Problem Set 1 Answer Sheet
Fangzhou Li

Short answer problems:

1. Let's say we have an image A of M*N size, and a Gaussian filter F = [1 2 1; 2 4 2; 1 2 1]. For each pixel of A, we need to calculate 9 times of operation (additions and multiplications). That is, the complexity is roughly O(9MN). Now, we find that F = [1 2 1]' * [1 2 1]. Due to the associativity, the result of F convolutes A is equal to [1 2 1]' convolutes [1 2 1] convolutes A. Now, our total complexity is reduced to O(6MN).

2. [1 1 1 1 1 0 1 1]

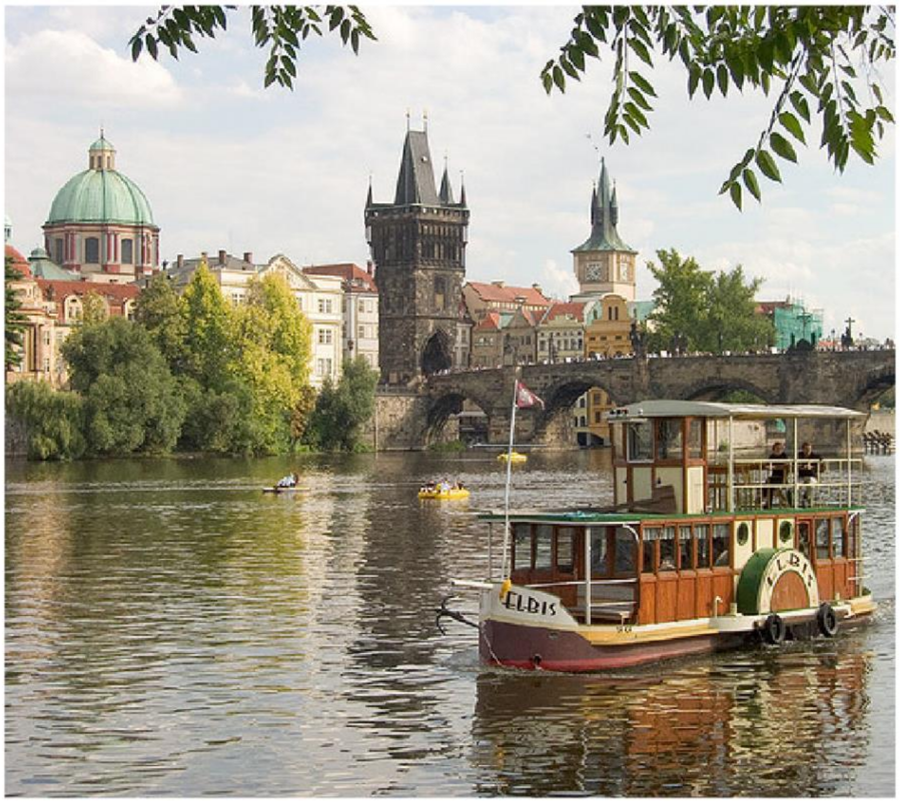
3. Two important factors of Gaussian distributions are mean value and standard deviation. For Gaussian noise, one possible flaw is not setting its mean value to 0. Another possible flaw is, if one has already added a Gaussian noise with STD σ_1 and wants to a new Gaussian noise so that the image will have the noise with σ, it is untrue to add to set the STD σ_2 of the new Gaussian noise to $\sigma_2 = \sigma - \sigma_1$. The relation between these terms should be $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$.

