CS 376 Computer Vision
Spring 2011

Problem set 1
Out: Tuesday Feb 1
Due: Monday Feb 14 11:59 PM

See the end of this document for submission instructions.

Visit us during office hours to discuss any questions on the assignment. Or, if sending a question via email, please submit to cv-spring2011@cs.utexas.edu with CS376 in the subject line.

I. Short answer problems [30 points]

1. Give an example of how one can exploit the associative property of convolution to more efficiently filter an image.

2. This is the input image: ([0 0 1 1 0 0 1 1]. What is the result of dilation with a structuring element [1 1 1]?

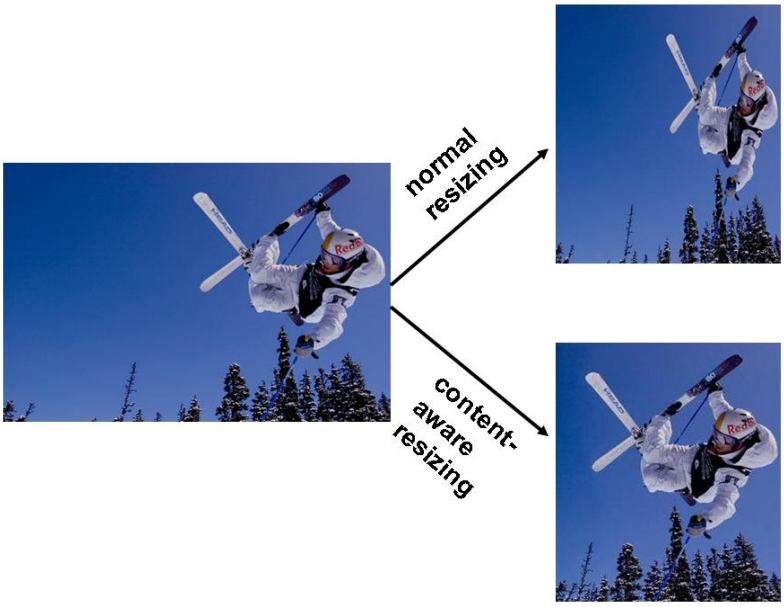
3. The filter $f' = [0, -\frac{1}{2}, 0, \frac{1}{2}, 0]$ gives an estimate of the first derivative of the image in the *x* direction. What is the corresponding second derivative filter f''? (Hint: asymmetric filters must be flipped prior to convolution.)

4. Name two specific ways in which one could reduce the amount of fine, detailed edges that are detected with the Canny edge detector.

5. Describe a possible flaw in the use of additive Gaussian noise to represent image noise.

6. Design a method that takes video data from a camera perched above a conveyor belt at an automotive equipment manufacturer, and reports any flaws in the assembly of a part. Your response should be a list of concise, specific steps, and should incorporate several techniques covered in class thus far. Specify any important assumptions your method makes.

II. Programming problem: content-aware image resizing [70 points]



For this exercise, you will implement a version of the content-aware image resizing technique described in Shai Avidan and Ariel Shamir's SIGGRAPH 2007 paper, "Seam Carving for Content-Aware Image Resizing". The paper is available off the course website. The goal is to implement the method, and then examine and explain its performance on different kinds of input images.

First read through the paper, with emphasis on sections 3, 4.1, and 4.3. Note: choosing the next pixel to add one at a time in a greedy manner will give sub-optimal seams; the dynamic programming solution ensures the best seam (constrained by 8-connectedness) is computed. Use the dynamic programming solution as given in the paper and explained in class.

Write Matlab code with functions that can do the following tasks:

- Compute the energy function at each pixel using the magnitude of the x and y gradients (equation 1 in the paper)
- Compute the optimal vertical seam given an image
- Compute the optimal horizontal seam given an image
- Reduce the image size by a specified amount in one dimension (width or height decrease)
- Display the selected seam on top of an image
- Functions with the following interface:

  ```
  [output] = reduceWidth(im, numPixels)
  [output] = reduceHeight(im, numPixels)
  ```

  These functions take an input image `im`, and a parameter specifying how many seams to carve, from the width or height, respectively. The image `im` will be a *h x w x 3* `uint8` matrix, which is what `imread` returns for a color image. Put these functions in file named `reduceWidth.m` and `reduceHeight.m`

Set up scripts so that you can play with the seam removal and specify different combinations of horizontal and vertical removals. Apply your system to the provided images. View the results in color, but note that the gradients should be computed with the grayscale converted image.

Matlab hints:

- Useful functions: `imfilter`, `im2double`, `fspecial`, `imread`, `imresize`, `rgb2gray`, `imagesc`, `imshow`, `subplot`;
- To plot points on top of a displayed image, use "`imshow(im);`" followed by "`hold on;`" followed by "`plot(…)`".
- Be careful with `double` and `uint8` conversions as you go between computations with the images and displaying them – filtering should be done with doubles.

