Solutions: 4F8 2012

ENGINEERING TRIPOS PART IIB

Monday 7 May 2012 9 to 10.30

Module 4F8

IMAGE PROCESSING AND IMAGE CODING

Version: JL02

1 (a) For rectangular sampling, our sampling function $s(u_1,u_2)$ is given by

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

where Δ_i is the sampling interval in the u_i direction.

For diamond sampling, our sampling function $s(u_1,u_2)$ is given by $s=s_1+s_2$ with

$$s_1(u_1,u_2)=\sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty}\delta[u_1-2n_1\Delta_1,u_2-2n_2\Delta_2]$$

and

$$s_2(u_1,u_2)=\sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty}\delta[u_1-(2n_1+1)\Delta_1,u_2-(2n_2+1)\Delta_2]$$

where Δ_1 and Δ_2 are as shown in fig 1

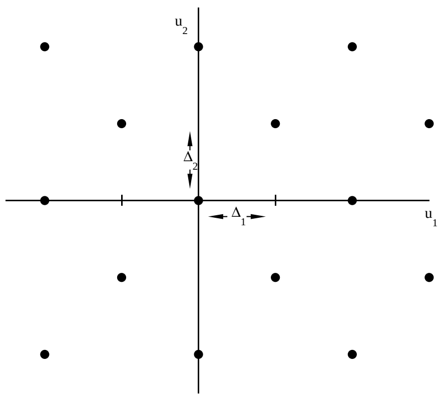

Fig. 1

[15%]

(b) For the rectangular sampling function s in part (a), we can write it as a Fourier series as it is periodic in both the u_1 and u_2 directions:

$$s(u_1,u_2)=\sum_{p_1=-\infty}^{\infty}\sum_{p_2=-\infty}^{\infty}c(p_1,p_2)\,e^{j(p_1\Omega_1u_1+p_2\Omega_2u_2)}$$

where $\Omega_1=\frac{2\pi}{\Delta_1}$ and $\Omega_2=\frac{2\pi}{\Delta_2}$

(cont.

To find the $c(p_1,p_2)$ we perform the standard integral

$$c(p_1,p_2)=\frac{1}{\Delta_1\,\Delta_2}\int_{-\frac{\Delta_2}{2}}^{\frac{\Delta_2}{2}}\int_{-\frac{\Delta_1}{2}}^{\frac{\Delta_1}{2}} s(u_1,u_2)\,e^{-j(p_1\Omega_1u_1+p_2\Omega_2u_2)}\,du_1\,du_2$$

which evaluates to give $c(p_1,p_2)=\frac{1}{\Delta_1\Delta_2}$ for all p_1,p_2.
We can then express the sampled image as

$$g_s(u_1,u_2)=g(u_1,u_2)\,\frac{1}{\Delta_1\,\Delta_2}\sum_{p_1=-\infty}^{\infty}\sum_{p_2=-\infty}^{\infty}\mathrm{e}^{j(p_1\Omega_1u_1+p_2\Omega_2u_2)}$$

and using the frequency shift theorem to take the Fourier transform, we get

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

From this we can see that the spectrum of the sampled 2D signal is the periodic repetition of the spectrum of the unsampled signal at every grid/sampling point. Thus, if our sampling intervals are less than twice the largest directional frequencies $(\Omega_{B1},\Omega_{B2})$, we will have overlap of the spectra and hence aliasing, as illustrated in fig. 2:

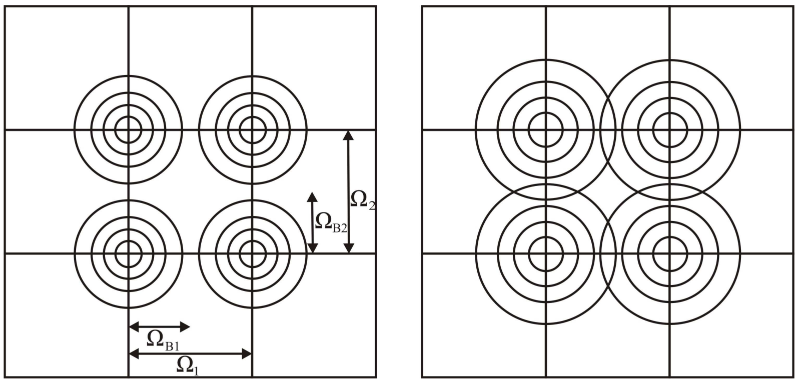

Fig. 2

[30%]