4. First of all, I will make some reasonable assumptions of the problem. Since all objects are on the convey belt, we can assume that the background of the image is the same. Also, we will assume that workers are required to put the product specifically on the middle of the belt, and every product will face up with the same angle. These assumptions make each pixel of video data represent a specific point of every product. Also, we want to assume that most products are decent. Now we will train our model. Assume there are many images of good products and bad products. We will use these images to do k-means clustering with each pixel of images being a feature. Next, we will see if our clustering is valuable: If we set k = 2, see if one group is consisted by majorly good products, and another group is consisted by majorly bad products. We pick the best clustering result and treat it as our trained model. Now, if our camera detects a product, we will calculate its distance to centroids in our model. If that product belongs to the bad cluster, that product will be treated as a flaw. In real life, the number of pixels maybe numerous, and it will be difficult to put all of them in the model. We can use [1, -1] filter to do edge detection so that we can divide an image to multiple parts (pixels in the same part have close values). Then we can assign a mean value to each part based on its member pixels. In this way, the

features of an image are reduced to the number of parts, and we can follow the precious training and detecting procedures.

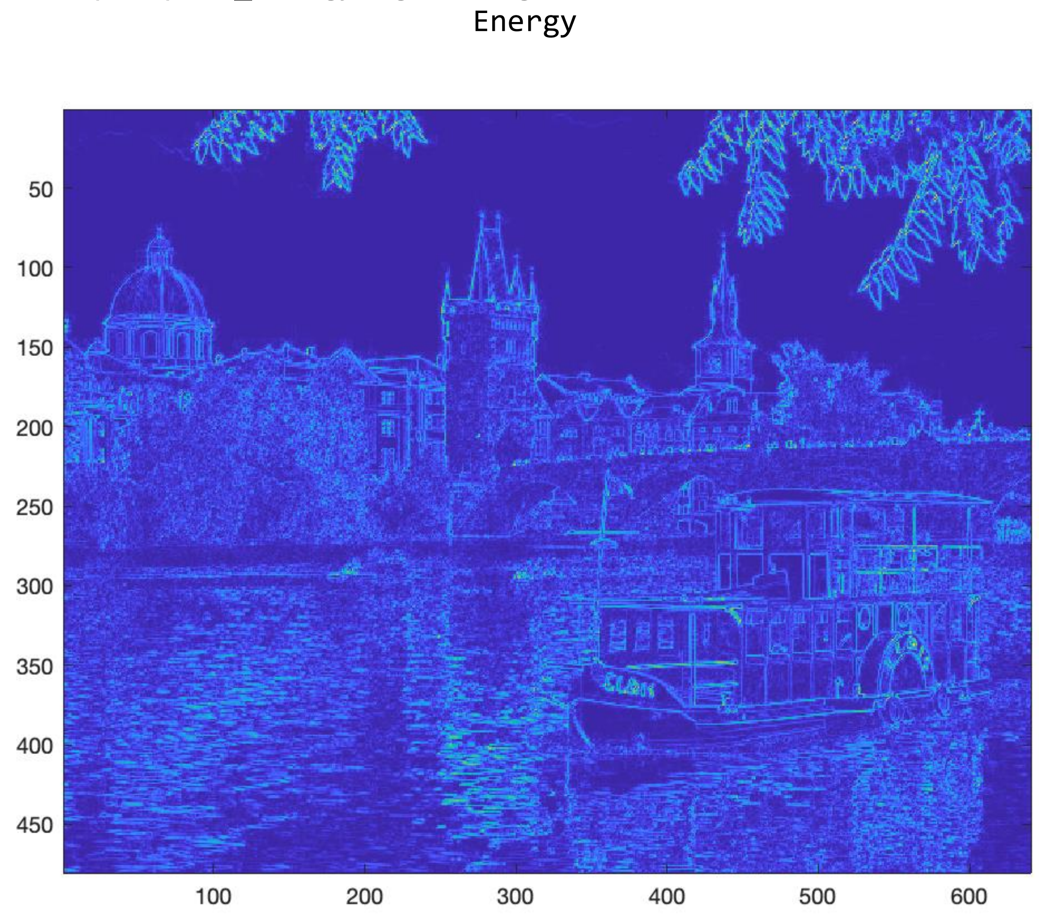
Programming problem: content-aware image resizing

1.

