Methods in Segmented Image Seam Carving

Weiss, Tomer
tweiss@cs.ucla.edu

Liu, Brian
liubrian7@ucla.edu

Abstract

Seam carving deals with the task of taking an input picture and resizing it to fit another screen type and aspect ratio. In this paper, we discuss possible methods to decrease the number of artifacts intoduced by seam carving, as well as decreasing run time of the original algorithm by using an image segmentation approach. Experimental results demonstrate that in every case our method resizes the image faster and in some cases reduces the number of artifacts introduced by the original seam carving algorithm.

CR Categories:

I.3.0 [Computing Methodologies]: Computer GraphicsGeneral—

I.4.10 [Computing Methodologies]: Image Processing And Computer Vision—Image Representation;

accuracy

Keywords: Image resizing, Image retargeting, Image seams, Content-aware image manipulation, Display devices

1 Introduction

With the recent advances in imaging technology, digital images have become an important component of media distribution. Images are frequently used in news stories, and people post their pictures online to be seen by family and friends. Images, however, are typically authored once, but need to be adapted for consumption under varied conditions. For example, pictures are often displayed on different screens, where the area available for the picture may have a different aspect ratio than the original image has for layout reasons. Dynamically changing the layout of web pages in browsers should take into account the distribution of text and images, resizing them if necessary. Also, the use of thumbnails that faithfully represent the image content is important in image browsing applications. A variety of displays are used for image viewing, ranging from high-resolution computer monitors to TV screens and low-resolution mobile devices. Recently, there has been a growing interest in media retargeting that is driven largely by the growing number of mobile devices used to view digital content. Even if technological advances allow for their resolution to increase, their physical area will still be small. Hence, rearranging the relative sizes of different objects in the image could still provide an improved viewing experience, despite the availability of more pixels.

This diversity of image consumption conditions introduces a new problem: images must be resized for optimal display or use in different applications. The process, also known as image re-targeting or image resizing, consists of modifying the image's aspect ratio and size in order to best satisfy the new requirements. The common approach for all media resizing works is first to define an importance map on the pixels of the media, and then use this map to guide some operator that reduces (or enlarges) the media size. However, straightforward image resizing operators, such as scaling, often do not produce satisfactory results, since they are oblivious to image content. To overcome this limitation, a class of techniques attempt to resize the images in a content-aware fashion, i.e., taking the image content into consideration to preserve important regions and minimize distortions. This is a challenging problem, as it requires preserving the relevant information while maintaining an aesthetically pleasing image for the user.

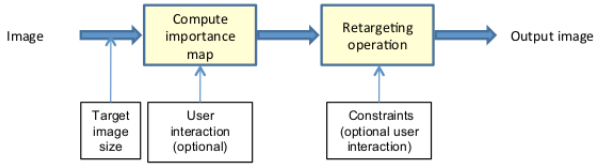

Figure 1: *Common image retargeting steps.*

There are numerous ways to define importance in media. [**?**] proposed an approach for changing the composition of objects in a given image in order to improve its aesthetic value, based on rules of thumb from photography such as the rule of thirds. Furthermore, other solutions have been contributed by the computer vision, computer graphics, and human-computer interaction communities. The definition of *important* can depend on the specific application being considered. There are different approaches for defining importance measures that specify the level of importance of pixels in the image. Also, the definition of what is important and what is unimportant is clearly subjective – there are situations where user interaction is unavoidable, and many techniques support the specification of important areas as an input provided by the user.

