Implementing the Seam Carving Algorithm for Content-Aware Image Resizing

19ter1, Thomas Edward Ragucci, ter1@williams.edu
19zl2, Zhiqi Li, zl2@williams.edu

I. Introduction

To resize an image, we often scale the image according to the specified height and width. However, simple scaling does not preserve the perspective of the content unless the new dimensions are proportional to the original dimensions. Therefore, the seam carving algorithm [Avidan, Shamir] becomes useful to resize images while preserving the perspective of the content. By repeatedly removing seams of lowest importance (i.e. where there is a minimum difference in color between neighboring pixels), the more important content is preserved after resizing. This paper attempts to reproduce the implementation of the seam carving algorithm to resize images and to remove specified objects in images.

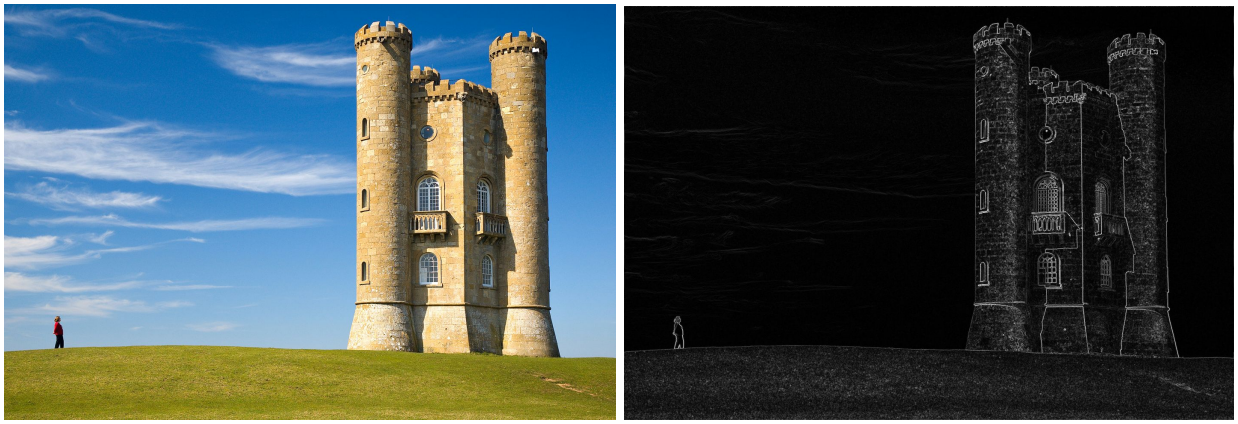
II. Methodology

We approach the implementation by dividing the seam carving algorithm into the following subtasks: (1) generate an energy map; (2) find the lowest energy seam via dynamic programing; (3) remove or add the seam.

1. Generate an energy map

The energy of each pixel is calculated by an energy equation, in which the energy is represented by the difference in grayscale value of the specified pixel with its four neighbors, top, bottom, left, and right. [Hug, Ginsburg, and Wayne] To account for the edge pixels, we pretend that there is a border of black pixels outside of the image.

$$e(x,\ y) = \sqrt{(g(x,\ y+1) - g(x,\ y-1))^2 + (g(x+1,\ y) - g(x-1,\ y))^2},\ \textit{where}\ g(x,y) = \textit{grayscale}(x,y)$$

