Seam Carving for Context-Aware Resizing

Matthew Bollom
ECE 533
December 19, 2012

Introduction

The amount of ways that we can consume digital content today has increased exponentially since the advent of the Internet. New phones, smartphones, tablets, computers, and other internet-connected devices are released at a staggering rate, allowing us to use a variety of devices within our daily lives. To account for the wide variety of screen sizes available to view content, designers have long had the ability to dynamically change the layout of content depending on the screen size of the device. For example, a web page may appear one way in a desktop computer's web browser, but the same page in a smartphone web browser will be organized differently to account for the smaller screen and additional space needed for the user to interact with the page via touch. However, images do not have the same ability to dynamically change size.

To resize images for use in different sized spaces, one must manually change the image size. The most common methods for doing so are through resizing and cropping.

Resizing is a commonly used method that uses the principles of resampling. An image that needs to be reduced in size can be downsampled by eliminating the number of pixels. A variety of operators can be performed; one could remove rows or columns from the image or one could reduce the overall resolution of the image (i.e. resample from 300 dpi to 150 dpi). Both methods are efficient and produce excellent results from some images. However, it is possible that the resized image will have distorted details depending on the amount of downsampling. In addition, the content within the image may become smaller, an undesirable side effect if the content was already small to start with.

To see the effect of resampling, consider the image in Figure 1 (courtesy of the author's personal image library). This original image is of size 480x320 px. Figure 2 demonstrates the effect of resampling the image so its width is now 240 px, a reduction of 50%. As expected, the image is now distorted. For example, the sun in the background is no longer a circle. In addition, the mast lines running at an angle at the right side of the image are no longer straight. Because of these effects, Figure 2 appears unnatural.

On the other side, an image can be upsampled to enlarge it. Upsampling inserts pixels into the original image and interpolates those pixel values based on known pixels using methods such as nearest neighbor interpolation, bilinear interpolation, or bicubic interpolation (Gonzales & Woods, 2008). This method can produce results that also distort the image with the appearance of stretching.

The other easily performed resizing operator is cropping. Rather than removing pixels from throughout the entire image, cropping removes pixels from the borders of the image. This has the desirable effect of minimizing distortion in the image because all the pixels of interest are preserved, but it will remove objects that appear at the border of the image. This may be undesirable if those objects provide details necessary to understand the image. For example, Figure 3 shows cropping the bottom 80 px off the image. The boats in the lower right corner are no longer present and the source of the wires running along the right side of the image is no longer known. Overall, this picture is not distorted, but without the content from the bottom, a viewer cannot understand the entire image.

A better approach would be to add or remove pixels from different parts of the image rather than along the same row or column. Avidan and Shamir have proposed an operator that ranks pixels based on an energy function that they title "seam carving" (2007). The net result of using the seam-carving operator is that it permits context-aware resizing of an image.

This project implemented Avidan and Shamir's seam carving operator and tested it on a suite of images. This operator is implemented in Adobe Photoshop as "Content-Aware Scale", but implementing the operator allowed for an understanding and study of the operator in detail.

Figure 1. The original image of size 480x320 px. Courtesy of the author's personal image library.

Figure 2. Image size changed via downsampling. This image is 240x320 px. Note the distortion of the sun and the mast lines on the right side of the image are no longer straight as compared to the original image in Figure 1.

Figure 3. Image size changed via cropping. This image is 480x240 px. Note that the boats in the lower right corner are no longer present and the source of the wires running along the right side of the image are no longer known.

Approach

To resize an image in a context-aware fashion, there must be a means to designate the importance of the pixels. Pixels that are important usually consist of pixels that contain the details of the image. Pixels that are less important are usually in constant intensity parts of the image and do not provide additional contextual detail. When it come time to resize the image, the operator should act over the less important pixels as modifying the content of the image will be noticeable. For instance, removing pixels that provide no additional detail to the image will not change the content of the image. In Figure 1, the human brain ranks the sun as important in this image, so removing the pixels of the sun would change the content of the image significantly. A function is therefore desired such that it ranks those pixels as more important.