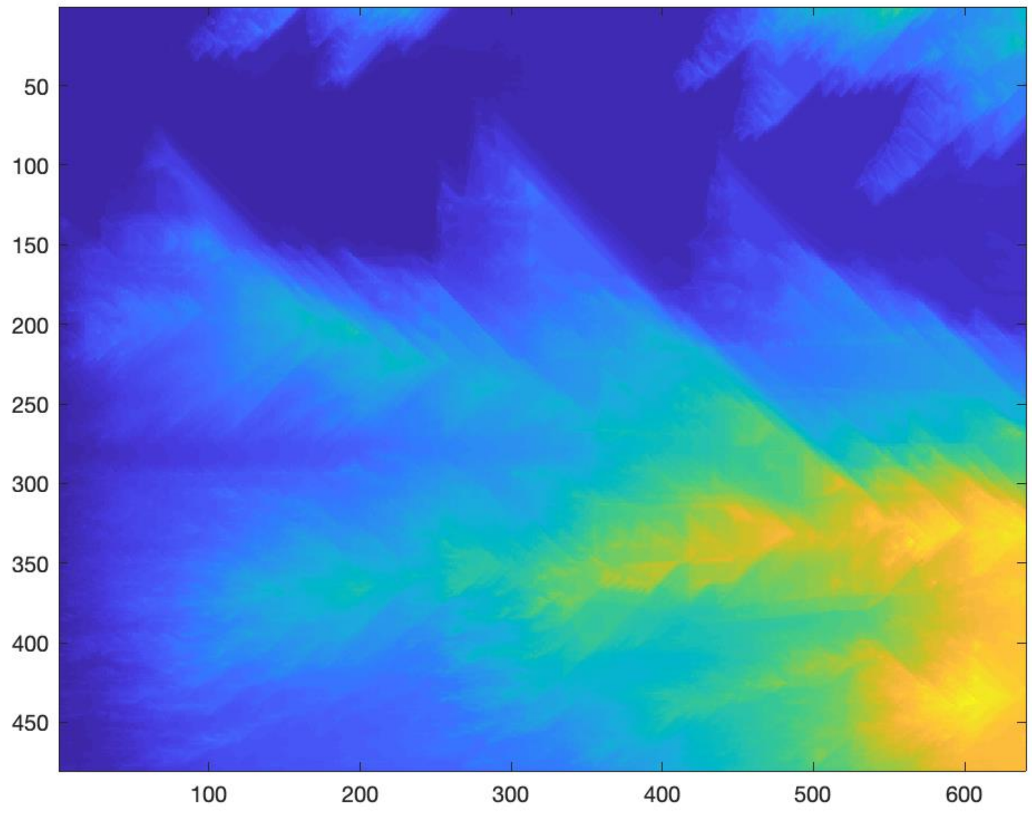


2.

