Muhammed Halbutogullari

913938544

4/25/19

PS1 ECS 174

PT 1:

1: Since convolution is associative, we can optimize the filtering of our image by grouping together all the filters into one filter. This would optimize our run time by making the application of a filter essentially one step instead of multiple steps.

2: Result: [1 1 1 1 1 0 1 1]

3: Gaussian noise may be too perfectly random. For example, if we are taking a picture with a dark background, the darker spots will likely have more noise (and vice versa, brighter spots usually have less noise). If we were to simulate this with gaussian noise, it would not be the same because the gaussian distribution is essentially random (does not take into account the underlying picture) and thus may not be accurate in all scenarios.

4: My assumptions:

-We can extract a non-blurry image for each part on the assembly line

-The orientation will always be the same, and the item will be perfectly centered in the image.

-The background is pitch black

-The parts on the assembly line are equal in size

-The parts come one by one, i.e. we won't have two items in one picture.

-We have a single binary image of the part we are currently expecting, and the image matches the exact size and orientation of the assembly line input video.

Steps:

-Pause conveyor belt and extract still frame from video.

-Convert image to grayscale, then threshold to create a binary image. Then first erode to remove small imperfections such as noise, then dilate to restore the general shape of the initial image.

-We should now have a smoothed out binary image of the item, we then compare this to our expected image, and if there is over 90% pixel mismatch, we flag the item for human inspection.

PT 2:

1: Width Reduction

Original "Mall"

Mall, width reduced by 100 pixels (note difference in rightmost tree)

Original Prague

Prague width reduced by 100 (note the gaps between the top leaves, most seams were taken from the middle)

2: Height reduction

Original mall

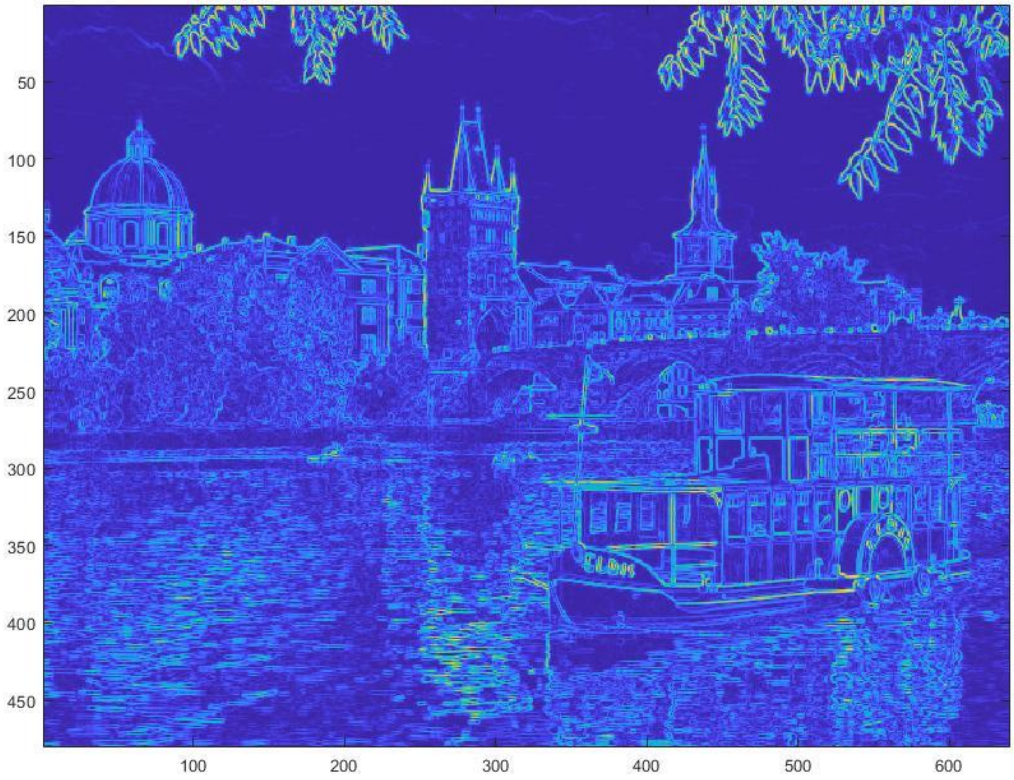
Mall, reduced height by 50 pixels (note the trimmed lawn):

