

## CMPT882 Assignment 1: Edge detection and Texture Recognition

Due date September 28

### 1. Implement the Canny edge detector (ref: Trucco and Verri, Ch. 4)

- Smooth input image  $I$  by convolving with a Gaussian  $G = \exp(-(x^2 + y^2)/\sigma^2)$ ,  $J = I * G$
- Compute image derivatives  $J_x, J_y$
- Compute edge strength  $E_s = \sqrt{J_x^2 + J_y^2}$  and direction  $E_o = \arctan \frac{J_y}{J_x}$
- Perform non-maximum suppression, setting to zero values of  $E_s$  that are not larger than their neighbours along the direction perpendicular to the edge orientation in  $E_o$ .
- Implement hysteresis thresholding: given high threshold  $t_h$  and low threshold  $t_l$  ( $t_h \geq t_l$ ), mark as edges all points with either:
  1.  $E_s$  larger than  $t_h$
  2.  $E_s$  larger than  $t_l$  and connected to an edge point  $\hat{e}$  with  $E_s(\hat{e}) > t_h$  by other edge points with strength  $E_s > t_l$ , in the direction of the edge at  $\hat{e}$

Helpful MATLAB functions (aside from `edge(..., 'canny')`) include `filter2`, `gradient`, and `fspecial('gaussian', ...)`. Experiment with running your edge detector on a couple of your favourite images, with different values for  $\sigma$ ,  $t_h$ , and  $t_l$ .

**Extra:** If you are interested in edge detection, perhaps for use in a course project, try downloading and running the “PB” code from <http://www.cs.berkeley.edu/projects/vision/grouping/segbench/> Please see the course webpage for more details.

### 2. Perform texture recognition using histograms of textons

- Download the training and test images from the course website
- Construct a filterbank consisting of 18+18+3 filters ( $L_1$  normalized), of three different types:
  1. Oriented odd-symmetric filters at 3 scales and 6 orientations, modeled as rotated copies of the horizontal filter  $f(x, y) = G'_{\sigma_1}(y)G_{\sigma_2}(x)$ . Use a ratio of 3 for  $\sigma_2 : \sigma_1$ . Set the 3 scales to be a “half-octave” apart, i.e.  $\sigma_1^{i+1} = \sqrt{2}\sigma_1^i$ .
  2. Oriented even-symmetric filters at 3 scales and 6 orientations, again rotated copies of a horizontal filter, this time  $f(x, y) = G''_{\sigma_1}(y)G_{\sigma_2}(x)$ .
  3. Radially symmetric center-surround filters at 3 scales, each modeled as a “Difference of Gaussians” (DOG),  $f(x, y) = \exp(-(x^2 + y^2)/\sigma_1^2) - \exp(-(x^2 + y^2)/\sigma_2^2)$ .
- Compute textons by filtering the training images and running kmeans to cluster the output. You will need to randomly sample points from these images, running kmeans using all pixels in all images will be very slow.
- Use these textons for texture recognition: for each test image, compute texton histogram and compare to histograms for training images using  $\chi^2$  distance. Assign label of closest matching training image as label for test image.

For debugging, it is helpful to look at all the filter kernels and the filter responses on an image.

```
% K is a cell array of filterbank kernel matrices
K = cell(39,1);
% R is the filter responses
R = zeros(N,M,39);

figure(1);
subplot(4,10,1);
for k_i=1:length(K)
    subplot(4,10,k_i);
    imagesc(K{k_i});
end

figure(2);
subplot(4,10,1);
for k_i=1:size(R,3)
    subplot(4,10,k_i);
    imagesc(R(:,:,k_i));
end
```

Experiment with a few different values of  $K$ , the number of textons computed using kmeans. Look at the texton maps computed for each image, and check that they are sensible.

Helpful MATLAB function: `imrotate(..)` to create rotated filters.

There is a kmeans routine in MATLAB stat toolkit, or you can use the one in Netlab (see website) if you don't have stat toolkit access (stat toolkit is available on CS machines).

### Submitting your assignment

Please schedule a time during my office hours (or other if you can't make them) to come and show me your assignment. Sign up on this wikipedia: <http://wiki.cs.sfu.ca/vml/AssignmentDemo> or send me email if you can't make it during the office hours.

You can either bring printouts of results, or a laptop (have the results generated and saved), to show me how your assignment works. E.g. For part 1, show me image derivatives, edge strength pre and post non-max suppression, thresholded edge strength, hysteresis thresholded edge strength.