This function is often implemented as an energy function. Avidan and Shamir recommended the gradient as an energy function. The gradient is defined as

$$\bigtriangledown f \equiv \text{grad}(f) = \begin{bmatrix} \text{g}_x \\ \text{g}_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad [1]$$

where *f* is the image with coordinates (*x*,*y*). This vector points in the direction of the greatest rate of change at location(*x*,*y*) in *f* (Gonzales & Woods, 2008). This means that the vector at location (*x*,*y*) will point in the direction perpendicular to an edge in the image. One can then further refine the gradient in an energy function by finding the value of the rate of change by summing the absolute values of the individual components of the gradient at pixel (*x*,*y*):

$$e(f) = \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right| = |\text{g}_x| + |\text{g}_y| \quad [2]$$

where *e* is the energy function. Thus, the larger the value of the energy function at pixel (*x*,*y*), the greater the change in value at that pixel, signifying content at that pixel.

Given this basic energy function, one can then traverse the image to look for the pixels with the smallest energy. Avidan and Shamir discuss a variety of methods for removing these pixels of lowest energy such as to remove the k smallest energy pixels, remove a row or column of lowest energy, or to remove k pixels from each row or column. The results of these operations results in images with holes or heavily distorted images, both unacceptable results.

To produce better results, they proposed an operator that is allowed to move between rows and columns of the image. This operator, called a seam, is an 8-connected path of pixels such that it only contains one pixel from each column of the image (a horizontal seam) or one pixel from each row of the image (a vertical seam). Given a *nxm* image, we can define a horizontal seam as

$$s^y = \left\{s_j^y\right\}_{j=1}^m = \{(j, y(j))\}_{j=1}^m$$

such that [3]

$$\vee j, |y(j) - y(j-1)| \leq 1$$

where *y* will be a mapping of each column to a row ([1,...,*m*] → [1,...,*n*]). If one exchanges the order of variables, *n* for *m*, and *i* for *j*, the equivalent definition for a vertical seam can be obtained.

Given the definition of a seam and an energy function, one can traverse the image looking for the seam of lowest energy. This seam will correspond with pixels that are least important. If one desires to find the horizontal seam of lowest energy, one can use

$$\begin{aligned} M(i,j) = e(i,j) + \\ \min(M(i-1,j-1), M(i,j-1), M(i+1,j-1)) \end{aligned} \quad [4]$$

where *M* is cumulative minimum energy. That is, add the current pixel (*i,j*) to the seam that has the minimum cumulative energy thus far. If *e(f)* is already computed, it is a trivial matter to start at the second column and perform this algorithm for each pixel in each column successively until the end of the image is reached. One can then find the lowest minimum energy at the last column and work backwards from that pixel to recover the seam. This seam will correspond to the seam of lowest energy in the image.

From there, the seam of lowest energy can then be removed from the image, resulting in an image that is now one pixel shorter in height. This process can be repeated many times until the desired size is met.

To remove a column of pixels (i.e. make the width smaller), one can compute the cumulative minimum energy of a vertical seam by starting at the top of the image and working to the bottom. This will result in a 8-connected seam that contains 1 pixel from each row. Removing this seam will reduce the width of the image by 1 pixel.

Work Performed

To implement this algorithm, Matlab was chosen as it offers significant ease in working with images. In addition, it provides functions such as computing the directional gradients, improving the ease of implementation.

To start, the energy function *e(f)* (Equation 2) was computed. Figure 4 shows the numerical results of evaluating *e(f)* on Figure 1.