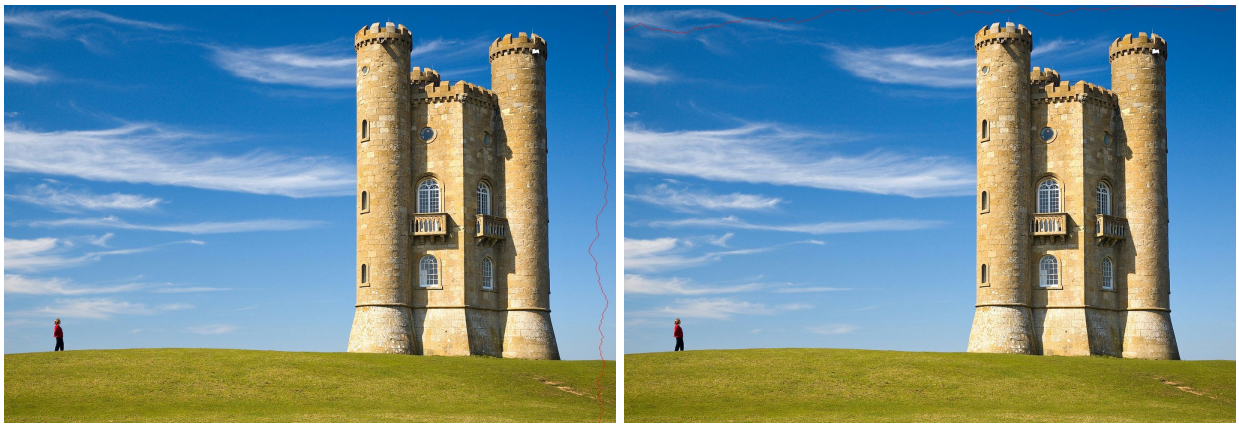


(a) (b)

Figure 1: (a) is the original image of the Broadway Tower in Worcestershire, UK (1428*968). (b) is a visualization of the energy of each pixel

2. Find the seam of lowest total energy

The lowest cumulative energy of all possible seams starting with a specified pixel can be found using dynamic programming [Avidan, Shamir]. The cumulative (vertical) energy at pixel (i, j) is shown as $M(i, j) = e(i, j) + min(M(i-1, j-1), M(i-1, j), M(i-1, j+1))$. By incrementing the energy of the pixel with the minimum cumulative energy of seams originating from the pixel's three neighbors in the next row (if looking for a vertical seam) or next column (if looking for a horizontal seam), the entries in the top row or column become the lowest total energy of all possible seams originating from the pixels. Therefore, the optimal seam can be found by starting from the pixel holding the minimum cumulative energy and recording the indices of pixels that have contributed to that total energy.



(a) (b)

Figure 2: In (a), the lowest energy vertical seam has been marked in red (far right of the image). In (b), the same has been done for the lowest energy horizontal seam (near the top of the image).

3. Removing & adding seams

After obtaining the optimal seam, it can be either removed or duplicated (added). To remove a seam, a copy of the image is created with the pixels specified in the seam removed, thus decreasing the height or width of the image by one pixel. This is repeated until the desired image height and width are achieved. Additionally, between each seam removal, the energy map of the image is recalculated to account for changing energies due to the deletion of the previous seam. This causes our program to take a longer time to process an image; however, it allows us to generate a more seamless (pun intended) and better quality image. Alternatively, to add seams, repeatedly inserting optimal seams does not work because the same seam will be added repeatedly and the enlarged image will be stretched. This makes image enlarging harder to implement than seam removal. One possible way to implement it is to find and insert the k most optimal seams (with no intersecting pixels) at once to increase the height or the width of the image by k pixels.

