Alex Johnson
CS4300 - Assignment 3
Due 10/19/14

Instagram Reverse Engineering

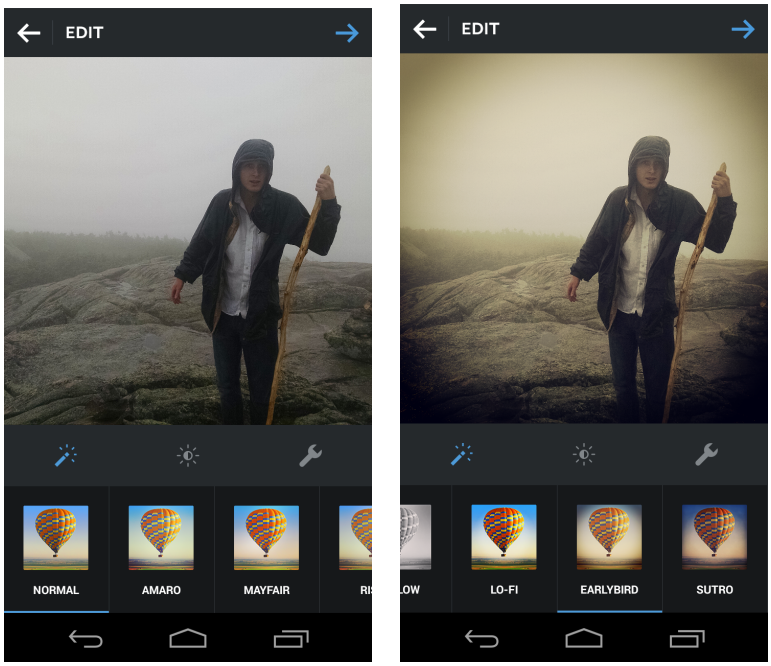
1. Abstract/Thesis & Introduction
2. Methods
3. Code
4. Results
5. Discussion
6. References

Abstract

"Write a recipe for an Instagram filter."

The goal of this assignment was to deconstruct and replicate a filter from the app Instagram. I chose to analyze and recreate Instagram's "EARLYBIRD" filter. EARLYBIRD applies a number of effects to images including color dampening, level adjustments, and a soft circular vignette [4]. The filter is designed to be similar to the image style of old photographs. Fig 0.1 shows two screenshots where the user is using Instagram to apply the EARLYBIRD filter.

Fig 0.1 - Instagram's EARLYBIRD Filter



I analyzed the raw image data of filtered images. I was able to approximate Instagram's filter into a few key attributes that it was manipulating. These attributes included levels, colors, and contrast. The *methods* section addresses how I isolated and implemented these adjustments into a custom image filter. My custom image filter was able to create images that were similar to the images filtered by Instagram. However, there were still a number of subtle differences between Instagram's filter and my implementation.