Original Prague

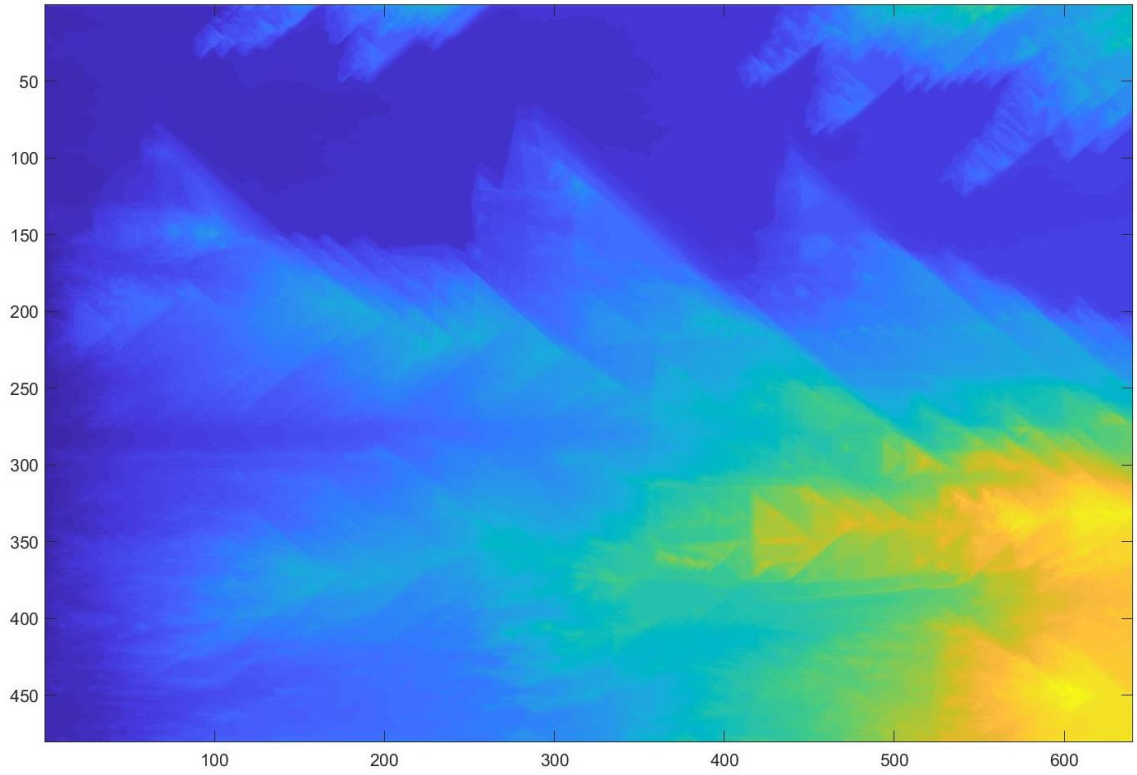
Prague, height reduced by 50 pixels (note the closed gap between the leaves at the top and the buildings)

3A)