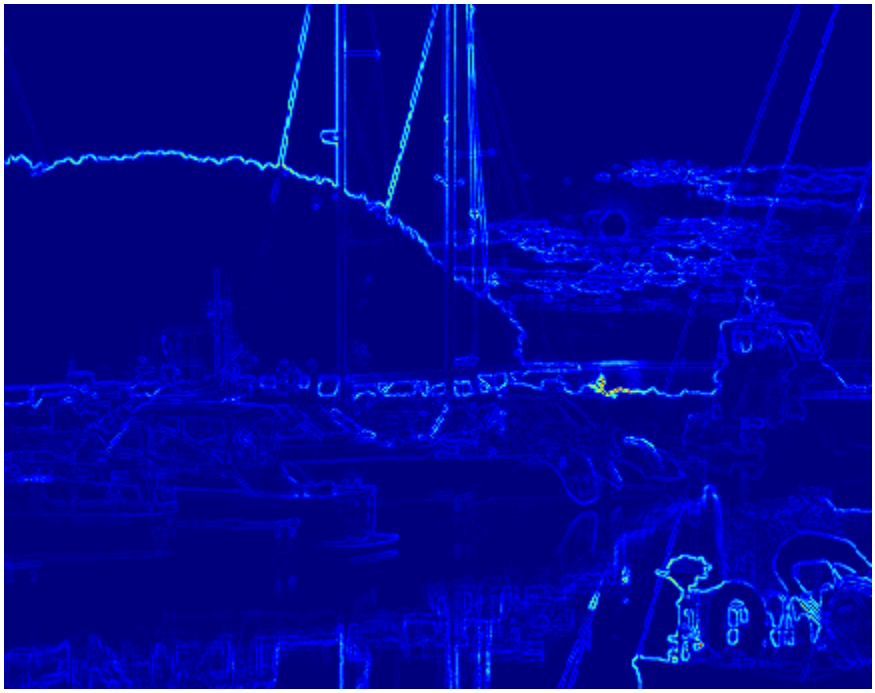

Figure 4: *e(F)* evaluated on Figure 1. Blue corresponds to a value of approximately 0 while brighter areas correspond to an energy of significant value.

The colormap used to display this image places energy values of approximately 0 at blue, while areas of significant energy are represented by brighter colors. When compared to Figure 1, the brighter colors correspond to the "edges", or content, of the image. Thus, when this image is resized, those pixels should not be removed if possible.

Given the energy function, dynamic programming was then used to arrive at the cumulative energy function *M*. Dynamic programming starts at the second column of the image, then computes *M* for each pixel in each column. When it reaches the last column of the image, it traces the path backwards corresponding to the minimum energy seam. M for the horizontal direction going left to right can be seen in Figure 5.

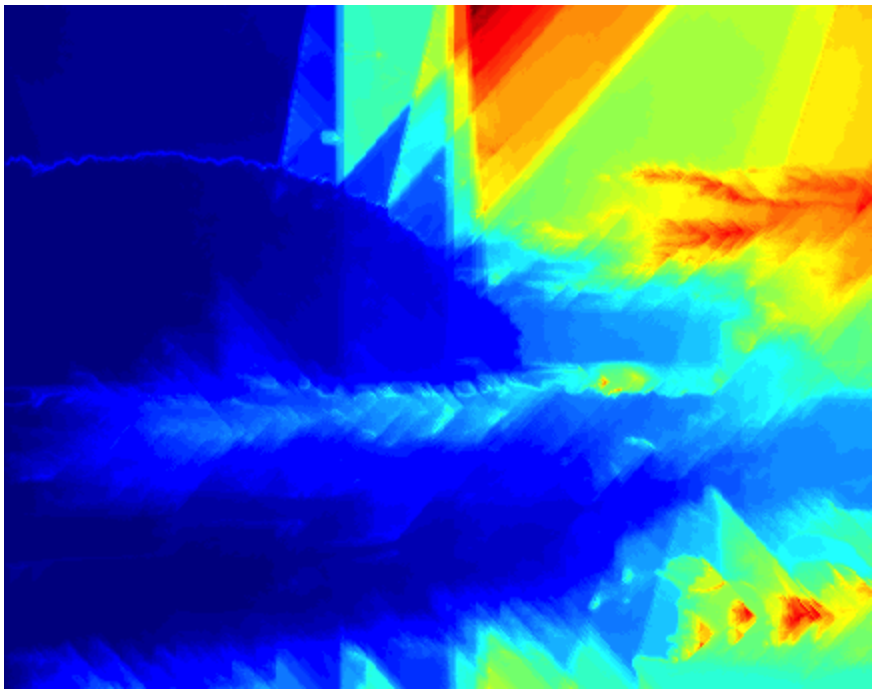

Figure 5: *M* for Figure 1. Blue corresponds to a cumulative energy of approximately 0 while red corresponds with the most energy.