One of the automatic image retargeting techniques is Seam Carving, where the general idea is to decrease or increase the image width (or height) one pixel at a time, by removing or adding a seam of minimal importance. Hence Seam Carving can be viewed as a generalized cropping or scaling method. A seam is defined as an 8-connected path of pixels (from top to bottom, or from left to right of the image, depending on which dimension is being reduced) that contains only one pixel per row (or column). When the importance map is based on gradient energy, the first removed seam will be in a homogeneous area. The image is then readjusted by shifting pixels left or up to compensate for the removed seam, resulting in an image which is one pixel smaller, either on width or height. So the image changes only at the seam region, while the other areas remain intact. This method is repeated until the picture is shrunk to its new ratio. When a picture is enlarged by *important* k pixels, the *important* k seams of lowest energy are found (as if the image is being shrunk), and then each of these seams, labeled *important* s are artifically added into the picture. Each particular seam *important* s is found by averaging the pixel intensities orthogonal to it's direction, e.g. if you were adding a horizontal seam, you would average pixel intensities above and below it.

To summarize, our goal is to experiment with different combinations of Seam Carving and image segmentation, and by so to improve the results of the original paper.

2 Prior and Related Work

The seam carving technique by [**?**] is a popular, recently developed approach for content-aware image resizing. The general idea is to decrease or increase the image width (or height) one pixel at a time, by removing or adding a seam of minimal importance. Intuitively, if the importance map is based on gradient energy, the first removed seam will be in a homogeneous area. The image is then readjusted by shifting pixels left or up to compensate for the removed seam, resulting in an image which is one pixel smaller, either on width or height. Enlargement of an image according to the seam carving algorithm follows a similar approach, duplicating the k weakest seams, and then finding their value by averaging the pixels around those seams. The image changes only at the seam region, while the other areas remain intact. [**?**] observe that using gradient energy as the importance map gives satisfactory results, but other importance measures could be used, such as saliency map, entropy, and histograms of oriented gradients. The optimal seams are computed using dynamic programming, and an algorithm for resizing in both dimensions by choosing between optimal vertical or horizontal seams is also presented.

Generally, the best outcomes are achieved when there are enough seams of low-importance to be removed, since artifacts and distortions are created when seams cut through areas with important features. Moreover, since the energy function reflects feature strength only along the axes of the image coordinates, it cannot protect some prominent shape boundaries of arbitrary orientations. Also, seam carving does not work well when the input image is feature-rich or has a noisy background. To illustrate the limitations of the existing seam carving framework [**?**], we present Figure **??**, which is obviously feature rich – the two basketball players have a lot of distinct features and take up a large region of the total image's size, not to mention that the crowd provides a noisy background in which to run the algorithm. Based on the method of [**?**], we attempt to resize this image both in the horizontal and vertical direction.

Figure 2: *Left – original image. Right – resized image, result of original Seam Carving method (enlarged).*

As you could see in Figure **??**, there are artifacts that result from the removal of pixels from both players hands and legs, resulting in gross deformation of Magic Johnson's legs and a misalignment in the right arm/shoulder of Larry Bird. This example shows us that there is much room for improving the original method, especially with images with noisy backgrounds.

In [**?**], the optimal seams are computed using dynamic programming, and an algorithm for resizing in both dimensions by choosing between optimal vertical or horizontal seams is also presented. The technique can be used for enlarging the image, by finding seams to be removed and duplicating them. It produces impressive results when there are enough low-importance seams to be removed, but creates distortions and artifacts when seams cut through important areas.

Motivated by the compelling applications and the challenges related to the problem, we proposed to use image segmentation to improve the seam carving algorithm. There are other methods that rely on segmentation to assign saliencies to different regions in the image. Previous work by [**?**] and [**?**] suggests to segment the image into regions and then assign saliencies to each region by considering heuristics such as the region size, position in the image, and relationships between neighboring regions. [**?**] proposed to segment the image using mean-shift and assign saliencies to the obtained regions by a combination of bottom-up and top-down features. Also, [**?**] ,authors of the original Seam Carving algorithm, suggested that users could scribble on salient areas, artificially raising their image energy values, making it so their pixels would not be removed in the seam removal process.

3 Technical details

3.1 Overview

Our method builds upon the basis of the original seam carving algorithm, where we use the Laplacian to find the energy levels of each pixel in the input image. After finding the energy of each pixel, we segment the image to smaller, mini pictures in a grid like fashion – vertical or horizontal. For example, if we segment the image to 3 by 3 segments, then we get a total of 9 segments. On each such segment we engage the next step of the seam carving algorithm – i.e. finding the seams. This step is done via the basic dynamic algorithm for finding seams, but on each segment separately. Furthermore, the number of seams in each individual segment to be removed is proportional to the total number of seams that must be removed in the entire image.