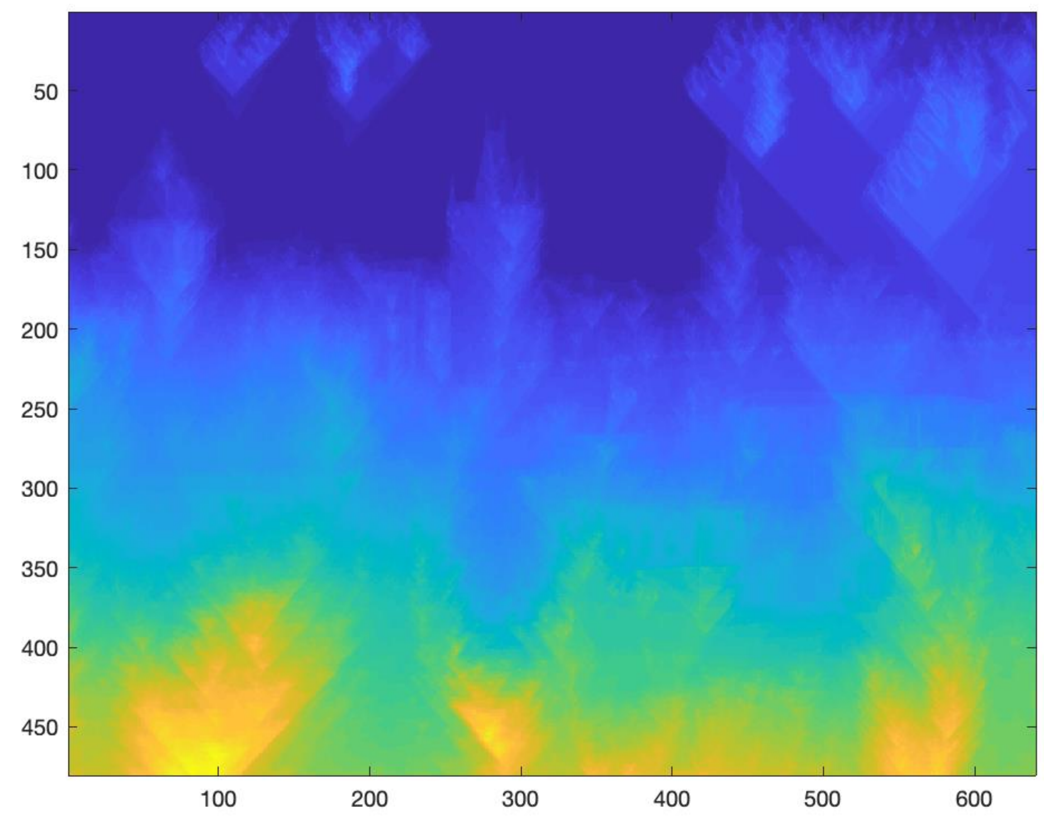
3. Run script, plot_energyImgs, to get these results.



Cumulative Horizontal



Cumulative Vertical



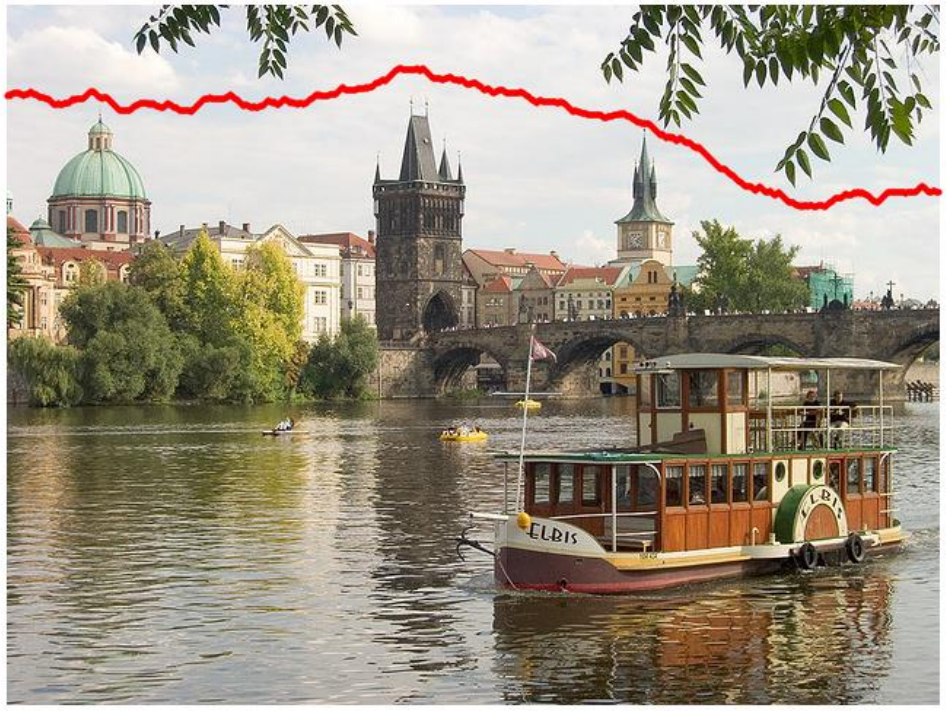
We can observe that the warmer pixels in cumulative energy maps are, the higher the energy they preserve. That is, seam should do its best to avoid passing warm places (yellow, orange, etc.). Knowing this, look at the horizontal cumulative energy map. Row 100 has a pretty continuous region of blue. This is corresponding to the sky of our original energy image. Now, we find that the vertical cumulative energy map does not have continuous seam that can perfectly avoid warm regions. This is because that when we draw a vertical seam on our energy image, it is not avoidable that the line will go across different objects, and different objects usually create edges which result in higher energy. Especially look at the water on the left bottom corner where waves create numerous edges, which is an evidence why it is yellow on the corresponding corner of vertical cumulative energy map.

