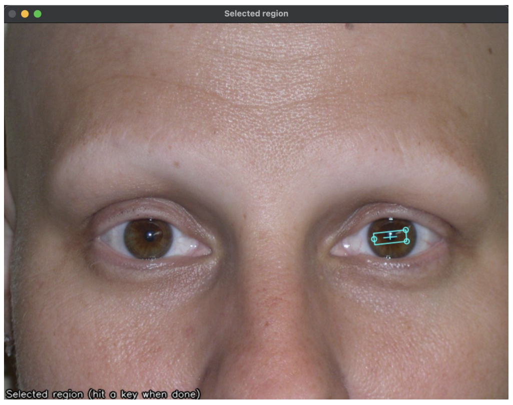
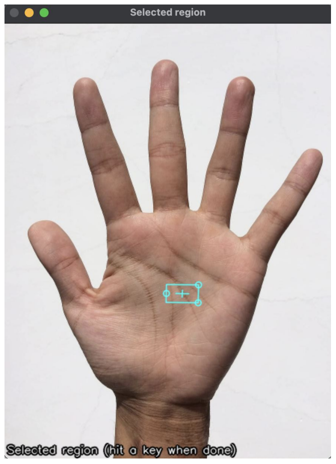
Great work!
Grade: A+

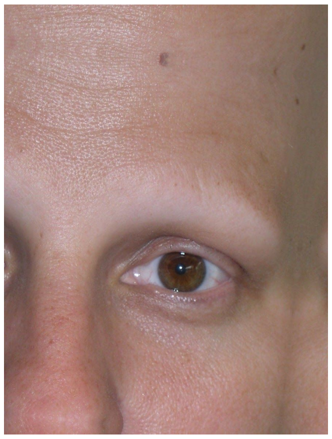
LAPLACIAN BLENDING

For the laplacian blending, I decided to use a picture of a hand and an eye. The hand picture was obtained from https://www.pikist.com/free-photo-skuau. The eye picture was obtained from https://hu.pinterest.com/pin/308426274465247127/. I selected horizontal regions of interest and treated the eye as a "horizontal face" by the following command: **python ../project2_util.py select ../lptest/eye.jpeg eye.json**
I then selected the same horizontal regions of interest for the hand as well by the following command: **python ../project2_util.py select ../lptest/h.jpeg h.json**





Then I warped the image of the eye to align with the image of the hand with the following command: **python ../project2_util.py warp ../lptest/eye.json ../lptest/h.json eye_warp.jpeg**



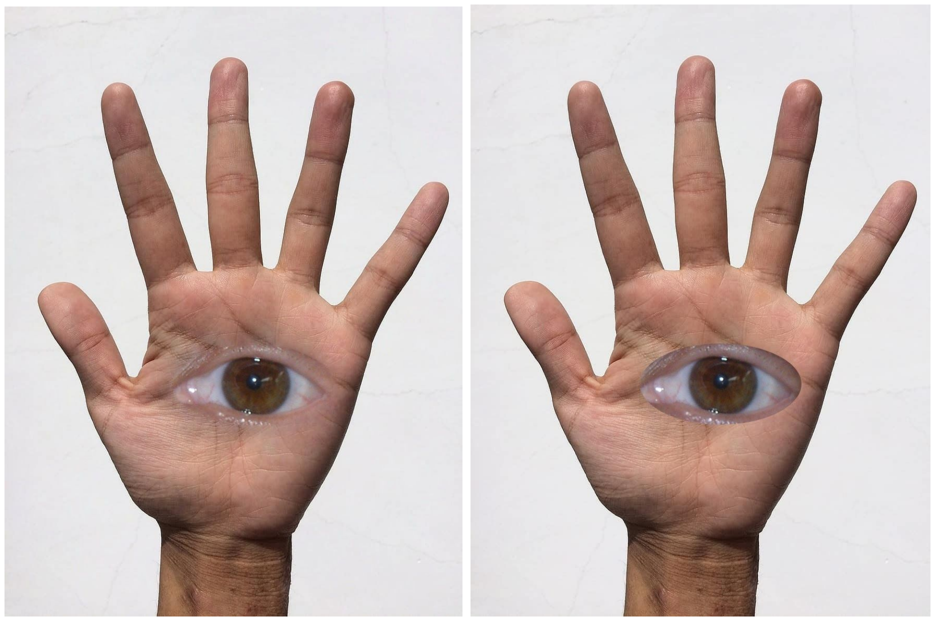
Then I created an ellipse mask with 1.8 as the vertical distance and 3.6 as the horizontal distance with: **python ../project2_util.py ellipse ../lptest/eye.json 1.8 3.6 eye_mask.png**
Now I just needed to run laplacian_blend.py with the two images and the mask. **python ../laplacian_blend.py ../lptest/h.jpeg ../lptest/eye_warp.jpeg ../lptest/eye_mask.png result.jpeg**
I ran this as well as the alpha blend command to compare and contrast the two methods. The left image is the result of the laplacian blending and the right is from alpha blending. There is clearly

