\underline{T^T Y T}}$$

b) If the rows of T are $\underline{t}_i$ for $i = 1 \to n$, then the operation $T^T Y T$ may be split into a linear sum of basis functions – one for each ~~non zero~~ element of Y.

$$\text{Hence } X = \sum_{i=1}^{n} \sum_{k=1}^{n} \underline{t}_i^T \, y_{ki} \, \underline{t}_k = \sum_{i=1}^{n} \sum_{k=1}^{n} y_{ki} \, \underline{t}_i^T \underline{t}_k$$

3. b) (cont) In this case there are only 4 non-zero elements in Y, so only 4 terms from the summation are required. Let us call these 4 components of X, $X_1 \cdots X_4$.

$$\therefore \quad X_1 = 30 \,.\, \underline{t}_1^T \underline{t}_1 \qquad k=1,\ i=1 \qquad \underline{\underline{n=8}}$$

$$X_2 = 9 \,.\, \underline{t}_2^T \underline{t}_1 \qquad k=1,\ i=2$$

$$X_3 = -5 \,.\, \underline{t}_1^T . \underline{t}_3 \qquad k=3,\ i=1$$

$$X_4 = 3 \,,\, \underline{t}_3^T . \underline{t}_2 \qquad k=2,\ i=3$$

The row vectors $\underline{t}_k$ are as given in part (a) of the question with elements $[t_{k1} \cdots t_{kn}]$.

In X_1, all elements are: $x_{1,k,i} = 30 . \sqrt{\frac{1}{8}} . \sqrt{\frac{1}{8}} = \underline{\underline{\frac{30}{8}}}$

In X_2, $x_{2,k,i} =$ ~~$9 . \sqrt{\frac{2}{8}} \cos(\pi$~~ $9 . t_{2,k} . t_{1,i}$

$$= 9 . \sqrt{\frac{2}{8}} \cos\left(\frac{\pi(2k-1)(2-1)}{16}\right) . \sqrt{\frac{1}{8}}$$

$$= \underline{\underline{\frac{9}{4\sqrt{2}} \cos\left(\frac{\pi(2k-1)}{16}\right)}}$$

In X_3, $x_{3,k,i} = -5 . t_{1,k} . t_{3,i}$

$$= -5 . \sqrt{\frac{1}{8}} . \sqrt{\frac{2}{8}} \cos\left(\frac{\pi(2i-1)(3-1)}{16}\right)$$

$$= \underline{\underline{\frac{-5}{4\sqrt{2}} \cos\left(\frac{\pi(2i-1)}{8}\right)}}$$

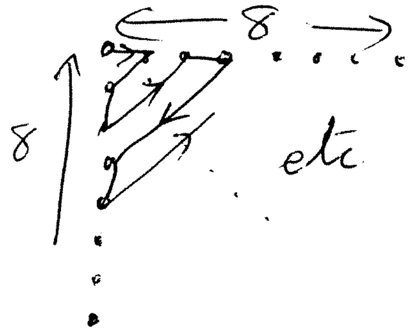
In X_4, $x_{4,k,i} = 3 . t_{3,k} . t_{2,i}$

$$= 3 . \sqrt{\frac{2}{8}} \cos\left(\frac{\pi(2k-1).(3-1)}{16}\right) . \sqrt{\frac{2}{8}} \cos\left(\frac{\pi(2i-1)(2-1)}{16}\right)$$

$$= \underline{\underline{\frac{3}{4} \cos\left(\frac{\pi(2k-1)}{8}\right) \cos\left(\frac{\pi(2i-1)}{16}\right)}}$$

$$x_{k,i} = \underline{\underline{x_{1,k,i} + x_{2,k,i} + x_{3,k,i} + x_{4,k,i}}}$$

3. (c) The 8 × 8 block of coefs is first scanned into a 64-element 1-D vector using a zig-zag scan



Run is used to code the no. of zero samples between each non-zero element in the 1-D vector.

Size is used to code $\lfloor \log_2(\text{magnitude}) \rfloor$ of each non-zero element. ← floor() operator.

Additional bits are used to code the polarity of each non-zero element, and its exact integer magnitude with the permitted range of Size.

