ISSN XXXXX XXXX © 2017 IJESC

Research Article Volume 7 Issue No.8

# Image Resizing using Seam Carving

Nagarathna .H .S[1], Sudha .B .S[2], Ramesh Kumar .K .R[3]
Assistant Professor[1], Associate Professor[2], Research Scholar[3]
Department of Electronics and Communication Engineering[1, 2], Department of Mechanical Engineering[3]
Dr Ambedkar Institute of Technology, Bengaluru Karnataka, India[1, 2]
NITK, Surthkal, Mangalore Karntaka, India[3]

**Abstract:**
Effective resizing of images should not only use geometric constraints, but consider the image content as well. We present a simple image operator called seam carving that supports content-aware image resizing for both reduction and expansion. A seam is an optimal connected path of pixels on a single image from top to bottom, or left to right, where optimality is defined by an image energy function. By repeatedly carving out or inserting seams in one direction we can change the aspect ratio of an image. By applying these operators in both directions we can retarget the image to a new size. The selection and order of seams protect the content of the image, as defined by the energy function. Seam carving can also be used for image content enhancement and object removal. By storing the order of seams in an image we create multi-size images that are able to continuously change in real time to fit a given size.

**Keywords:** Cost Function, Seam Carving, Seam Deletion, Seam Insertion, Sobel Edge Detection.

## I. INTRODUCTION

A technique called seam carving is introduced for content-aware image resizing. Seam carving is a method to change the size of an image by moderate amounts by recursively reducing or increasing the horizontal/vertical dimension one row/ column at a time. This process is performed by respectively deleting or inserting a set of pixels which is referred to as a seam. Each seam is associated with its own cost which can reflect scene content. The seam with larger costs has more important content. Therefore using these seams one can achieve the desired aspect ratio that is by deleting or inserting seams with the least cost in a recursive manner. The recursive procedure together with the cost function tries to minimize the amount of content degradation caused by the insertion or deletion of seams. In a rectangular image which is W pixels wide and H pixels tall, a vertical seam is a set of pixels which:

- Extend from the top to the bottom of the image.
- Have different vertical coordinates.
- Are path-connected.

Thus a vertical seam is a vertically oriented curve which partitions the image into left and right parts. Deleting the pixels in a vertical seam reduces the width of an image by exactly one column. A vertical seam is shown in Fig. 1(a) in red. A horizontal seam is shown in Fig.1 (a) in yellow. Deleting the pixels in a horizontal seam reduces the height of an image by exactly one row. The key idea of seam deletion for content-aware image downsizing is to associate content-aware costs with seams and then recursively remove seams one at a time, recalculating the costs if needed, until the image is reduced to the desired size. Typical content-aware costs used in [10] are based on weighted intensity gradients. The total number of vertical and horizontal seams grows exponentially with the height and width of the images respectively. Thus, an exhaustive search for a minimum-cost seam for even moderate image sizes is computationally prohibitive. To overcome this difficulty [10] uses cost functions which are additive over the pixels in a seam, e.g., the sum of the absolute values of the local intensity gradients over all pixels in a seam. An additive cost function together with the structure of a seam makes it possible to reformulate the search for a minimum-cost seam in terms of a dynamic programming algorithm which is guaranteed to find a minimum-cost seam and has a computational complexity which is linear in the number of pixels in the image.

(A) (B) (C) (D)

**Figure.1. Illustration of seam carving: (a) original image with horizontal and vertical seams superposed; (b) cropped image; (c) uniformly scaled image; (d) seam-carved image. [11]**

Figure 1 illustrates unique properties of image down-sizing by means of seam carving, cropping and uniform scaling. While cropping removes pertinent parts of the original image, uniform scaling distorts object appearance. On the other hand, seam carving largely preserves the relevant content while avoiding excessive geometric distortions. Standard image scaling is not sufficient since it is oblivious to the image content and typically can be applied only uniformly. Cropping is limited since it can only remove pixels from the image periphery. More effective resizing can only be achieved by considering the image content and not only geometric constraints. The project proposes a simple image operator, that is seam-carving, that can change the size of an image by gracefully carving-out or inserting pixels in different parts of the image. Seam carving uses an energy function defining the importance of pixels. A seam is a connected path of low energy pixels crossing the image from top to bottom, or from left to right. By successively removing or inserting seams we can reduce, as well as enlarge, the size of an image in both directions. For image reduction, seam selection ensures that while preserving the image structure, we remove more of the low energy pixels and fewer of the high energy ones.

