I400/B490 Intro to Computer Vision - Programming assignment #1

Problem 1: Seam Carving

0. Download the test images from Canvas: seam_carving_input1.jpg, …, seam_carving_input3.jpg

1. Read slides 32-44 of lecture2_2_gradients.pptx. For more details, check Sections 3.1, 3.2, and 4 of the original paper: http://perso.crans.org/frenoy/matlab2012/seamcarving.pdf

2. Implement the function `energyImage = energy_image(im)` . For each pixel, you essentially need to compute the magnitude of x_gradients and y_gradients: sqrt(dx^2 + dy^2). Convert the image into a grayscale image and use it to compute x_gradients and y_gradients. The output `energyImage` should be a 2D array of datatype double. (slide 36).

- Save the `energyImage` of the three test images as “energy_image1.jpg”, ...

3. Implement the function `cumulativeEnergyMap = cumulative_minimum_energy_map(energyImage,seamDirection)` . The parameter `seamDirection` is either 0 or 1 (0=horizontal and 1=vertical). The output `cumulativeEnergyMap` must be a 2D array of datatype double. Check slides 39-41 for the implementation. We are essentially coming ***M*** from the slides. Treat all boundary values (e.g., i<0 or j<0) to be 0.

- Save the `cumulativeEnergyMap` of horizontal and vertical directions. Name them “cem_horizontal1.jpg”, “cem_vertical1.jpg”, ...

4. Implement the function `verticalSeam = find_optimal_vertical_seam(cumulativeEnergyMap)` . Check slides 39-41. The output `verticalSeam` must be a vector containing the column indices of the pixels which form the seam for each row.

5. Implement the function `horizontalSeam = find_optimal_horizontal_seam(cumulativeEnergyMap)` . Check slides 39-41. The output `horizontalSeam` must be a vector containing the row indices of the pixels which form the seam for each row.

6. Implement the function `display_seam(im,seam,seamDirection)` , to display the selected type of seam on top of an image. The input `im` should be an image of type jpg. `seam` can be the output of `find_optimal_vertical_seam` or `find_optimal_horizontal_seam` . `seamDirection` is either 0 or 1 to indicate the seam type (0=horizontal and 1=vertical). The function should display the input image and plot the seam on top of it.

- Visualize the vertical/horizontal seam of each of the three images. Save them as “seam_v1.jpg”, seam_h1.jpg”, ...

7. Implement the function `greedyVerticalSeam = find_greedy_vertical_seam(energyMap)` and `greedyHorizontalSeam = find_greedy_horizontal_seam(energyMap)` . Check slide 38. For the greedy vertical seam, iterate through each row of the image, and simply choose the pixel with the minimum energy connected from the previous row.

- Visualize the greedy vertical/horizontal seam of each of the three images. Save them as “greedy_seam_v1.jpg”, greedy_seam_h1.jpg”, ...

8. Implement the functions `reducedColorImage, reducedEnergyImage = reduce_width(im,energyImage)` and `reducedColorImage, reducedEnergyImage = reduce_height(im,energyImage)` . The function will reduce the width (or height) of the image by 1 pixel, and obtained the new reduced image and `energyimage`.

9. In your script, reduce the width of the assigned image file by 100 pixels. Try the three files.

- Save the results as “output_width1.jpg”, ...

10. In your script, reduce the width of the assigned image file by 100 pixels. Try the three files.

- Save the results as “output_height1.jpg”, ...

11. Try seam carving with your own files. Select three files of your own. Apply the steps 8 and 9.

- Save the results as “output_yours_width1.jpg”, “output_yours_height1.jpg”, ...