Dassie Galapo
Somayeh Keshavarzdarolkalaei
Computer Vision
09/22/2022

Project #1 Hybrid Image

The goal of this project was to superimpose low frequency and high frequency components of two separate images to produce a hybrid image that appeared one of two different ways based on human perception and viewing distance. In a high-level overview, I first transformed the two 2D images into their frequency domains using Fourier transforms, then multiplied each of the images in frequency domain with either a low or high pass filter, took the inverse of the Fourier transform to transform the images back into spatial domain, and overlapped both images in spatial domain to produce a hybrid image.

In order to complete these tasks, I first loaded in the images and resized them to match each other's sizes. Then, I created the following functions to produce the low and high pass images used to create the hybrid image.

The **low_pass_filter()** function takes in the image's number of rows, columns, and a cutoff frequency, and returns a Gaussian low pass filter. The cutoff frequency has a mathematical relationship to the standard deviation in the Gaussian distribution, namely that Sigma is inversely proportional to cutoff frequency. Therefore, in order to make the image sharper, one would use a higher cutoff frequency which corresponds to a smaller sigma value, thus giving less weight to pixels further away from the mean, and one would use a lower cutoff frequency to make an image appear more blurry. This was useful in producing the desired low pass Gaussian filter and after much experimenting to find the optimal hybrid output, I chose a low cutoff frequency of 10, which served to suppress all frequencies higher than the cutoff value and keeps values of lower frequencies. This function iterates over each pixel value, computing the Gaussian distribution formula at each pixel to produce a Gaussian lowpass filter. Values in the Gaussian low-pass filter lie between 0 and 1, where a value of 1 corresponds to low frequency, and a value of 0 corresponds to high frequency. Rather than creating a separate function for the high-pass filter, I simply computed it by subtracting 1 from the Gaussian low-pass filter that was created from this function.

The **low_pass_fft()** function takes an image and transforms it into its frequency domain using a Fourier transform and returns a re-transformed image in the spatial domain. First, it uses Numpy's Fourier transform function to convert the image into frequency domain, and then performs a shifting to center the low frequency pixels and place high frequencies furthest from the center. By transforming the image into frequency domain, I was able to perform a multiplication operation between the transformed image and the Gaussian lowpass filter to achieve a low-pass image in the frequency domain (as opposed to needing to perform a convolution in the spatial domain). Once this operation is performed, the image is reshifted so that its higher frequencies are recentered and lower frequencies are furthest from the center, and

then an inverse fourier transform is performed on the image to transform it back into spatial domain. I chose to pass my cat image to the low pass filter function and it produced an image that appeared blurry, as desired.

The **high_pass_fft()** function performs similarly to the function above, but rather than multiplying the transformed image in the frequency domain by a low pass filter, it multiplies it with the high-pass filter that was obtained by subtracting 1 from the low-pass filter, and returns a high-pass image in the spatial domain. When I passed the Trump image to this function, it returned an image with distinctive, highlighted edges, as expected.

In order to add color to the hybrid image, I iterated each color channel of the image that I wanted to be low-passed through the low_pass_fft() function, and iterated each color channel of the other image that I wanted to be high passed through the high_pass_fft() function. I decided to use color for both the high and low frequency components, but I ended up adjusting the r,g,b values that came out of the highpass filtered image since the hybrid image was initially too dark. I retained the color information in both frequencies since it did not cause interference in the spatial domain when overlapping both images. However, it is clear that most of the color information comes from the low frequency image, whereas high frequencies provide information about fine details and textures.

The hybrid image that I created displays the low frequencies of the cat image overlapped with the higher frequencies from the Trump image, which produces the effect of being able to see Trump from a close-up view of the image, and seeing the cat from a further distance. I used a weighting function to handle the overlaying of both images. After experimenting with different values, I tuned it so that the low-pass (cat) image would be less opaque than the high-pass (Trump) image by a 2:3 ratio. I found that this provided a smooth balance between seeing the fine details of Trump when peering close to the image, and seeing the cat from a far distance with its features still visible.