Answer each of the following, and include image displays where appropriate:

1. [10 points] Run your `reduceHeight` function on the provided `prague.jpg` with `numPixels = 100` (in other words, shrink the height by 100 pixels). Run your `reduceWidth` function on the provided `mall.jpg` with `numPixels = 100` (in other words, shrink the width by 100 pixels). Display the outputs.

2. [10 points] Display (a) the energy function output (total gradient magnitudes $e_1(\mathbf{I})$) for the provided image `prague.jpg`, and (b) the two corresponding cumulative minimum energy maps (M) for the seams in each direction (use the `imagesc` function). Explain why these outputs look the way they do given the original image's content.

3. [10 points] For the same image `prague.jpg`, display the original image together with (a) the first selected horizontal seam and (b) the first selected vertical seam. Explain why these are the optimal seams for this image.

4. [10 points] Make some change to the way the energy function is computed (i.e., filter used, its parameters, or incorporating some other a prior knowledge). Display the result and explain the impact on the results for some example.

5. [30 points] Now, for the real results! Use your system with different kinds of images and seam combinations, and see what kind of interesting results it can produce. The goal is to form some perceptually pleasing outputs where the resizing better preserves content than a blind resizing would, as well as some examples where the output looks unrealistic or has artifacts.

   Include results for the **two provided images**, plus at least **three images of your own** choosing. Include an example or two of a "bad" outcome. Be creative in the images you choose, and in the amount of combined vertical and horizontal carvings you apply. Try to predict types of images where you might see something interesting happen. It's ok to fiddle with the parameters (seam sequence, number of seams, etc) to look for interesting and explainable outcomes.

   For each result, include the following things, clearly labeled (see `title` function):
   (a) the original input image,
   (b) your system's resized image,
   (c) the result one would get if instead a simple resampling were used (via Matlab's `imresize`),
   (d) the input and output image dimensions,
   (e) the sequence of enlargements and removals that were used, and
   (f) a qualitative explanation of what we're seeing in the output.

III. [OPTIONAL] Extra credit [up to 10 points each, max possible 20 points extra credit]

Below are ways to expand on the system you built above. If you choose to do any of these (or design your own extension) include in your writeup an explanation of the extension as well as images displaying the results and a short explanation of the outcomes. Also include a line or two of instructions telling what needs to be done to execute that part of your code.

1. Allow a user to mark an object to be removed, and then remove seams until all pixels on that object are gone (as suggested in section 4.6 of the paper). Either hard-code the region specific to the image, or allow interactive choices (Matlab's `ginput` or `impoly` functions are useful to get mouse clicks or draw polygons).

2. Design an alternate energy function, instead of the gradient magnitude. Explain your choice, and show how it can influence the results as compared to using the gradient magnitude. Choose an image or two that illustrates the differences well.

3. To avoid warping regions containing people's faces, have the system try to detect skin-colored pixels, and let that affect the energy map. Try using the hue (H) channel of HSV color space (see Matlab's '`rgb2hsv`' function to map to HSV color space). Think about how to translate those values into energy function scores.

4. Implement functions to *increase* the width or height of the input image, blending the neighboring pixels along a seam. (See the Seam Carving paper for details.) Demonstrate on an image that clearly shows the impact.

5. Implement the greedy solution, and compare the results to the optimal DP solution.

Submission instructions: what to hand in

Electronically:

- Your documented Matlab code (including functions `reduceWidth.m` and `reduceHeight.m`)
- A first pdf file named `file1.pdf` containing the following:
    - Your name at the top
    - Your answers to Part I, numbered.
    - A brief explanation of your implementation strategy: a short paragraph or two describing in English what you have computed.
    - Your responses and image results for questions 1 through 4 in Part II, numbered. Insert image figures in the appropriate places for these questions.
- A second pdf file named `file2.pdf` containing the following:
    - Your name at the top
    - Your responses and image results for question 5 in Part II.
    - (optional): any results and descriptions for extra credit portions.
    - This file will be posted online – be sure to credit any photo sources.

Tip: How to save as pdf? If you have a pdf printer installed on your computer, you can convert a document prepared in Word to pdf. The CS machines have `openoffice` installed, which will also allow you to Save as… a pdf file. Or, if you work in Latex, you can use `pdflatex`, or compile to a ps and then convert.

Submit all the above with one call to turnin:

```
>> turnin --submit shalini pset1 file1.pdf file2.pdf
reduceWidth.m reduceHeight.m   <otherFunction.m>  …
<otherFunction.m>  etc.
```

Hardcopy:

- Also drop a hardcopy in the CS homework dropbox in PAI 5.38. Attach a cover page with the course number CS376 and your name, to make it easy to find in the dropbox. Please save paper by concatenating shorter functions into a single page before printing.

Image acknowledgements: Thanks to the following Flickr users for sharing their photos under the Creative Commons license: mall.jpg is provided by hey tiffany! prague.jpg david.nikonvscanon