T-61.3010 Digital Signal Processing and Filtering

(v. 1.0, 13.3.2009), Matlab #5 (7.-17.4.2009)

Registration in WebOodi. Bring your own **headphones** if you have. The assistant will guide you through the exercises, but you may go on your own speed. Feel free to ask the assistant, if you have troubles. You can also consult `http://www.cis.hut.fi/Opinnot/T-61.3010/how_to_start_with_matlab.shtml` or `kuinka_aloitan_matlabin.shtml`.

Getting started: In **Windows** just click `Programs - Matlab`. Write down the code into separate files in your working directory (e.g. `Z:\DSP\`) for future use. Set the "Current Directory" in Matlab to point to the working directory (or type `cd <workdir>`).

The problems marked with [**Pxx**] are from the course exercise material (Spring 2009).

In the end of this session **you should know**: (a) how to compute energy in a time frame of a signal, (b) how to apply ideal filtering to 2D image, and see how ideal filtering can be a bad choice (instead of non-ideal).

1. [**M3013**] Speech (audio) signal is often analyzed in small time frames or windows. This is essential if Fourier analysis is applied. The signal should be stationary when computing Fourier transform, see [**P16**]. A typical way of seeing frequency contents of a signal in function of time is to draw a spectrogram, see Figure 1(b) and Matlab Round #3.

   The signal energy of a frame is an example of a feature which can be easily computed, see Figure 1(a). It can be applied, for instance, to finding out whether the signal is silent or not at certain time moment. Another feature of a frame is energy at different frequency bands.

   Task: Download a script file `M3013.m` and `kiisseli.wav` (or any other speech / audio file). Go through a demo in `M3013.m` and find answers to given questions.

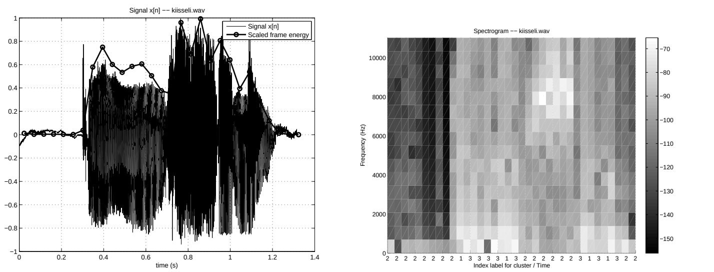

Figure 1: Problem 1: kiisseli.wav, (a) waveform and frame energy, (b) spectrogram and labels for clusters.

2. [**M4003**] Consider a Matlab demo picture `cameraman.tif` seen also in Figure 2(b). It is 2-dimensional matrix with gray-scale values $s(x, y)$, often called as pixels, which are, e.g., integers from 0 to 255, where 0 corresponds black and 255 white. The spatial variables x and y express the position of a pixel (compare to t or n in 1-D case). The size of the image is 256×256 pixels. Its discrete Fourier transform $S(u, v)$ can be computed using `fft2`, and it is also a matrix of 256×256 complex values.

   Let us filter the picture with an ideal 2-D lowpass filter, and transform it back to spatial domain using `ifft2`. Ideal lowpass filtering can be considered in transform domain as multipling passband ("area") terms by 1 and stopband terms by 0.

   Note that in 1-D case `fft` computes angles $(0 \dots 2\pi)$, see Figure 2(a). In 2-D case angle values of `fft2` run $((0 \dots 2\pi),\ (0 \dots 2\pi))$, which can be "shifted" around the origo, see Figures 2(c) and (d), because DFT-2 is 2π-periodic in both dimensions. Shifting from (c) to (d) can be done using `fftshift`, if needed. Hence, in lowpass filtering the passband area, called often mask, containing 1s has to cover $(0 \dots \omega_c,\ 2\pi - \omega_c \dots 2\pi)$ for both dimensions.

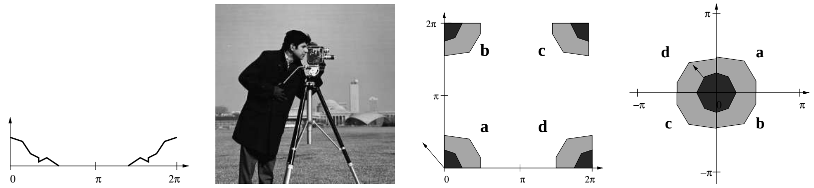

Figure 2: Problem 2: (a) 1-D spectrum in range $(0 \dots 2\pi)$, (b) `cameraman.tif` in spatial domain, (c) 2-D spectrum in range $((0 \dots 2\pi),\ (0 \dots 2\pi))$, (d) the same 2-D spectrum in range $((-\pi \dots \pi),\ (-\pi \dots \pi))$. Remember that discrete-time Fourier transform is always 2π-periodic: in (c) left-top corner "b" is $((0 \dots \pi),\ (\pi \dots 2\pi))$ whereas in (d) "b" is $((0 \dots \pi),\ (\pi \dots 2\pi) - 2\pi) = ((0 \dots \pi),\ (-\pi \dots 0))$, etc.

```
%% Read and draw a picture (Matlab demo picture)
I  = imread('cameraman.tif');   % type 'uint8' = unsigned integer 8bit
figure(1); clf;
imshow(I); axis on;
N  = size(I, 1);                % size is (N x N)
%% Compute 2D-DFT and plot spectrum
Id = double(I);                 % convert to 'double' type
IF = fft2(Id);                  % 2-D discrete Fourier transform
figure(2); clf;
imagesc(log10(abs(IF)));        % axis 0..2pi, 0..2pi like in Figure (c)
%imagesc(log10(abs(fftshift(IF)))); % shifted like in Figure (d)
colormap(gray);                 % grayscale
axis equal                      % interval of x and y equal
%% Filter mask of an ideal lowpass filter
wc = 0.3*pi;                    % cut-off frequency wc, 0 < wc < pi
M  = round(wc*N/(2*pi));        % corresponding index, (M/N) == (wc/2pi)
filterMask = zeros(size(I));    % initialize to zeros
filterMask(1:M,          1:M)        = 1;  % left-top corner,
filterMask(end-M+2:end,  1:M)        = 1;  % left-bottom corner, ...
filterMask(end-M+2:end,  end-M+2:end) = 1;
filterMask(1:M,          end-M+2:end) = 1;
figure(3); clf;
mesh(filterMask); colorbar;
%mesh(fftshift(filterMask)); colorbar  % shifted like in Figure (d)
%colormap(gray);                % grayscale
%% 2-D ideal filtering
```

```
IFfilt = IF .* filterMask;      % ideal filtering in transform domain
figure(4); clf;
imagesc(log10(abs(IFfilt)));
%imagesc(log10(abs(fftshift(IFfilt)))); % shifted like in Figure (d)
colormap(gray);
%% Inverse 2D-DFT and plot figure
Ifiltered = real(ifft2(IFfilt));  % may contain small complex values
figure(5); clf;
imshow(Ifiltered, [min(Ifiltered(:)) max(Ifiltered(:))]);
```

Task: Try filtering with a few different $\omega_c \in (0 \ldots \pi)$ (`wc`). How does the lowpass filtering affect a picture? How is it analogue to audio signals? What kinds of side-effects can be seen? What kind of side-effects do you believe to happen for audio signals?