Seam Carving: Content Aware Image Resizing

Viktor Gorte

July 25, 2019

1 Introduction

Changing image dimensions by cropping or by rescaling can often result in a loss of information or distorted image content. The sc package provides an implementation of the Seam Carving algorithm to solve this task. Seam Carving is the process of calculating an energy and cost value for every pixel of a given image and removing or adding a 6-connected path of pixels, vertical or horizontal, to reduce or increase the dimensions of the given image with respect to the content of the image. The algorithm consists of three major parts:

- Generate an energy map of the image
- Calculate a cost map, where each value represents the minimal cost to reach a pixel from the image top
- Either:
- Remove lowest cost seam, by:
  - Finding lowest cost value in bottom pixel row
  - Backtrack lowest cost seam
  - Remove all pixels in that seam
  - Reshape image to fit new dimensions
- Duplicate lowest cost seam, by:
  - Finding lowest cost value in bottom pixel row
  - Backtrack lowest cost seam
  - Duplicate lowest cost seam

A more indepth description of the algorithm can be found in Shai Avidan et al. (2012).

2 Process

2.1 Seam Removal

The sc package supports RGB, three channel, PNG, JPG and TIF files. In this example, an image provided by the EBImage package will be used:

```
> imagePath <- system.file("images", "sample-color.png", package="EBImage")
> img <- readImage(imagePath)
> dim(img)

[1] 768 512   3

> display(img)
```

Figure 1: Original Image: 768x512 pixels and 3 channels

First, an energy map needs to be calculated. For that, a vertical and horizontal Sobel kernal is used which provides a good estimate about contrasts and edges in the image. The cost of a pixel inside a high contrast area will be higher and therefore more likely effect the primary content of the image. The energy map for figure 1 can be seen in the following figure:

```
> energy_map <- sc_mark_leastCSeam(imagePath)$energy_map
> display(energy_map)
```

Figure 2: Energy Map

The cost map can now be generated in the following manner:

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

Where M is an Array with the minimum cost of all pixels, i is the current row, j is the current column and $e(i,j)$ is the energy of the current pixel. Essentially, the cost of any given pixel is the energy value of said pixel and the minimum value of the three pixels above it.

The resulting cost map can be used to find the lowest cost seam, the pixel path with the lowest calculated impact on the content of the image:

```
> seam_image <- sc_mark_leastCSeam(imagePath)$seam_image
> display(seam_image)
```

Figure 3: Image with lowest cost seam (red)

The marked pixel seam in figure 3 can be removed and the image can be resized to fit the reduced dimensions.

This whole process has to be conducted for every iteration of removing a seam because once a connected path of pixels is removed from the image, the energy map and the cost map are no longer accurate and have to be calculated again. The introduced process focused on vertical seam removal and works for horizontal seam removal by rotating the image by 90°.

2.2 Seam Duplication

The increase of the image dimensions works by finding the lowest cost seams in the image and duplicating them. Since the lowest cost seam has the lowest potential influence on the primary image content, duplicating it and therefore increasing the image dimensions has as little impact as possible.

Duplicating a single seam multiple times would result in stretched artifacts across the image. Therefore, the lowest k seams, where k is the number by which the width or height has to be increased, are duplicated.

```
> increased_img <- sc_increase(imagePath, 10, 10)$increased_img
> display(increased_img)
> dim(increased_img)

[1] 778 522   3
```

Figure 4: Image with increased dimensions

Figure 4 shows an image, where the dimensions were increased by 10 for both width and height.

When duplicating multiple seams, this process potentially produces artifacts because it can happen that multiple lowest cost seams share common pixels and when duplicating them, those sections of pixels are duplicated multiple times, resulting in stretches in the image. This is more likely to occure when the image dimensions are increased by a significant amount.

3 Usage

The previous section showed figures that represented every step of the Seam Carving process, which were generated with the sc package. This section will explain the functions that the package provides.

3.1 sc_reduce

This function reduces the dimensions of an RGB image. The function expects the following arguments:

- Path to the image
- Number of pixels by which the width has to be reduced
- Number of pixels by which the height has to be reduced

It is also possible to reduce an image only in width or height by providing only one of those two arguments.

The function returns an object of class $SC - Image$. This class object contains the reduced image which is of class Image (EBImage).

```
> result <- sc_reduce(imagePath, ncols = 8, nrows = 12)
> attributes(result)

$names
[1] "img_reduced"

$class
[1] "SC-Image"

> dim(result$img_reduced)

[1] 760 500   3

> display(result$img_reduced)
```

Figure 5: Image with reduced dimensions: From 768x512 to 760x500

```
> result <- sc_reduce(imagePath, ncols = 18)$img_reduced
> dim(result)

[1] 750 512   3

> display(result)
```

Figure 6: Image with reduced width only: From 768x512 to 750x512

3.2 sc_increase

This function increases the dimensions of an RGB image by duplicating lowest cost pixel seams. The following arguments are expected:

- Path to the image
- Number of pixels by which the width has to be increased
- Number of pixels by which the height has to be increased

It is also possible to reduce only the width or height of the image.

The function returns an object of class $SC - Image$. This class object contains the increased image which is of class Image (EBImage).

```
> result <- sc_increase(imagePath, ncols = 10, nrows = 10)
> attributes(result)

$names
[1] "increased_img"

$class
[1] "SC-Image"

> dim(result$increased_img)

[1] 778 522   3

> display(result$increased_img)
```

Figure 7: Image with increased dimensions: From 768x512 to 778x522

3.3 sc_mark_leastCSeam

This function marks the lowest cost seam in a given image and returns an object of class $SC - SImage$ which contains the energy map used to calculate the lowest cost seam and the image with the marked pixels. The function expects one argument:

- Path to the image

```
> result <- sc_mark_leastCSeam(imagePath)
> attributes(result)

$names
[1] "energy_map" "seam_image"

$class
[1] "SC-SImage"

> display(result$seam_image)
```

Figure 8: Image with marked lowest cost seam

References

[1] Shai Avidan & Ariel Shamir: *Seam Carving for Content-Aware Image Resizing*. CMU Graphics, 2012
`http://graphics.cs.cmu.edu/courses/15-463/2012_fall/hw/proj3-seamcarving/imret.pdf`