(c) The side of the hexagonal grid is d, so we can form the whole grid by summing 4 rectangular grids as follows:

(i) $s_1(u_1,u_2)$ is centred on the origin and has a horizontal spacing, Δ_h, of $3d$ and a vertical spacing, Δ_v, of $d\sqrt{3}$ – this grid is picked out with large circles in fig 3.

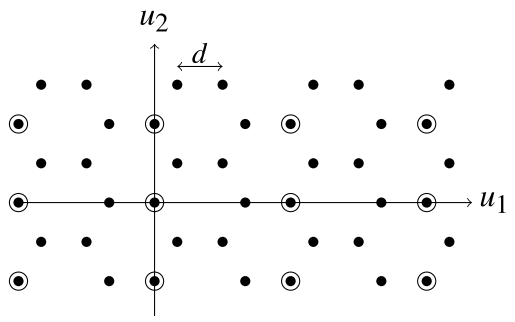

Fig. 3

$s_2(u_1,u_2)$ is displaced by $(2d,0)$ from the origin and has a horizontal spacing, Δ_h, of $3d$ and a vertical spacing, Δ_v, of $d\sqrt{3}$ – this grid is picked out with large circles in fig 4.

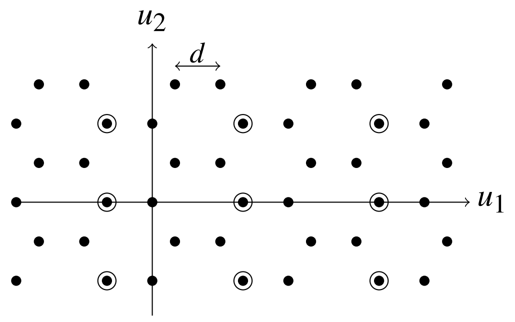

Fig. 4

$s_3(u_1,u_2)$ is displaced by $(3d/2, d\sqrt{3}/2)$ from the origin and has a horizontal spacing, Δ_h, of $3d$ and a vertical spacing, Δ_v, of $d\sqrt{3}$ – this grid is picked out with large circles in fig 5.

$s_4(u_1,u_2)$ is displaced by $(d/2, d\sqrt{3}/2)$ from the origin and has a horizontal spacing, Δ_h, of $3d$ and a vertical spacing, Δ_v, of $d\sqrt{3}$ – this grid is picked out with large circles in fig 6.

The sampled signal is therefore

$$g_s(u_1,u_2) = g(u_1,u_2)(s_1+s_2+s_3+s_4)$$

Since s_2, s_3, s_4 are just shifted versions of s_1, if s_1 has Fourier

(cont.

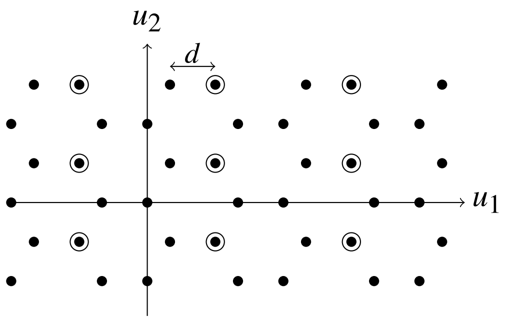

Fig. 5

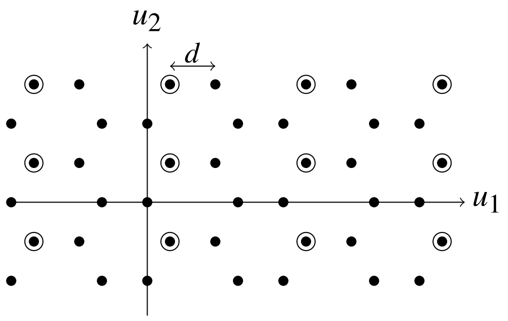

Fig. 6

coefficients $c_1(p_1,p_2)$, we know that s will have Fourier coefficients, c_s, given (via the shift theorem) by

$$c_s(p_1,p_2) = c_1(p_1,p_2)\left(1+\mathrm{e}^{-j\frac{4}{3}\pi p_1}+\mathrm{e}^{-j(\pi p_1+\pi p_2)}+\mathrm{e}^{-j(\frac{\pi}{3}p_1+\pi p_2)}\right)$$