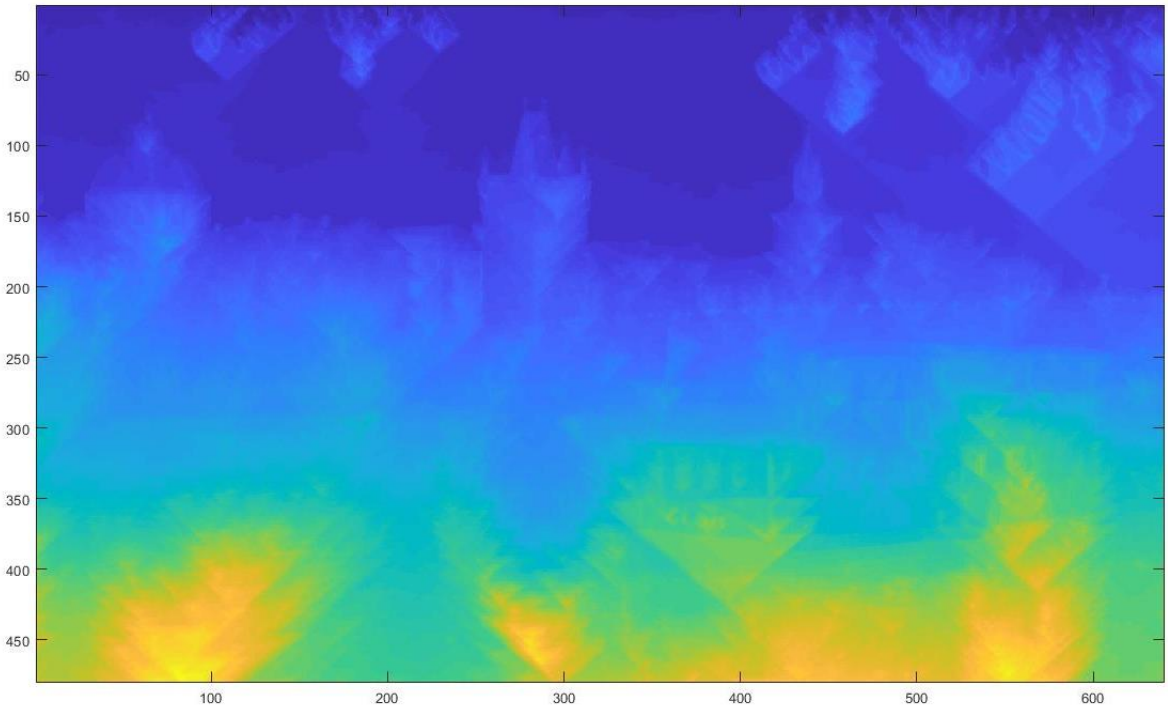
Energy Image of Prague (the sky is dark corresponding to its low imporantance, vs the much brighter edges of the boat, which are more important feautures of this image)



B) Horizontal Cumulative Energy Map: We can see clearly inexpensive paths from left to right in the sky area, which is what the code ends up removing. On the other hand, we see that the brightest portions of the image are right around where the boat is,mainly because of all the edges on the boat our filters have detected. These higher energy spots will be avoided and thus the boat will be preserved as it is seen as a much more central part of the image.