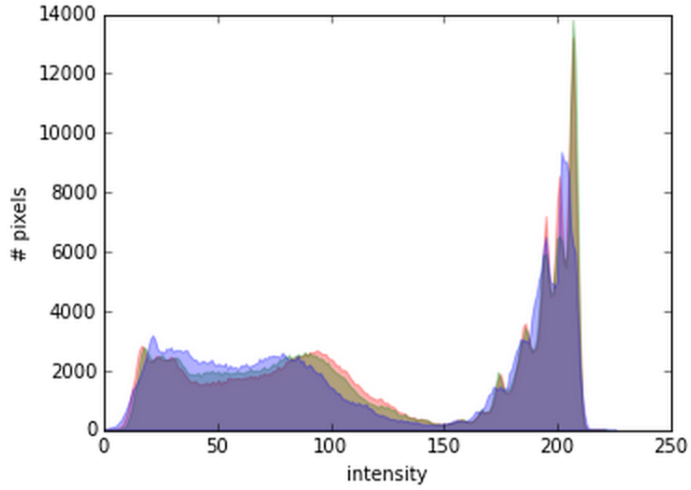
Methods

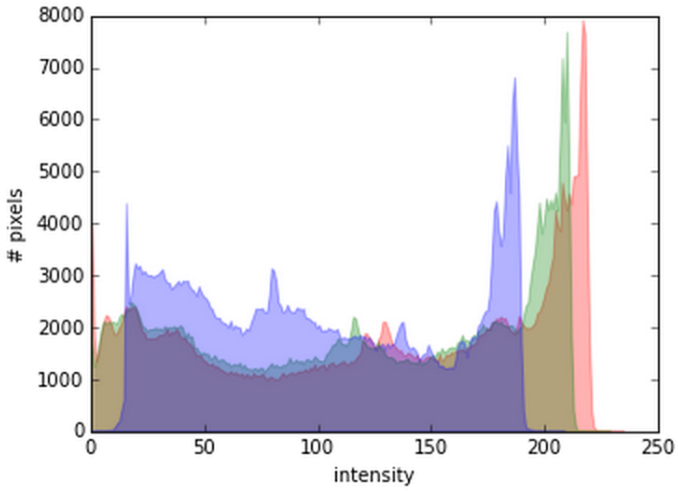
The first step to replicating an Instagram filter was to deconstruct the filter. I compared the original image to the image data after Instagram had applied the filter. This comparison helped gain insight into how the effects could be recreated. The images in Fig 1.1 show the comparison between the original image's histogram, and the histogram from the image after Instagram applied the filter. The histograms reveal two key differences:

1.) **Color scaling** - Instagram's EARLYBIRD filter scales down the Blue and Green channels. For example, the Blue channel in the original image peaks at an intensity ~210. However in the filtered image, the Blue channel now peaks at ~190. By decreasing the Blue and Green channels, EARLYBIRD applies a yellow tint to the images.

2.) **Brightness** - Instagram's EARLYBIRD filter adjusts the brightness of images. The histograms in Fig 1.1 show how the average intensity of each color has been increased. In the original image, we can notice that there is almost no blue at intensity ~150. However, in the filtered image we can see that the RGB colors have been increased "across the board". These changes in overall color intensity would pull the RGB color values towards brighter colors.

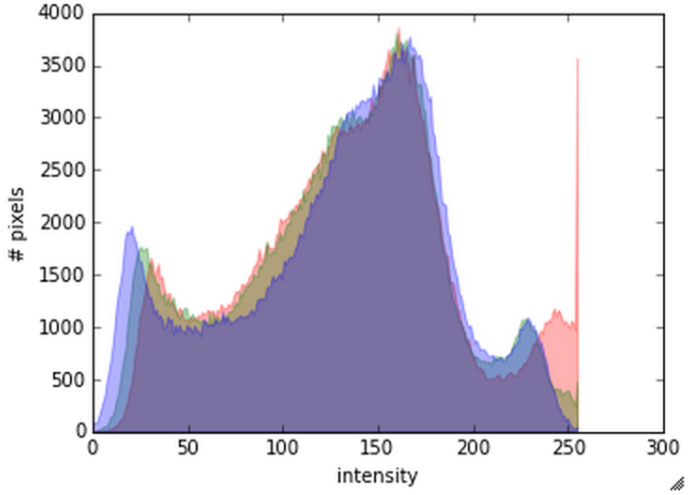
Fig 1.1 - Comparing the Histograms of the original image, and an image where the EARLYBIRD filter has been applied.

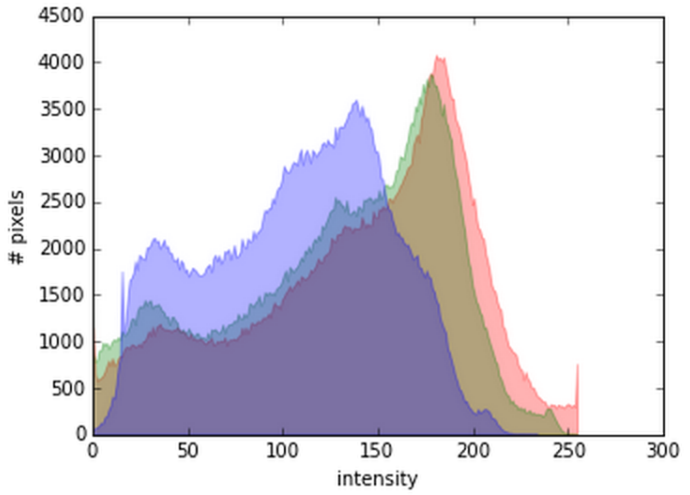




I ran tests on a few sample images. Fig 1.2 shows the histogram comparisons for another example image. Notice the differences between the original and the filtered image:

Fig 1.2 - Original vs. Instagram





Using python, and the skimage library, I was able to break down the filter into a number of steps. Fig 1.3 shows a number of functions utilized to recreate the filter.