a better transition from the masked eye to the hand resulting from the laplacian blending, and we can’t see the outline of the ellipse mask in the laplacian blend.

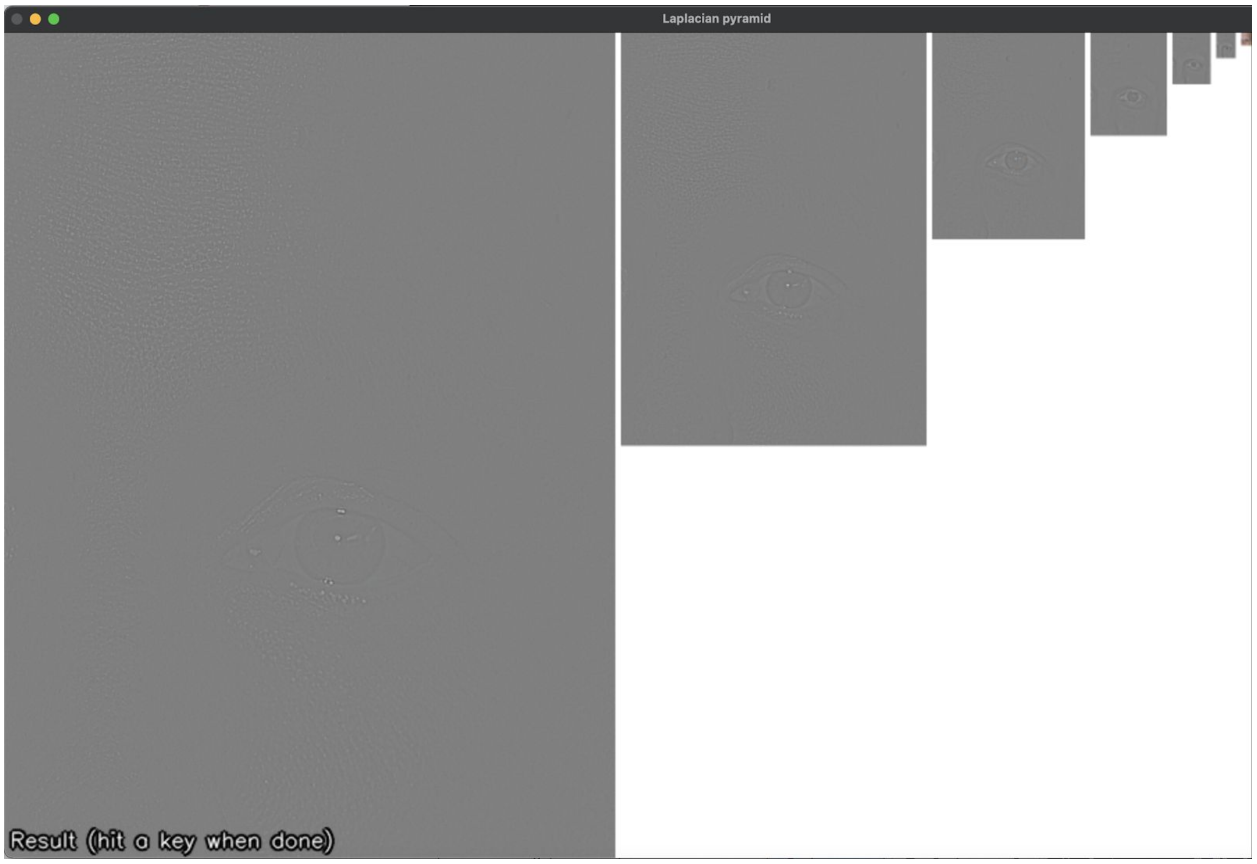
Now I’m not going
to sleep well at night.

