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 improve the algorithm, and also make it work faster. We approached this problem using the segmentation approach. Experimental results demonstrate in some cases our method resizes the image faster and with creating the same or less amount of artifacts as the original 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. As an 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. The use of thumbnails that faithfully represent the image content is important in image browsing applications. In addition, a variety of displays can be 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.

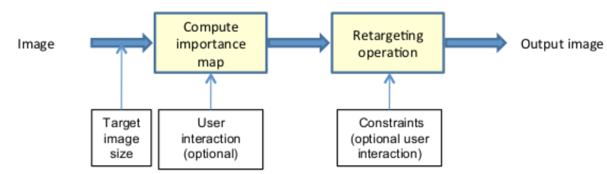

Figure 1: *The image retargeting steps.*

There are numerous ways to define importance in media. [Liu et al. 2010] 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 those techniques is image retargeting techniques Seam Carving, where the general idea is to decrease the image width (or height) one pixel at a time, by removing a seam of minimal importance. Hence Seam Carving can be viewed as a generalized cropping 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.

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 [Avidan et al. 2007] is a popular, recently developed approach for content-aware image resizing. The general idea is to decrease the image width (or height) one pixel at a time, by removing 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. The image changes only at the seam region, while the other areas remain intact. [Avidan et al. 2007] 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. The technique can also be used for enlarging the image, by finding seams to be removed and duplicating them.

The best outcome is when there are enough low-importance seams to be removed, since it creates distortions and artifacts when seams cut through important areas. Moreover, since the energy function reflects feature strength only along to the axes of image coordinate, it cannot protect some prominent shape boundaries of arbitrary orientations. Also, seam carving does not work well when the input image is feature-rich. To illustrate the limitations of the existing seam carving framework [Avidan et al. 2007], we prepared an image which is obviously feature rich. Based on the method of [Avidan et al. 2007], let us attempt to resize this image both in the horizontal and vertical direction:

Figure 2: *this is the caption*

Here, we use $I(x)$ to denote the input image, where $x = (x, y)$. As you could see, there are artifacts from result from pixels removed from both players hands and legs.

In [Avidan et al. 2007], 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 [Liu et al. 2007] and [Hasan et al. 2009] 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. [Setlur et al. 2005] 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, [Avidan et al. 2007] ,authors of the original Seam Carving algorithm, suggested that users could scribble on salient areas.

3 Technical details

3.1 Overview

Our method builds upon the basis of the original seam carving algorithm, where we the laplacian to find the energy levels of each pixel in the input image. After finding the energy levels, 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 the 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.

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:

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

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:

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

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:

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

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

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):

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:

$$M(i,j) = e(i,j) + min(M(i-1,j-1), M(i-1,j), M(i-1,j+1)) \tag{4}$$

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 [Avidan et al. 2007]:

1. Calculate energy map $O(mn)$
2. Find seam of lowest energy $O(mn)$
3. Remove seam / Add seam $O(mn)$
4. Repeat 1-3 until image is resized

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

- Divide the original image into equally sized images and run the original seam carving algorithm.
- From each segmented image remove half the number of seams from the original image. This can be done horizontally, vertically, or both, creating a grid like segmentation.
- 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 $o(smn)$ 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 $ss_m s_n$, where s_m and s_n are the number of segments in a column and the number of segments in a row, respectively. So the total number of segments equals to $s_m s_n$, and total number of seams in each segment is $\frac{s}{s_m s_n}$, 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 results with the original seam carving method. All experiments were run on a standard home laptop, equivalent to MacBook Air.

Figure 3: *Left – Input image. Right – output of segmented seam carving. Beach Image*

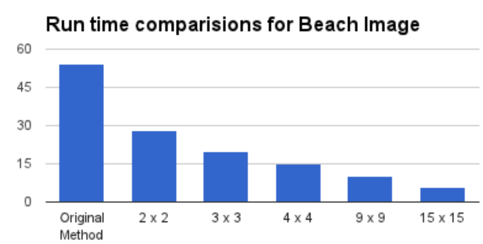

Figure 4: *Run-times*

As we see in Figure 4, 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 5.

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 could yet confirm it fully.

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, in the figure below:

We used 3 by 3 segmentation to produce Figure 6. The artifacts are the results of the segmentation lines going through the coast line.

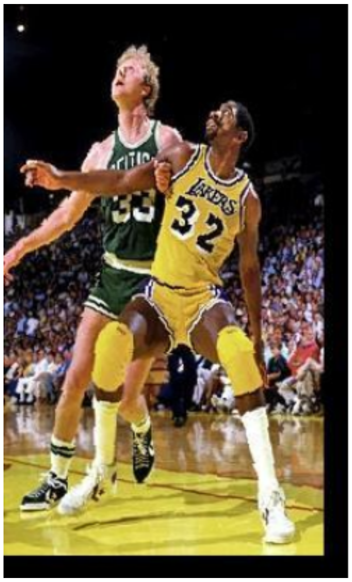

Figure 5: *Segmented Seam Carving(30x30)*

Figure 6: *Left – input image. Right – output of segmented seam carving, artifacts created are circled.*

Figure 6 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 [Avidan et al. 2007]. 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

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Add ti bib:

M. Rubinstein, A. Shamir, and S. Avidan, "Multi-operator media retargeting," ACM Transaction on Graphics, vol. 28, no. 3, pp. 23:1-23:11, 2009. J. Stultz, Seam Carving, Parallelizing a novel new image resizing algorithm R. Achanta and S. Susstrunk, Saliency Detection for Content-aware Image Resizing, IEEE International Conference on Image Processing, 2009.

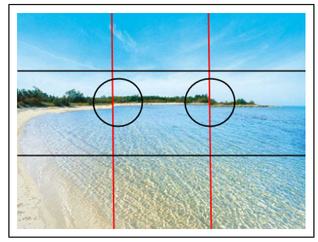

Figure 7: *Left – input image. Right – output of segmented seam carving, artifacts created are circled.*

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: 305–308.

HASAN, M. A. AND KIM, C. 2009. An automatic image browsing technique for small display users Intl. Conf.on Advanced Communication Technology: 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: 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)*, 469–478.

PARK, S. W., LINSEN, L., KREYLOS, O., OWENS, J. D., AND HAMANN, B. 2006. Discrete sibson interpolation. *IEEE Transactions on Visualization and Computer Graphics 12*, 2 (Mar./ Apr.), 243–253.

PELLACINI, F., VIDIMČE, K., LEFOHN, A., MOHR, A., LEONE, M., AND WARREN, J. 2005. Lpics: a hybrid hardware-accelerated relighting engine for computer cinematography. *ACM Transactions on Graphics 24*, 3 (Aug.), 464–470.