Midterm Research Project

Replication of “Seam Carving for Content-Aware Image Resizing”

Nitish Sanghi

Abstract

This replication project investigates and implements a content-aware image resizing algorithm which uses a simple image operator called seam carving [1]. Seam carving can be used for both reduction and expansion of images. To accomplish the tasks of the project, a thorough review of the paper was conducted to understand the seam carving algorithm presented. The report documents implementation of seam carving algorithm for seam removal, image enlargement, and image retargeting. A brief description of the algorithm is presented along with replicated results of Fig 5, Fig8, and Fig 7 from [1]. These replicated results have been compared for similarities and differences. Further, due to certain ambiguities present in the paper [1], assumptions made and difficulties are also discussed.

Seam Carving Algorithm

The fundamental idea behind the seam carving algorithm is to leverage the image *content constraints* and not only the geometric constraints to resize an image. The algorithm uses an energy function to determine and define the importance of pixels in an image. A *seam* in an image generated using the algorithm is defined as a connected path of *low energy* pixels which cross the image form top to bottom or left to right [1]. A number of energy functions can be used and there is no general optimal energy function. The “optimality” of an energy function depends on the image being resized.

As part of this project, I investigated the use of L1 energy functions for gradient magnitude for resizing Fig 5. The goal for Fig 5 was to resize the image from 466 x 700 pixels to 466 x 350 pixels. This tasks involved removing pixels from each row. To pick pixels the most intuitive criteria is pixels which are relatively unnoticeable and blend with the surrounding. For this purpose the energy function used characterizes the energy value of each pixel. Pixels with high energy are important and need to be preserved and low energy pixels can be removed. In order to maintain the structure of the image equal number of low energy pixels were removed from each row. A constrain of using 8-connected pixels in generating the seam which implies that the distance between pixel in 2 different row is less than equal to 1.

The energy function used as recommended by the authors [1] was:

$$e_1(\mathbf{I}) = |\frac{\partial}{\partial x}\mathbf{I}| + |\frac{\partial}{\partial y}\mathbf{I}|$$

Algorithm Implementation

The algorithm has been implemented for finding **vertical seams** by breaking down the algorithm into specific functions. The same code can be used for finding **horizontal seams** by transposing the input image before and after passing it through the seam finding pipeline. The transpose operation will convert a **466** x 700 (Fig 5) image into a 700 x **466** image. Remove a “vertical seam” from the 466 pixel dimensional edge and then transpose it back into a **465** x 700 image. The steps and functions are discussed below:

Step 1 Step 1 was implemented in 2 different way to check for robustness of the gradient function.

Method 1: Functions *x_gradient_gray(grayscale image)* and *y_gradient_gray(grayscale image)* used to calculate gradients in the height and width directions and applied on a grayscale image. Both gradient functions convolve a gradient matrices with the image. The absolute values of the gradient were added using l*1_energy_function(x_gradient,y_gradient).*

Method 2: The gradients can also be found using the Sobel() operator from opencv and the absolute values of the gradients added to find the *energy map*.

Step 2 Function *cumulativeenergy(img_energy, matrix_energy)* is used to calculate the minimum cumulative energy of every pixel. The function leverages dynamic programming to determine the min cumulative energy of every pixel. The function also keeps account of the pixel energy location used to achieve minimum cumulative energy at the next pixel for backtracking.

Step 3 Function *lowestcumulativeenergyseam(cumulative matrix_energy)* was used to backtrack and create a list of pixels which constitute the minimum energy seam. The backtracking starts from the last row and tracks the path back to the top of the image.

Step 4 *The last important function is removeverticalseam(img_gray,img_rgb,seampixellist)* which takes in the gray image, color image, and the list of pixels which form the lowest energy seam and removes those pixels to return an image which is narrower by 1 pixel.

Step 5 Finally, steps 1 - 4 are repeated to achieve the desired resized image. For Fig 5, the steps were repeated 350 times.

Note: *Transpose_image()* was used along with the vertical seam extraction pipeline to remove horizontal seams.