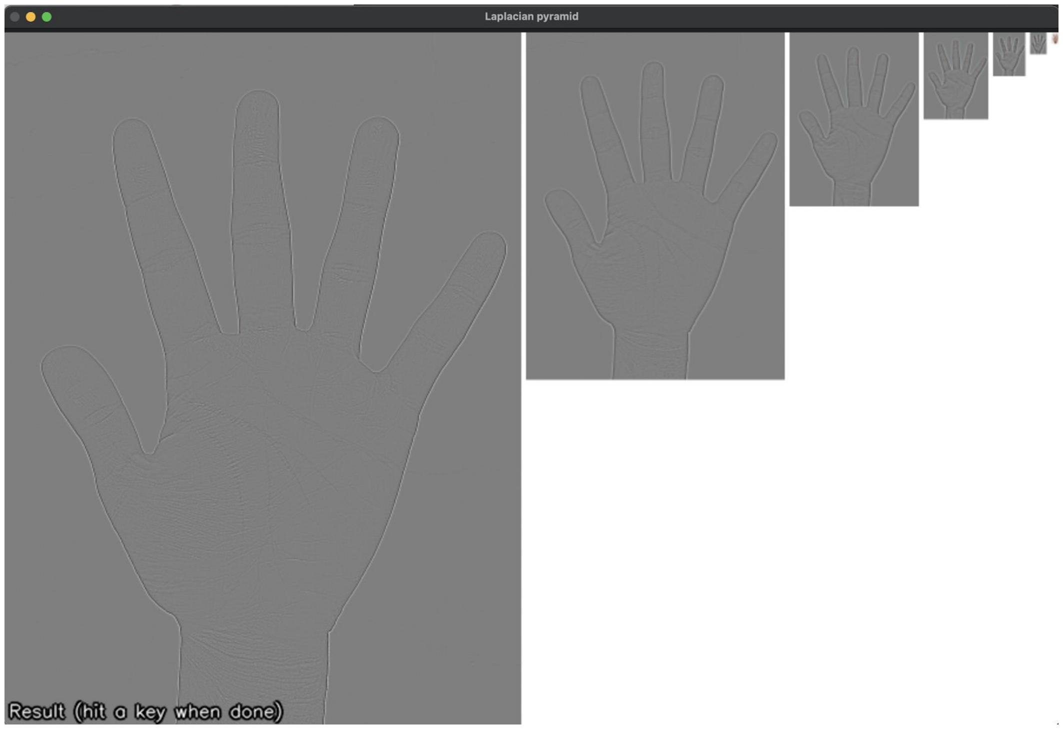
Thanks a lot Patrick!

(This is great)



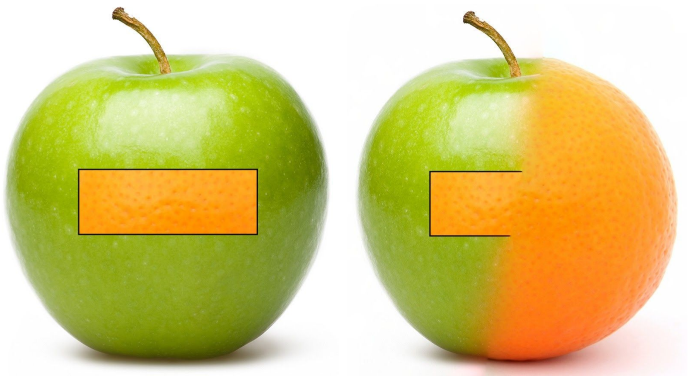
The constituent laplacian pyramids were outputted and are shown below. The left hand side is the eye and the right hand side is the hand.





The hallmark of Laplacian blending is that it blends low frequency features over a larger distance than high frequency features of the two images. Essentially, we are taking the gaussian filters of the binary mask itself repeatedly and alpha blending the mask to one image, and 1-mask to the other image. These are then combined. Since, in the early iterations, the edges are still very prominent in the laplacian pyramids, and very few or no gaussian iterations have been applied to the mask, the edges are still sharp and we get normal alpha blending for the high frequencies of the two images. However, we get more of the low frequencies as the pyramid builds up. In these later iterations, the binary mask has also been through many gaussian blurring iterations. The laplacian pyramid of the image at this point also contains more of the lower frequencies. Hence when we apply alpha blending to this gaussian blurred mask to these levels of the laplacian pyramid, we get a more smoothed image across the two images. Consider the modified apple picture in example1. The sharp black lines of the rectangle should not be blended smoothly across the mask border as shown below. Where the orange and green color started blending, the black line has not even started to fade yet, and then just abruptly disappears.