## II. ALGORITHM TO IMPLEMENT SEAM CARVING FOR IMAGE RESIZING

### A. Algorithm

**Step1:** Read the input image.
**Step2:** Converts an RGB color map input image to an HSV color map.
**Step 3:** Involves calculating the gradient image for the original image.
**Step4:** The next step is to calculate the energy map of input image.
**Step 5:** Finds the seam map from which the optimal seam can be calculated.
**Step 6:** Returns a column vector of coordinates for the pixels to be removed.
**Step7:** The seam vector calculated from step 6 is superimposed on the input RGB image.
**Step8:** For the given input image, apply the seam vector deletion to obtain the desired size.

### B. Sobel Edge Detection

In this part, the Sobel edge detection algorithm is described to find the gradient of an image from a given RGB or gray scale image. In general, an algorithm of edge detection finds the sharp intensity variation of an image and in this way it obtains the edges of the objects contained on the image. There are various methods to detect the edges which use discrete gradients, Laplacians, etc. The most common methods used in the detection of edges are Roberts, Sobel, Prewitt, Laplacian, Canny, etc. Their operators are masks of 3x3 windows (2x2 windows in the Roberts algorithm) which are convolved with the incoming image to assign each pixel a value of 0 or 255. To obtain better results each method applies between two and four masks to find edges in the image. The implementation of the project uses one of the most commonly used methods for edge detection called Sobel edge detection algorithm. This algorithm uses 4 operators (also called masks or kernels) of 3x3 windows which measure the intensity variation of the image when they are convolved with it in four directions that are horizontal, vertical, right diagonal and left diagonal. In this case only vertical and horizontal gradient is sufficient therefore only two operators horizontal and vertical are used. Now we describe the Sobel Edge Detection algorithm, consider an input image of size as shown in figure 2.

| | | | | |
|---|---|---|---|---|
| A(1,1) | A(1,2) | A(1,3) | … | A(1,n) |
| A(2,1) | A(2,2) | A(2,3) | … | |
| A(3,1) | A(3,2) | A(3,3) | | |
| A(m,1) | | | … | A(m,n) |

**Figure. 2.Arrangement of Pixels for an Image.**

Since the masks are of dimension 3x3 as shown in figure 3, it is necessary to select a 3x3 window of the image to convolve it with each mask.

H=

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

**(a)**

V=

| | | |
|---|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

**(b)**

**Figure.3. A) Horizontal Mask b) Vertical Mask**

The operator uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical. If we define A as the source image, and $G_X$ and $G_Y$ are two images which at each point contain the horizontal and vertical derivative approximations, the computations are as follows:

$$G_X = A * H \quad (1)$$

$$G_Y = A * V \quad (2)$$

At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using equation (3)

$$G = \sqrt{G_x^2 + G_Y^2} \quad (3)$$

**Figure. 4.A Input Image**

The figure 4B shows the Sobel edge detected image for a given input image.

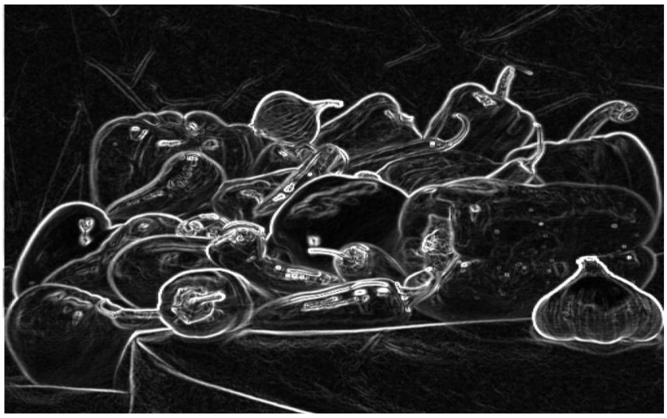

**Figure .4.B Sobel Edge Detected Image**

### C. Cost Function

Once the gradient image is calculated, the next step is to calculate the energy map image. The energy map image needs to be calculated separately for either vertical or horizontal seams and also needs to be recalculated after every seam removal. To find the cost function of an image, content based image resizing is carried out. In the content based image resizing, the energy (cost) of the content will be more than the cost of the background. Using this method, entire cost of the image can be found and the low cost seam is selected and deleted to resize the image. The energy is calculated by the following process for the vertical seam case (a horizontal energy image can be calculated using the same function) for each pixel (i,j) in the gradient image as shown in figure 5, the value at (i,j) in the energy map is the sum of the current value at (i,j) from the gradient image and the minimum of the three neighboring pixels in the previous row, i.e. *min*((i-1,j-1),(i-1,j),(i-1,j+1)), from the energy map. For i=1 (the initial row), the values in the energy map image are set to those in the gradient image, and for when the pixel (i,j) is along the edge of an image, only (i-1,j) and either (i-1,j-1) or (i-1,j+1) are used depending on if (i,j) is on the right or left edges, respectively.