The above factors are obtained from the fact that $s_1(u_1-a_1,u_2-a_2)$ will acquire a factor of $\mathrm{e}^{-j(a_1p_1\Omega_1+a_2p_2\Omega_2)}$ relative to the Fourier coefficients of $s_1(u_1,u_2)$. As before, $c_1(p_1,p_2)=\frac{1}{3\sqrt{3}d^2}$. [30%]

(ii) From part (b) we know that we can write the FT of the sampled signal G_s as

$$G_s(\omega_1,\omega_2) = \frac{1}{3\sqrt{3}d^2}\sum_{p_1=-\infty}^{\infty}\sum_{p_2=-\infty}^{\infty}\left[1+\mathrm{e}^{-j\frac{4}{3}\pi p_1}+\mathrm{e}^{-j(\pi p_1+\pi p_2)}+\mathrm{e}^{-j(\frac{\pi}{3}p_1+\pi p_2)}\right]G(\omega_1-p_1\Omega_1,\omega_2-p_2\Omega_2)$$

since $\Delta_1=3d$, $\Delta_2=d\sqrt{3}$. As usual, $\Omega_1=2\pi/\Delta_1$ and $\Omega_2=2\pi/\Delta_2$.

Thus, $f(d)=\frac{1}{3\sqrt{3}d^2}$, $\alpha_1=\frac{2\pi}{3d}$, $\alpha_2=\frac{2\pi}{d\sqrt{3}}$ and $Z=\mathrm{e}^{-j\frac{4}{3}\pi p_1}+\mathrm{e}^{-j(\pi p_1+\pi p_2)}+\mathrm{e}^{-j(\frac{\pi}{3}p_1+\pi p_2)}$. [25%]

2 (a) Histogram equalisation:

(i) The probability of X lying between x and $x+\delta x$, must be the same as the probability of $g(X)=Y$ lying between y and $y+\delta y$. Therefore

$$Pr\{x \leq X < x+dx\} = Pr\{y \leq Y < y+dy\}$$

so that

$$p_Y(y)dy = p_X(x)dx \implies p_Y(y) = \frac{p_X(x)}{\frac{dy}{dx}}$$

If $p_Y(y)$ is to be constant over the greylevel range 0 to L, we have $p_Y(y) = 1/L$, which leads to

$$\frac{dy}{dx} = Lp_X(x) \implies y = g(x) = \int_0^x Lp_X(u)du$$

If the input image is quantised into N_L levels x_i spaced by Δx_i, we can approximate the above integral by a sum

$$y_k = \sum_{i=1}^{k} Lp_X(x_i)\Delta x_i \text{ for } k = 1,..,N_L$$

We now approximate $p_X(x_i)\Delta x_i$ by referring to the image histogram and counting the number of occurrences, N_i, in x_i to $x_i+\Delta x_i$, so that $p_X(x_i)\Delta x_i = N_i/(N \times M)$. The mapping rule therefore becomes

$$y_k = \sum_{i=1}^{k} L\frac{N_i}{NM}$$

[20%]

(ii) The image histogram is shown in fig. 7.

We see that the greylevels are concentrated towards the ‘black’ and ‘white’ regions, with very little in the middle ‘grey’ area. [10%]

(iii) To find the set of transformed values set up a table which tabulates N_i, $C_i = \sum_{j=1}^{i} N_j$ and y_i. Note that in this case $N \times M = 36$ and $L = 6$, so that $L/(NM) = 1/6$

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
N_i	6	4	3	2	1	1	1	0	0	0	0	1	1	1	2	3	4	6
C_i	6	10	13	15	16	17	18	18	18	18	18	19	20	21	23	26	30	36
$C_i/6$	1	1.67	2.17	2.5	2.67	2.83	3	3	3	3	3	3.17	3.33	3.5	3.83	4.33	5	6
$y(i)$	1	2	3	3	3	3	3	3	3	3	3	4	4	4	4	5	5	6

The new image histogram is shown in fig. 8.

