I Short Answer

1.The associative property is (f * g) * h = f * (g * h). If the image f is large, convolving the image f with filter g then with filter h would be more computationally costly than computing the dot product of the two filters g * h before applying it to the image. For example, suppose we use two Gaussian filters, we can "separate" the 2-D Gaussian convolution into 2 1-D convolutions with great computational cost savings.

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

3.Additive Gaussian noise is adding a zero mean random vector to an image. It assumes that the image has a normal distribution, which means that when the Gaussian noise is added, the result is also Gaussian. If the image is not subject to the Central Limit Theorem, the result will not be Gaussian, thus it will not be an effective smoothing.

4. We assume that the object on the conveyor belt is distinguishable from its background by at least one of them: color, brightness and texture. Another important factor in accurate detection would be the uniformity of the objects. For example, we are tuning hyperparameters to detect a particular kind of edge, it wouldn't work if the coming part is drastically different from what we expect.

1. First, I smooth the image with a Gaussian filter to reduce noise. Use dilation and erosion to simplify the image.

2. Next, I will use Canny filter and adjust the low threshold and high threshold parameters to find edges with high gradient without discarding too many edges.

3. Then, I would create a binary mask of the two colors that characterize the most salient features of the object. It doesn't necessarily have to be black and white.

4. In order to find a region of interest that fully contains the object but not the conveyor belt and floor, I would have to find several vertices that enclose the

object and draw lines connecting the vertices. We exclude areas outside the region of interest by applying a mask.

The lighting and the change in the background would affect step two if the changes introduce edges with high gradients. Another flaw is that if the objects are spaced in an erratic way, our dilation, erosion and region of interest might be off.