Really nice illustration!

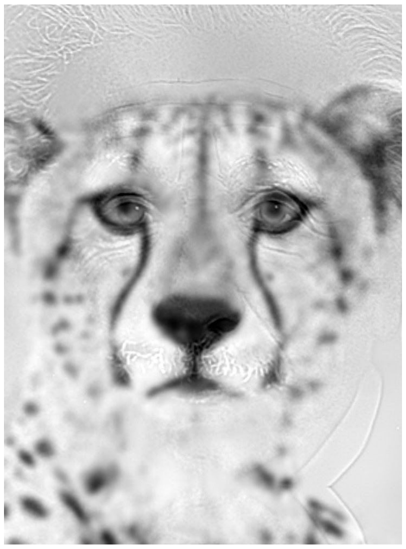


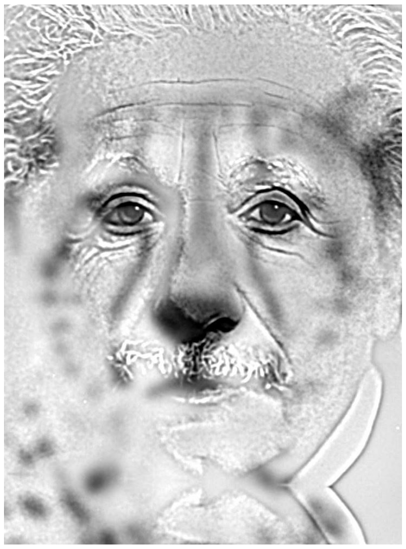
HYBRID IMAGES
For the hybrid image, I chose Einstein from https://hu.pinterest.com/pin/463800461603519809/, and a picture of a cheetah from https://www.wallpaperflare.com/cheetah-monochrome-grayscale-photo-of-cheetah-animals-wild-wallpaper-ctgnr. I selected the regions of interest by selecting the eyes and the mouths, then cropped the einstein image: **python ../project2_util.py crop ../testhybrid/ein.json 2.2 3.0 1000 0 50 ein_crop.jpg ein_crop.json**

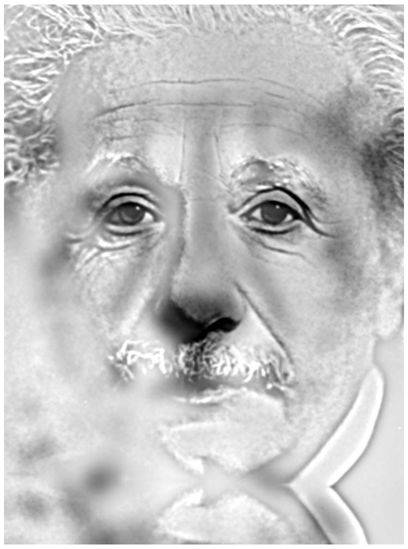
I then warped the cheetah image to the cropped einstein image: **python ../project2_util.py warp ../testhybrid/chee.json ein_crop.json chee_crop.jpg**

Now that they're aligned, I called the hybrid program to make a hybrid image: **python ../hybrid.py ../testhybrid/chee_crop.jpg ../testhybrid/ein_crop.jpg 8 1.5 result815.jpg**