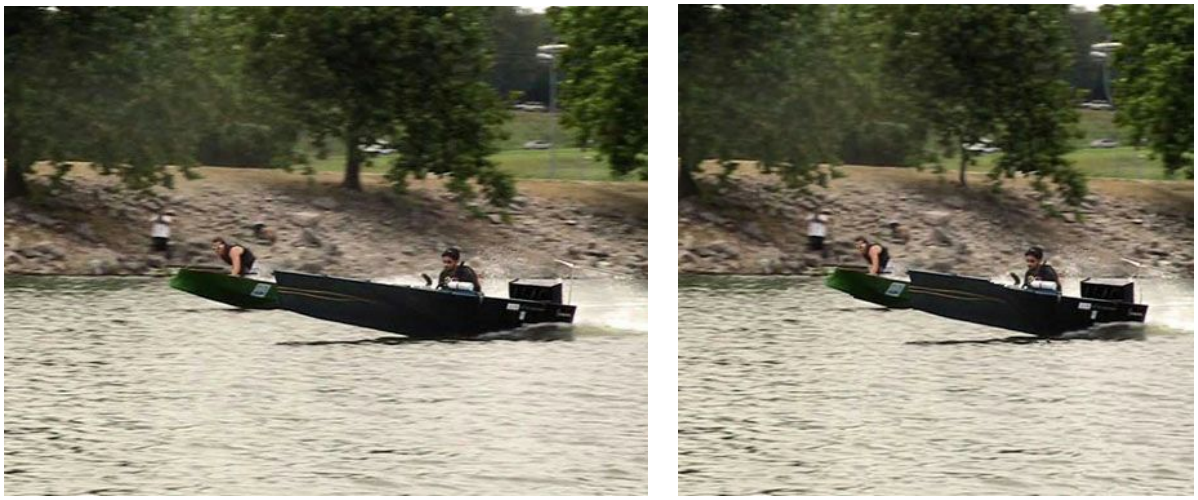
4. Object conservation & deletion:

In order to specifically remove or conserve an object or section of an image, we can artificially deflate or inflate, respectively, the energy values of the pixels that comprise the given object or section, which will therefore lead the algorithm to favor or disfavor said pixels for removal. [This is not in our

implementation, but would be somewhat easy to implement, assuming that a sufficiently convenient way to specify pixels for energy level modification is created.]