3.2 Image Energy

Our initial approach, similar to the original paper, is to preserve the image's energy, by removing unnoticeable pixels. That leads to the following energy function that was used in the original paper. We use $I(x)$ to denote the input image, where $x = (x, y)$.

$$e_I(I) = |\frac{\partial}{\partial x}I| + |\frac{\partial}{\partial y}I| \quad (1)$$

We need a resizing operator that will be less restrictive than cropping or column removal, but can preserve the image content better than single pixel removals. This leads to our strategy of seam carving and the definition of image seams. Let I be and $n \times m$ image and define a vertical seam to be:

$$s^x = \{s_i^x\}_{i=1}^n = \{(x(i), i)\}_{i=1}^n, s.t. \forall i |x(i) - x(i-1)| \leq 1 \quad (2)$$

where x is a mapping $x : [1, ..., n] \longrightarrow [1, ..., m]$. That is, a vertical seam is an 8-connected path of pixels in the image from top to bottom, containing one, and only one, pixel in each row of the image. Similarly, if y is a mapping y : [1, . . . , m] [1, . . . , n], then a horizontal seam is:

$$s^y = \{s_j^y\}_{j=1}^m = \{(y(j), j)\}_{j=1}^m, s.t. \forall j |y(j) - y(j-1)| \leq 1 \quad (3)$$

The pixels of the path of seam s (e.g. vertical seam $\{s_i\}$) will therefore be $I_s = \{(I(s_i)\}_{i=1}^n = \{I(x(i), i))\}_{i=1}^n$. Note that similar to the removal of a row or column from an image, removing the pixels of a seam from an image has only a local effect: all the pixels of the image are shifted left (or up) to compensate for the missing path.

3.3 Dynamic Programming Algorithm

Similar to the original method, we find the optimal seam for each segment using dynamic programming. The first step is to traverse the image from the second row to the last row and compute the cumulative minimum energy M for all possible connected seams for each entry (i, j), and S specifies the seam.

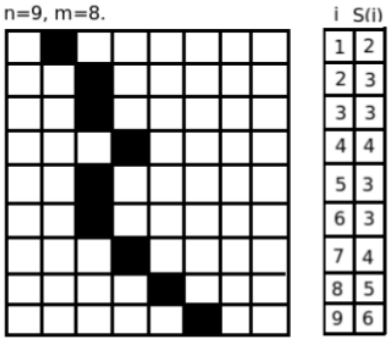

Figure 3: *Seam finding example.*

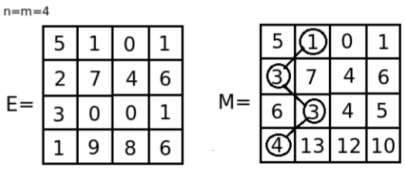

Figure 4: *Example of finding seams using dynamic programming.*

At the end of this process, the minimum value of the last row in M will indicate the end of the minimal connected vertical seam. Therefore, in the second step we backtrack from this minimum entry on M to find the path of the optimal seam. The definition of M for horizontal seams is similar:

$$M(i, j) = e(i, j) + min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)) \quad (4)$$

We repeat the above process for each segment, and then remove the newly found seams for each of them, hence resizing the picture.

3.4 Algorithm Outline

We build upon the original method [**?**]:

1. Calculate energy map $O(m * n)$
2. Find seam of lowest energy $O(m * n)$
3. Remove seam / Add seam $O(m * n)$
4. Repeat 1-3 until image is resized total run time $O(s * m * n)$

By using the same algorithm on different segments of the image:

- Divide the original image into equally sized images and run the original seam carving algorithm.
- From each segmented image remove the number of seams in ratio to the original image. For example, if we divide the image into four horizontal segments, and we must remove 40 horizontal seams, we will remove 40/4 or 10 seams from each of our segments.
- This can be done horizontally, vertically, or both, creating a grid like segmentation.