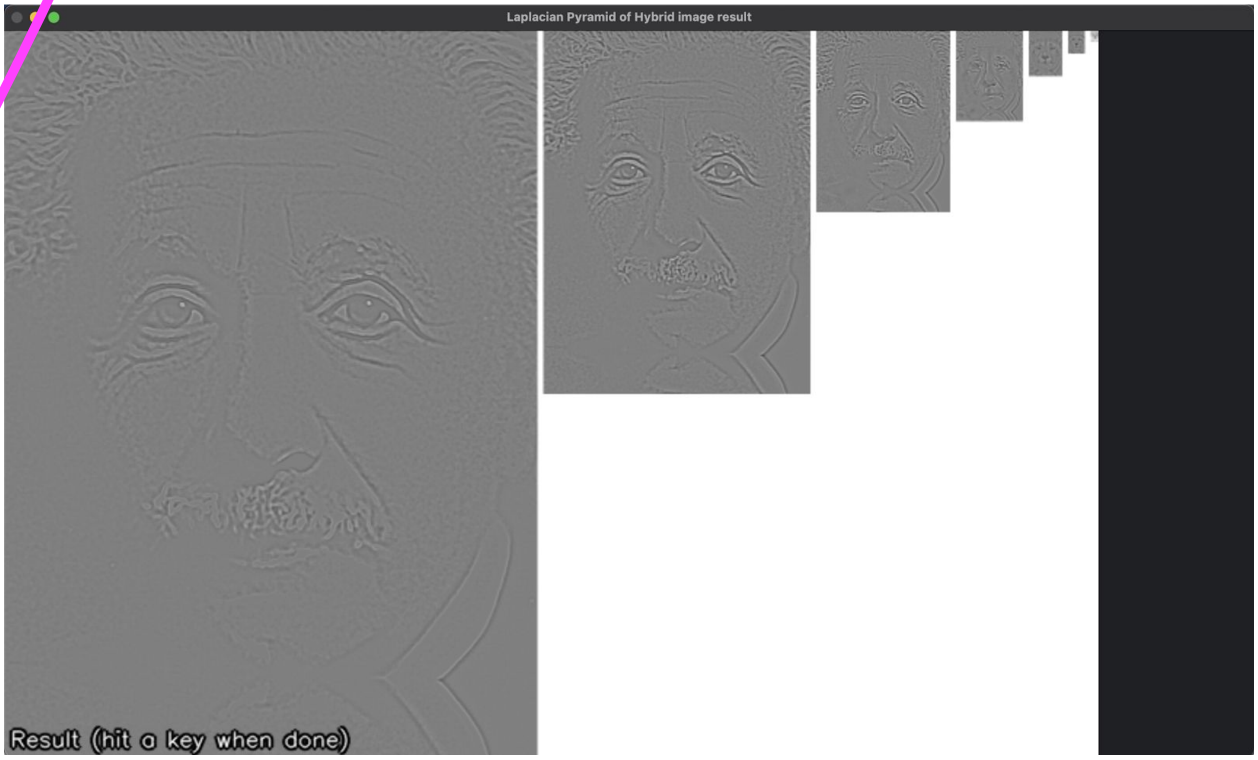
Image B, the Einstein image, was not very noticeable so I increased both the sigma and k values from sigma = 2 and k = 2 to sigma = 4.5 and k = 3. By doing this we are getting a "fatter" gaussian curve since the standard deviation is being increased. This allows for the higher frequencies of image B to pass, allowing us to see the edges of the Einstein image better close up. I then increased the value of sigma even more up to 8 so that image A is blurred more so the cheetah spots are not too prominent. Then I had to decrease the K value since the edges on the Einstein image were too prominent and I had to zoom out too much until almost all the edges went away. The final values that I chose were sigma = 8 and k = 1.5. The modifications are shown below.


Sigma, k = 2, 2


Sigma, k = 4.5, 3


Sigma, k = 8, 1.5

If I display my image at full size on my computer screen, I can clearly see image B at a normal distance from the screen, which is around 1.5 feet away from the screen. To clearly see image A, I would have to be around 17 feet away from the image at full size on my computer screen. The thumbnail of the image also looks like image A. These distances do seem to be consistent when shown without any context to friends and family members.


Laplacian Pyramid of Hybrid image result
Result (hit a key when done)

The early iterations of the pyramid resemble image B, Einstein, and the later layers of the pyramid resemble image A, the cheetah. Since our image is just I = A_low + k*B_high, the constituents of our image is composed of the low frequencies from A, the cheetah, and the high frequencies from B, Einstein. We can think of the laplacian pyramid as the differences between successive gaussian blurs on our image. Since the laplacian pyramid just filters out low frequencies, in the early iterations, the edges are very prominent, which corresponds to the high frequencies present in image B. Therefore, we see more of image B, Einstein. Afterwards, the difference between successive gaussian blurs no longer contains very prominent edges since there have been many iterations of gaussian blurs. Since the Laplacian of the Gaussian is a band pass filter, at this point, it's allowing high frequencies to pass that's relative to the current gaussian differences. These "high frequencies" are actually low frequencies relative to the original image. Hence, more of the high frequencies are suppressed from the original image. Therefore we see more of the low frequencies, which corresponds to image A, the cheetah. This could also be because the image is simply getting smaller, hence we detect less of the high frequencies from image B, and see more of image A, the cheetah.