Tracing the seam backwards with the lowest energy results in the following seam:

Figure 6: The seam of lowest cost for Figure 1. Note that the seam does not go through any major content of the image, the desired effect.

This seam is 8-connected and it corresponds with the cumulative energy function computed in Figure 5. Also notice that the seam does not pass through any major content of the image, the desired effect.

This seam can then be removed and the process will repeat until the image has reached the desired size. Figure 7 shows the context-aware scaled image that was reduced in height by 50 px. The content of the image is preserved, and comparison of this image to Figure 1 shows that most of the pixels that were taken out in the shadows of the water and at the horizon line. Close examination of the image will also show a break in one of the mast lines along the horizon at the right side of the image.

To remove pixels from a vertical direction, another algorithm can be implemented to work along that direction. However, an easier approach is to rotate the image 90º, seam carve the image, then rotate the image back. This was the approach taken since it would allow for more time for other experimentation. Figure 8 shows M for Figure 1 in the vertical direction ($e(f)$ is identical).

Clearly, the order of seam carving the image will have an effect on the resultant image. As each seam is removed, the energy of the image will change, requiring the need to recalculate the energy each time. A variety of different approaches could be taken here, such as evaluating M for both directions and then removing the seam that will have the lowest energy. However, such an approach will require normalizing the energy based on the length of the seam as seams that are longer will naturally have a larger energy. Another approach, and the method taken here, is to perform all the seam carving along one direction and then the other. While this may produce sub-optimal results for some images, the approach worked for all test cases attempted.

With the basic algorithm complete, some additional modifications were attempted, namely with the energy function. A brief literature review indicates many possible energy functions to choose from. Two additional energy functions chosen include entropy and orienting the gradient along edges of the image.

Figure 7. Context-aware scaled image resized to a height of 291 px (reduced by 50 px) using seam carving. The content of the image is preserved.

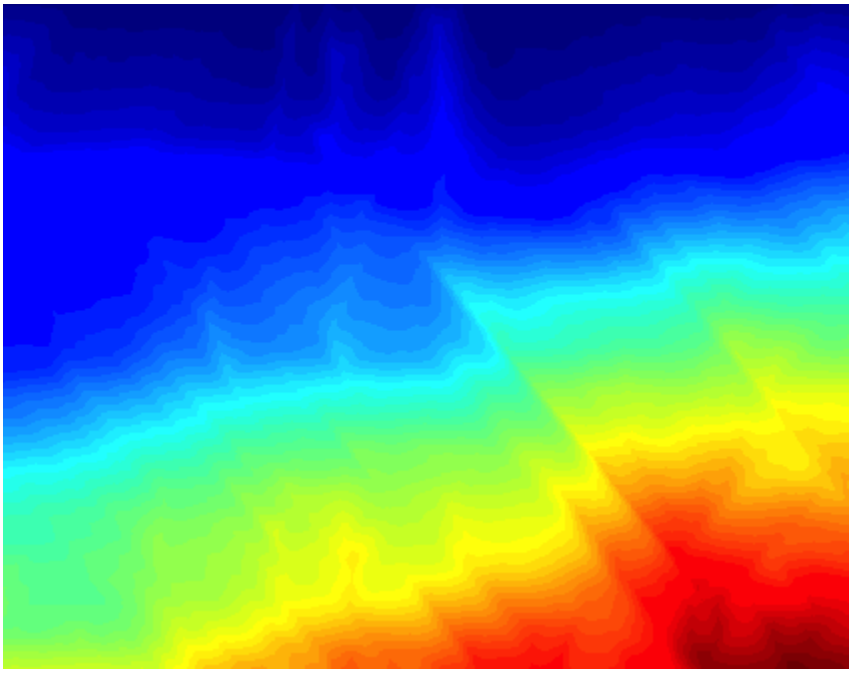

Figure 8. *M* for Figure 1 along the vertical direction. Blue corresponds to a cumulative energy of approximately 0 while red corresponds with the most energy.

