ECE5/7382 – Fundamentals of Computer Vision

Homework-3

Assigned: Sep 18

Due: Sep 29

Maximum Possible Grade: 55 Points (ECE5382), 65 Points (ECE7382)

General guidelines

* Please upload your response to Canvas as a ZIP file with the following filename convention
* Name of ZIP file FIRSTNAME\_LASTNAME\_Homework3.zip
* Replace FIRSTNAME with your first name
* Replace LAST NAME with your last name

**Failure to adhere to the filename convention will result in deduction of points.**

* **Provide a detailed response to each question in the** [**Deliverables & Questions**](#_Deliverables_&_Questions) **section, including supporting mathematical arguments**.

Submitting your solutions

Please ensure that the ZIP file uploaded to Canvas, includes the following components:

* Completed Python code for **Homework-3.ipynb**
* Response to all questions embedded in this document. Please convert this document into PDF and include it in the ZIP archive.

**For 7382 students:** include your sample images in the ZIP archive.

Content aware image resizing

As part of this assignment, you will be developing Python routines to implement a simplified 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 here: <http://www.faculty.idc.ac.il/arik/SCWeb/imret/index.html>. Please read the paper, with emphasis on sections 3, 4.1, and 4.3.

For this assignment, we will restrict attention to shrinking images or making images smaller. The goal is to implement the algorithm and analyze its performance on multiple images.

### Seam Carving Example: Please refer to the lecture slides/videos for an example.

### Basic Idea

As described in the lecture, seam carving seeks to resize an image by removing pixels characterized by low gradient energy (relatively smooth areas), while preserving more informative pixels characterized by higher gradient energy. Central to the effort is the computation of the per-pixel gradient energy, using one of two approaches

* The term represents the image function associated with the image to be resized.
* The term represents a spatial location in the image.
* The terms represent the outcome of derivative filtering in the and directions respectively. The OpenCV function **Sobel(…)** may be used to compute these terms

#### Seams

The seam carving algorithm proceeds by identifying single pixel wide horizontal and vertical trajectories (called seams) characterized by minimum cumulative gradient energy.

* A vertical seam is an 8-connected path of pixels from top to bottom that contains one and only one pixel in each row.
* An optimal vertical seam is characterized by least accumulated gradient energy in the vertical direction (accumulate gradient energy across rows).
* A horizontal seam is an 8-connected path of pixels from left to right that contains one and only one pixel in each column.
* An optimal horizontal seam is characterized by least accumulated gradient energy in the horizontal direction (accumulate gradient energy across columns).

#### Finding Optimal Vertical seams using dynamic programming

The following approach inspired by dynamic programming is used to find the optimal vertical seam in two stages.

#### Stage-1:

In the first stage, we seek to find the cumulative gradient energy map that specifies the energy/cost of a vertical seam traversing each pixel in the image. The recursion relation disclosed below is used to populate the entries of the matrix .

Illustrated below are the individual steps of the recursion for a sample energy function .

A picture containing calendar

Description automatically generated

#### Stage-2:

In the second stage, the pixels that make up the optimal vertical seams are identified by backtracking. The minimum value in last row of 𝑴 signifies the end of optimal vertical seam. Beginning with this pixel location, we identify the next pixel in the optimal vertical seam by selecting the minimum of the 3 nearest neighbor pixels in the previous of. The process iterates until we reach the first row of the image, and is illustrated below:

Graphical user interface, application, table

Description automatically generated

### Implementation Caveats

* It is highly recommended that the input images to the seam carving algorithm be converted to grayscale. This may be done using **im2double(…)** supplied with the starter code.
* To compute the gradient energy , you will need to compute image gradients on the X and Y direction, at each pixel. Please use the OpenCV function **Sobel(…)** function to accomplish this task.

|  |
| --- |
| Help: [Image Gradients — OpenCV-Python Tutorials beta documentation (opencv24-python-tutorials.readthedocs.io)](https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_gradients/py_gradients.html) |