Part II

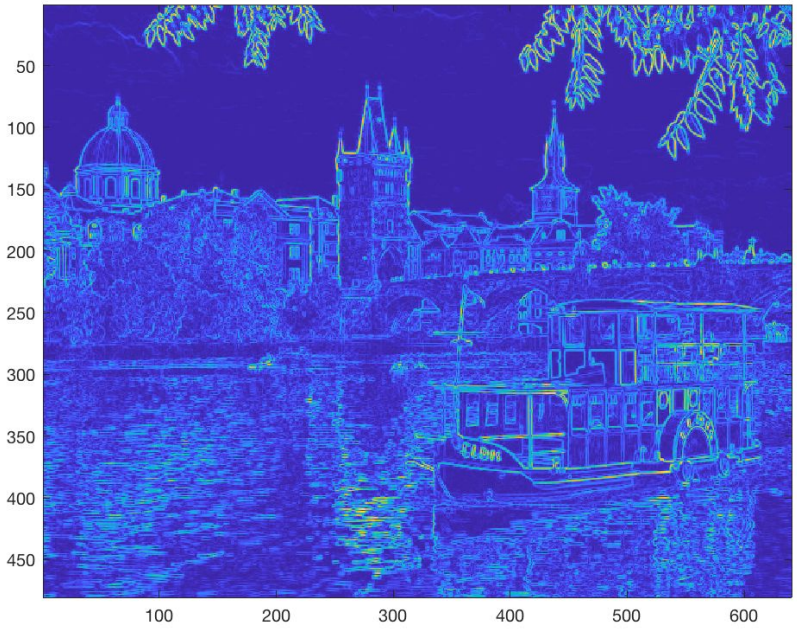
Question 1 and 2

outputReduceWidthPrague.png outputReduceHeightPrague.png

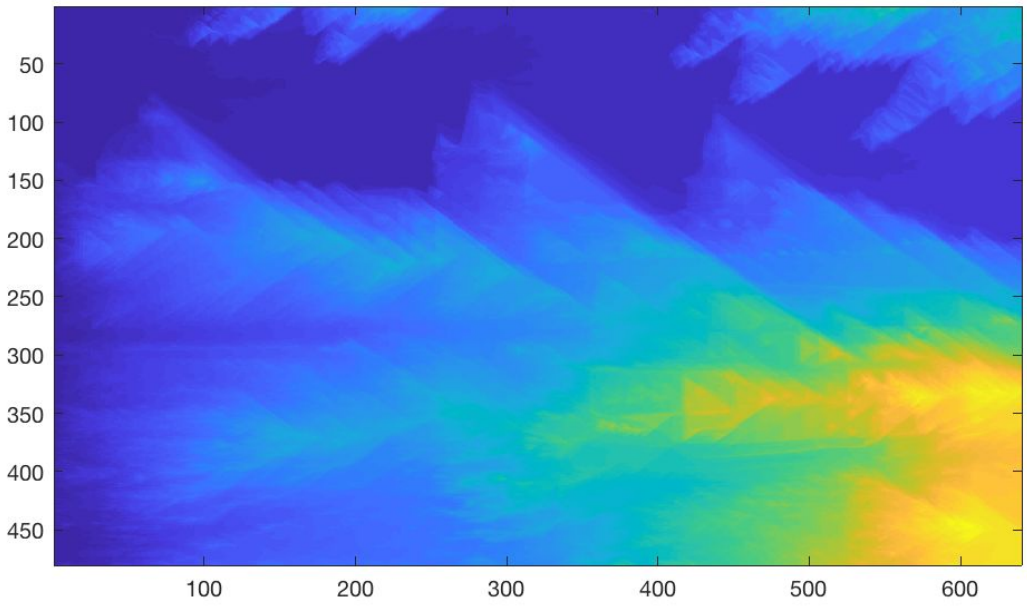
outputReduceWidthMall.png outputReduceHeightPrague.png

3.

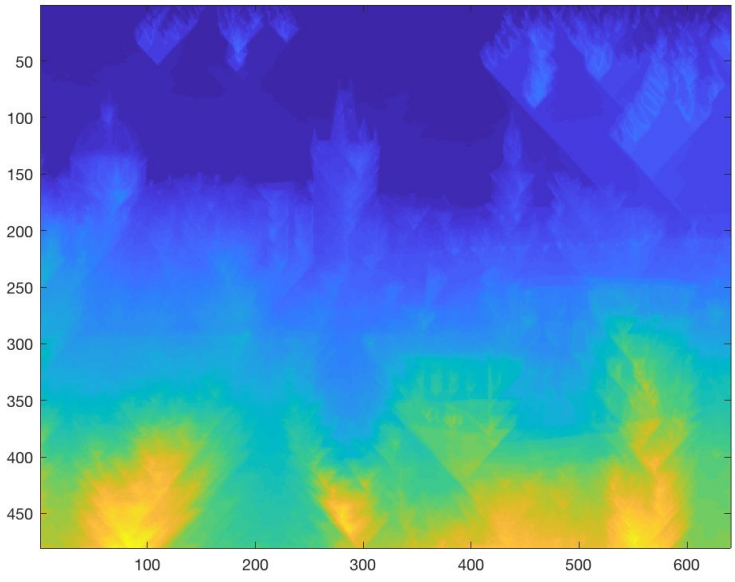
a) Energy function output



b) Horizontal cumulative minimum energy maps



c) Vertical cumulative minimum energy maps



The energy function computes the energy at each pixel by taking its gradient. Taking the first derivative of both x and y allows us to see the direction of the edge. Thus, the strong edges can be distinguished from background.

The yellow area denotes high energy cost. As you can see, it is at the bottom of the vertical energy map and on the right of the horizontal energy map. Both energy maps show high energy cost in the boat area. The reason is that the boat has the strongest edges.

A key difference between the horizontal energy map and vertical energy map is that the vertical map has high energy cost in the reflections of the water whereas the horizontal map doesn't.

4.