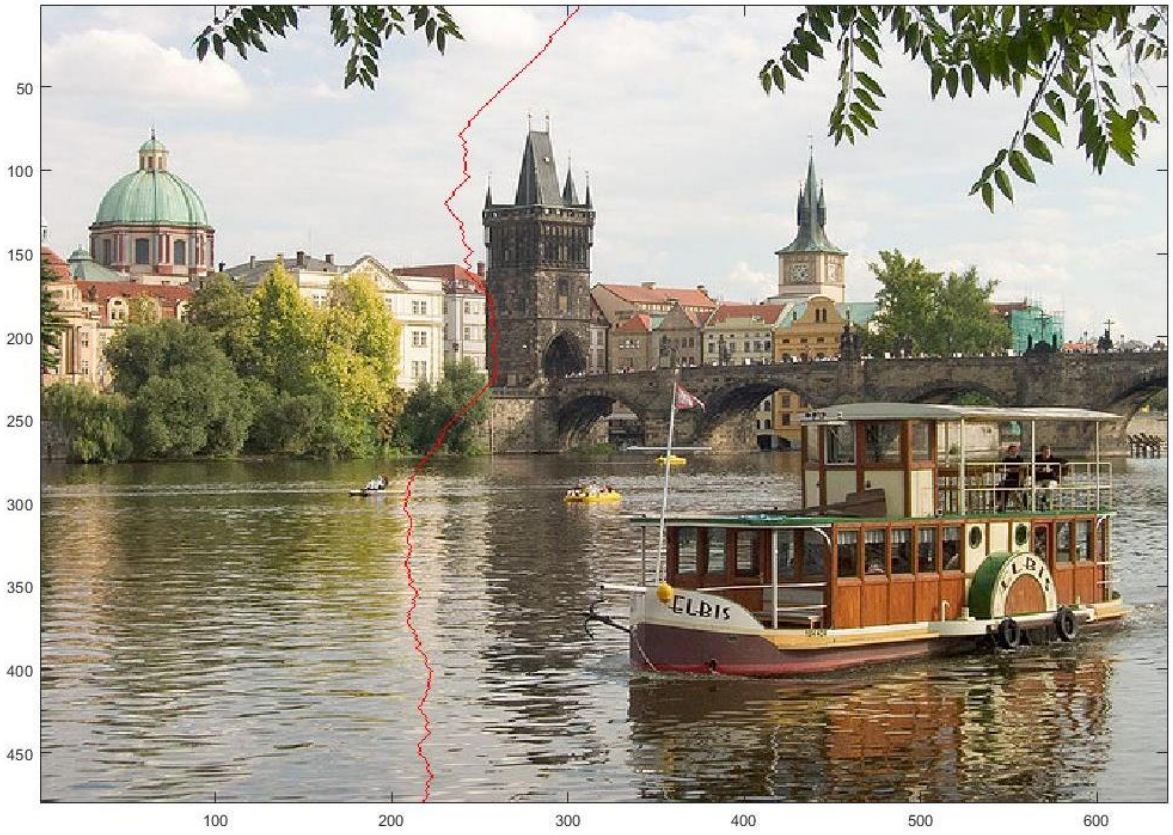
C) Vertical Cumulative Energy Map: Going vertically is a little trickier since there is more going on in the lower half of the image. Since the boat has a bunch of edges, we can see that the areas under the boat are very bright and thus will be picked last as to not disort the boat. The left side also has quite some brightness, mostly accumulated energy from the buildings and bushes above. We can see that the darkest vertical path seems to be around the 200 pixel index, which looks like it is avoiding the bushes and the boat. The darker portions mean accumulated energy is lower in these areas, and they will be removed first.



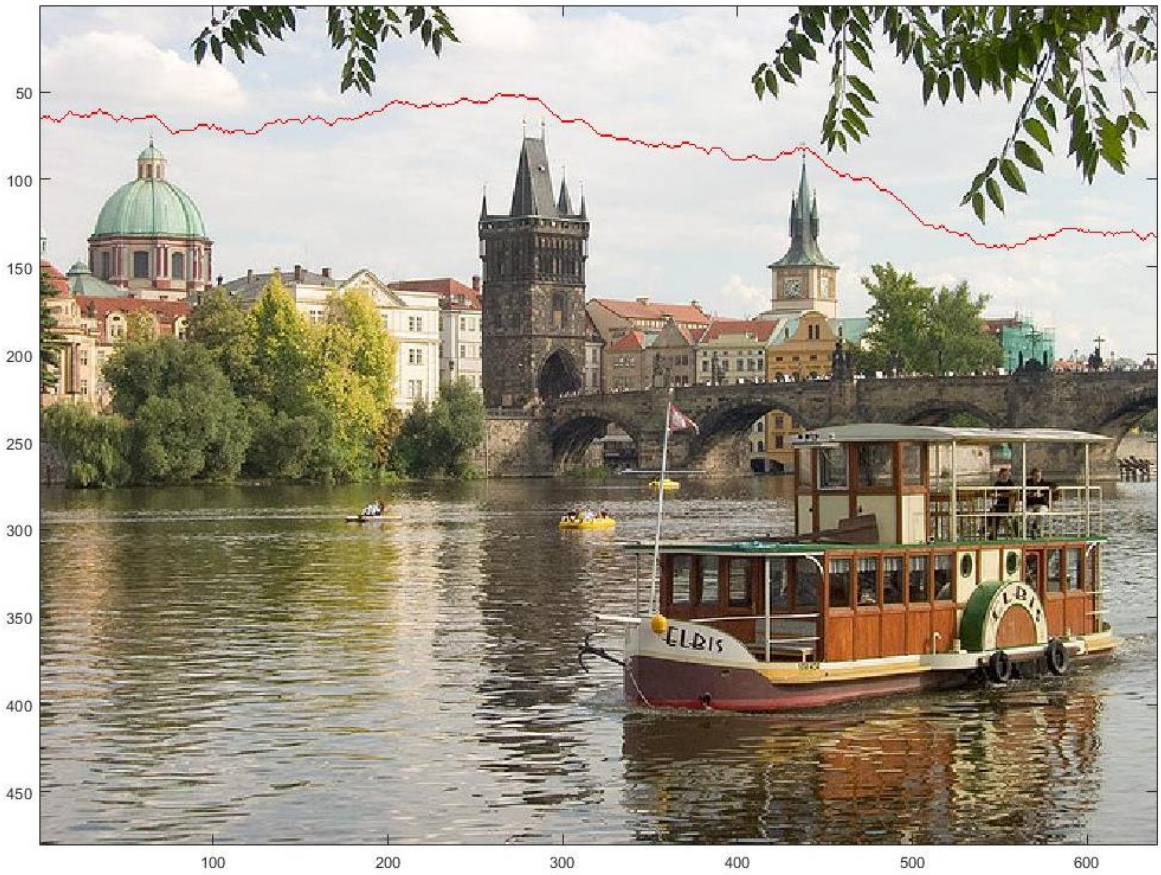
4)