4. In order to get thw following images, run

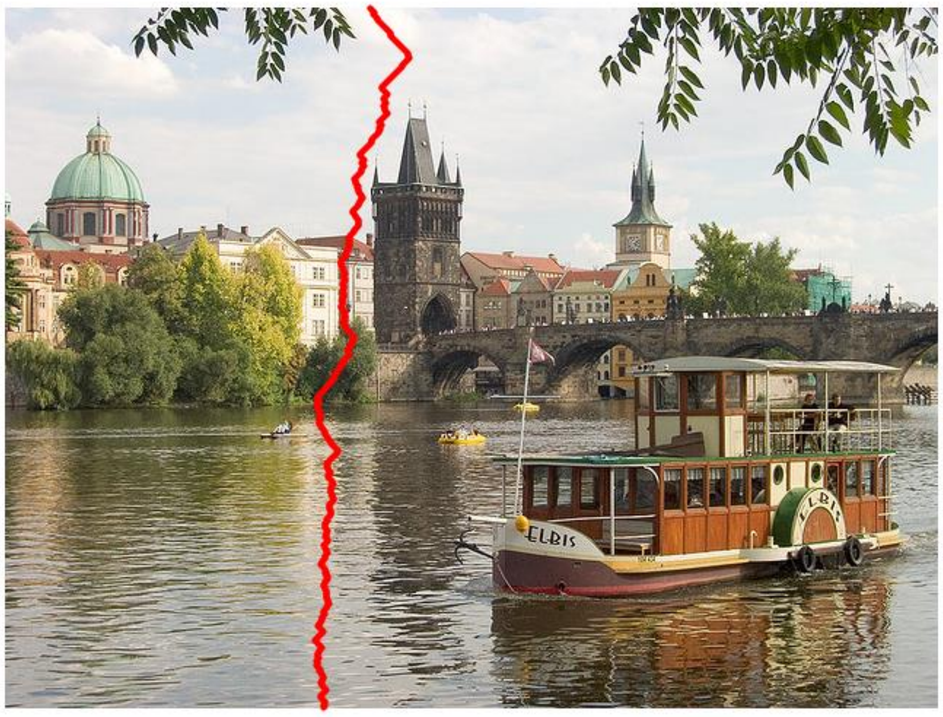
```
>> imshow('inputSeamCarvingPrague.jpg')
>> plot_first_horizontal_seam
>> plot_first_vertical_seam
```