| | | |
|---|---|---|
| (i-1,j-1) | (i-1,j) | (i-1,j+1) |
| (i,j-1) | (i,j) | (i,j+1) |
| (i+1,j-1) | (i+1,j) | (i+1,j+1) |

**Figure.5. Pixel indices**

If attempting to compute a vertical seam (path) of lowest energy, for each pixel in a row we compute the energy of the current pixel plus the energy of one of the three possible pixels above it. This is better described by figure 6

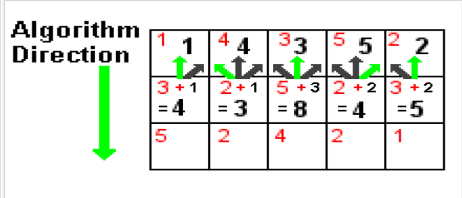


**Figure.6. Each Square Represents a pixel, with the top-left value in red representing the gradient value of that said pixel. The value in black represents the cumulative sum of energies leading up to and including that pixel.**

The first row in figure 6 has no rows above it, so the sum (black) is just the energy value of the current pixel (red).The second row, if we look at the second pixel for example, we see its energy value is 2 (red). If we look above it, it has a choice of 1, 4, or 3 (black). Since 1 is the minimum number of the three values, we ignore the other two and set the sum of the pixel to its energy value which is 2 (red) plus 1 (black). After the above operation is carried out for every pixel in the second row, we go to the third row as shown in figure 7.

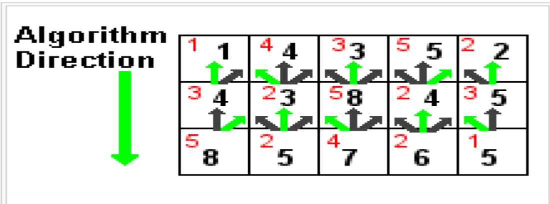


**Figure .7. Operation Is Carried Out For Every Pixel in the Third Row**

Repeat the process in row two, in row three to end up with the final cumulative sums for the seams/paths. The lowest value or values are the seams with the lowest energy, which would be in this example the seams with '5' in the last row. The vertical seam energy map for a given input image is as shown in figure 8.

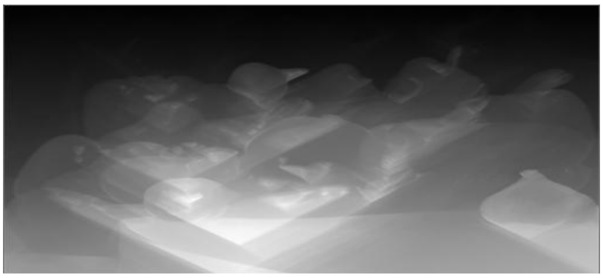

**Figure.8. The vertical seam energy Map.**

**D. FINDING LOW COST SEAM**

A seam is a monotonic and connected path of pixels going from the top of the image to the bottom, or from left to right. By removing one seam from an image, the image size is reduced by one either in the horizontal or the vertical dimension. Seam carving uses an energy function defined on the pixels and successively removes minimum energy paths from the image. Seams can be either vertical or horizontal. A vertical seam is a path of pixels connected from top to bottom in an image with one pixel in each row. A horizontal seam is similar with the exception of the connection being from left to right. The importance/energy function values a pixel by measuring its contrast with its neighbor pixels. Computing the seam consists of finding the path of minimum cost from one end of the image to another. This can be done via Dijkstra's algorithm, dynamic programming. Dijkstra's algorithm, is a graph search algorithm that solves the single-source shortest path problem for a graph with non-negative edge path costs, producing a shortest path tree. This algorithm is often used in routing and as a subroutine in other graph algorithms. The algorithm finds the path with lowest cost (i.e. the shortest path) between that vertex and every other vertex. It can also be used for finding costs of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined. Dynamic programming is a programming method that stores the results of sub-calculations in order to simplify calculating a more complex result. Dynamic programming is used in seam carving for computing seams. If attempting to compute a vertical seam (path) of lowest energy, for each pixel in a row we compute the energy of the current pixel plus the energy of one of the three possible pixels above it. Once the energy map is calculated as shown in above figure 8, the method to find the optimal seam is to first find the minimum value in the last row (which becomes the (i,j)'th pixel), saving the pixel location for use in removal, then working backwards by finding the minimum of the 3 neighboring pixels of (i,j) in the (i-1)'th row and saving that pixel to the seam path. This process is repeated until the first row is reached and results in the optimal seam, an example of which is shown in Figure 9.