As we can see, the process has done a reasonable job of spreading out the distribution of pixels across the chosen greylevels. Reducing the number of levels to 6 has made it easier to produce a more uniform distribution – but at the cost of reducing the dynamic range in the image, which would not produce an aesthetically pleasing effect. [20%]

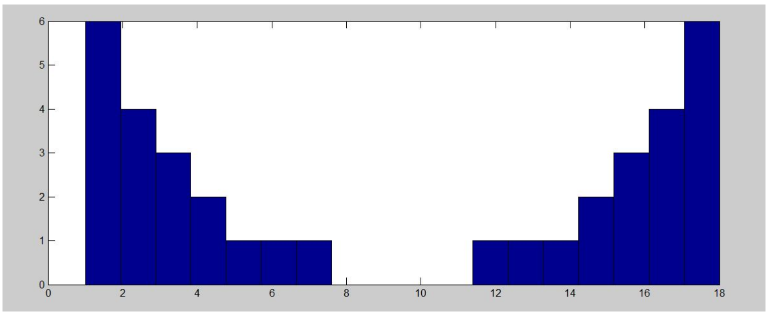

Fig. 7

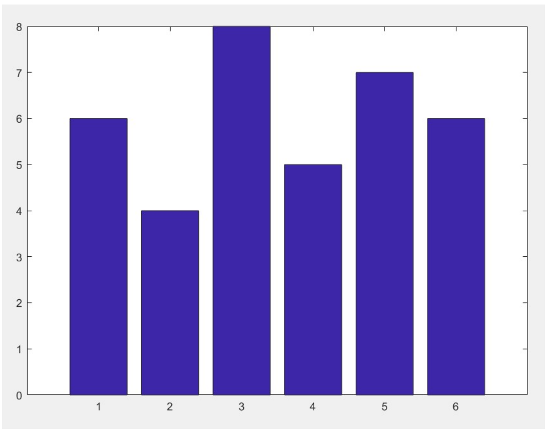

Fig. 8

(b) Wiener filter etc

(i) In the expression

$$P(\mathbf{x}|\mathbf{y}) \propto \mathrm{e}^{-\frac{1}{2}[(\mathbf{y}-L\mathbf{x})^T N^{-1}(\mathbf{y}-L\mathbf{x})+\mathbf{x}^T C^{-1}\mathbf{x}]}$$

The prior is $e^{\mathbf{x}^T C^{-1}\mathbf{x}}$ and represents a scenario where $\mathbf{x}$ is a Gaussian random variable described by a known covariance matrix $C = E[\mathbf{x}\mathbf{x}^T]$.

The likelihood is $P(\mathbf{y}|\mathbf{x}) = e^{(\mathbf{y}-L\mathbf{x})^T N^{-1}(\mathbf{y}-L\mathbf{x})}$, where we are making the assumption that our noise is Gaussian and described by the covariance matrix $N = E[\mathbf{d}\mathbf{d}^T]$, where $\mathbf{y} = L\mathbf{x} + \mathbf{d}$. [10%]

(ii) This is effectively an exercise in completing the square:

$$(\mathbf{x}-\hat{\mathbf{x}})^T M^{-1}(\mathbf{x}-\hat{\mathbf{x}}) = \mathbf{x}^T M^{-1}\mathbf{x} - \mathbf{x}^T M^{-1}\hat{\mathbf{x}} - \hat{\mathbf{x}}^T M^{-1}\mathbf{x} + \hat{\mathbf{x}}^T M^{-1}\hat{\mathbf{x}}$$

Equate the quadratic term in $\mathbf{x}$ between the two expressions for the posterior:

$$\mathbf{x}^T M^{-1}\mathbf{x} = \mathbf{x}^T L^T N^{-1} L\mathbf{x} + \mathbf{x}^T C^{-1}\mathbf{x} = \mathbf{x}^T \left(L^T N^{-1} L + C^{-1}\right)\mathbf{x}$$

So that

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

If the Wiener filter is given by $\hat{\mathbf{x}} = W\mathbf{y}$, then equating the terms in $\mathbf{x}^T(..)\mathbf{y}$ gives:

$$M^{-1}W = L^T N^{-1}$$

[Note that equating terms in $\mathbf{y}^T(..)\mathbf{x}$ gives $N^{-1}L = W^T M^{-1}$, which is equivalent to the above due to the fact that N and C are covariance matrices and therefore $N^{-1} = (N^{-1})^T$, $M^{-1} = (M^{-1})^T$.]