FURTHER CONSIDERATIONS

Eulerian Video Magnification

Upon searching for where laplacian pyramids can be useful, I stumbled upon Eulerian Video Magnification. We take some video as an input. Then, we process each frame in the video and create a laplacian pyramid from all the respective frames. We can then process the pixel values over time and apply temporal filtering, resulting in an extraction of the frequencies of interest. We can then amplify these frequencies of interest such as motion or color changes that are vital but unseen to the naked eye so they are more apparent. We then combine it to the original pyramid frames and the original frames are then reconstructed from the pyramids. We are then left with a video whose features of interest are enhanced, such as seeing prominent changes in the color of someone's face in relation to their heart rate, or tracking minor facial movements and ticks in response to some environmental stimulus. There are various biomedical applications of this method, one of which is noninvasive monitoring of heart failure. Pulsation can be monitored by enhancing the motion detected from the patient, which can provide information on the current health status of the patient.

✓

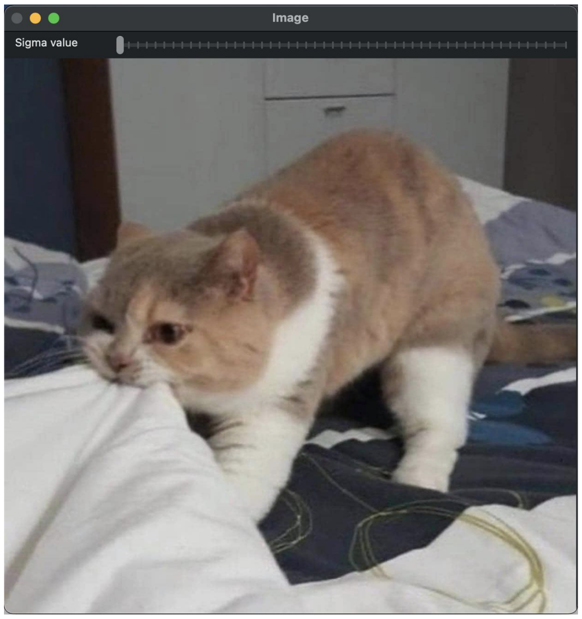
There are some considerations and questions that arise when I was considering this method. The relative redness representing blood flow in the face could be affected by lighting, and more importantly, skin tone. I am curious as to how these factors would be dealt with. Lighting could be dealt with by taking the relative redness of the pixels through time, but different skin tones could present difficulties as blood flow may not be as visible for darker skin tones.

Nice discussion

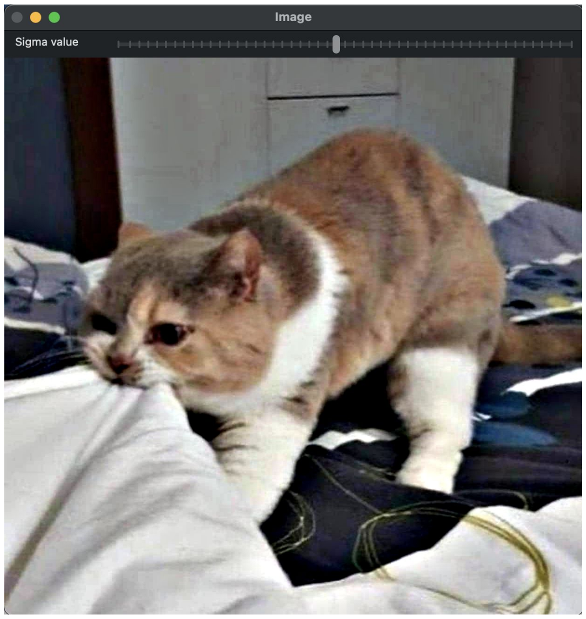
Sharpening Images

I decided to play around with extracting different frequencies from the images and tried extracting high frequency filters from a photo and amplifying them and adding them to the original photo to make the edges more prominent, or to "sharpen" the photo. To do this, I created a short program sharpen.py, which gets the gaussian blur of the image, and subtracts it from the original image to get the high frequency image. Then I just add that image to the original to amplify the edges. I added a slider which varies the values of sigma for blurring the original image. I saw that the bigger sigma was, the sharper the image became since I am filtering out more and more low frequency signals. I found this cat image in my photos and could not figure out where I screenshotted it from. The other image that I used is a picture of my friend that I took years ago.

Yes, this is a thing people do - see also "unsharp mask" filtering



Sigma = 1



Sigma = 26