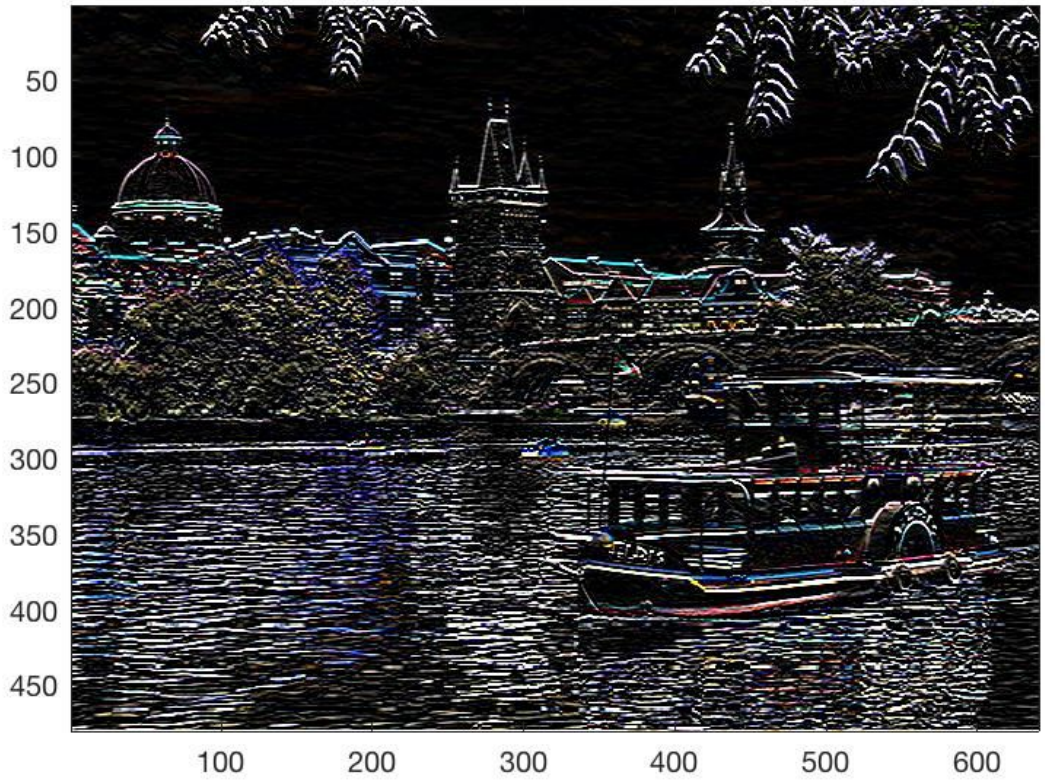
a)the first selected horizontal seam

b)the first selected vertical seam

The first horizontal seam lies entirely within the sky above all the architectures and steers away from the leaves. The brightness and color of the sky is relatively homogeneous compared to the surrounding regions, which means that they are low-energy pixels in the horizontal cumulative energy map.

The first vertical seam is the line that separates the black building and the white building. The separating line does not contain important information about either building, It also goes through the minimum amount of architectures. The the upper region of the seam is the sky and the lower region is in the lake.

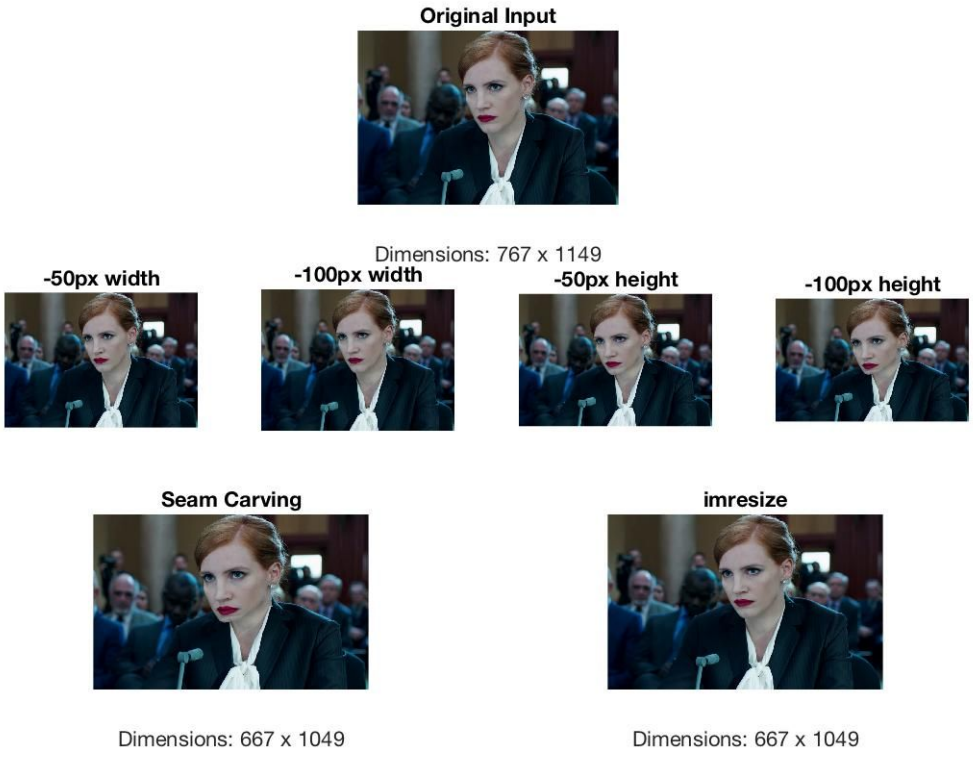
5.