Therefore, $W = ML^T N^{-1}$, so that

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

[30%]

(iii) Since the world is not as simple as the Wiener filter assumes, we are forced to consider alternative *priors*. One such prior which has been widely and successfully used is the *entropy prior*.

This produces the Maximum Entropy Method (MEM) which is applied to positive, additive distibutions (PADS). Let $\mathbf{x}$ be the (true) pixel vector we are trying to estimate, $Pr(\mathbf{x})$ is given by

$$Pr(\mathbf{x}) \propto e^{\alpha S}$$

where one version of the *entropy* S (sometimes known as the *cross entropy*) of the image is given by

$$S(\mathbf{x},\mathbf{m}) = \sum_i \left[x_i - m_i - x_i \ln\left(\frac{x_i}{m_i}\right)\right]$$

where **m** is the *measure* on an image space (*the model*) to which the image **x** defaults in the absence of data. (Can see global maximum of S occurs at $\mathbf{x} = \mathbf{m}$.)

[Other image priors could be discussed] [10%]

3 (a) The block diagram of a basic image coding system is shown in fig 9.

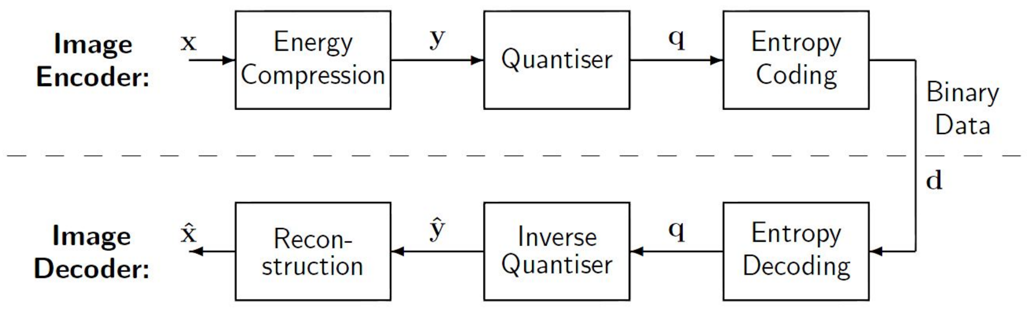

Fig. 9

Encoder:

(i) Energy compression compresses most of the energy of image $\mathbf{x}$ into a small proportion of coefficients in matrix $\mathbf{y}$.

(ii) The quantiser represents $\mathbf{y}$ by integers $\mathbf{q}$ using some pre-defined quantising strategy. Ideally a high proportion of elements of $\mathbf{q}$ should be zero.

(iii) The entropy coder converts the integers of $\mathbf{q}$ into a binary data-stream $\mathbf{d}$, so that all the information in $\mathbf{q}$ is retained in $\mathbf{d}$ while minimising the number of bits in $\mathbf{d}$.

Decoder:

(i) The entropy decoder recovers $\mathbf{q}$ (exactly) from $\mathbf{d}$.

(ii) The inverse quantiser generates $\hat{\mathbf{y}}$ from $\mathbf{q}$, in such a way that the mean quantising error energy of $(\mathbf{y} - \hat{\mathbf{y}})$ is minimised.

(iii) The reconstruction block converts $\hat{\mathbf{y}}$ back into $\hat{\mathbf{x}}$, such that if $\hat{\mathbf{y}} = \mathbf{y}$, then $\hat{\mathbf{x}} = \mathbf{x}$.