The first term ~~of the list~~ of the 1-D vector is the D.C. coefficients & this is always coded using just Size and Additional bits. Hence the given matrix produces the following list:

Value	Run	Size	Add. Bits
30	—	5	11110 ~~00010~~ (2nd term from ~~left~~ right)
9	0	4	1001
−5	1	3	~~101~~ 010
3	3	2	11
End-of-block	0	0	—

3 d) A straight Huffman code for the coefficient values would be very inefficient because the probability of '0' is so high ($\frac{60}{64}$ in this example). By using Run-length coding the '0' state is split into many less probable events which code many 0's in a row with a single codeword each time. This is much more efficient, especially if an 'End-of-block' codeword is used after the last non-zero coef.

Combining <u>Run</u> with <u>Size</u> into a 2-D Huffman code takes advantage of the natural correlations between length of run and expected size of the next coef. This gives further efficiency gains & makes the code well adapted to handle widely varying levels of image activity.

Using <u>Size</u> with <u>Additional Bits</u> allows the 2-D Huffman code to be only modest in size, and the Add. bits do not need Huffman coding, since all states for these (given <u>Size</u>) have similar probabilities.

4. a) $\alpha = \sinh\left(\frac{q}{\sqrt{2}\,\sigma}\right) \qquad r = e^{-\sqrt{2}\,q/\sigma}$

$$p_0 = 1 - \sqrt{r}$$

$$p_k = \alpha\, r^{|k|} \quad \text{for } k \neq 0$$

$$H = \cancel{p_0} - \sum_{k=-\infty}^{\infty} p_k \log_2 p_k = -p_0 \log_2 p_0 - 2\sum_{k=1}^{\infty} p_k \log_2 p_k$$

(due to symmetry of p_k about 0.)

$$\sum_{k=1}^{\infty} p_k \log_2 p_k = \sum_{k=1}^{\infty} \alpha r^k \log_2(\alpha r^k) = \sum_{k=1}^{\infty} \alpha r^k (\log_2 \alpha + k \log_2 r)$$

$$= \alpha(\log_2 \alpha)\sum_{k=1}^{\infty} r^k + \alpha(\log_2 r)\sum_{k=1}^{\infty} k r^k$$

$$= \alpha(\log_2 \alpha)\,\frac{r}{1-r} + \alpha(\log_2 r)\,\frac{r}{(1-r)^2} \qquad \left(\begin{matrix}\text{standard}\\ \text{results}\end{matrix}\right)$$

$$= \frac{\alpha r}{1-r}\left[\log_2 \alpha + \frac{\log_2 r}{1-r}\right]$$

$$\therefore H = \underline{\underline{-(1-\sqrt{r})\log_2(1-\sqrt{r}) - \frac{2\alpha r}{1-r}\left[\log_2 \alpha + \frac{\log_2 r}{1-r}\right]}}$$

b)

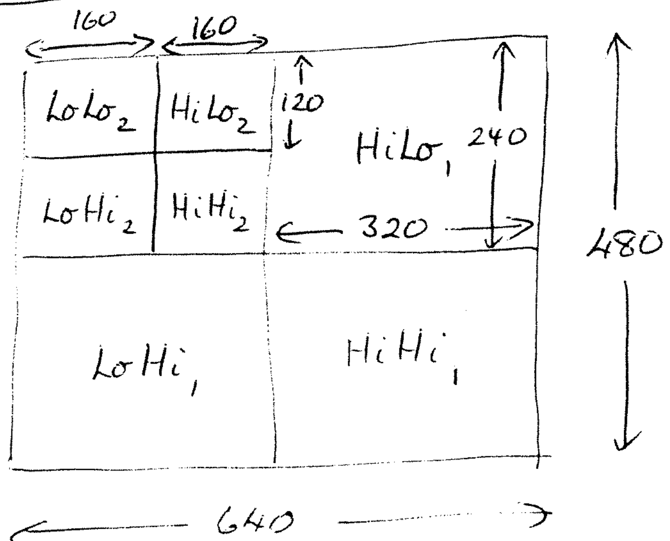


Image is first decomposed into 4 subimages at level 1, of size 320 × 240, & then the $LoLo_1$ subimage is ~~&~~ again decomposed into the 4 level-2 subimages $LoLo_2 \ldots HiHi_2$ as shown, each of size 160 × 120.

Lo & Hi refer to Low- & High-pass filtering, both horizontally & vertically.