Original Prague

First Vertical Seam, we can see here that my hypothesis in part 3 was correct. The seam seems to have avoided the boat and also the bushes as it considered these as the most important parts of the image



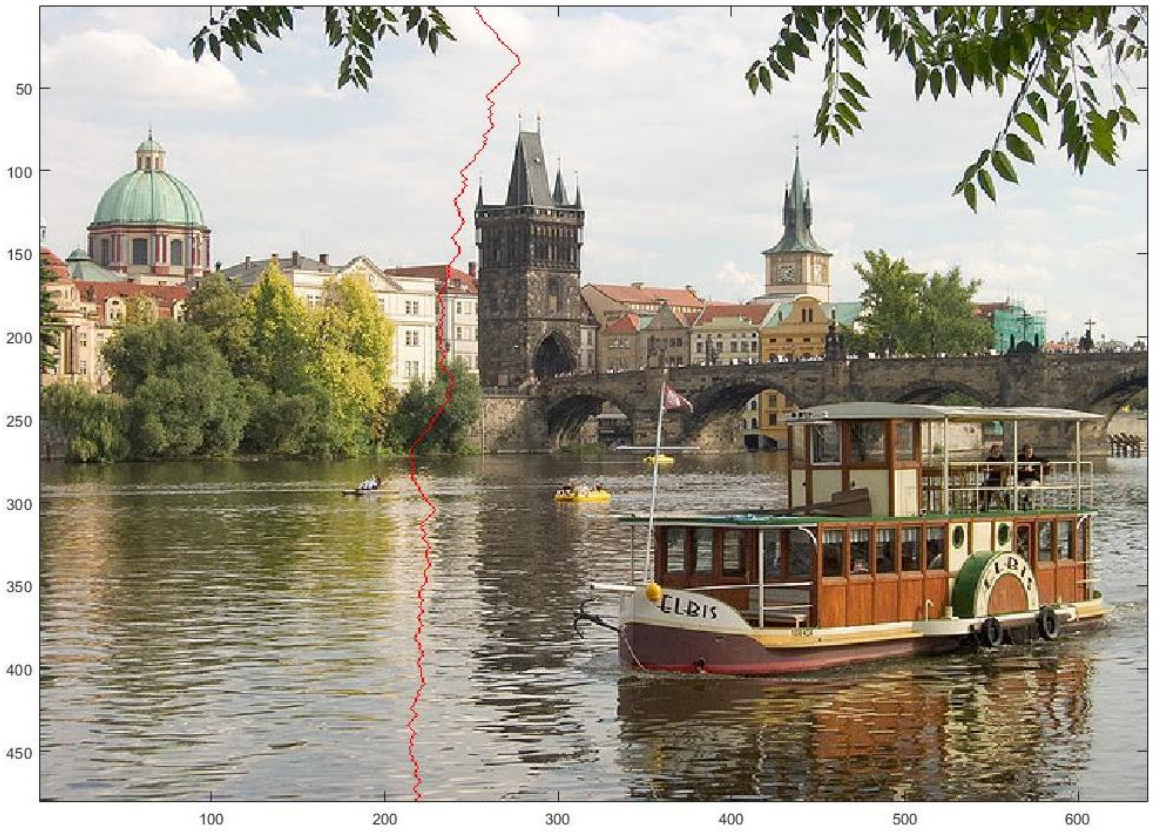
The first horizontal seam is also as expected, cutting through the parts with the least energy (the mostly solid colored [no edges] background sky)