[25%]

(b) The two main classes of image transform used in the energy compression and reconstruction blocks of a typical coder are:

•The Discrete Cosine Transform (DCT)

•The Discrete Wavelet Transform (DWT)

The DCT is a block transform which operates in 2D on an $N \times N$ block of pixels. Usually each $N \times N$ block in an image is non-overlapping with other blocks. It is relatively simple and fast to compute, but strongly DCT-compressed images often suffer from *blocking* artifacts caused by significant discontinuities at the boundaries of the $N \times N$ blocks.

The DWT is a filtering-based transform in which convolution allows the transform basis functions to overlap in a smoothly decaying way. Hence *blocking* artifacts are largely avoided. However, the DWT is computationally more complex than the DCT. The DWT is also multi-scale, so the high frequency basis functions are much smaller than the low frequency ones. This helps to minimise visibility of high frequency artifacts as they are localised better than in the DCT. [25%]

(c)

$$T = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Forward transform:

In 1D, $\mathbf{y} = T\mathbf{x}$ where $\mathbf{x}$ and $\mathbf{y}$ are 2×1 vectors.

In 2D, $Y = TXT^T$ where X and Y are 2×2 matrices. Left multiplication by T transforms columns of X and right multiplication by T^T transforms rows of X.

Inverse transform:

In 1D, $\mathbf{x} = T^{-1}\mathbf{y} = T^T\mathbf{y}$ if T is orthonormal.

In 2D, $X = T^{-1}YT = T^TYT$ if T is orthonormal.

The Haar matrix for T is orthonormal because the dot product of any two columns or of any two rows is zero, and the energy of each row or column is unity. It is well known that multiplying a vector or matrix by an orthonormal (or unitary) matrix preserves the energy of the input vector or matrix. This is because

$$\mathbf{y}^T\mathbf{y} = \mathbf{x}^T T^T T\mathbf{x} = \mathbf{x}^T I\mathbf{x} = \mathbf{x}^T\mathbf{x}$$

since $T^TT = I$. [25%]

(d) A 1-level Haar transform of an $N \times M$ image matrix results in four separate $\frac{N}{2} \times \frac{M}{2}$ image matrices after coefficient reordering. These 4 subbands are known as $Hi-Lo$, $Lo-Hi$, $Hi-Hi$ and $Lo-Lo$.

A 2-level Haar transform applies the same process again to the $Lo-Lo$ subband from level 1 and generates 4 subbands of size $\frac{N}{4} \times \frac{M}{4}$, to replace the $\frac{N}{2} \times \frac{M}{2}$ $Lo-Lo$ subband from level 1.

Hence for the 1024×768 image given, we have

Band	Level	Size	Entropy	No. of bits (rounded up)
Hi-Lo	1	512×384	1.2	235930
Lo-Hi	1	512×384	1.2	235930
Hi-Hi	1	512×384	0.8	157287
Hi-Lo	2	256×192	2.5	122880
Lo-Hi	2	256×192	2.5	122880
Hi-Hi	2	256×192	1.8	88474
Lo-Lo	2	256×192	5.6	275252
Total bits				1238633

Hence approx 1.24 Mbits would be needed to code this image (approx 1.58bit/pixel). [25%]

4 (a) A sketch of a two-band analysis/reconstruction filter bank system is shown in figure 10 ((a) shows analysis and (b) shows reconstruction).

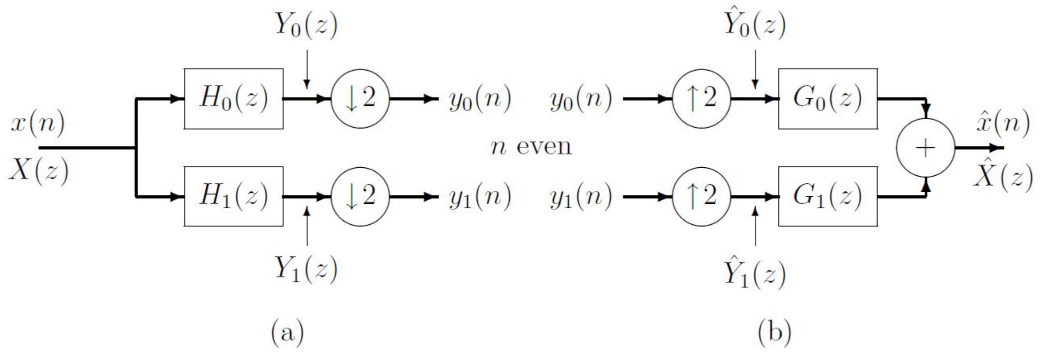

Fig. 10

The downsamplers by 2, omit all samples $y(n)$ when n is odd.