The entropy of an image will tell how much contrast is in the image. An image with high entropy will have high contrast and most likely significant content. An image with low entropy will have many areas of constant intensity. Entropy can be calculated by

$$H = -\sum_{k=1}^{L-1} p_r(r_k)\, log_2\left(p_r(r_k)\right) \quad [5]$$

where H is entropy of an image with L intensity levels and $p_r(r_k)$ is the probability of intensity level r_k occurring. This equation assumes that the intensity levels of the image are independent of each other (Gonzales & Woods, 2008).

Since calculating the entropy over the entire image would result in a single number, a result that is not useful, the entropy was calculated over a 9x9 neighborhood of each pixel. The local entropy was then added to $e(f)$ to further refine the value of the energy

$$e_2(f) = e(f) + H_{9x9} \quad [6]$$

Evaluating $e_2(f)$ over Figure 1 yields the result in Figure 9.

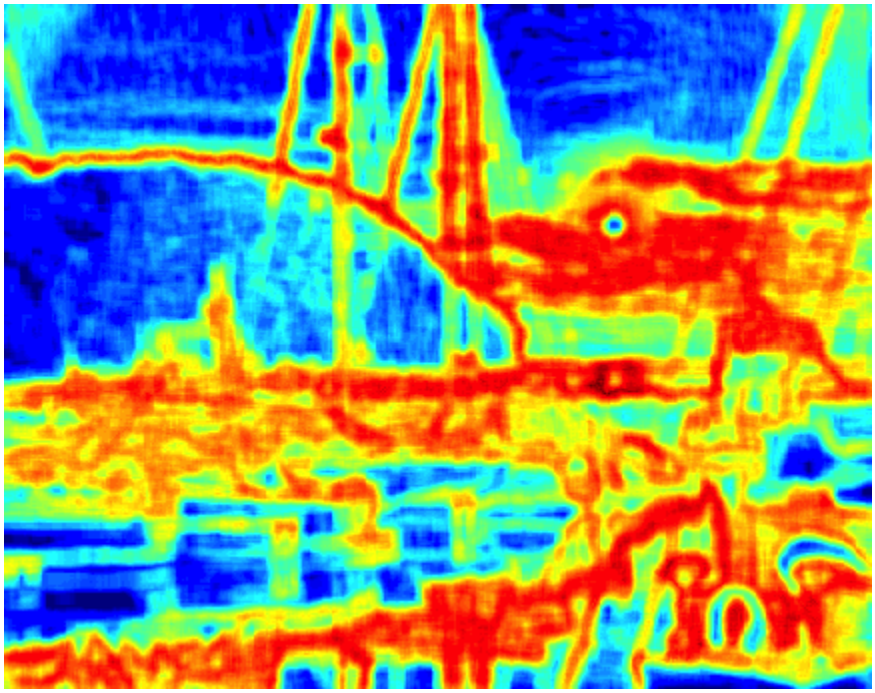

Figure 9. $e_2(f)$ evaluated on Figure 1. Blue corresponds to areas of low energy and red corresponds to areas of high energy.

This energy image also captures the essence of the content as well, but it does so by widening the locations of the content. Because of this, the cumulative energy function M should have a significantly different look than that of Figure 5. It does, as shown in Figure 10.