Fig 1.3 - Sample Code

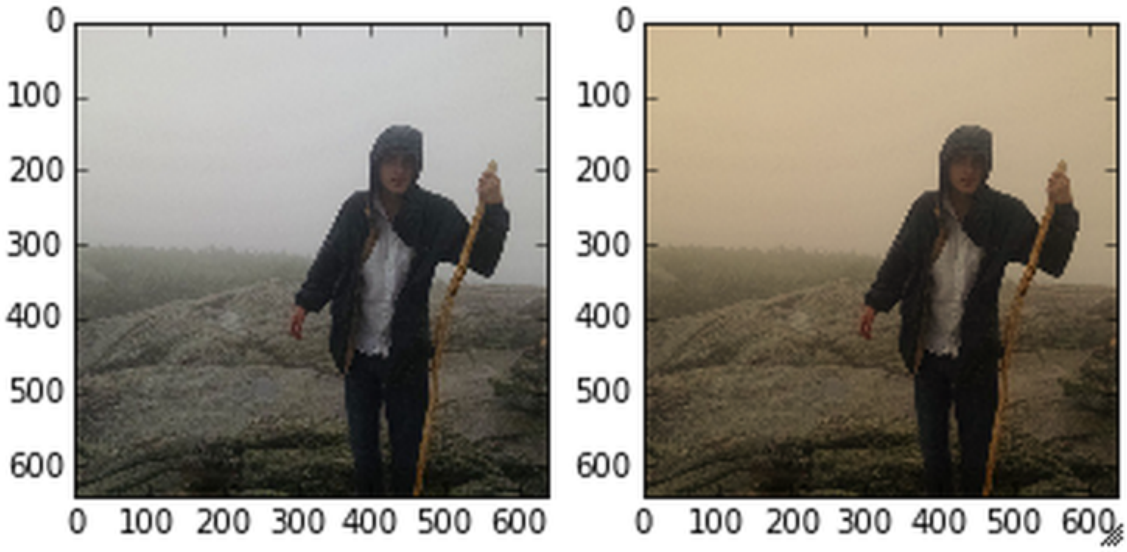
```
# Create Custom Filter to Mirror "EARLYBIRD".
# Applies multiple image effects:
# - Apply Yellow Tint
# - Increase Contrast
# - Decrease Gamma (+ Brightness)
# - Apply Vignette (Faded Border)

def filter(image):
    image = tint(image);           # Tint
    image = contrast(image, 6.8); # Contrast
    image = gamma(image, .6);      # Gamma
    image = vignette(image);       # Vignette
    return image
```

1.) Tint

The first step in constructing my custom filter was to adjust the color of the image. According to photodoto.com[1], EARLYBIRD tints the subject photo with the color #b39f77. I had difficulty matching the color scheme, but with some experimenting I was able to edit the RGB values of the image. Fig 1.4 shows an example of my tint function being applied to a sample image:

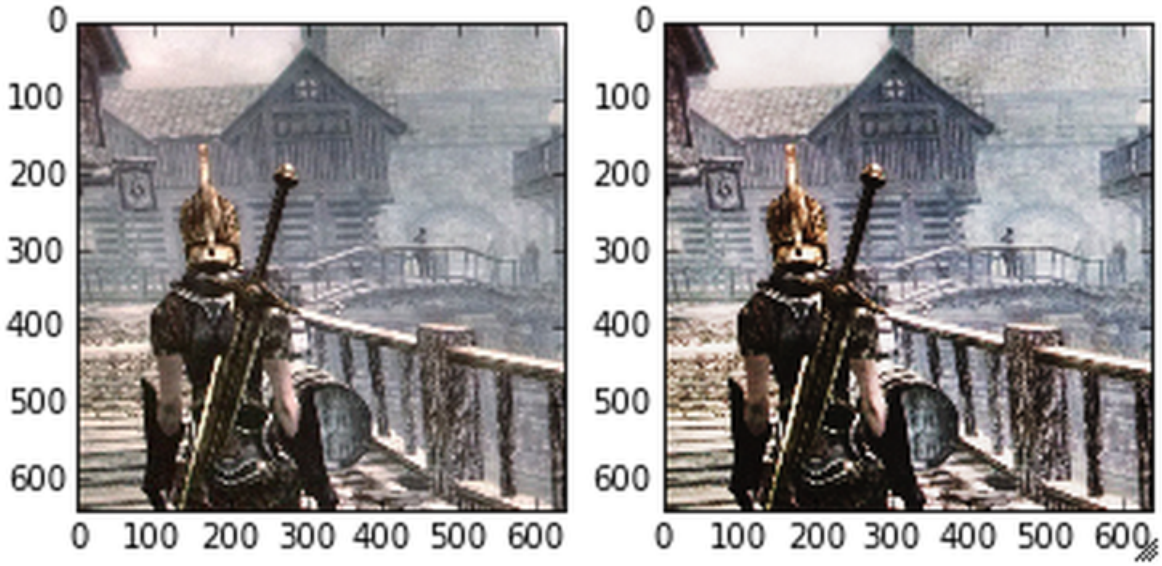
Fig 1.4 - Tint & Color Correcting



2.) Contrast

The second step to emulating the EARLYBIRD filter was to adjust the contrast. Instagram's EARLYBIRD filter makes the whites whiter and the darks darker. I was able to use skimage's exposure library[3] for their adjust_sigmoid function. This function helped to adjust an image's contrast. Fig 1.5 shows an image before and after I applied the contrast function.

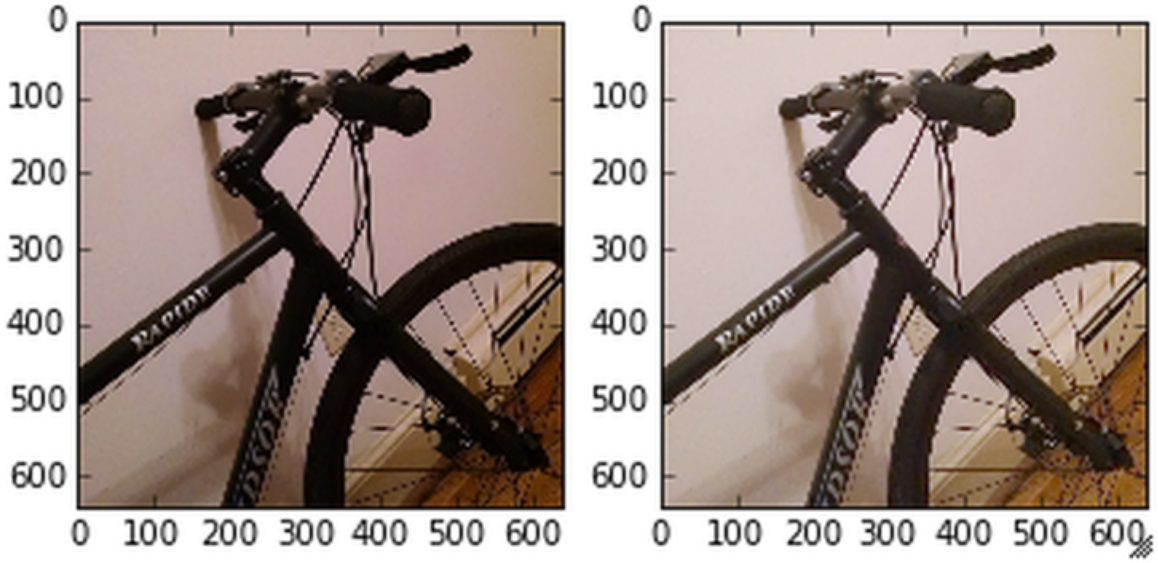
Fig 1.5 - Contrast Adjustment



3.) Gamma

It was also necessary to adjust the brightness of the image. In images that have been subject to the EARLYBIRD filter, you may notice that light areas of the photo are washed out. Once again, using skimage's exposure library[3] I was able to adjust the gamma of an image. As seen in Fig 1.6, the gamma adjustment increases the brightness, and washes out some of the lighter areas.

