Trevor Carter

6.815: Make Your Own Assignment

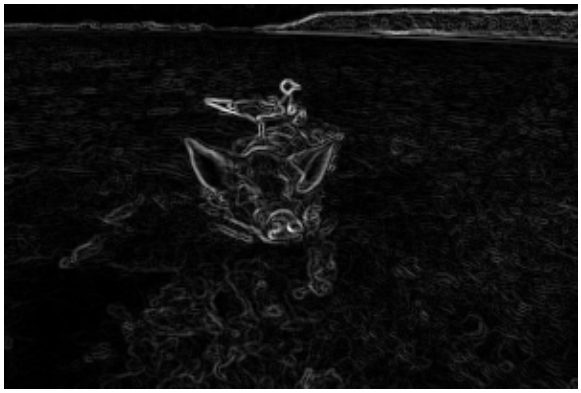
For this assignment, I chose the Seam-Carving problem set from previous years. When we did an introduction to Seam-Carving techniques my sophomore Fall semester in 6.009, it was one of the events that got me interested in image processing and image algorithms.

In summary, seam-carving is a method of image resizing that is *content-aware*, meaning it utilizes the input image to make informed decisions about resizing. In the case of my implementation of seam-carving, we use an energy function to get a mapping of energy in the image, then choose the path of minimum energy so that important features of the image are preserved after the carving. Below is a more detailed step-by-step of the process:

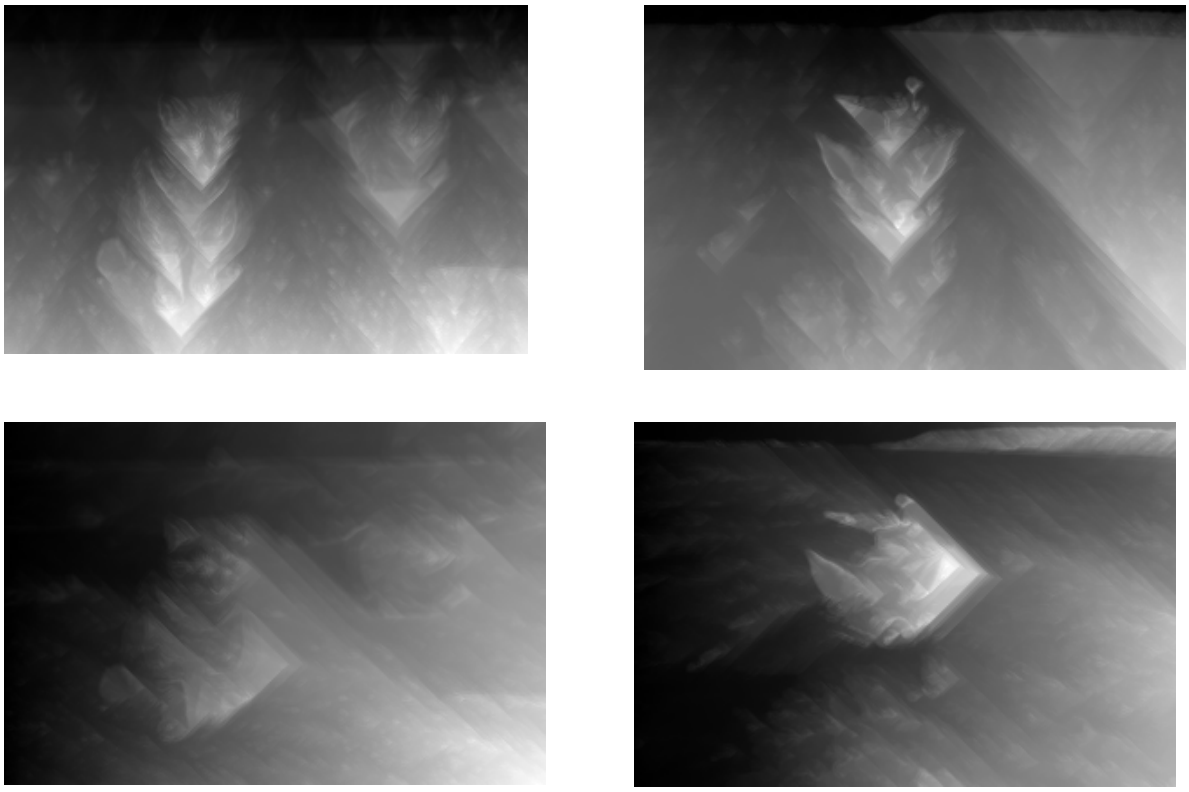
Seam Carving Process

Prior to the explanation of seam-carving, below are the two images I will be demonstrating in the write-up. Many more examples will be shown at the end, but for the step-by-step process, I have used the Pigbird and Twocats images that were used in 6.009, since they are very good examples of the ability of seam-carving. Here are the two raw images:

For this version of seam-carving, I am utilizing the L1 energy of the image, defined by the sum of the magnitudes of the directional gradients we've used in previous PSETS (the equation E(I) = |I_x| + |I_y|, where E(I) is the energy at pixel I and I_x is the x-direction gradient and I_y is the y-direction gradient). In my code, I have a function called L1_energy that calculates a one-channel image of energy values given a three-channel image input. Below is the L1 energy for both example images:



I have implemented two versions of seam-carving, one in the horizontal direction and one in the vertical direction. Thus, the next part requires two functions (I choose not to do it based on rotation, just two explicit functions). We must now calculate the cumulative energy map. For the vertical case, the function starts at the top row, and for each pixel below the top row, finds the path with the lowest amount of energy to reach that pixel. The value at that pixel becomes the amount of energy it takes to get there. Thus, as we progress downwards, the values increase. In the horizontal case, we go from left edge to right edge, so values increase as we reach the right border. The functions are named *vert_cumulative_energy_map()* and *hori_cumulative_energy_map()*. Here are four images, the top representing the vertical cumulative energy maps of our two images, and the bottom representing the horizontal cumulative energy maps of our two images:



The next step is calculating the seams themselves. In this vertical case, we start by finding the minimum energy in the bottom row of the cumulative energy map, then backtracking a path to the top row by following the pixels with lowest energy. By doing this, we find the minimum-energy seam in the image. This method is guaranteed to find the lowest energy seam in the entire image, since we recall that the cumulative energy map values correspond to lowest cumulative energy over a path. Thus, the minimum of the bottom row is the final pixel in the path containing lowest energy. In the horizontal case, we start the search on the right column, and progress back towards the left column. Below are four images, the first two representing the

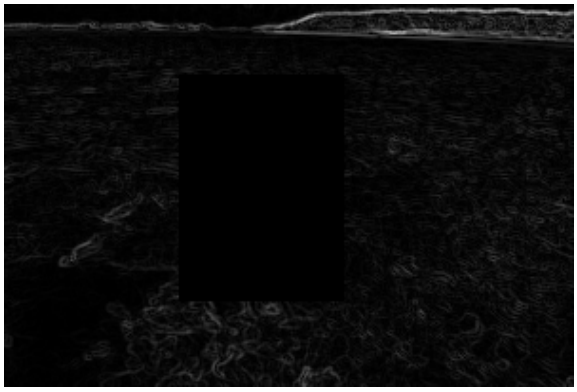
minimum seam found in the vertical direction, and the bottom two representing the minimum seam found in the horizontal direction. The seam is represented as a line of red pixels:

The final task is to carve the image. I have implemented two functions, *vertical_seam_carving()* and *horizontal_seam_carving()*. Both take an input three-channel image and an integer N, the number of pixels we want to reduce the image by in the associated direction (i.e. how many seams we must remove). Rather than choosing to work recursively, I set up the functions iteratively. We iterate N number of times. In each iteration, we first calculate the L1 energy and cumulative energy map in the associated direction, and then get the minimum seam. We then initialize a new image, decreased by 1 pixel in the dimension we carve in. Iterating over the current image, we move the pixels to the new image unless they are present in the minimum seam, where we ignore them. The loop then iterates, and our new current image is the one we've just carved. At the end of the loop, the current image becomes our output that we return. Below are four images, my test cases for both vertical and horizontal seam carving. In the top two, we seam carve the two images with vertical seams, reducing their width by ⅓. In the bottom two images, we seam carve the two images with horizontal seams, reducing their height by ⅓. It can be seen that important features, the animals themselves, are preserved through this process:

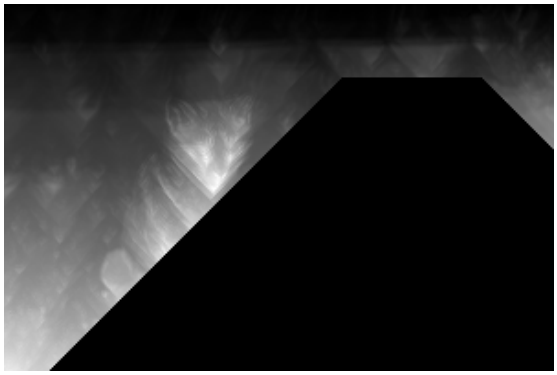
Advanced Technique

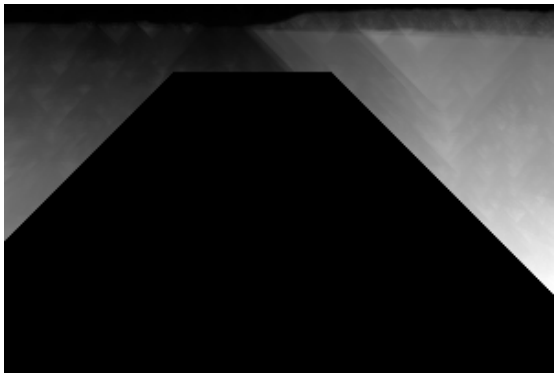
The advanced technique I chose to add from the paper was the object removal. Their object removal works in a slightly different manner, using a simple UI for region selection and alternates between vertical and horizontal carving. For mine, I implemented a function *object_removal_seam_carving().* This function takes as input a three-channel image, a vector of integers, and a boolean determining whether you want vertical or horizontal seam carving. The vector of integers represents the coordinates of a rectangular region you would like to intelligently remove from the input image. This function works recursively, calling itself on successively smaller regions until the entire region has been removed from the image. This is similar to the algorithm described in the paper, which continuously runs until all marked pixels have been removed. The base case for recursion is when the region has elapsed to a size of zero in the associated direction of carving. For my example images, we will be removing the right cat in the Twocats image, and both the pig and bird in the Pigbird example.

The first step is calculating the L1 energy. After calculating, we mark the region we want to remove with L1 energies of -1000, regardless of what the function determined for that region. This way, we can guarantee the minimum seam goes through the region. Here is what they look like after this calculation:



As you can see, the regions we mentioned we would be removing have now become black squares in the image, since their value is below zero. Next, we calculate the cumulative energy map, and we traverse the image as before, starting from the top in the vertical case or the left side in the horizontal case, traversing to the other side as we add up the energy. Here are example images from the vertical direction:





One can see that any path that crosses through our rectangular regions now has an incredibly negative seam, showing up black in the image since its total energy is far below zero. Lastly, we calculate the minimum seam, then perform the carving. Carving and seam finding is done just as before. At the conclusion of carving, we increment the region by subtracting 1 from the X direction if it is vertical carving, or 1 from the Y direction if it is horizontal carving. Since we have removed a seam going through the rectangular region, the region shrinks by 1 in the associated size. Thus, when we recurse, we use this updated region and our input image is the one we have just carved by one seam. The base case then returns the image which has been carved to fully remove the region. Here are the results:

The right cat has been fully removed, and the pig and bird have been removed. The images are not without slight visual artifacts, which could be solved with further optimization or a smart decision-making algorithm that combines both horizontal and vertical carves for area removal. As a whole, the removal is robust enough to make the output images acceptable. That is the end of the description of my code/work.