3.5 Complexity

Running the Seam Carving algorithm on each segment separately, we achieved an improvement in runtime. First, let us remember that finding s seams takes $O(s * m * n)$ in the original method, where m, n define the image's size and s is the number of seams.

With our segmented approach running on a single thread, the runtime time is equal to the total number of segments multiplied by the running time for each segment. So by segmenting the image we can concurrently reduce the running time of the algorithm by a factor of $s_m s_n$, where s_m and s_n are the number of segments in a column and the number of segments in a row, respectively. The total number of segments we divide the image into equals to $s_m * s_n$, and total number of seams in each segment is $\frac{s}{s_m}$, where s is the absolute difference between the images old width and new width, or equivalently the total number of seams that we add or remove. Total running time for each segment is $\frac{smn}{(s_m * s_n)^2}$.

Total running time of each segment serially equals to $s_m s_n \frac{O(smn)}{(s_m s_n)^2}$. For example, when we segment the image into 4 parts, we get $s_m, s_n = 2$, and $\frac{s}{4}$. So in this case: $\frac{O(s*m*n)}{16}$, and the the total running time for each segment is $4 * \frac{O(smn)}{16} = \frac{O(smn)}{4}$, which runs serially. But there is potential for a significant speedup if we run the algorithm concurrently – up to $\frac{O(s*m*n)}{16}$.

3.6 Implementation

We initially implemented the original Seam Carving algorithm in Java, but encountered some difficulties when trying to expend the code to support image segmentation. Therefore we rewrote the original and segmentation expansion of in python.

Access to the code is available through our paper website website

4 Experimental Results

We took a number of images, some of which were used previous papers, varying from feature rich images to images that show great re-

sults with the original seam carving method. All experiments were run on a standard home laptop, equivalent to MacBook Air.

Figure 5: *Left – Input image. Right – output of segmented seam carving. Beach Image*

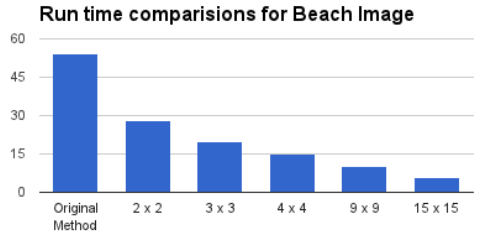

Figure 6: *Run-times*

As we see in Figure **??**, segmenting images into equally sized segments results in a much faster running algorithm, even if the algorithm runs serially. We also observed that by increasing number of grid segments we speed up the algorithm, and decrease artifacts within an image, as seen in Figure **??**.

Finally, we notice that as we segment to smaller and smaller parts, black artifacts appear in the images edges. We hypothesize that this is a byproduct of our implementation, but couldn't confirm it.

5 Limitations

The proposed method is dependent on where the image is segmented. So if the segmentation cut through feature rich parts of the image, some details might be lost and artifacts might be created.

For example, we used 3 by 3 segmentation to produce Figure **??**. The artifacts are the results of the segmentation lines going through the coast line.

Figure **??** illustrates the segmentation we used. We shall discuss how to pick better segmentation lines in the next section.

6 Conclusions and Future Work

In this paper we proposed a content-aware im- age resizing algorithm which builds upon the previous work by [**?**]. Our segmentation approach shows that there is much to be done in improving the speed runtime and performance of the original method. We didn't have enough time to investigate other methods to segment a picture, like image saliency as shown in [**?**] and [**?**]. Here are some points that we think are worth investigating in future work:

- Allowing segmentation into non-rectangular regions could lead to better results, especially if the segmentation fits the features and image energy map, so the seams that we create will only carve through these regions, thereby reducing artifacts in the output resized image.
- Flexible user input for segmentation is also one possible improvement for segmented Seam Carving – the user will be able to arrange the segments, and also mark areas of importance in the image so that the seam will skip them, thus avoiding artifacts.
- Parallelizing our python implementation for greater speed increase.
- As wee in Figure **??**, some artifacts are created between the different segments. Applying a small smoothing operator to areas in between segments to reduce artifacts could lead to better results.
- Using forward instead of backward energy, so that instead of removing seams with lowest energy, we remove seam which leaves the image with highest energy.
- Applying the saliency filter before seam carving to emphasize areas to minimize distortion.