The upsamplers by 2, insert zeros in place of the missing odd-numbered samples. [20%]

(b) We are given that $\hat{y}(n) = y(n)$ when n is even and $\hat{y}(n) = 0$ when n is odd. We know that

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

[20%]

(c) For the filter banks in part (a) we have

$$Y_0(z) = H_0(z)X(z) \quad \text{and} \quad Y_1(z) = H_1(z)X(z)$$
$$\hat{Y}_0(z) = \frac{1}{2}\left[Y_0(z) + Y_0(-z)\right] \quad \text{and} \quad \hat{Y}_1(z) = \frac{1}{2}\left[Y_1(z) + Y_1(-z)\right]$$

and

$$\hat{X}(z) = G_0(z)\hat{Y}_0(z) + G_1(z)\hat{Y}_1(z)$$

Combining these expressions we have:

$$\begin{aligned}
\hat{X}(z) &= \frac{1}{2}G_0(z)\left[H_0(z)X(z) + H_0(-z)X(-z)\right] + \frac{1}{2}G_1(z)\left[H_1(z)X(z) + H_1(-z)X(-z)\right] \\
&= \frac{1}{2}X(z)\left[G_0(z)H_0(z) + G_1(z)H_1(z)\right] + \frac{1}{2}X(-z)\left[G_0(z)H_0(-z) + G_1(z)H_1(-z)\right]
\end{aligned}$$

For antialiasing, the $X(-z)$ term must be zero and so we require that

$$G_0(z)H_0(-z) + G_1(z)H_1(-z) = 0$$

For perfect reconstruction, the $X(z)$ term must be multiplied by unity, so we require that

$$G_0(z)H_0(z) + G_1(z)H_1(z) = 2$$

[25%]

(d) To satisfy the antialiasing condition and make $G_1(z)$ and $H_1(z)$ highpass when $G_0(z)$ and $H_0(z)$ are lowpass, we let

$$G_1(z) = z^k H_0(-z) \quad \text{and} \quad H_1(z) = z^{-k} G_0(-z)$$

with k being an odd integer (usually ± 1).

Then the perfect reconstruction (PR) condition becomes

$$G_0(z)H_0(z) + G_0(-z)H_0(-z) = 2$$

or

$$P(z) + P(-z) = 2$$

since $P = G_0 H_0$.

For symmetric left/right and up/down filter behaviour in images, we usually assume linear-phase filters, so that $p_{-n} = p_n$.

The PR condition causes all odd coefficients p_n in $P(z)$ to be cancelled when it is added to $P(-z)$, so it only constrains the even coefficients.

Hence $p_0 = 1$ and $p_2, p_4, p_6, \ldots$ are all zero. Thus $P(z)$, with symmetry, will be of the form

$$P(z) = \ldots + p_5 z^5 + p_3 z^3 + p_1 z^1 + 1 + p_1 z^{-1} + p_3 z^{-3} + p_5 z^{-5} + \ldots$$

The design process is to find a good set of coefficients, $\{p_1, p_3, p_5, \ldots\}$, such that $P(z)$ is a well-shaped lowpass filter that can be factorised into two lowpass filters $H_0(z)$ and $G_0(z)$.

To ensure symmetry, it is usual to let $Z = \frac{1}{2}(z + z^{-1})$ and write $P(z) = P_t(Z)$, where $P_t(z)$ contains only positive odd powers of Z as well as a constant term of unity. It is usually easier to find the coefficients of P_t than those of P, as there are only half as many of them.

If the factors of $P(z)$ are such that $H_0(z)$ is more smooth than $G_0(z)$, the reconstructed image will tend to contain artifacts that are more visible because of the 'roughness' of G_0. It is therefore better to swap the factors so that G_0 is always smoother than H_0, and the reconstructed image is then as smooth as possible. [35%]

END OF SOLUTIONS