Background Research

Here are three summaries of papers I looked at relating to applications and usage of seam-carving:

- In this paper, researchers apply optimizations to pairs of stereo images (same content, different viewpoints). They develop energy equations that take into account discrepancies in images. For example, in one energy function, it takes into account the discrepancy between a seam in one image and the same seam in another. The seam goes through different points on top of the same object in the image, so the function utilizes the distance between those points as a factor in it's energy calculation. Another energy function they test utilizes differences in color along a seam, rather than distance between points like the previous. They conclude by demonstrating how seams created taking into account stereoscopic information in an image can be used to manipulate depth between objects in an image or sets of images. Link: https://ieeexplore.ieee.org/abstract/document/5506316

- In this paper, researchers improve the seam-carving algorithm's ability to preserve edges. They do this primarily by adjusting how energy is calculated by introducing a line-detection algorithm that increases energy where seams would intersect with straight lines. In addition, they take into account cases where multiple seams would border one another and remove large chunks of a line, creating a discontinuity visual artefact on the line. Their algorithm adjustments offer a much improved quality of edges, curves, and lines on image content like buildings, vehicles, roads, and other objects that take up space and use long lines/curves in their depiction. Link: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/7542/75420G/Seam-carving-with-improved-edge-preservation/10.1117/12.840263.full?SSO=1

- In a different vein than the previous two papers, in this one researchers propose a convolutional neural network model that detects images that have undergone seam carving (either seam insertion or seam removal) based on the detection of image artefacts. This can be used for image forgery and other forensic applications. They feed the network numerous images that have undergone various forms of cropping, scaling, and seam-carving, so that it gains an ability to detect artefacts specific to a seam-carving case. Their neural network achieved an overall 97.64% accuracy of detecting seam-insertion correctly and 95.90% on detection of seam-removal. Their computational complexity was als proven to be much lower than existing neural networks on a non-CNN based approach. Link: https://ieeexplore.ieee.org/abstract/document/9257477

Additional Image Examples:

Object removal algorithm on soccer players, seams made in the vertical direction.

Object removal algorithm on skydivers, seams made in the horizontal direction.

Performance

All test cases mentioned below perform their function on both the Pigbird and Twocats images.

- The L1 energy test case takes 0.0497 seconds.
- The vertical and horizontal cumulative energy map test cases take ~0.06 seconds.
- The vertical seam and horizontal seam calculation test cases take ~0.084 seconds.
- The vertical seam carving test case takes 2.00 seconds.
- The horizontal seam carving test case takes 1.45 seconds.
- The object removal test case takes 1.68 seconds.

Ethical Issues

The first change I would propose to the professor's task is adjusting the goals of the system to be through an objective lens. A concept such as 'attractiveness' is a subjective function, whose inputs are based on the person determining the attractiveness. They have their own ideas, biases, and opinions of attractiveness that make it impossible for us to make a system that adjusts attractiveness in a vacuum. To make the task objective, we must instead focus on adjusting face features such as height and width of the head, warping of facial features, or removal of facial features. This would make the task more responsible in terms of removing some of the subjectivity of the phrase 'attractiveness'.

In a vacuum, the task seems unethical. Even if we boil down to objective features like eyes, mouth, nose, etc. and the warping of them, the amount we warp or change is still a version of the user adjusting the face based on a subjective interpretation of attractiveness. It seems we are effectively talking about what editors for beauty magazines, exercise magazines, or other media that requires the view of the human body use. Everyone knows that the glamour shots are always edited towards a culture's definition of 'traditional attractiveness'.

To rescope the project to make it avoid ethical problems, I think all forms of subjective nature have to be taken away from it. If we introduce any form of opinion to it using terms like attractiveness, it ruins the responsibility and ethicality of the project. One can utilize the framework posed above, boiling down facial editing to individual subsections like ears, nose, mouth, etc., but the ultimate outcome of editing these things should be in the spirit of something more objective, i.e. making the eyes look different directions, doubling the size of the nose, adjusting these parameters, etc. Or, rather than using real datasets, you could use one of the machine-generated image datasets of people that don't actually exist. These AI take features from datasets of real people and map them into artificial faces.