Another energy function suggested by Avidan & Shamir is to orient the gradient along edges. To do so, Dalal & Triggs (2005) have developed an operator they call "Histogram of Oriented Gradients" (HoG). This operator quantizes the gradient values into discrete bins. The bin that holds the most values is then the dominant direction. An 8 bin HoG was used over an 11x11 neighborhood of each pixel. The gradient at that pixel was then divided by maximum value of the HoG to arrive at a new energy function $e_3(f)$:

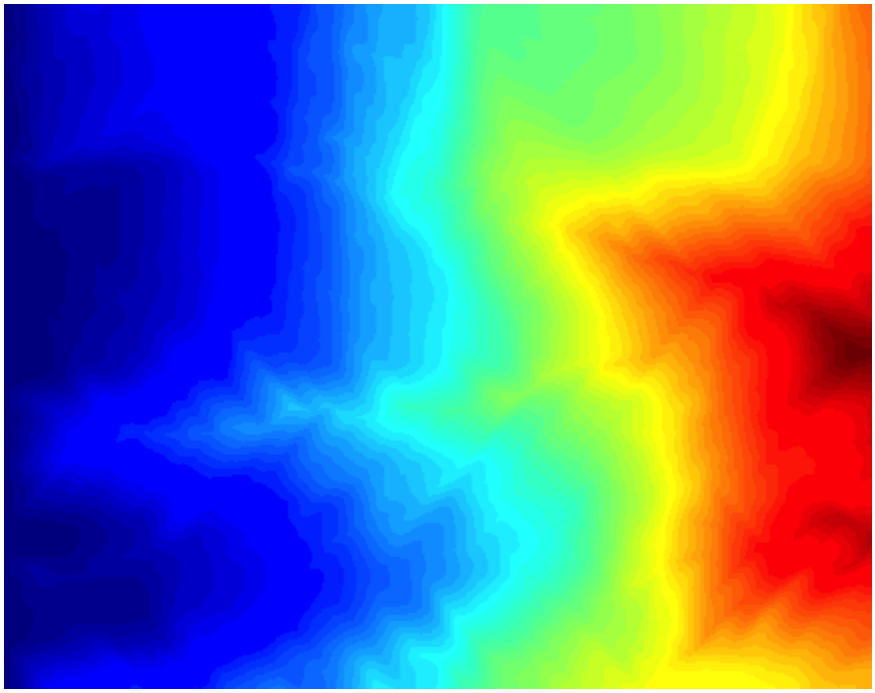

Figure 10. *M* calculated for Figure 1 using $e_2(f)$.

$$e_3(f) = \frac{e(f)}{\max(HoG_{11x11}(f))} \quad [7]$$

To implement the HoG, existing code was used. This code is available at http://www.mathworks.com/matlabcentral/fileexchange/33863-histograms-of-oriented-gradients.

Figure 11 shows the result of evaluating $e_3(f)$ on Figure 1 and Figure 12 shows the result of calculating M.

