Dept. of CST, Tsinghua University

Image Retargeting
Project Report

Name	Yuxin Wu
Student No.	2011011271
Class	CST-14
Mail	ppwwyyxxc@gmail.com

Contents

1 Introduction

This is an image retargeting program using Seam Carving[1] algorithm.

1.1 Compilation

Dependencies:

1. gcc >= 4.7
2. GNU make
3. Magick++

Compilation:

```
$ make
```

1.2 Run

The program supports following command line arguments:

```
  -i <file>   input image file
  -o <file>   output image file
  -w <num>   target width. can be the pixel number in integer, or a relative number in (0, 1]
  -h <num>   target height. same as above
  -e <file>   energy file, optional
  -m <file>   mask image, red to discard, green to keep. optional
  -c <type>   convolution type, can be one of 'prewitt', 'vsquare', 'sobel', 'laplacian'
  -p    use optimized seam carving.
  -f <file>   feature file. every line is a coordinate.
  -v    output intermidiate results to generate video demo.
```

1.3 Examples

1. Output with every intermidiate result:

```
$ ./image_resize -i ./sea.png -o result.png -w 0.8 -v
$ feh path*.png
```

The original image and the result is as followed:

2. Use mask:

```
$ ./image_resize -i mike.jpg -o result.png -w 0.8 -m mike-m.png
```

2 Algorithms

2.1 Seam Carving

This section will briefly introduce the main processes of Seam Carving algorithm as well as the optimizaitons I made. For the details of this algorithm, please refer to the original paper[1].

1. **Energy Map**

   An energy map of the original image must be first calculated to measue the "importance" of every pixel. In our implementation, four types of convolution kernel are supported to calculate energy map:

   (a) Prewitt:

   $$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

   (b) vsquare

   $$(\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix})^2$$

   (c) sobel(default)

   $$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

   (d) laplacian

   $$\begin{bmatrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{bmatrix}$$

2. Mask The program allows users to use an image mask to declare the region they prefer to keep or discard. Users can modify the output energy grey image and adding red or green regions to it. Red region will have lower energy value, and therefore more likely to be discarded. Green region will be kept.

3. Optimized Seam Carving

   [2] suggested an optimized version of seam-carving, by integrating seam-carving with scale. Their method first uses seam-carving to cut of certain number of seams, than apply a direct scale. They try all the possible number of seams to carve, and choose the best result measuring by their "Patch-Level Image Dist Function":

   $$D(P,Q) = \sqrt{\sum_{i=1}^{nm}\sum_{j=1}^{nm} g_{ij}(p_i - q_i)(p_j - q_j)}$$

   Actually, from my observation, this formula is just a simple combination of the two factor: "Coordinate Distance" and "Color distance", measured by the two terms g_{ij} and $(p_i - q_i)(p_j - q_j)$, respectively.

   There is a tradeoff between the two distance: if "Color Distance" is more important, the result tends to cut more seams and use less scaling. Since seam-carving can

keep lots of patches in the image unchanged. If "Coordinate Distance" is more important, the algorithm is more likely to use scaling rather than seam-carving, since scaling keep the relative coordinate of every pixel unchanged.

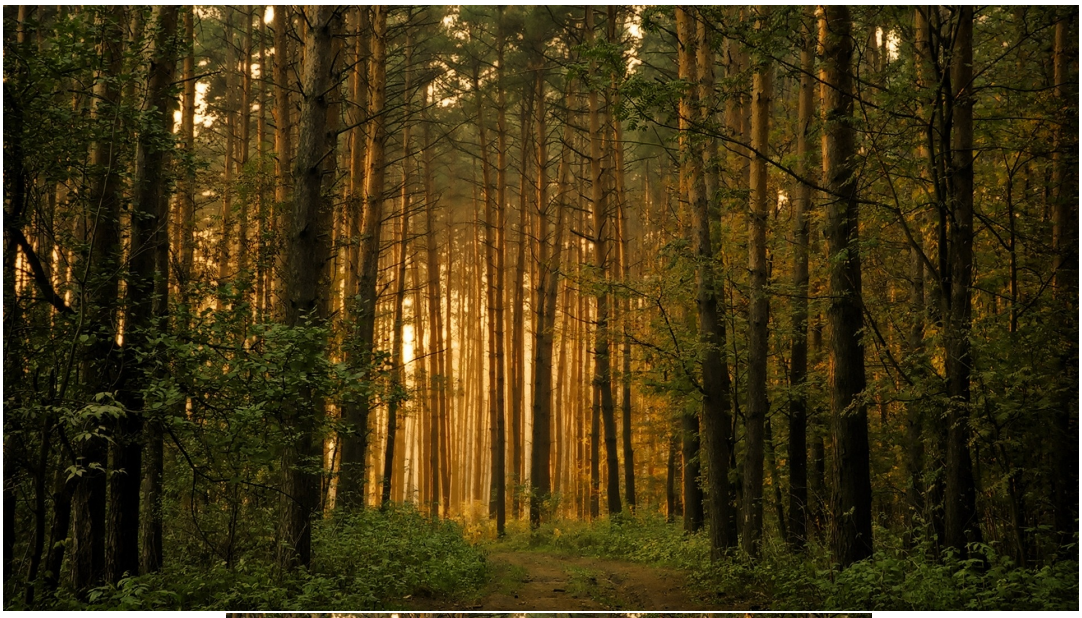
Therefore, the performace of this algorithm is strongly dependent on the choice of parameter, weighting the two types of distance mentioned above, i.e. the choice of σ in g_{ij}. From my experiements, it works well only when σ is carefully selected. But most of the time the algorithm will only naively choose to use only seam-carving or only scaling.

Also, this algorithm, although with parallel support, is still quite ineffective in computation, since the image distance takes too much time to calculate. One optimization can be made on the patch distance computation: I first assumed that all input image is needed for horizontally resize (since we can always transpose the image to get the same result). Then, when comparing a pair of patches, we can directly discard them if their vertical coordinate difference is larger than a threshold. This is reasonably because the "Coordinate Distance" mentioned above shall only account for horizontal coordinate distance.

4. Feature Map

   I tried detecting keypoints in SIFT features, and add them to the energy map, since these are points that shall be kept. Keypoints coordinate will be Gaussian-blured before added to the energy map. However, this didn't give a better result than the original seam-carving, since SIFT keypoints are similar to those points previously detected by convolution.

3 More Results



4 References

[1] Shai Avidan and Ariel Shamir. "Seam carving for content-aware image resizing". In: *ACM Transactions on graphics (TOG)*. Vol. 26. 3. ACM. 2007, p. 10.

[2] Weiming Dong et al. "Optimized image resizing using seam carving and scaling". In: *ACM Transactions on Graphics (TOG)* 28.5 (2009), p. 125.