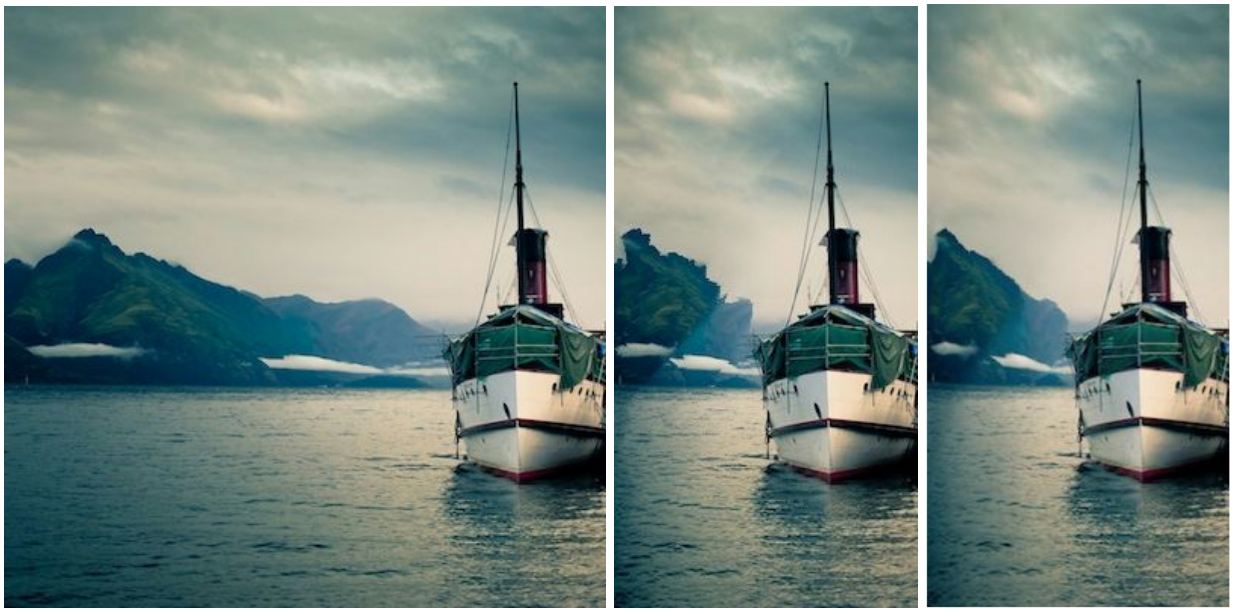
III. Results

1. Image resizing by removing seams:



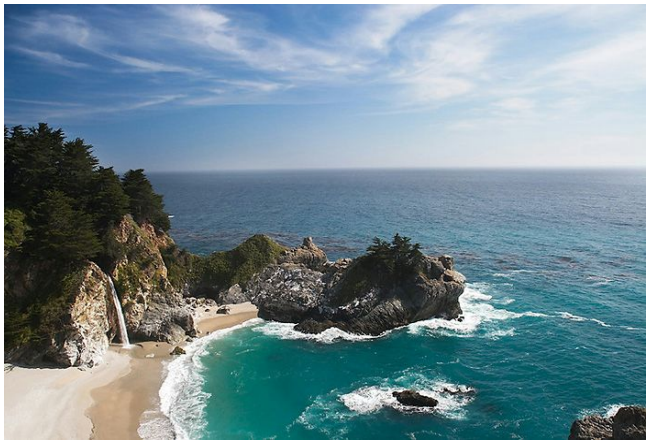
(a) (b)

Figure 3: (a) is the original image of a speed boat (500*380). (b) is the image after removing 100 vertical seams via our seam carving algorithm (400*380).

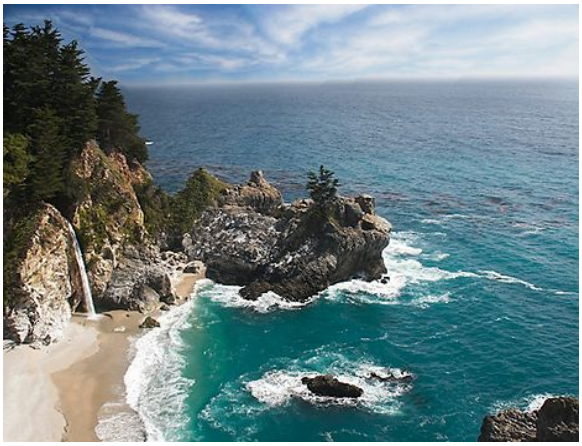


(a) (b) (c)

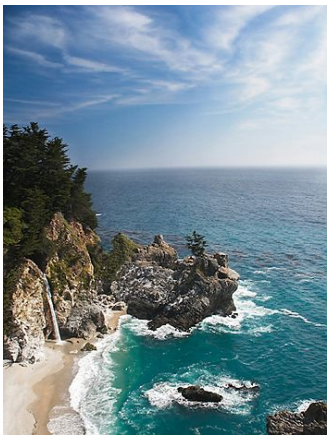
Figure 4: (a) is the original image of a boat. Images (b) & (c) are the same image after removing 200 pixels, with the algorithm we implemented and the algorithm implemented by previous students, respectively.



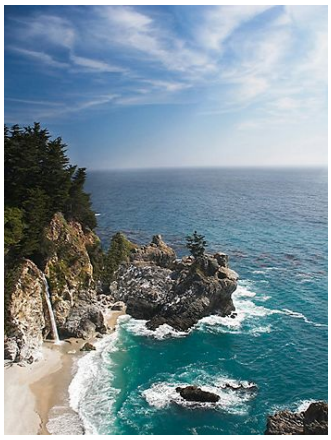
(a)



(b)

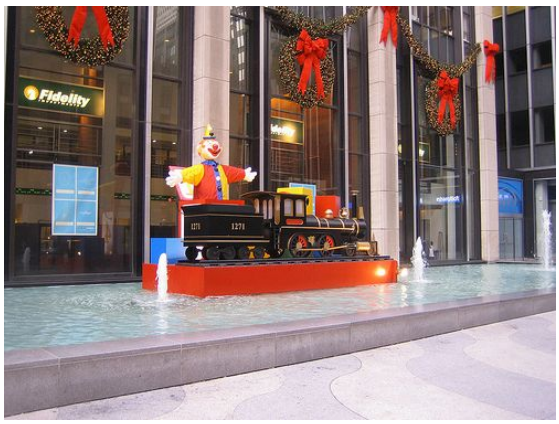


(c)