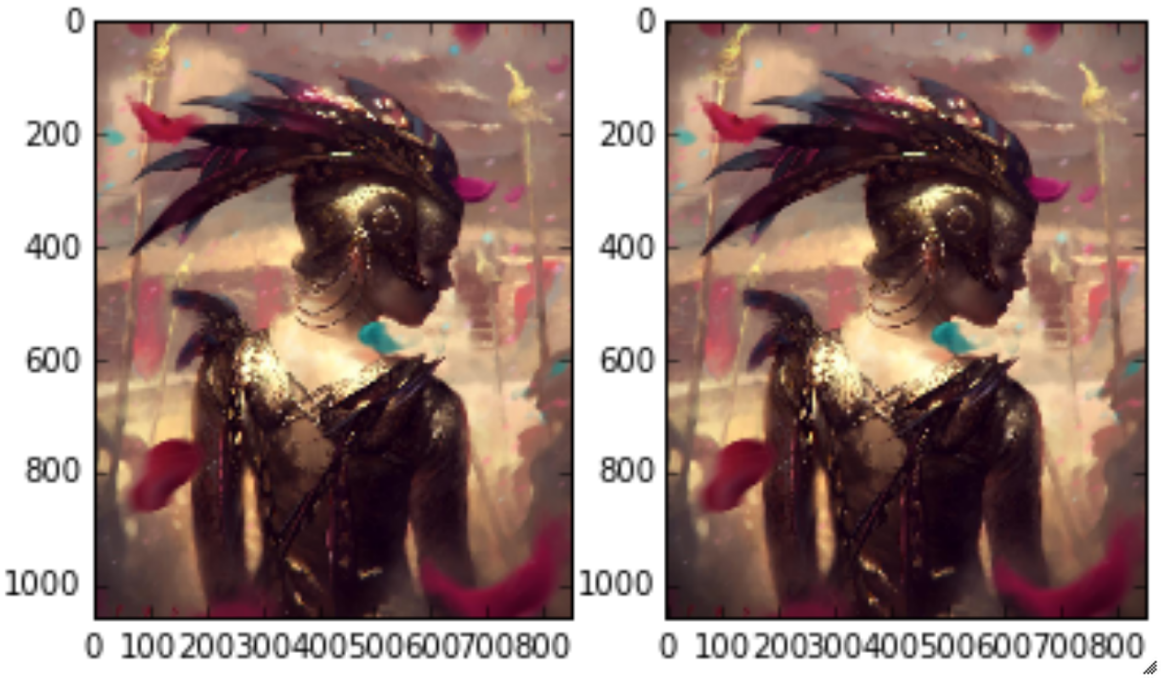
Fig 1.6 - Gamma Adjustment



4.) Vignette

The last effect I used was a slight vignette. I wrote a function that iterates through the pixels of an image and darkens them based on their proximity to the center pixels of the image. I added an offset, as to avoid manipulating the center of the image. Fig 1.7 shows my ever-so-slight vignette applying a slight faded darkness to the edges of the image.

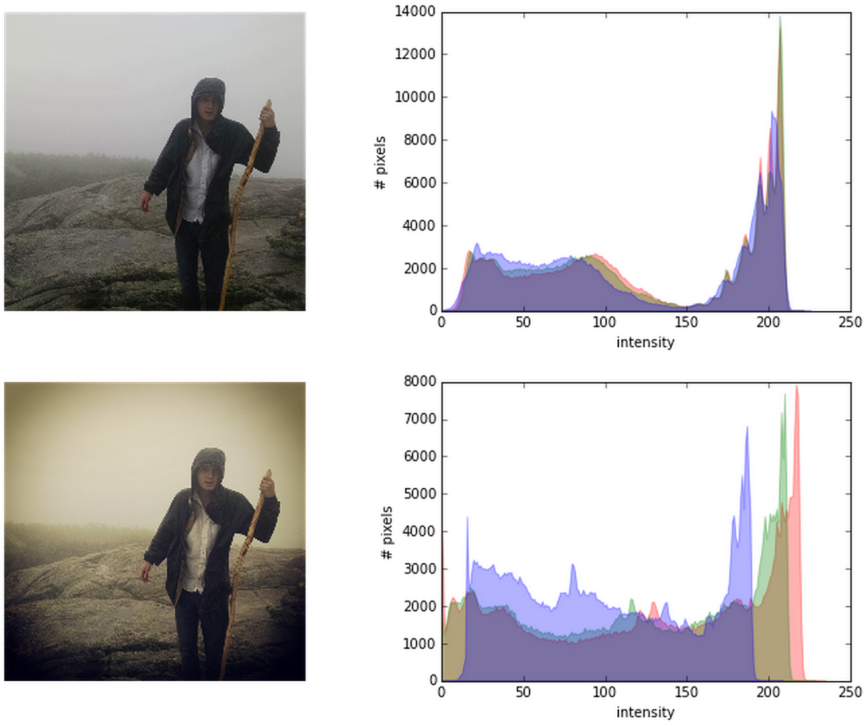
Fig 1.7 - Vignette Effect



Code

Please see the attached '*image-filter.ipynb*' for the source code. The source code contains each of the functions used to implement the filter. The code also contains more comments addressing implementation. I've also included a number of tests and functions for image analysis in an attempt to interpret the results. Two key functions I wrote are the *'tint'* and the *'vignette'* function. The *'tint'* function takes in the RGB image array, and returns an image with the yellowing hue. Fig 2.1 shows an example of how the Instagram filter scales and manipulates the color channels of the image.