* Prior to computing the matrix , it is recommended that the gradient energy image be padded (using **copyMakeBorder** function in OpenCV) with **Inf**. Padding avoids complications when implementing Eq.(5) at the image boundary. Please refer to the lecture slides/videos for additional details.

|  |
| --- |
| Help: [OpenCV: Adding borders to your images](https://docs.opencv.org/3.4/dc/da3/tutorial_copyMakeBorder.html),  [Python OpenCV | cv2.copyMakeBorder() method - GeeksforGeeks](https://www.geeksforgeeks.org/python-opencv-cv2-copymakeborder-method/) |

### Python/OpenCV Code to implement

The starter code **HWK3\_SeamCarving.ipynb** is intended to help you get started with Part-II of the assignment.

* The code invokes a Python function **shrink\_width** with the following interface

**[outp\_im] = shrinkWidth(inp\_im, numPixels)**

The function accepts a grayscale input image **inp\_im**, and a parameter **numPixels** specifying the number of vertical seams to carve. You will use this function to reduce the size of an input image by a prescribed number of pixels.

* The function **shrinkWidth** iteratively shrinks the width of the image, by making repeated calls to a function

**[optVertSeamPath,EnergyFn,mtxM] = findVerticalSeam(im\_old)**

that should accept an input image **im\_old**. The function should output the vertical seam associated with the image **im\_old**, the energy function (**EnergyFn**) and the accumulated gradient energy (**mtxM**).

NOTE: The column is zero-indexed consistent with Python’s convention.

This information is used by the starter code to shrink the width of the image by 1 pixel. Lines 29-32 in the Helper Functions cell implements this capability.

* As a part of identifying the optimal vertical seam, you will have to additionally implement functions or code- fragments to accomplish the following tasks
  + Computing the energy function at each pixel location. Use the definition of the energy function defined in Eq.(3).
  + Computing the matrix for implementing the forward recursion step in seam carving.

### Deliverables & Questions

* Completed source code for **findOptVertSeam()** (20 points)

1. Question-1 in **Homework-3.ipynb** (15 points)

* “*What is the difference between the two calls to* ***findVerticalSeam()****”*

Please furnish your response here

* “*What are the two calls to* ***np.transpose()*** *trying to accomplish?”*

Please furnish your response here

* “*Why is it that the energy function E remains unchanged in the two calls to* ***findVerticalSeam()****”*

Please furnish your response here

Insert screenshot of Energy function E

Insert screenshot of matrix M for vertical seam

Insert screenshot of matrix M for vertical seam

* *“What do you think is happening in lines 24, 30? Why are the variables reversed between the two calls?”*

Please furnish your response here

* *“Comment on why you think the seams look the way they do given the original image’s content.”*

Please furnish your response here

1. Question-2 (10 points)

Invoke **shrinkWidth** function on the Lake.jpg with numPixels = 100 (in other words, shrink the width by 100 pixels).

Insert screenshot of original image

Insert screenshot of smart resize

1. Question-3 in **Homework-3.ipynb** (10 points, 2 points each)

“*Does the result seem reasonable to you. If not comment on what you think appears to be happening.”*

Please furnish your answer here

Insert screenshot of Energy function E

Insert screenshot of matrix M for vertical seam

Insert screenshot of matrix M for vertical seam

Insert screenshot of smart resize for numPixels = 50

Question-4: ECE-7382 students only (10 points)

Attempt seam carving on an image of your choosing.

Insert screenshot of original Image

Insert screenshot of matrix M for vertical seam

Insert screenshot of matrix M for vertical seam

Insert screenshot of Energy function E

Insert screenshot of smart resize for numPixels = 50

Acknowledgments

The seam carving assignment has been adapted from multiple sources: CS 376 at UT Austin, CSE 455 at University of Washington and CS 131 at Stanford.