Comparison: Original vs Replicated result

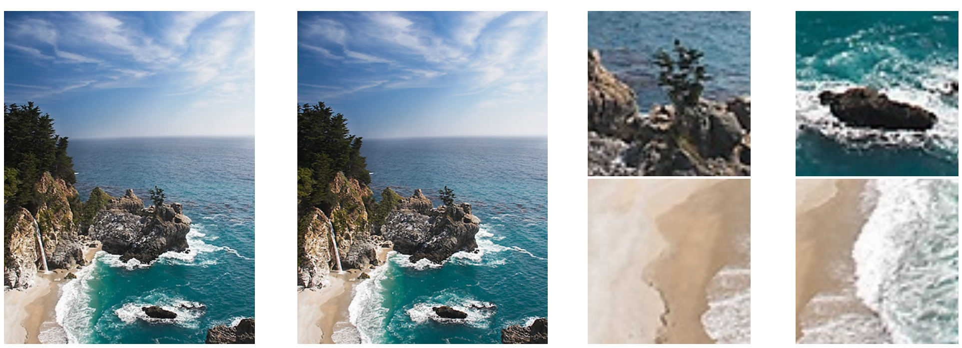

Figure 1. Fig 5 replicated from seam carving paper; Left image is replicated, Center image is the original, and right images are the comparison points

Fig 5 was replicated using the vertical seam pipeline outlined previously. The image on the left in Figure 1 is the replicated image and image on the right is the original image. The image are virtually identical and no differences were found. This was confirmed by comparing features like the shape of the tree top at the edge of the cliff, the lone rock shape in the middle, the contour of the wet portion of the sand on the beach, and the frothy white tide contour.

Figure 2. Fig 8 replicated images with seam insertion to enlarge the image. Left image is step 1 50% enlargement and Right image is step 2 with additional 50% enlargement

The insertion algorithm used the vertical seam removal algorithm to determine minimum energy seams for the same number of seams as required for the enlarging the image. The minimum energy pixels were identified and stored in a data container to be accessed for enlarging the original image. The enlargement was completed in 2 steps. enlarging wight wise from 239 pixels to 359 pixels in step one and then 479 pixels in step 2. One important caveat in expansion was to keep track of the pixel locations with respect to the original image to ensure when the pixels are used for enlargement the correct locations are targeted. The replicated results are close to the original results but have a slight secondary artifact in the highlighted area. This artifact is possible there due to the assumptions made to replicate the images.

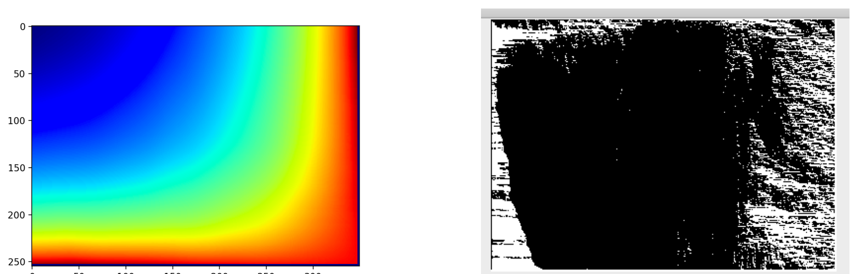

Figure 3. The transport minimum energy map and 1 bit mask for optimal retargeting constructed using seam carving algorithm.

Ambiguities and Assumptions

There are definitely some ambiguities in the paper and a few assumptions had to be made to implement the seam carving algorithm. The paper does not clearly mention the final energy functions used to generate the Figure for replication. A few energy functions are mentioned in the paper like gradient magnitude, entropy based, visual saliency based, and segmentation + l1 norm based. I made the assumption that the energy function to be used is the gradient magnitude with L1 norm of the gradients in width and height dimensions added together. Secondly, for the image enlargement, the paper does not clearly mention how to handle the location of the average pixel, i.e. whether to place it before or after the seam line pixels. I placed the pixels before the seam line pixels. Thirdly, for the transport map the paper mentions that T(0,0) is equal to 0 but does not clarify if the T function wraps around the image if the indices have a negative value. For instance what does T(0,-1) point to. I assumed that T function values with negative indices were equal to 0 as well. This led to the conclusion that the 0th row and 0th column are composed of energy values of vertical and horizontal seams respectively.

The video link for this project: https://www.youtube.com/watch?v=uAp3X809VTo&feature=youtu.be

References: [1] http://www.faculty.idc.ac.il/arik/SCWeb/imret/index.html