Fig 2.1 - Histogram Analysis comparing the original image to the Instagram Filter




```
for x in range(0, xValues - 1):
    for y in range(0, yValues - 1):

        # Scale each color channel.
        # Reduce Blue, while keeping
        # Red and Green (yellow) close
        # to their original intensities
        acc[x][y][0] = acc[x][y][0] * 1
        acc[x][y][1] = acc[x][y][1] * .9
        acc[x][y][2] = acc[x][y][2] * 0.75
```

Fig 2.2 - Code Snipped from *'Tint'* Function.

Fig 2.2 shows how the custom filter attempts to replicate the Instagram filter by scaling the intensities of Red, Green, and Blue channels by different amounts. My implementation takes influence from Professor Daniel R. Schlegel's explanation of how to scale color channels in an image [2].

The second function I wish to highlight is the *'vignette'* function. The *'vignette'* function takes an image array, and darkens pixels that are farther away from the center of the image. After applying an

offset from the center, this function adds a faded border to the outsides of the image. For more details regarding code and filter implementation, please see the attached file '*image-filter.ipynb*'.

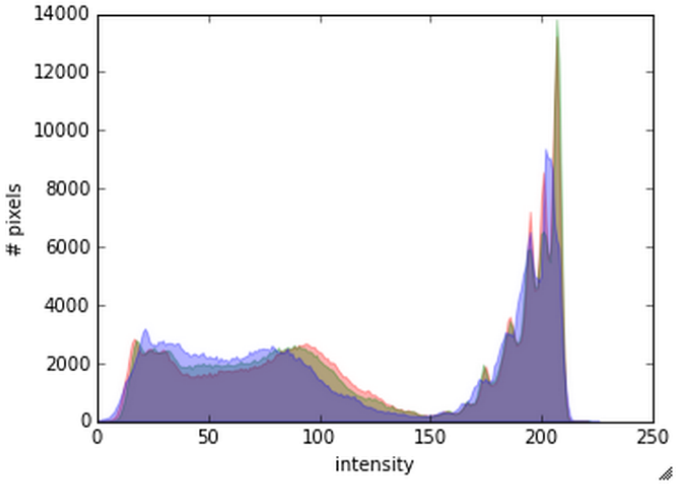
Results

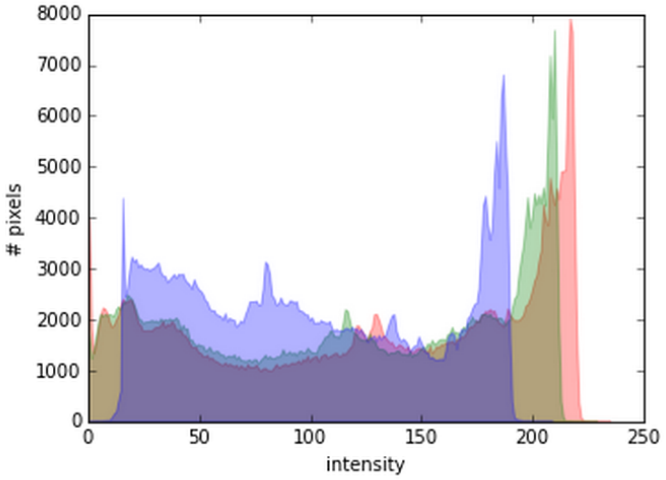
Figures 3.1, 3.2 and 3.3 are the results for three different sample images. Each image set contains the outputs for:

a.) The original image
b.) The image after Instagram's 'EARLYBIRD' filter was applied
c.) The image after the custom algorithm was applied.

