MIT CSAIL

6.869 Advances in Computer Vision

Spring 2011

Problem Set 2: Filtering

Posted: Wednesday, February 9, 2011 **Due:** Wednesday, February 16, 2011

You should submit a hard copy of your work in class, and upload your code (and all files needed to run it, images, etc) to stellar.
Your report should include images and plots showing your results, as well as pieces of your code that you find relevant.

This problem uses pyramid image processing. Download and install the pyramid image processing toolbox by [4]. When forming pyramid decompositions for these problems, you may always use the default decomposition filters.

Problem 2.1 *Image blending*

(a) Build a Laplacian pyramid of one image and show you can reconstruct back the original image. Code for the Laplacian pyramid is available in the pyramid image processing toolbox.

(b) Implement the function `PyrBlend(im1,im2,mask)` that takes as input two images and a binary mask (determining which part to use from each image) and produces the Laplacian pyramid blend of the two images. Use your function to blend two images of your favorite pets, friends or objects. Include in your report the original images, their Laplacian pyramids, the blending mask, and the resulting blended image.

Problem 2.2 *Visual acuity*

Subjectively, our visual world appears to us to be high resolution everywhere. However, we have much higher spatial resolution in the center of our field of view than in the periphery. In this problem, we will synthesize an image approximating our visual resolution as a function of eccentricity (angle form the center of fixation).

Approximate acuity, a, in minutes of arc (60 minutes to a degree) as a function of eccentricity, e, in degrees, by the expression

$$a = 0.23e + 0.7 \tag{1}$$

We will create an image with the effective spacing of the pixels equal to the angular size of the acuity limit. Assume that the image (or monitor) is square, and that you view it from a distance of two times the length of one side of the image. Where convenient, you may assume angles are small enough so that $\tan(\theta) \simeq \theta$.

(a) How many evenly spaced pixels per side does the image need to have so that the highest resolution part of the image has one pixel per length of finest acuity?

(b) Let the image be represented by a unit square with the upper left corner at location (0,0) and the lower right corner at location (1,1). Also assume that the upper left corner is the center of fixation. Using those units, derive the equation for the effective pixel spacing as function of the eccentricity (at each pixel) which causes the pixel spacing to equal the spatial acuity.

(c) We can approximate images of this resolution by using a Gaussian pyramid, which generates images at different numbers of pixel samples, dividing the number of pixels by two at each level of the pyramid. Start from an image at the full resolution of part (a). Each pyramid level increases the effective size of its pixels by a factor of two in each dimension. As a function of the coordinate system used in (b), by how many factors of two should the resolution of the original image be reduced as a function of position in the image in order to simulate the human visual acuity, assuming the viewer stares at the upper left corner of the image?

(d) The expression in (c) involves fractional pyramid levels. We can visually approximate images at those intermediate resolution levels by linearly interpolating between the Gaussian pyramid levels.
With this problem set is a 2000x2000 image `prob2.jpg`, which should be more than enough pixels for this experiment. Crop that image to the desired resolution such that the upper left corner will be at half the maximum visual acuity, when viewed from two picture lengths away. Write a function `simIm = VisAcuity(im)` that simulates the fall-off in visual acuity, assuming the fixation point is at the upper left corner of the given image. At any given pixel, determine the coefficients for interpolating between the pyramid levels by linearly interpolating the corresponding pixel dimensions.

Hint: You will want to use the upBlur function to transform the Gaussian pyramid levels to all have the same number of pixels. Assume that a pyramid level after upBlur has effectively the same number of pixels (in terms of picture content) as the original pyramid band before the upBlur operation.

Problem 2.3 *Steerable filters*

Take a photograph of a face. We'll process it in greyscale, so you can either convert to luminance, or just use the green channel of a color image. Using the G2, H2 steerable filters, find the dominant orientation everywhere in the image.
(a) Construct an image filtered with G2 steered to be along the dominant orientation everywhere.
(b) Construct an image filtered with G2 steered perpendicularly to the dominant orientation everywhere.
(c) and (d): construct contrast-normalized versions of each of those two images, by dividing (a) and (b) by the square root of the average angular energy in G2, H2 at each point (add a small constant to that denominator to avoid divisions by zero).

You'll want to use tables III, IV, V, VI, XI of [2]. Submit your code, and all resulting images.

Problem 2.4 *Hybrid images*

In this problem you will create hybrid images as described in [3].
Take two images, A and B, that you'll want to have blend from one to the other. Try to make the objects in the two images occupy more or less the same region. Construct a hybrid image from A (to be seen close-up) and B (to be seen far away) as follows:

$$\text{out} = \texttt{blur}(\text{B}) + (\text{A-}\texttt{blur}(\text{A}))$$

Where `blur` is a function that low-pass filters the image. You can write your own blur function, or use the `upBlur` and `blurDn` functions supplied in [4] (which go up and down Gaussian pyramid levels). You will want to blur by more than just one Gaussian pyramid level. How does the blurring level affect your perception of the results?
Submit your images, results and code.

Problem 2.5 *DFT*

(a) *Definition:* An image is said to have n-fold symmetry with respect to 2D point p if it is invariant under $2\pi/n$ rotation about p.
Show that if an image has n-fold symmetry, then its corresponding Fourier transform does, too.

(b) Take, or find on the web, 5 photographs of different scenes. Measure the (horizontal) spectral fall-off for the five images. Characterize by their visual appearance which images have a fast fall-off and which have a slow fall-off. Submit your photos, the spectral plots, and code.
You can use the MATLAB function `fft` to compute the Fourier transform. Note that `fft` places the lowest frequency components in the upper left part of the returned image. `fftshift` can be used to shift them to the center of the spectrum.

Research problem [optional] *Energy for motion*

In class we mostly concentrate on spatial analysis of images – snapshots of the world taken at instantaneous moments in time. It is often useful however to think about the world as captured though time, as appears in videos. Although videos too sample the world (albeit in a higher rate), when we watch a movie or a tv show we can sense the motions as if we are observing a continuously moving scene. Temporal analysis can give us a lot of information on events occurring in that scene, such as trajectories of objects, their velocities, and so on.

Motion can be analyzed through the 3D Fourier spectrum corresponding to the 3D video volume, as well as 2D analysis of XT (spatiotemporal) slices of the video using techniques mentioned in class [1]. A possible project is to explore those different representations, and use them to characterize different types of videos. Motion estimation and event recognition are some of the possible applications that can be investigated in this context.

References

[1] Edward H. Adelson and James R. Bergen. Spatiotemporal energy models for the perception of motion. *J. OPT. SOC. AM. A*, 2(2):284–299, 1985. `http://web.mit.edu/persci/people/adelson/pub_pdfs/spatio85.pdf`.

[2] William T. Freeman and Edward H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:891–906, 1991. `http://people.csail.mit.edu/billf/papers/steerpaper91FreemanAdelson.pdf`.

[3] Aude Oliva, Antonio Torralba, and Philippe G. Schyns. Hybrid images. In *ACM Transactions on Graphics (Proceedings SIGGRAPH)*, pages 527–532. ACM, 2006. `http://cvcl.mit.edu/publications/OlivaTorralb_Hybrid_Siggraph06.pdf`.

[4] Eero Simoncelli. matpyrtools. `http://www.cns.nyu.edu/~eero/software.php`.