CS 422/522: Digital Image Processing
Homework 5 (Fall '13)

1 Theory

1. Let $f(t) = e^{-\pi t^2}$, $f''(t) = 2\pi e^{-\pi t^2}(2\pi t^2 - 1)$, and $g(t) = at + b$. Prove or disprove the following:
$$\langle f'', g \rangle = 0$$
for all a and b.

2. The n-th moment of Ψ is defined to be $M_n\{\Psi\} = \int_{-\infty}^{\infty} t^n \Psi(t) dt$. Let $f(t) = e^{-\pi t^2}$, $f'(t) = -2\pi t e^{-\pi t^2}$, and $f''(t) = 2\pi e^{-\pi t^2}(2\pi t^2 - 1)$. Prove the following:
   (a) $M_0\{f'\} = 0$.
   (b) $M_0\{f''\} = M_1\{f''\} = 0$.

3. The six vectors, $\mathbf{f}_1 = \begin{bmatrix} \cos(\pi/3) & \sin(\pi/3) \end{bmatrix}^T$, $\mathbf{f}_2 = \begin{bmatrix} \cos(\pi/3) & -\sin(\pi/3) \end{bmatrix}^T$, $\mathbf{f}_3 = \begin{bmatrix} -1 & 0 \end{bmatrix}^T$, $\mathbf{f}_4 = \begin{bmatrix} -\cos(\pi/3) & -\sin(\pi/3) \end{bmatrix}^T$, $\mathbf{f}_5 = \begin{bmatrix} -\cos(\pi/3) & \sin(\pi/3) \end{bmatrix}^T$, and $\mathbf{f}_6 = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$ form a frame $\mathcal{F}$ for $\mathbb{R}^2$. Draw the frame.
   (a) Give two representations for the vector, $\mathbf{x} = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$, in $\mathcal{F}$.
   (b) Prove that $\mathbf{x}$ has an infinite number of representations in $\mathcal{F}$.
   (c) Give a matrix which transforms any representation of a vector in $\mathcal{F}$ into its representation in the standard basis for $\mathbb{R}^2$.
   (d) Give a matrix which transforms a representation of any vector in the standard basis for $\mathbb{R}^2$ into its representation in $\mathcal{F}$.

4. The continuous representation of the Haar highpass filter is
$$h_1(t) = \frac{1}{2}[\delta(t + \Delta t) - \delta(t - \Delta t)].$$
The continuous representation of the Haar lowpass filter is
$$h_0(t) = \frac{1}{2}[\delta(t + \Delta t) + \delta(t - \Delta t)].$$

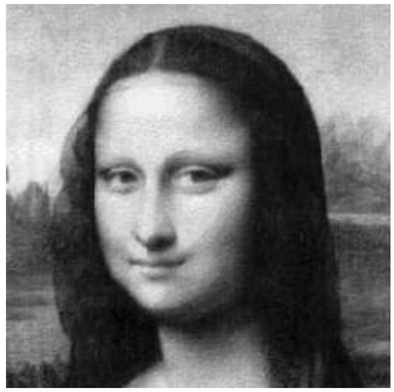

Figure 1: The Mona Lisa.

Prove that

$$H_0(s)H_0^*(s) + H_1(s)H_1^*(s) = 1$$

where $H_0(s)$ and $H_1(s)$ are the Fourier transforms of $h_0(t)$ and $h_1(t)$.

5. The $N+1$ channel Haar transform matrix can be recursively defined as follows:

$$\mathbf{H}_N = \frac{1}{\sqrt{2}} \left[\begin{array}{c|c} \mathbf{I}_{N-1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{H}_{N-1} \end{array} \right] \left[\begin{array}{c} \mathbf{U}_N \\ \hline \mathbf{L}_N \end{array} \right]$$

where $\mathbf{U}_N$ convolves a length 2^N signal with the Haar highpass filter followed by downsampling, $\mathbf{L}_N$ convolves a length 2^N signal with the Haar lowpass filter followed by downsampling, $\mathbf{I}_N$ is the identity matrix of size $2^N \times 2^N$ and

$$\mathbf{H}_1 = \frac{1}{\sqrt{2}} \left[\begin{array}{c} \mathbf{U}_1 \\ \hline \mathbf{L}_1 \end{array} \right] = \frac{1}{\sqrt{2}} \left[\begin{array}{cc} 1 & -1 \\ 1 & 1 \end{array} \right].$$

(a) Using the above definitions, derive expressions for $\mathbf{H}_3$ and $\mathbf{H}_3^{-1}$.

(b) Compute the Haar transform of the vector $\left[\begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \right]^{\mathrm{T}}$.