**Figure.9. To trace the seam/path, work from the last row and follow the green arrow.**

The optimal or low cost vertical seam energy map for a given input image as shown in figure 10.

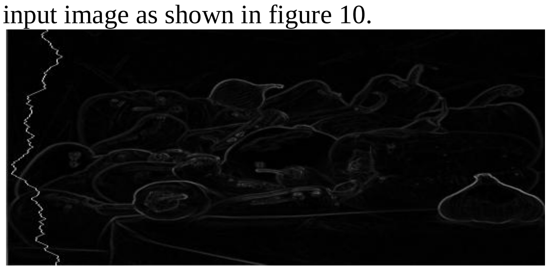

**Figure.10. Low cost vertical seam line superimposed upon the energy map image.**

**E. SEAM DELETION**

After the optimal seam is found, the path of pixels that make up the seam are removed from both the gradient image and the original RGB image and the remaining pixels are shifted right or up to form a continuous image. As shown in figure 11

**Figure .11. Low cost vertical seam line superimposed upon the input image**

The process can be repeated to remove a set of seams, horizontally or vertically and will result in an image with reduced dimensions, but with the overall scene content intact. An example of this is included as Figure 12, where the image was resized to 370x370 pixels, from 520x520 pixels and as can be seen, the resulting image will have artifacts if a large number of seams are removed.

**Figure .12.The Image Resized to 370x370 pixels.**

**F.SEAM INSERTION**

For the case of seam insertion (increasing the image size), a seam can be calculated along a given direction, and the average of the two neighboring pixels along the seam can be inserted. If the desired image size is to be increased by *N* pixels in a given direction, the computation of the first *N* seems to be removed along that direction must first be completed and then averaged pixels are inserted along each successive seam, hence the limitation on the maximum increase in image size in my implementation noted earlier in the features and functionality section. This method of calculating *N* seams is used to avoid inserting pixels along the same seam repeatedly.

## III. EXPERIMENTAL RESULTS

**Input the first frame:**

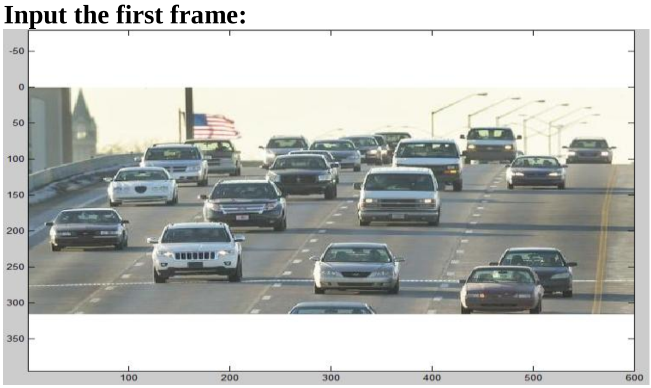


**Figure.13. Input frame.**

Involves calculating the gradient image for the original image.

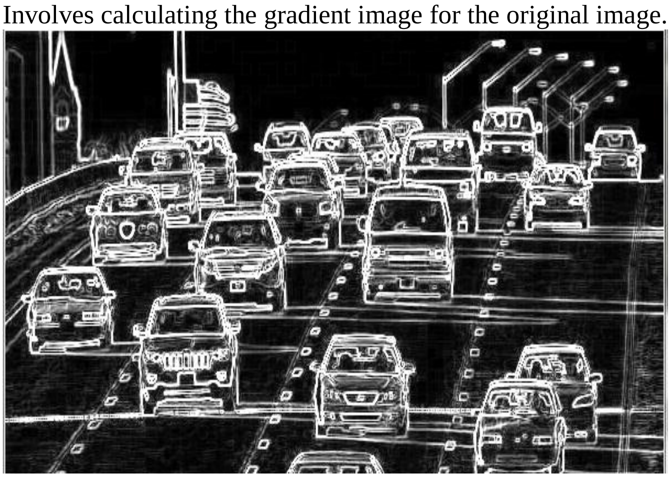

**Figure.14. Gradient calculated image for a given input image.**

The next step is to calculate the energy map image.

