Introduction to Computer Vision  
Exercise 4 (last exercise)

Deadline: Thursday, Jan 29th, 2015

This programming exercise is a simplified version of the paper Eulerian Video Magnification for Revealing Subtle Changes in the World. The paper shows a powerful application of simple tools. You will need to read and partially implement the paper, and add in some cool experiments of your own. It is meant to be fun!

In this exercise you will need to submit videos. Because of their large size please submit a link to google drive, dropbox or any other way you prefer. This link should include all of your code, results, and written explanations. It should also include the input movie you chose by yourselves in section 3. The exercise may be done in pairs. Note that we will use the code you submitted to reproduce your results, so make sure beforehand that it indeed works as you reported.

Please submit your exercise to ita.lifshitz@weizmann.ac.il or michal.yarom@weizmann.ac.il.

Section 1: General questions on the paper
Please read the paper: Eulerian Video Magnification for Revealing Subtle Changes in the World, Wu et al, (paper website) and answer the following:

1. Explain the idea of temporal filtering. If we want to magnify a small movement, why do we magnify the same pixel over time?
2. What advantage do we achieve by using the Laplacian pyramid?

Section 2: Building the magnification method
In order to read and write a movie in matlab, you will find the VideoReader, VideoWriter functions useful. Please read their documentation.

1. Write a function filtered_matrix = temporal_filter(spatial_time_matrix, frequency_mask) which gets a matrix that each row corresponds to a time vector of each pixel (that is, the intensity over time of a specific pixel) and a filter. The filter should be a vector of size of the number of frames (which is also the number of frequencies in the Discrete Fourier Transform of a time vector of a pixel), such that each value represents a specific frequency magnification. The function scales each frequency according to the given filter, and returns a matrix
such that each row corresponds to a time vector of a pixel after the magnification.
You may want to use some of the following matlab functions: fft, ifft, fftshift, conv
(read their documentation).
Note: Since matlab is slow with “for” loops, you may want to consider vectorizing
the code for greater efficiency.

2. Write a matlab function magnify(filename_in, filename_out, m0, m1, m2, m3, m4)
which magnifies all the pixels read from the video in the path filename_in and
saves it to filename_out. The magnification should be done for each level in the
Laplacian pyramid independently using its own specific magnification filter (m0
for the level L0, m1 for L1, etc.)
Reminder: the DFT of a signal also contains negative frequencies (after fftshift).
When you design the filter don’t forget to take it into consideration.

Section 3: Results
1. In the paper website you can find 3 movies, showing small movements of
different kinds. We want to magnify the following motions/changes:
   a. face.mp4 – Magnify the change in the face color and motion.
   b. baby.mp4 – Magnify the breathing motion of the baby.
   c. camera.mp4 – Magnify the motion of the camera.
   d. Your own fourth movie, not from the authors' website. (Either download an
      interesting movie, or record one yourself.) Surprise us - Be creative!
2. For each of the movies, describe the following:
   a. What range of frequencies did you magnify? Note that the range of
      frequencies should be given in Hz (i.e. number of cycles per second).
      Remember that in Fourier transform, the distance \( \Delta u \) between samples
      in the frequency domain is given by:
      \[
      \Delta u = \frac{1}{N\Delta t}
      \]
      where \( N \) is the number of frames, and \( \Delta t \) is the time difference between
      two frames. (Use the properties of the VideoReader in matlab to retrieve
      the needed parameters).
   b. Explain how and why you chose this particular frequency range.
   c. Explain any difficulties and possible reasons for bad results.
3. For each movie, add to the link the magnified movie and add a script, called
   run_movie_name.m which reproduces the results you reported.
For those interested in further reading:
The implementation above will tend to produce noisy output videos. However, a later paper by the same group, *Phase-Based Video Motion Processing*, Wadhwa et al, (paper website), describes a more advanced method which handles noise much better than the linear Eulerian motion magnification technique.