I used Matlab built-in function Sobel filter. Sobel filters works especially well for edge detection. In fact, it is at the heart of the Canny edge detection. It obtains edges of the buildings by performing 2-D convolution of the original image. The amount of change is captured by gradient magnitude.The operations gives gradient along x- and y- direction, aka, the image's intensity. As we can see in the image, the edges are highlighted because color and intensity change the fastest at these areas . What's good about Sobel is that we can use different threshold for the magnitude and change the direction of the gradient for every image.

6.

Miss Sloane

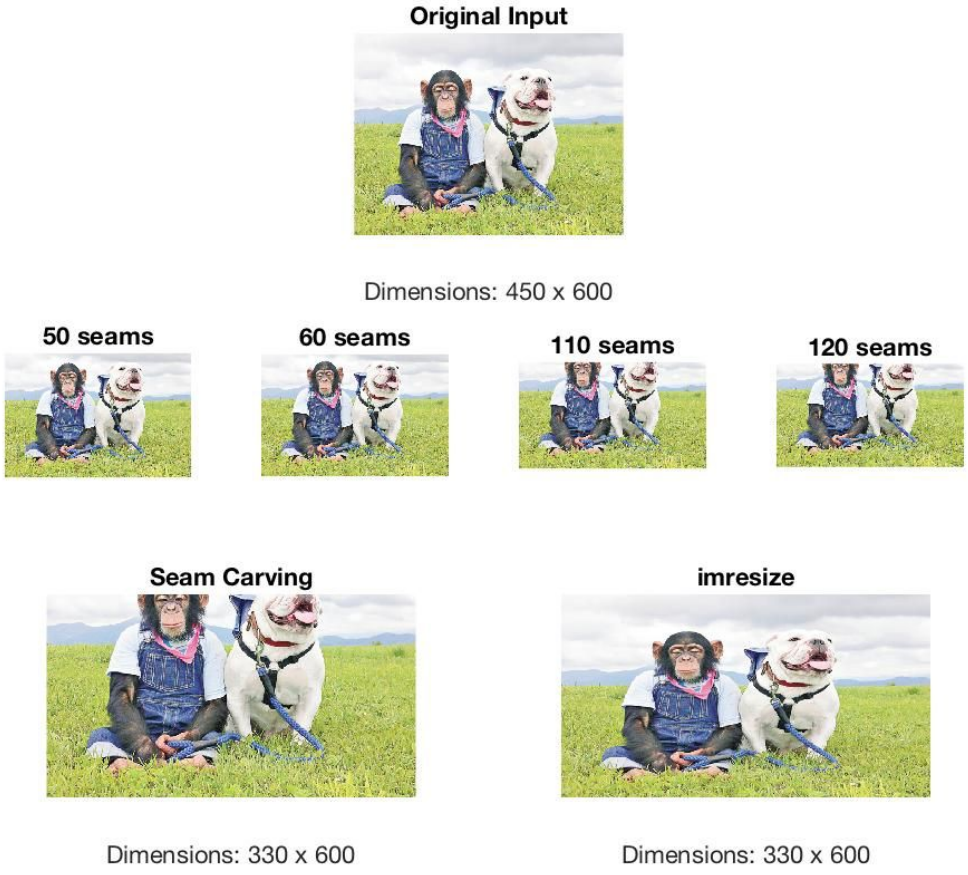


Source
http://freebeacon.com/culture/gun-control-thriller-miss-sloane-one-worst-opening-weekends-ever/