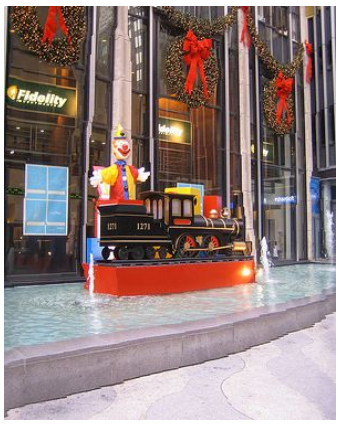


(d)

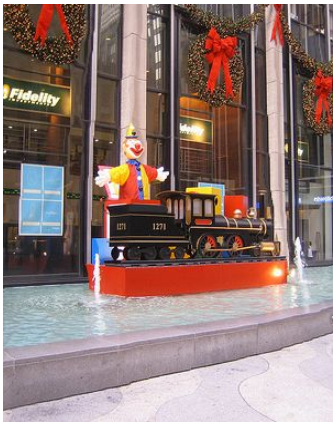
Figure 5: (a) is the original image of some landscape (700*466). (b) is the landscape resized to 466*350 by removing vertical and horizontal seams by our implementation. There's slight distortion of horizontal lines. (c) & (d) are the same image after reducing its width by 50%, with the algorithm we implemented and the algorithm in the original paper.



(a)



(b)

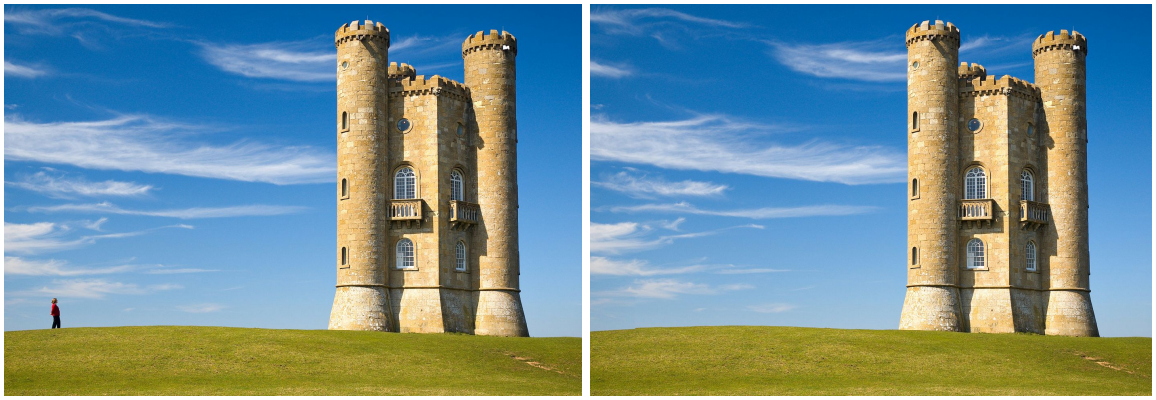


(c)

Figure 6: (a) is the original image of a Christmas scene. Images (b) & (c) are the same image after removing 200 pixels, with the algorithm we implemented and the algorithm in the original paper. Note that though both results do not preserve the vertical lines in the background, but our result has slightly more distortion in both foreground and background. Our implementation most likely has a slightly worse