4. b) (cont) Highpass filters pick out edges normal to the direction of filtering and corners. Lowpass filters produce blurring in the direction of filtering. HiLo subbands are highpass filtered in the horizontal direction & so pick out vertical edges, while LoHi subbands pick out horizontal edges. Diagonal edges tend to get picked out by both HiLo & LoHi filters. Corners and other random fine textures are picked out by the HiHi subbands. The LoLo subband produces a blurred version of the original image.

c) At level 1, the energies split between the subbands as follows (assuming 95% & 5% split throughout):

$HiHi_1 = 0.05 \times 0.05 = 0.0025$

$HiLo_1$ & $LoHi_1 = 0.95 \times 0.05 = 0.0475$ each

$LoLo_1 = 0.95 \times 0.95 = 0.9025$

At level 2:

$HiHi_2 = 0.9025 \times 0.0025 = 0.002256$

$HiLo_2$ & $LoHi_2 = 0.9025 \times 0.0475 = 0.04287$

$LoLo_2 = 0.9025^2 = 0.8145$

The std. dev σ of a subimage of size $m \times n$ with energy E is: $\sigma = \sqrt{\frac{E}{mn}}$ since $E = mn\sigma^2$

$\therefore$ Energy of 640×480 input image $= 640.480.50^2 = 7.68.10^8$

(cont)

4. c) For $HiHi_1$ image :

$$\sigma = \sqrt{\frac{640.480.50^2 \times 0.0025}{320.240}} = 2.50.\sqrt{0.0025} = 5$$

$q = 20$

$$\therefore \alpha = \sinh\left(\frac{20}{\sqrt{2}.5}\right) = 8.4299$$

$$r = e^{-\sqrt{2}.20/5} = 0.003493$$

$H = 0.0827 + 0.3023 = 0.3850$ bit/pel

(using formula from part (a)). No of bits $= H \times 320 \times 240$

$= 29,567$

For $HiLo_1$ + $LoHi_1$ images

$$\sigma = 2.50.\sqrt{0.0475} = 21.794$$

$\therefore \alpha = 0.6954$

$r = 0.2731$

$H = 0.5093 + 1.6201 = 2.1294$ bit/pel

$\therefore$ No. of bits $= 163,538$ per subimage

For $HiHi_2$ image

$$\sigma = \sqrt{\frac{640.480.50^2 \times 0.002256}{160.120}} = 4.50.\sqrt{0.002256}$$

$= 9.4995$

$\therefore \alpha = 2.1029$

$r = 0.05092$

$H = 0.2857 + 0.7794 = 1.0651$ bit/pel

$\therefore$ No. of bits $= H.160.120 = 20,449$

4(c) (cont) For ~~Lo~~ $HiLo_2$ & $LoHi_2$ images

$\sigma = 4.50\sqrt{0.04287} = 41.410$

$\therefore \alpha = 0.3482$

$r = 0.5051$

$H = 0.5177 + 2.4967 = 3.0144$ bit/pel

$\therefore$ No of bit = 57,877 per subimage

For $LoLo_2$ image

$\sigma = 4.50.\sqrt{0.8145} = 180.50$

$\therefore \alpha = 0.07843$

$r = 0.8550$

$H = 0.2811 + 4.8369 = 5.1180$ bit/pel

No of bits = 98,266

$\therefore$ Total bits for all subbands

$$n_{TOT} = 29{,}567 + 2 \times 163{,}538 + 20{,}449 + 2 \times 57{,}877 + 98{,}266 = \underline{\underline{591{,}112}} \text{ bits}$$

$$\text{Mean bit rate} = \frac{n_{TOT}}{640 \times 480} = \underline{\underline{1.924}} \text{ bit/pel.}$$

(d) The number of bits ~~per pel~~ used for the $LoLo_2$ subimage is 98,266, which is ~17% of the total. If split down further the entropies of the level 3 subbands would be at least as big as those at level 2 (probably approx 1 bit/pel larger) so the average bit/pel for all level 3 subbands would be at least

4 (d) (cont) $\frac{1}{4}\left(1.06 + 2\times 3.01 + 5.12\right) = 3.05$ bit/pel

& probably more like 4 bit/pel. Hence we might save ~ 1 bit/pel in the $LoLo_2$ subband (from its current value of 5.12 bit/pel) & this would save

$1 \times 160 \times 120$ bits $\doteq$ 19,000 bits.

This is ~ 3% of the total for the image.

~~Hence one further~~
Additional levels will save much less.

So one further level of decomposition is probably worthwhile.