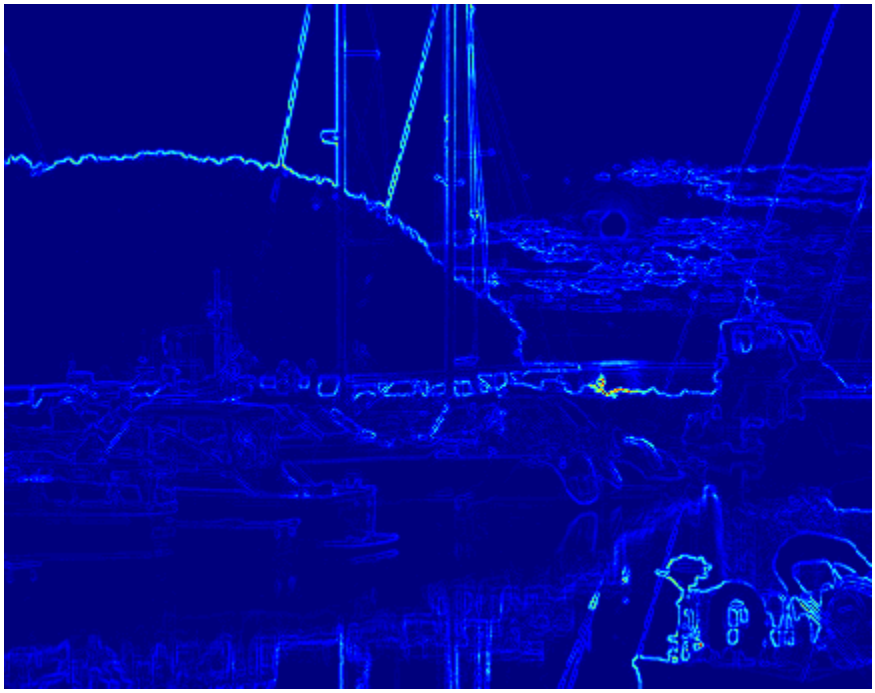

Figure 11. $e_3(f)$ evaluated on Figure 1. Blue corresponds to areas of low energy and red corresponds to areas of high energy.

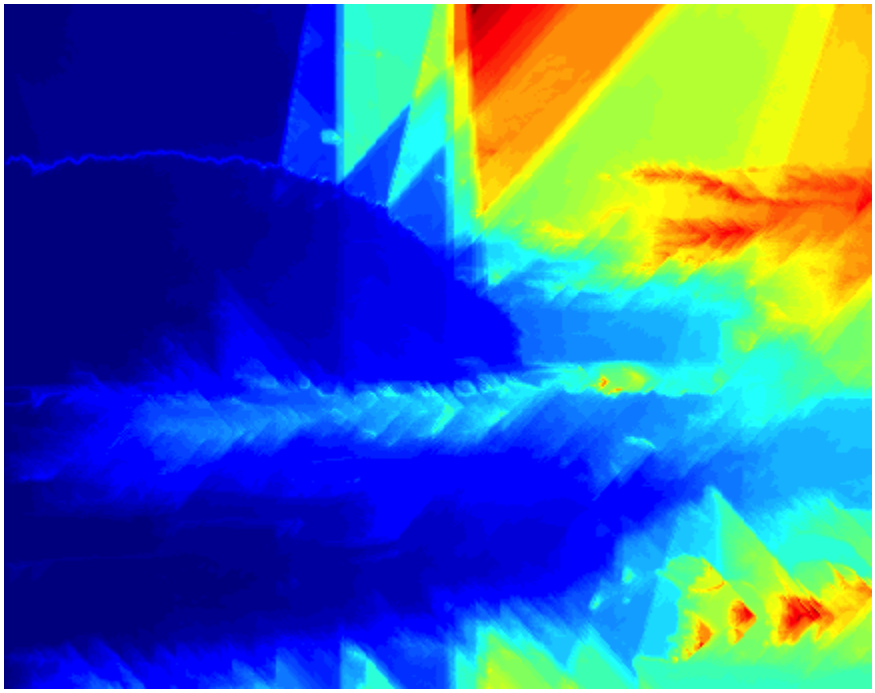

Figure 12. *M* calculated for Figure 1 using $e_3(f)$.

The result of $e_3(f)$ is nearly identical to that of $e(f)$. A scaling factor is the only difference between the two equations, so the maximum and minimums should appear in nearly the same place. Since these images use dynamic scaling, the minimum value is assigned blue and the maximum value is assigned red. Thus, the energy images appear nearly identical, but the numbers are scaled differently. In addition, close examination shows that some some of the values of the energy function are aligned along edges, the desired result.

For the rest of this paper, *e(f)* was used. This energy function had the fastest computational time and yielded good results for all test cases. However, some images may work better using a different energy function, so one must always make a decision as to which energy function to use at all times.

Results and Discussion

To test the functionality of the implemented algorithm, a variety of images were used. Figure 7 depicts the result when Figure 1 is reduced in height by 50 pixels using *e(f)*. Figure 13 is the result when both the height and width are reduced by 50 px. As discussed earlier, the order of seam removal makes a difference, but all images in this paper will present themselves by removing rows first, then columns.

Close examination of Figure 13 shows that the end result is still realistic even though there are some interesting artifacts. Namely, the horizon to the left of the boat has been shifted down and one of the mast lines at the right of the image is broken. The boats and sun are still intact, showing that the algorithm does leave important content in the image. Part of the reason why the mast line may have been broken is that it is a very small part of the image and it takes up very few pixels in each row. It may have cost less for the algorithm to remove some pixels rather than going around the mast lines. Once the mast line was cut, the algorithm continuously found that break and the lowest energy seam always went through the break.