Miss Sloane

I use decrease_height and decrease_width to reduce the image size by 50 pixels incrementally.

The final seam_carving image is reduced by a hundred pixels along both dimensions. I predict that the proportion of Miss Sloan's face would be slightly off due to the fact that there are many colors and no obvious region of homogeneous color and intensity. The result is surprising. Some of Miss Sloane's facial features are distorted. For example, her right jaw is chipped off in a slanting manner. The reason is that seam carving does not distinguish objects of strict proportion from the background. It treats all pixels equally, so instead of preserving Miss Sloane's facial feature, it cuts them out because those seams have low energy in the energy map.



Monkey

Source

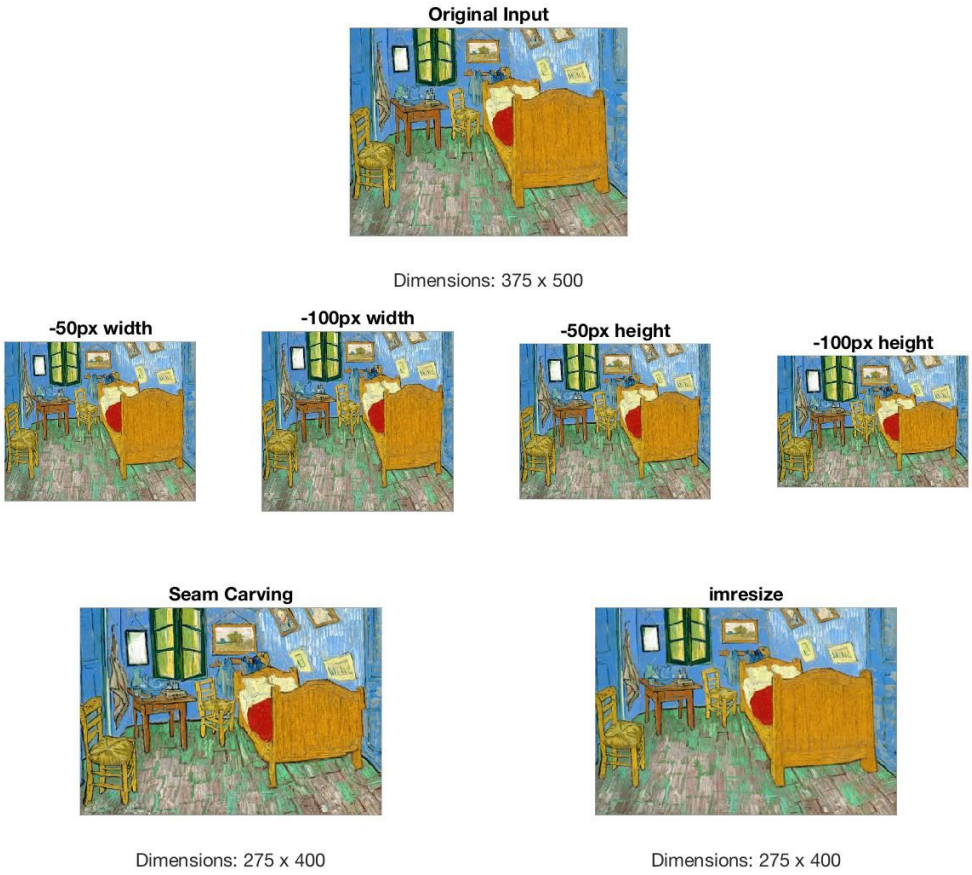
http://hk.on.cc/hk/bkn/cnt/entertainment/20160313/bkn-20160313144117101-0313_00862_001.html

I remove horizontal seams from the top of the image incrementally.