Original

First Horizontal Seam



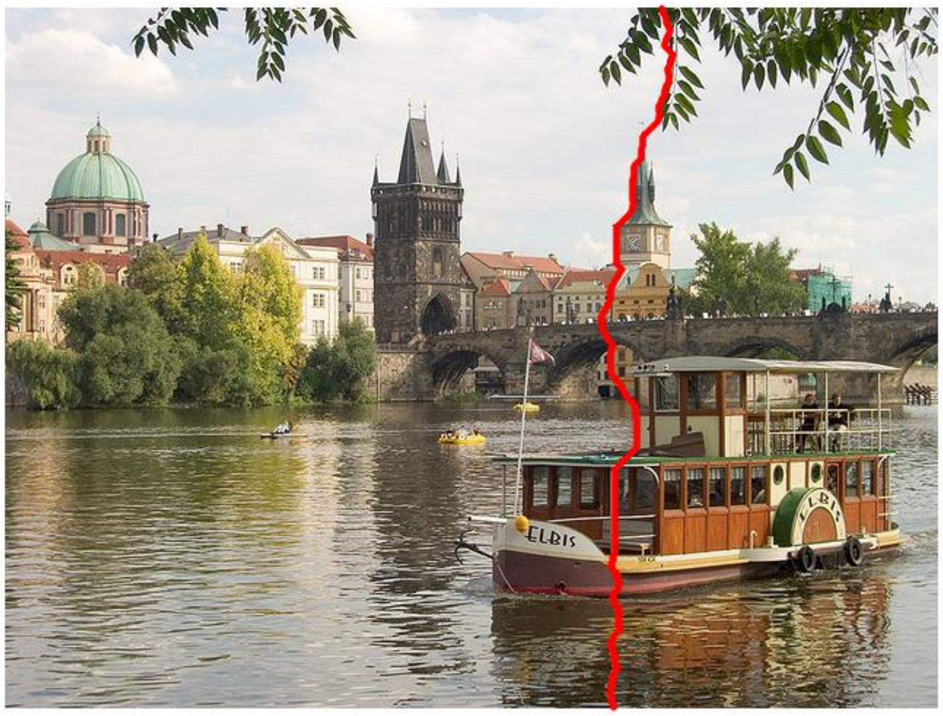
First Vertical Seam



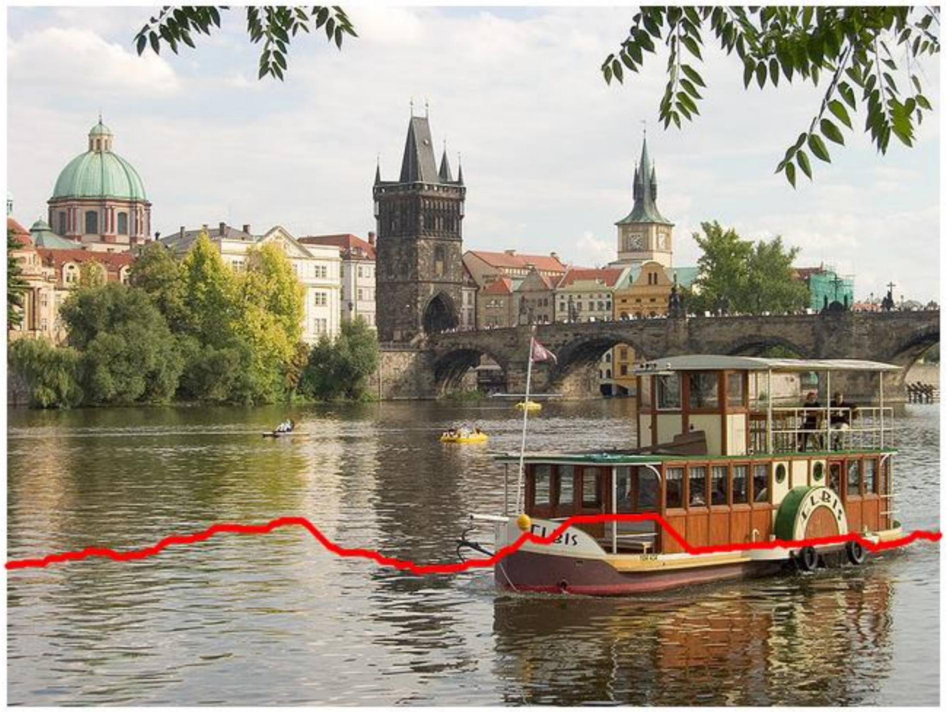
As I mentioned in the last problem, seam should prioritize its path across bluest region. It is not hard to see why the horizontal seam choose sky region in the original image since the blue region in horizontal cumulative energy map is distinguishable. Now observe the vertical cumulative energy map, we can notice that for the last row, pixel 200~300 is the bluest place comparing to other pixels. This is understandable because in the original image, although the waves of water create edges, we can see that in the particular water region along the vertical seam, the water reflects a white building, making the waves not easily detectable. Also, the path goes the particular building which is white, minimizing the variance of its color and that of cloud and sky.