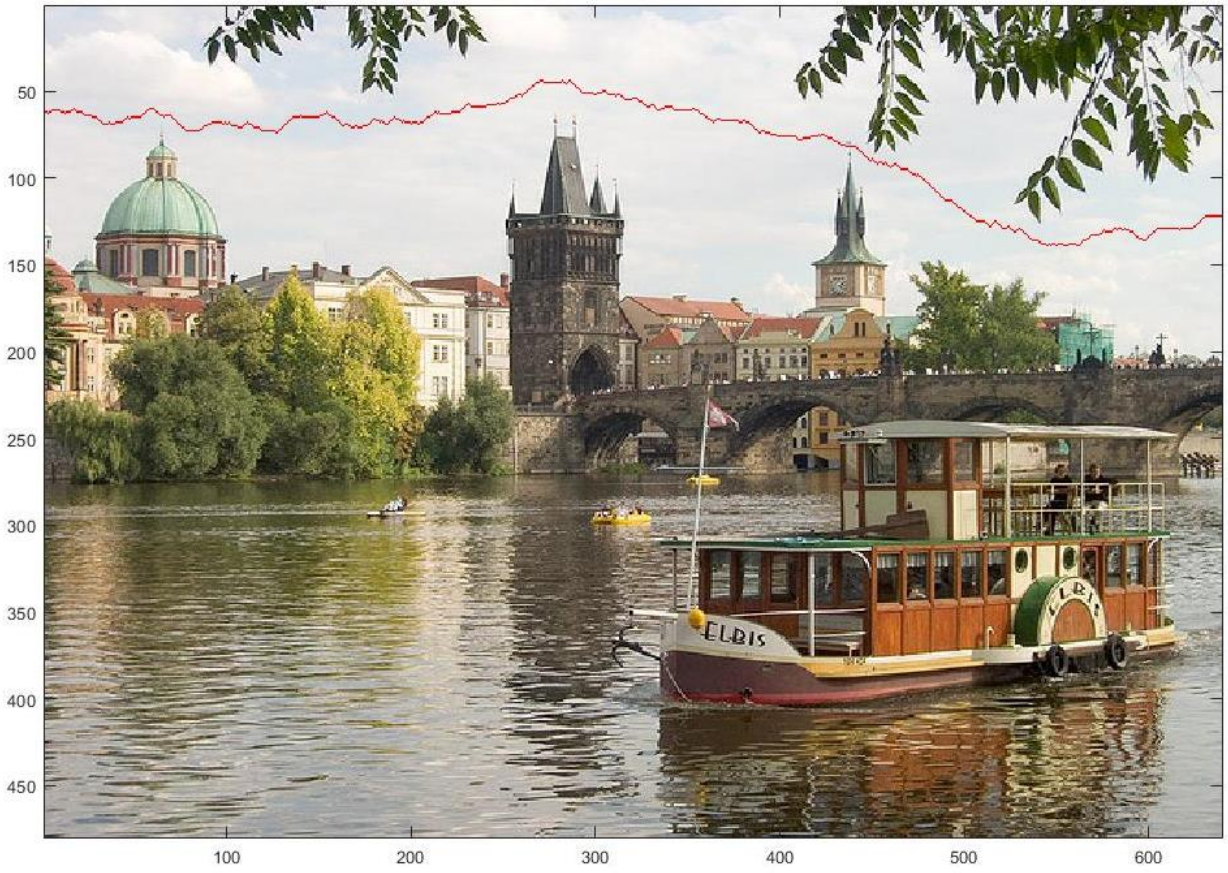
5) I changed my code from a sobel filter to a simpler filter ([-1,1] and [-1;1]), then reused my prob4 script

Original Prague

The first vertical seam comes down the same general region as the first, but it takes a path that is a little more to the left. I think the simpler filter avoided the areas near the dark building because of the immediate color change which the filter [-1,1] would see as a much higher energy edge, vs the more complex sobel filter.