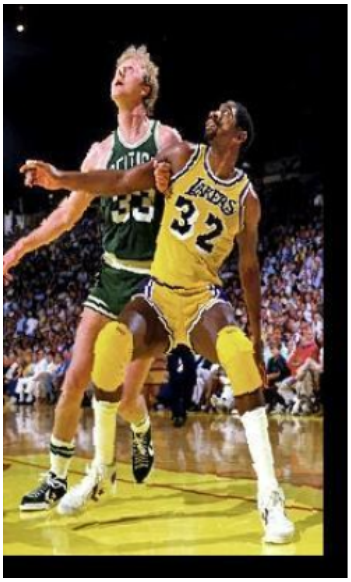

Figure 7: *Segmented Seam Carving(30x30)*

Figure 8: *Left – input image. Right – output of segmented seam carving, artifacts created are circled.*

Finally, there is always room for testing on more images, and trying different energy filters, since the results tend to be subjective by nature. There are further ways to utilize image segmentation to improve image resizing in large and specifically the Seam Carving method, and conducting more experiments and getting input from

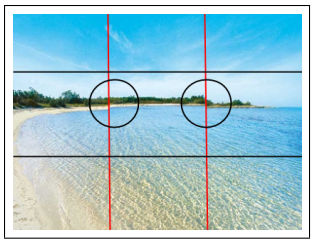

Figure 9: *Left – input image. Right – output of segmented seam carving, artifacts created are circled.*

users is essential in refining this technique.

References

LIU, H., JIANG, S., HUANG, Q., XU, C., AND GAO, W. 2007. Region-based visual attention analysis with its application in image browsing on small displays. In ACM Intl. Conf. on Multimedia: pp. 305–308.

HASAN, M. A. AND KIM, C. 2009. An automatic image browsing technique for small display users Intl. Conf.on Advanced Communication Technology: pp. 2044–2049

SETLUR, V., TAKAGI, S., RASKAR, R., GLEICHER, M., AND GOOCH, B. 2005. Automatic image retargeting ACM Intl. Conf. on Mobile and Ubiquitous Multimedia: pp. 59–68

AVIDAN, S. AND SHAMIR, A. 2007. Seam carving for content-aware image resizing Proc. of SIGGRAPH 26(3), 10

LIU, L., CHEN, R., WOLF, L., AND COHEN-OR, D. 2010. Optimizing photo composition *Computer Graphics Forum (Proc. of Eurographics) 29(2)*, pp. 469–478.

M. RUBINSTEIN, A. SHAMIR, AND S. AVIDAN. 2009. Multi-operator media retargeting *ACM Transaction on Graphics (Proc. of Eurographics) 28(3)*, pp. 23:1-23:11.

Y. PRITCH, E. KAV-VENAKI AND S. PELEG. 2009. Shift Map Image Editing *Proc. 12th IEEE International Conf. Computer Vision*, pp. 151–158.

J. STULTZ, A. EDELMAN 2008. Seam Carving: Parallelizing a novel new image resizing algorithm 6.338/18.337 Final Project Report `http://beowulf.lcs.mit.edu/18.337-2008/projects/reports/stultz-6338.pdf`

R. ACHANTA AND S. SUSSTRUNK. 2009. Saliency Detection for Content-aware Image Resizing *IEEE International Conference on Image Processing*

L. WOLF, M. GUTTMANN, AND COHEN-OR. D. 2007. Non-homogeneous content-driven video-retargeting *IEEE International Conference on Computer Vision (ICCV)*, pp 1–6