The author believes this is a difficult test case for the algorithm because of the amount of detail in the image. If the image had more open space in the image, the seam carving algorithm will produce consistent, near-optimal resizing.

Figure 13. Result of running the seam carving algorithm on Figure 1 to reduce both the height and width of the image by 50 px.

To test the previous hypothesis, another image (Figure 14) was obtained. This image has areas of constant intensity, most notably in the sky. The results of reducing the height and width by both 100 px is shown in Figure 15. Careful inspection of the image shows there are no visible artifacts. Comparing the reduced image to the original image shows that most of the rows of pixels have been removed above the sun so the sun is closer to the top of the image. The columns have been removed from a few places - mostly on the right side of the image and to the left of the wave in the ocean. Removing these pixels also removed some of the beach near the water, but the result appears natural.

This algorithm can fail for certain types of images. For example, images with a lot of parallel lines that are not parallel to the edge of the image. When the lowest energy seam is found, it often must go through a parallel line. When enough seams are removed, the parallel lines are no longer continuous and appear broken. Figures 16 and 17 emphasize this effect.

Many additional test cases were conducted, but for the sake of brevity, they are not included here.

Although this project has focused mostly on reducing the size of images via seam carving, it is possible to enlarge an image with the same concept. Rather than removing a seam, one can duplicate the seam and average across the inserted pixel's neighbors. This creates a desirable effect of not breaking up the original content of the image and inserting pixels in areas where they will not be likely to be noticed. This will also help reduce the appearance of visual artifacts.

However, the way the seams are inserted needs to be carefully considered. If a seam is inserted, the next iteration of the algorithm will most likely choose the same seam again. This will cause a stretching of the same row of pixels across the image. Avidan & Shamir suggest finding many seams for removal, then duplicating each of them to increase the size of an image. Because of time constraints, the author was not able to implement this portion of the algorithm; but they are interested in doing so in the future.

Figure 14. Another source image for testing the seam carving algorithm. This image is 512x341 px. Note the areas of mostly constant intensity in the sky. Courtesy of the author's personal image library.

Figure 15. Figure 14 with the height and width of the image reduced 100 px each via seam carving. There are no visible artifacts in this image.

Figure 16. Another test image exhibiting parallels lines that are not parallel to the edge of the image. This image is 540x341px. Courtesy of the author's personal image library.

Figure 17. The results of seam carving Figure 16 by reducing the height and width of the image by 100 px each. The parallel lines are no longer continuous. However, the words "Franklin Barbecue" are still completely intact, showing the seam carving algorithm correctly recognized the words as image content.

Conclusion

In this paper, Avidan & Shamir's context-aware seam carving algorithm was discussed and implemented. Adding and removing pixels of an image via resampling or cropping can cause the image to become distorted. A better solution is use the energy of the image such that pixels of content are ranked as more important. A 8-connected path, called a seam, can then be found by tracing the minimum cumulative energy of the image as it is traverse. Removing this seam will effectively reduce the size of the image without affecting the content.

A variety of test cases were performed with the implemented algorithm. Although several energy functions were implemented, it was found that adding the absolute value of the gradients at each pixel provided sufficient results in a computational efficient manner. Images that had areas of low content, such as patches of constant intensity, had better results than images with a lot of content. Although the algorithm will still work for images with details, artifacts may be introduced, an undesirable effect. In either case, it was shown that this operator produces useful results.

References

Avidan, S., & Shamir, A. (2007, August). Seam carving for content-aware image resizing. In *ACM Transactions on graphics (TOG)* (Vol. 26, No. 3, p. 10). ACM.

Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (Vol. 1, p. 886-893). IEEE.

Gonzales, R. C., & Woods, R. E. (2008). Digital Image Processing (3rd ed.). Upper Saddle River, NJ: Pearson Prentice Hall.