The horizontal seam is almost identical to the sobel filtered horizontal seam, as I expected. When using both filters, there is very little energy in the sky as there aren't any edges, so the horizontal seams from both filtering methods are naturally very similar to one another.



6) I wrote a script called prob6 to do everything. For the next 3 images "resized" means resized by my seam carving. My first image was a BMW M2. I chose this image because the floor had interesting shadows and the background also had some changes. I reduced the height by 150 pixels. Unfortunately, the seam carving failed pretty badly, and morphed the car terribly. The imresize results made the m2 quite compressed, but it was still better than the distortion from the seam carving resize.

Original M2

623x1000

M2 resized

473x1000

M2 imresized

473x1000

Next, I chose a picture of pikachu to be modified. I wanted to see how my algorithm would alter the facial features. I reduced the height by 100 pixels. In this case, the algorithm mostly removed seams from the upper part of his head, leaving his facial features relatively unaltered. Compared to the imresize, which had a noticeable stretching effect all over, my seam carving did a slightly better job at resizing this image.

Original Pikachu

800x1200

Pikachu resized

700x1200

Pikachu imresized

700x1200

Finally, I chose the UCD logo for the final image. I chose this because I wanted to see how the seam carving would avoid the text, or if it would totally fail. I chose to reduce the width this time, by 50 pixels. The results of seam carving were astounding! The algorithm removed most of the whitespace between each letter, the left and right sides, and also between the words and the logo. This was a very ideal resizing vs the imresize which left a significant, ugly compression on the words and overall image.

Original Davis Logo

265x564

Davis resized

265x514

Davis imresized

265x514

EXTRA CREDIT #3: I implemented the method explained in the paper to increase the height of an image. My code can be run using the script extra_credit.m

My first test subject was classic old Prague. At this point I clearly knew the sky would be expanded the most, and not surprisingly that is what it did. There was a little distortion to the clouds (due to averaging values), but the clouds were probably the least important feature anyway, so it worked as intended.

Prague, Normal

Prague, height increased by 100 pixels

I also wanted to test my code on pikachu, since pikachu worked so well with reduction. I found that increasing the height of pikachu also worked very well. After comparing with the original, it seems like most of the seams were added in the forehead region, which has the least features in respect to the rest of pikachu's face. The shading on his head was also preserved really well. I don't think I would be able to tell this image was stretched if I hadn't coded & stretched it myself. Overall my implementation of height increasing has been a success.

Normal Pikachu

Pikachu, height increased by 100 pixels (minimal distortion noted in left ear)

I wanted to test my width increaser too, so I ran it on the m2 since I had so many problems with this image before. I was interested to see what would happen. I increased the width by 50 pixels, and the results were suprisingly not that bad. The stretched car looks pretty good at first glance, but if you look a little closer you begin to notice some minor disortion at the front tire. Besides the minor issues with the tire the rest of the image is quite realistic, and not a flat out failure like it was when we decreased height. Overall my increasing pixels by averaging seems to work pretty well.

Original M2

M2, width increased by 50 pixels

IMAGE SOURCES:

Pikachu: https://cdn.vox-cdn.com/thumbor/1Evq57t9d53K2iHHjc6AkWRSKGA=/0x0:1280x960/1200x800/filters:focal(538x378:742x582)/cdn.vox-cdn.com/uploads/chorus_image/image/57601275/60861120_1280x960.0.0.jpg

UCD: http://mediaro.info/wp-content/uploads/2018/12/uc-davis-logo-uc-davis-logos.jpeg

M2: https://www.driving.co.uk/s3/st-driving-prod/uploads/2016/05/bmw-m2.jpg