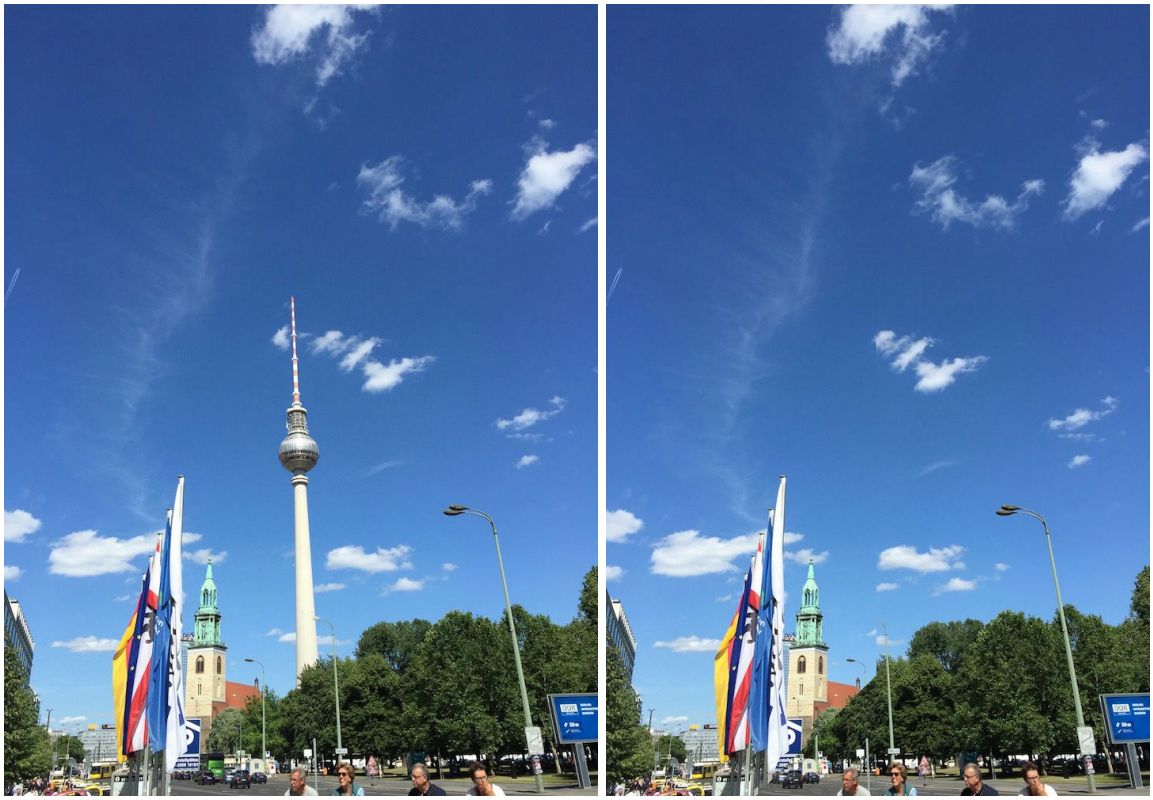
result because we used a different energy function. Note that vertical seam removal in general does not preserve the vertical shapes in images very well and the same applies with horizontal seam removal and horizontal shapes.

2. Object Removal



(a) (b)

Figure 7: (a) is the original image. (b) is the same image after removing the person standing in the field with the seam carving algorithm. Note that (b) was not cropped.



(a) (b)

Figure 8: (a) is the original image of a blue sky. Because we didn't want the Berlin TV tower to ruin the shot of the lovely sky, we used seam carving to remove it (b).

IV. References

Avidan, S., Shamir, A., *Seam Carving for Content-Aware Image Resizing* (2007)
http://www.eng.tau.ac.il/~avidan/papers/imretFinal.pdf
http://www.cs.princeton.edu/courses/archive/fall13/cos226/assignments/seamCarving.html
https://en.wikipedia.org/wiki/Seam_carving
http://cs.brown.edu/courses/cs129/asgn/proj3/
http://inst.eecs.berkeley.edu/~cs199-ap/
http://www.faculty.idc.ac.il/arik/SCWeb/imret/