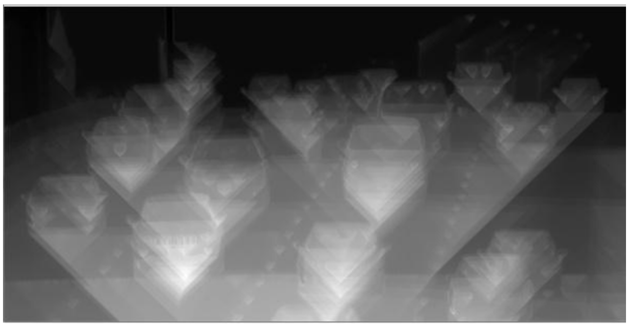

**Figure.15. Energy map image of given input image.**

After the energy of the image is calculated, generating a list of seams. Seams are ranked by energy, with low energy seams being of least importance to the content of the image. Calculating low cost seams via the dynamic programming approach. Figure 4. Shows low cost Seams shown with the energy function:

**Figure.16. Gradient image with superimposed low cost frame to be deleted.**

**Figure.17. RGB Image with superimposed low cost frame to be deleted**

Repeating the process to remove the low cost seams from the image until desired size of the image is reached. In this example, the image was resized to 500x235 pixels from 600x315 pixels. The condensed frame is as shown in figure 4.6.

**Figure.18. Low cost seam deleted condensed image.**

## IV. CONCLUSIONAND FUTURE WORK

The operator presented for content-aware resizing of images using seam carving. Seams are computed as the optimal paths on a single image and are either removed or inserted from an image. This operator can be used for a variety of image manipulations including: aspect ratio change, image retargeting, content amplification and object removal. The operator can be easily integrated with various saliency measures, as well as user input, to guide the resizing process. There are numerous possible extensions to this work. The approach can be extended to other domains, the first of which would be resizing of video. Since there are cases when scaling can achieve better results for resizing, we would like to investigate the possibility to combine the two approaches, specifically to define more robust multi-size images. The concept of seam carving for image resizing can be used to resize the video and the process is called video condensation. In video condensation, the length of video is reduced while preserving the activities that appear in video. Thus we can extend the idea of a seam to a ribbon; the approach of video condensation results by removing the ribbon with minimum cost. For example if there is a segment of N consecutive video frames where each frame is W pixels wide and H pixels tall. After condensation, the process needs to end up with a new segment of N' consecutive frames with $N' \leq N$. The condensation procedure should hopefully preserve all the important events. This can achieve by mimicking the idea of seam deletion for image down-sizing.

## IV. REFERENCES


[1].A. Rav-Acha, Y. Pritch, D. Lischinski, and S. Peleg, (Oct. 2007) “Dynamosa icing: Mosaicing of dynamic scenes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 29, no. 10, pp. 1789–1801.

[2].B. Chen and P. Sen, (2008) “Video carving,” presented at the EUROGRAPHICS.

[3].H.-W. Kang, Y. Matsuhita, X. Tang, and X.-Q. Chen, (Jun. 2006), “Space-time video montage,” in Proc. IEEE Conf. Computer Vision Pattern Recognition, pp. 1331–1338.

[4].J. Oh, Q. Wen, J. Lee, and S. Hwang,(2004) “Video abstraction,” in Video Data Mangement and Information Retrieval, S. Deb, Ed. Hershey, PA: Idea Group, Inc./IRM Press, , pp. 321–346, ch. 3.

[5].J. Konrad, (Jan. 2007), “Videopsy: Dissecting visual data in space-time,” IEEE Commun. Mag., vol. 45, no. 1, pp. 34–42.

[6].M. Irani, P. Anandan, J. Bergen, R.Kumar, and S. Hsu, (May 1996) “Efficient representations of video sequences and their applications,” Signal Process., Image Commun., vol. 8, no. 4, pp. 327–351.

[7].M. Ristivojevic´ and J. Konrad, (Feb. 2006) “Space-time image sequence analysis: Object tunnels and occlusion volumes,” IEEE Trans. Image Process., vol. 15, no. 2, pp. 364–376.

[8].N. Petrovic, N. Jojic, and T. Huang, (Aug. 2005) “Adaptive video fast forward,” Multimedia Tools Appl., vol. 26, no. 3, pp. 327–344,.

[9]. Rav-Acha, Y. Pritch, and S. Peleg, (Jun. 2006) “Making a long video short: Dynamic video synopsis,” in Proc. IEEE Conf. Computer Vision Pattern Recognition, pp. 435–441.

[10].S. Avidan and A. Shamir,(2007) “Seam carving for content-aware image resizing,” ACM Trans. Graph., vol. 26, no. 3.

[11].Y. Pritch, A. Rav-Acha, and S. Peleg,( Nov. 2008.) “Non-chronological video synopsis and indexing,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 11,

[12].Zhuang Li, Prakash Ishwar and Janusz Kornad,(2009) “Video Condensation by Ribbon Carving,” IEEE transactions on image processing, vol. 18, no.11.