2 Practice

1. Write a function *reduce* which takes a square image, *im*, of size 2^k for integer k as input, and convolves the rows and columns of *im* with the kernel,

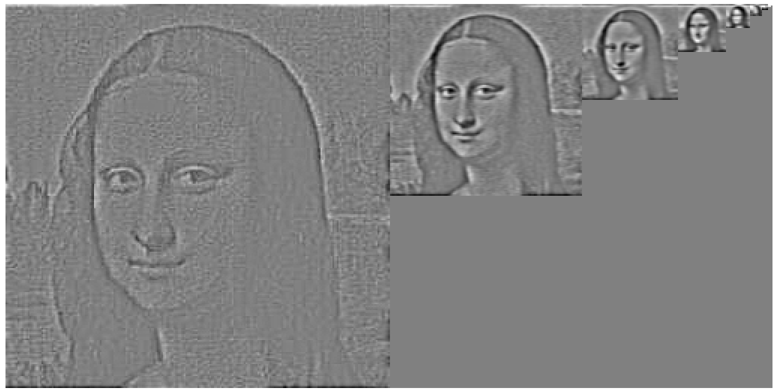

Figure 2: Laplacian pyramid transform of the Mona Lisa.

$\frac{1}{20}\begin{bmatrix} 1 & 5 & 8 & 5 & 1 \end{bmatrix}^{\mathrm{T}}$, and then downsamples it. Demonstrate your function on an image of your choice.

2. Write a function *project* which takes a square image, *im*, of size 2^k for integer k as input, upsamples it and then convolves the rows and columns of the upsampled image with the kernel, $\frac{1}{10}\begin{bmatrix} 1 & 5 & 8 & 5 & 1 \end{bmatrix}^{\mathrm{T}}$. Demonstrate your function on an image of your choice.

3. Write a function *laplacian-pyramid* which takes a square image, *im*, of size 2^k for integer k as input, and returns a list of k images representing the k levels of a two-dimensional Laplacian pyramid transform of *im*.

4. Write a function *inverse-laplacian-pyramid* which takes a list, *ls*, of k images representing the k levels of a two-dimensional Laplacian pyramid transform of a square image of size 2^k for integer k as input, and returns the reconstructed image. Demonstrate your function's ability to invert a Laplacian pyramid you compute with *laplacian-pyramid* for an image of your choice.

5. Write a function *display-laplacian-pyramid* which takes a list, *ls*, of k images representing the k levels of a two-dimensional Laplacian pyramid transform of an image of size 2^k for integer k as input, and returns an image depicting the Laplacian pyramid using the recursive scheme shown in Figure 2. Demonstrate your function on an image of your choice. Note: The images representing the Laplacian pyramid levels must each be normalized

to the range [0-255] with grey level 0 mapped to grey level 128 prior to constructing the display.

6. Write a function *daubechies4* which takes a square image, *im*, of size 2^k for integer k as input, and returns a list of length four representing the two-dimensional $x - y$ separable Daubechies 4 wavelet transform of *im*. The last three elements of the list are the level 1 wavelet subbands and the first element is (itself) a list of length four (recursively) representing levels 2 through k of the wavelet transform.

7. Write a function *inverse-daubechies4* which takes a list of length four representing a two-dimensional $x - y$ separable Daubechies 4 wavelet transform of a square image, *im*, of size 2^k for integer k as input, and returns the reconstructed image. Demonstrate your function's ability to invert a wavelet transform you compute with *daubechies4* for an image of your choice.

8. Write a function *display-wavelet-transform* which takes a list of length four representing a two-dimensional $x - y$ separable Daubechies 4 wavelet transform of a square image, *im*, of size 2^k for integer k as input, and returns an image depicting the wavelet transform using the recursive scheme shown in Figure 3. Demonstrate your function on an image of your choice. Note: The images representing the wavelet subbands must each be normalized to the range [0-255] with grey level 0 mapped to grey level 128 prior to constructing the display.

9. Write a function *denoise-color-image* which takes a color image, *cim*, as input and returns a denoised color-image computed by:

   - Converting *cim* to HSI.
   - Computing the Daubechies 4 wavelet transform of the saturation (S) and intensity (I) components.
   - Soft-thresholding the the S and I wavelet subbands.
   - Computing the inverse Daubechies 4 wavelet transform.
   - Converting the HSI representation back to RGB.

10. Find a noisy color image on the internet, *i.e.*, an image which has been degraded by aliasing from downsampling or contains visible JPEG blocking, film grain, or other additive noise. If you cannot find a suitable image, then start with a high quality color image and degrade it yourself, *e.g.,* using *xv*.

Figure 3: (a) Bill Clinton. (b) Recursively displayed two-dimensional $x-y$ separable Daubechies 4 wavelet transform.

11. Use *denoise-color-image* to denoise your image. Use a threshold for shrinkage which you judge to be optimum and one which is too large. Show your results for both thresholds.