Jacob Fund
Computational Photography
Assignment 6
Summer 2015

Black Image

White Image

Mask

Final Output

Black Image

My good friend. Don’t worry I got his permission. I had to rescale the image to fit the mask and my other friends head size.

White Image

My other good friend - He reluctantly let me use this, but I just love how ridiculous he looks in it. Sort of like a flamingo. My other friend likes to poke fun of this picture which is why I thought it'd be interesting to mix things up a bit.

Mask

Originally, I kept the mask to fit my friend's face in the black image. I realized that I needed to add a little more to the neck. Honestly, I a little more black on the sides would probably have returned better results, but I was happy enough with the out as it was.

Algorithms.

Reduce: The suggested convolution function for the image and the kernel was used. To make sure the reduce functioned returned a halved image, I checked whether the rows and columns had odd or even numbers like so:

```
if (rows %2) == 0:
   yStart = 1
  else:
   yStart = 0
  if (columns % 2 ) == 0:
   xStart = 1
  else:
   xStart = 0
```

This gave where the starting index would begin. Then I would put every other element until the end starting from these elements.

Expand: Likewise, the expand function used the same convolution function as the reduce function did. But first the image is extended to twice the size for rows and columns. Then the original image values are spread into the new expanded image like a checkerboard. Once the convolution is done, all the elements must be multiplied by four in order to compensate for the empty values in the expanded image.

gaussPyramid: Simply, for each level add the reduced version of the previous reduced image and put it in an array.

laplPyramid: Now that we have the Gaussian pyramid, we can find the Laplacian Pyramid by taking the difference between each layer and the expanded version of the next layer. To make sure the expanded level matches the current layer, I simply indexed the expanded image to the same dimension size of the current layer.

Blend: It is important to match the same dimensions as the reduce function so the similar indexing check was used as the reduce function. The mask needed to be normalized so that all values where 0 to 1. I used the sciimage function, img_as_float. Then for each layer of the Laplacian Pyramid, I did like the directions suggested, I did a computation for the two images: *gaussPyrMask * laplPyrWhite[layer] + (1 - gaussPyrMask) * laplPyrBlack[layer].*

Collapse: Finally, The blended pyramid needs to be brought back down the pyramid to the same size of the original two images and blended. To do this I iterated over the blended pyramid starting from the last element and moving backwards, and when the function got to the first element I had it return the summation of the entire pyramid.

Final Output:

Obvious caveats that appear in the final solution is the fact of a lighting difference. Although seaming to blend in somewhat, There is still a difference in intensities between the two pictures and that give its rather humorous but unsettling difference. Perhaps a darkening of the black image would have helped with the results.