Implementing Seam Carving for Content--Aware Image Resizing

Russell Jones, rjk2@williams.edu, 17rjk2 Miller Zijie Zhu, zz4@williams.edu, 17zz4,
Julia Goldman, jsg2@williams.edu, 18jsg2 Yiheng Zhang, yz4@williams.edu, 18yz4

1. Introduction

The tremendous variety and flexibility of display devices in our modern world has put pressure on digital media producers to be able to readily adapt to changing specifications. In this paper, we tackle the problem of image resizing in relation to these issues. Common resizing methods today include scaling and cropping, but these methods do not operate at optimal efficiency as neither takes into account the content of the image (Figure 1). To resize images while effectively preserving the original image content, one can use a different approach called seam carving. This resizing method was first introduced by Shai Avidan and Ariel Shamir [Avidan and Shamir 2007][i] and our paper aims to reproduce and implement their seam carving method.

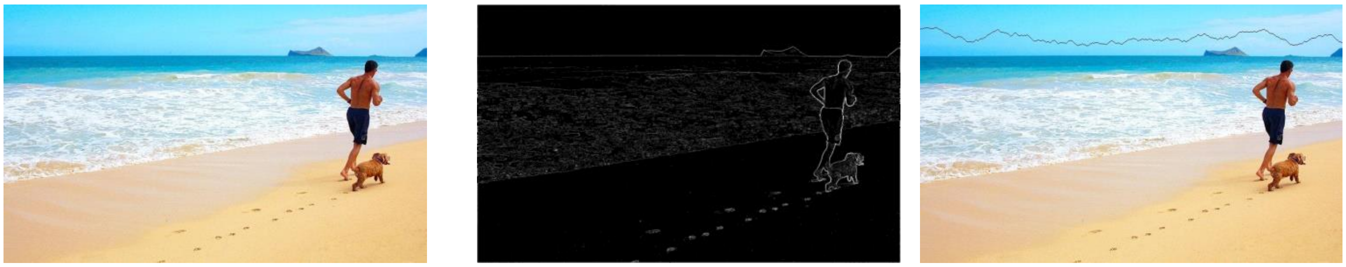
Figure 1: Scaling versus Seam Carving. Left picture is the original image at 400 x 400 pixels; middle picture is scaled to 200 x 400 pixels; right picture is seam-carved to 200 x 400 pixels. Note how the seam-carved image conserves the boat

2. Methodology

2.1 Approaching the Problem

Seam carving operates by removing pixels with low importance from the image, thereby conserving its key features while also changing the display specifications. The importance of a pixel is measured by its energy, which is calculated with an energy function. There are many different energy functions based on different criteria. For our project we chose a function similar to the Sobel operator and found that it worked well for most images [Krishnamurthi 2012][ii].

To balance the goals of removing the lowest energy pixels and of conserving the overall structure of the image, we remove seams of low energy pixels from the image. A seam is a vertical or horizontal path of connected pixels that crosses the image. Horizontal seams have one pixel in each column of the image, and vertical seams have one pixel in each row of the image. With dynamic programming, we can calculate the seam with the lowest total energy. By then adding or removing vertical and horizontal seams with low energy, we can reduce or enlarge images while preserving their important parts.



(a) (b) (c)

Figure 2: (a) shows the original image, (b) shows the energy map of the image, and (c) shows the horizontal seam with lowest energy based on the energy map

2.2 Energy Mapping

The first step in the seam carving process is to transfer the data in an image file into data that we can manipulate. After the image data is retrieved, the energy of each pixel is calculated using an energy equation [Krishnamurthi 2012] that looks for the difference in RGB values between a pixel and its 8 neighbors. The Energy function is:

$$Energy(x,y) = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2}$$

As seen in (figure) the energy values can then be mapped in grayscale to create an energy map of all of the pixels.

2.3 Seam Carving Sum Algorithm

Once the energy map is calculated, dynamic programming is used to calculate the energy sums of the seams. The dynamic programming algorithm works by finding the minimum energy path to get to each pixel on a row by row basis. For the vertical seams, the vertical sum fields of the top row of pixels are set to their energy value. Then for the next row of pixels, the vertical sum field is set to the pixel's energy value added to the minimum of the energy sum of the three possible "parent" pixels that are above them. This process is continued until the vertical sums for the last row of pixels are set. By tracing upwards the pixel with smallest energy sum, we can find the vertical seam to remove. An almost identical process happens during horizontal sum calculation, with the seams starting on the left edge of the image and moving right.