5.

Revised vertical seam
(‘outputRevisedVerticalSeamPraque.png’)



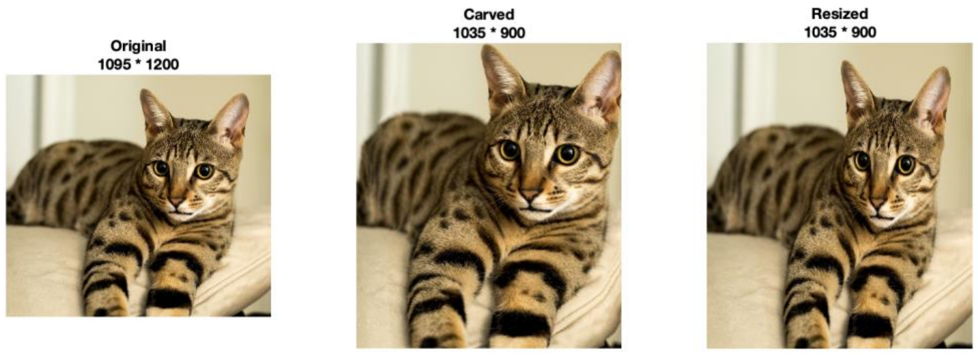
Revised horizontal seam
(‘outputRevisedHorizontalSeamPraque.png’)



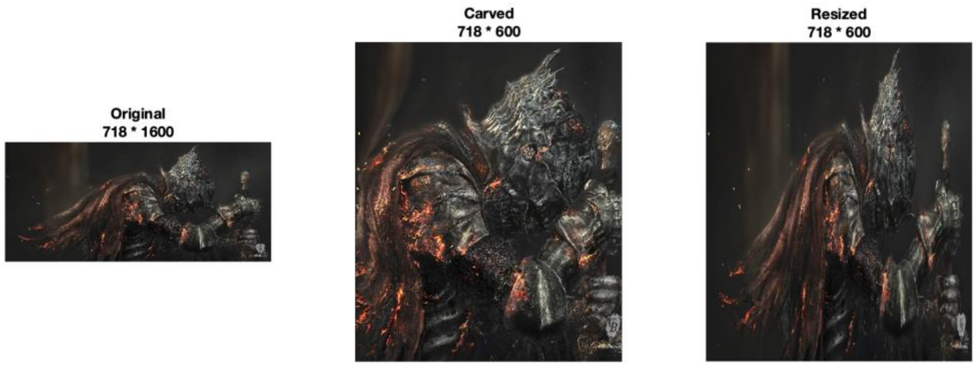
The revised energy function uses the filter, [-1 -1 -1; -1 8 -1; -1 -1 -1], to calculate the energy of the picture. Now, our energy function not only takes the next row/column pixel into calculation, but all adjacent pixels. Now seams tend to follow edges of objects. For the vertical seam, it follows the edge of tower, and for the horizontal seam, it follows the edge of the ship. The reason behind this seam behavior is because of the property of Gaussian filter. When the middle pixel of the kernel is close to the edge, the energy tends to be small because the pixels surrounding the middle pixel of the kernel contributes negative values.

6. *Great outcome*: myOutput_cat.png
   Image credit: John K Mulholland
   Usage: >> mainCat

Performed width and length reduction by 60 and 300 pixels respectively. Seam carving method succeeds to preserve the face and the major body of the cat. The typical resized image changes the ratio of cat’s body, making the picture unnatural.