The assumption is that the sky has the lowest energy because it is relatively homogeneous in color and pattern.I predict that the seam carving would not affect the foreground-- our cute couple.

Indeed, when we remove 50 seams, part of the sky is removed and the proportion of the monkey and dog is preserved. However, when we remove more than 100 seams, the topmost regions of the monkey and dog are also removed. It's like using a scissors to cut the blank upper region of the photo, but you can go overboard and cut the important part out.

imresize squashed the monkey and dog but do not remove any part of them.



Vincent van Gogh Van Gogh's Bedroom
Source: https://www.posterlounge.co.uk/van-goghs-bedroom-at-arles-pr122869.html

I use decrease_height and decrease_width to reduce the image size by 50 pixels incrementally. The final seam_carving image is reduced by a hundred pixels along both dimensions. I predict that the image will preserve all the furnitures but part of the door will be cut out.

Observation 1:

Indeed, the rightmost part of the image, the door, is cut, since it is is uniformly blue.

Observation 2:

However, the dimension of the bed shrinks disproportionally to the other furnitures. The bed board in the seam carving image is smaller than the one in the imresize image.The reason is that the bed board has a long stretch of yellow region both horizontally and vertically, thus seam carving cuts it out.

Observation 3:

The window is bigger and closer to the viewer than the one in the imresize image and. The window has complex pattern and thus is preserved; when other parts shrink, the window appears to be bigger.

Extra Credit

Problem 4: Implement the greedy solution

The greedy implementation of seam carving:

1. Start at the pixel with the lowest energy at the first level
2. At every level, choose the lowest energy pixel among its three down-neighbors.

Dynamic programming chooses pixel according to the cumulative energy map, which has each pixel look upwards to determine which seam to extend. In contrast, greedy solution only looks at local minimum.

Since we don't have to accumulate the energy sum of seams looking upwards, we don't need **cumulativeEnergyMap** for greedy solution.

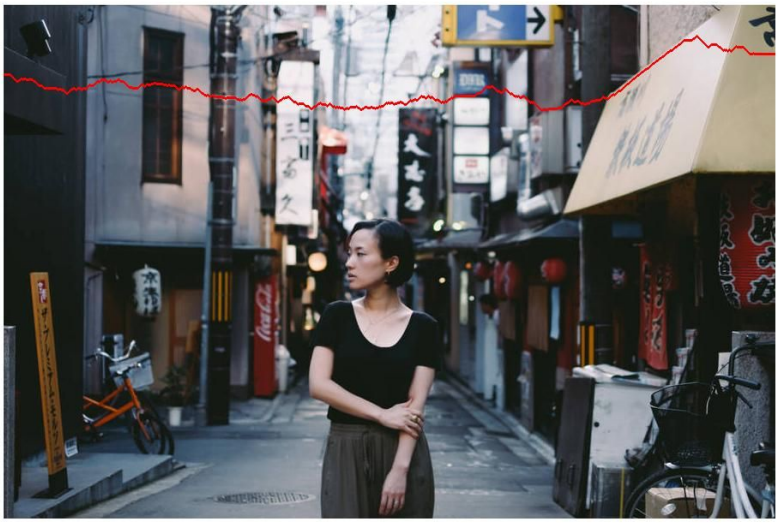
We replace the cumulative energy map with the energy map ,then the rest is the same as dynamic programming implementation.

Instruction

Run greedy.m.

Note that in the last picture, I decrease the height by 100 pixels. To speed up the grading process, I set it to 10 pixels in greedy.m.

Greedy:



Dynamic:

Japan street
Source: http://www.fubiz.net/2015/09/21/japan-street-photography/

Greedy Horizontal Seam

Dynamic Programming Horizontal Seam

Greedy VERTICAL Seam

Dynamic Programming VERTICAL Seam

DP decrease height -100

GD decrease height -100