The above sum calculation method is a "backward" method of finding seams, as it removes a seam only according to the current energy distribution of the image. We also implemented a "forward" sum calculation [Rubinstein et al. 2008][iii] that calculates sum fields based on what will happen if the pixel is removed. The algorithm works in a very similar way to the backward one, but instead of looking at the energy values for each pixel, the forward approach is concerned with the energy that is introduced to the image when a seam is removed, due to pixels having new neighbors. We use a very similar dynamic programming algorithm to calculate the forward sums for each pixel, and in our command-line interface, the user can choose to use either forward or backward seam carving.

2.4 Seam Removing and Adding

Once the sum fields have been calculated, seams can either be removed or added to the image. For seam removal the pixels are simply taken out of the image one seam at a time. After every seam removal the energy and sum fields of the remaining pixels are reset to create a better final image. For vertical seams, this is done by erasing the individual pixels in the seam one at a time, but for horizontal seams we bring the seam of pixels to be removed down to the bottom row of the image before we delete them due to reasons related to our underlying data structure. For seam addition, the lowest energy seam is found and then a new parallel seam is added to the image directly below it for horizontal seams and to the right of it for vertical seams. The color for the added pixels is then set to an average of the color of pixels directly above and below them for horizontal seams and to the left and the right of them for vertical seams. For added seams, the energy of the new seam is then artificially increased using a special energy field, as this design makes it less likely that new seams will congregate in one area, and thus create artifacts in the image.

2.5 Marking Specific Image Portions

In our implementation we also give the user the ability to mark areas of the image to be conserved or erased. This is accomplished by using the special energy field mentioned above. If the user marks an area to be conserved, then the special energy field of those pixels will be kept artificially high, making sure that seams do not pass through those areas. If the user marks an area as to be erased, then the special energy field of those pixels will be set artificially low, giving incentive for low energy seams to be found passing through those areas.

3. Results

3.1 Man and Tower

(a) (b)

Figure 3: (a) is the original image with 1428 x 968 pixels, and (b) is the resized using backward seam carving to 1128 x 968 pixels

3.2 Man and Dog on Beach

(a) Original “Beach”

(b) Shrinked “Beach”

(c) Backward Amplified “Beach”

(d) Forward Amplified “Beach”

(e) Preserve Man And Remove Dog

(f) Successful Preservation and Removal

3.3 Japanese Couple

(a) Remove Girl

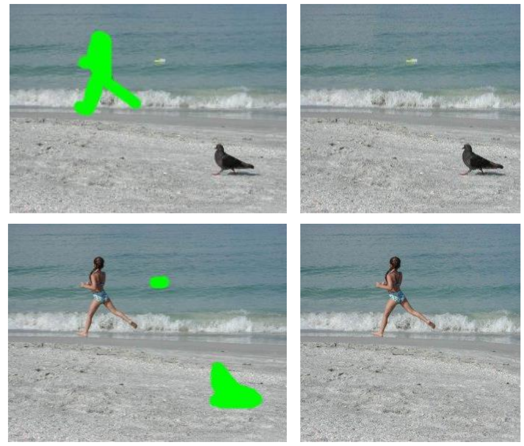
(b) Original Japanese Couple

(c) Remove Guy

(d) Marks

3.4 Girl, Pigeon, Frisbee

(a) Original Girl, Pigeon, Frisbee



(b) Remove Different Objects

3.5 Three People on Beach

(a) Original Three People on Beach

(b) Remove Different People

3.6 Seashore

(a) Original Seashore

(b) Shrinked Seashore

3.7 Lake Tree

(a) Original Lake Tree

(b) Backward Amplified Lake Tree

(c) Forward Amplified Lake Tree

4. References

[i] Avidan, S., Shamir, A.: "*Seam carving for content-aware image resizing*," In: *SIG-GRAPH* (2007)

[ii] Rubinstein, Michael, Ariel Shamir, and Shai Avidan: "*Improved Seam Carving for Video Retargeting*," *ACM Transactions on Graphics* 27.3 (2008)

[iii] Krishnamurthi, Shriram. "*Programming with Data Structures and Algorithms,*" *Seam Carving – CS19 Assignments* (2012)