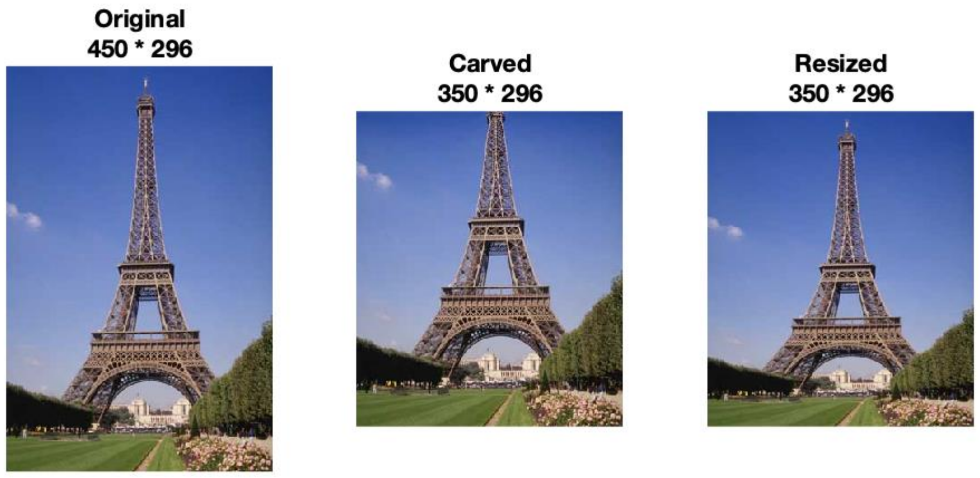


Surprising outcome: myOutput_souls.png
Image credit: From Software
Usage: >> mainSoul



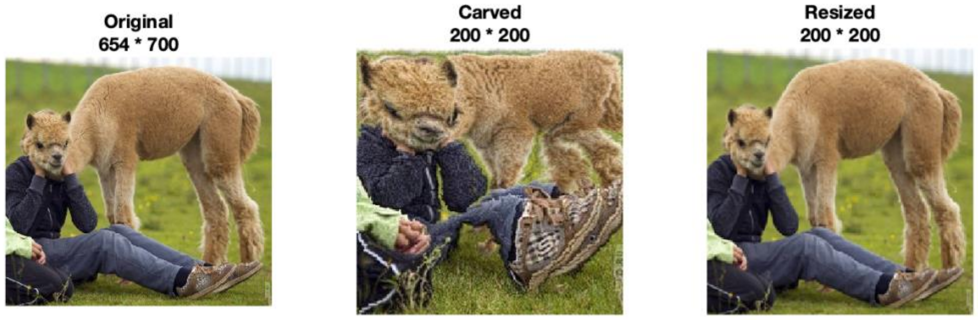
This result shows how practical the seam carving method can be in the real world. The width of the original picture is reduced by 1000 pixels, which is a very large size change. It is obvious that normal resizing will ruin the effect of the picture. However, seam carving successfully preserves the human torso in the picture and the hands holing the sword. This is unexpected by me because I was expecting this image as a ‘bad’ outcome.

Bad outcome: myOutput_tower.png
Image credit: Royalty-Free/CORBIS
Usage: >> mainTower



The height of the original picture, Eiffel Tower, is reduced by 100 pixels. This was originally what I expected to be a good outcome, but apparently the chosen seams removed the top of the tower. Since the tower is visually tall, using normal resizing method does not ruin the picture a lot.

Interestingly curious outcome: myOutput_llama.png
Image credit: funalive.com
Usage: >> mainLlama



Performed reduction, [w200, h154, w50, h50, w50, h50, w50, h50, w50, h50, w50, h50, w50, h50], where ‘w200’ denotes reducing width by 200 pixels, and ‘h154’ denotes reducing height by 154 pixels. Seam carving tries its best to preserve the important parts, but the outcome is hilariously bad. It seems that after removing the background at the top of llama’s head, it is no way to carve without removing some meaningful pixels.