Fig 3.1 - Showing a.) Original, b.) Instagram Filter, and c.) Custom Filter

The Original Image, Instagram Filter, Custom Filter





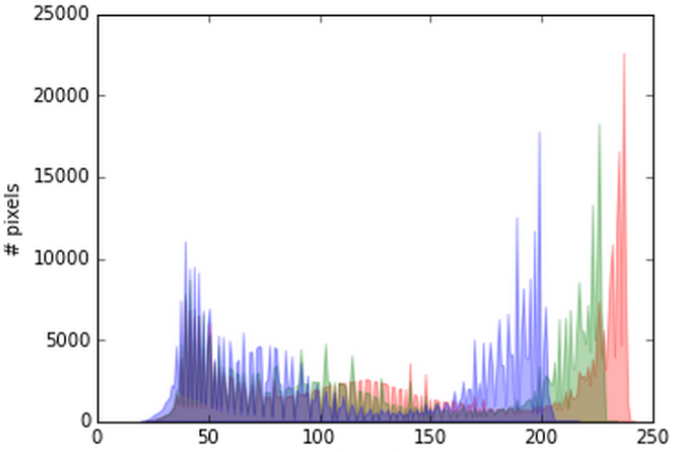


Fig 3.2 - Showing a.) Original, b.) Instagram Filter, and c.) Custom Filter

The Original Image vs. The Instagram Filtered Image

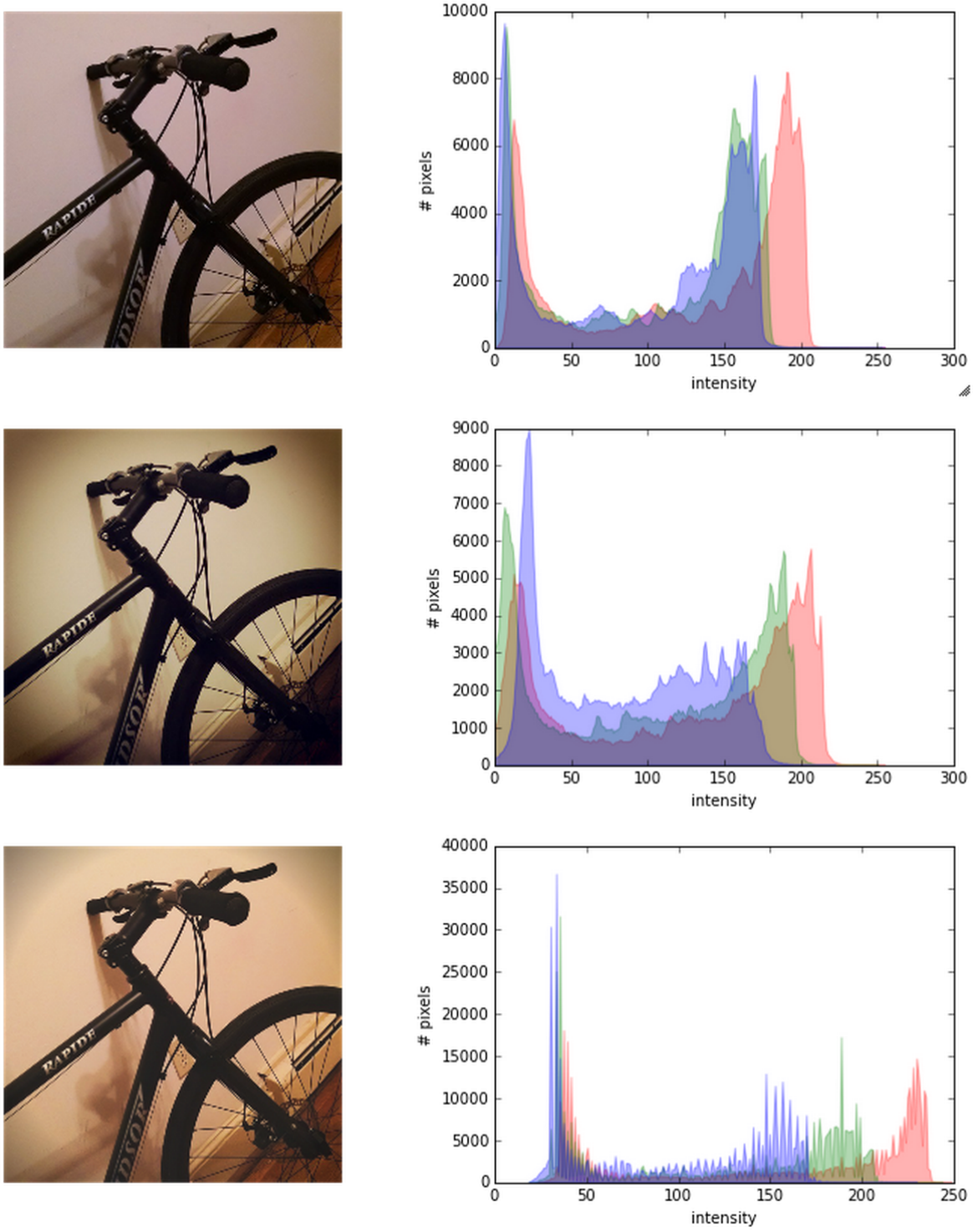
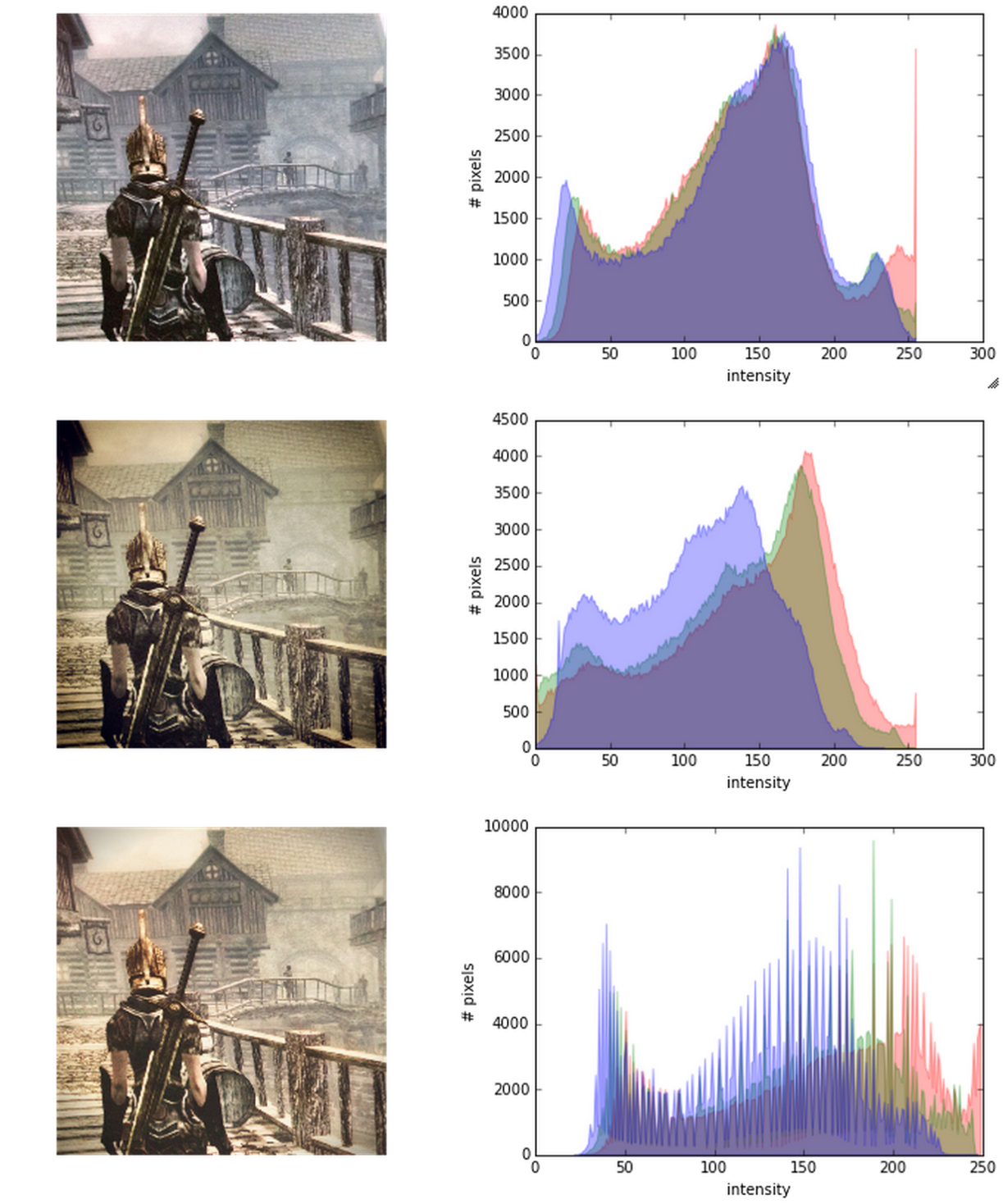


Fig 3.3 - Showing a.) Original, b.) Instagram Filter, and c.) Custom Filter



Discussion

I encountered difficulty developing a filter that was identical to Instagram's EARLYBIRD filter. Upon closer analysis, it was surprising how many subtle modifications each Instagram filter made to the original image. While I was able to isolate a number of steps to replicate EARLYBIRD, such as tint, contrast, gamma, and vignette, I struggled to manipulate the colors and vignette such that the results would be identical to Instagram's. Looking at Fig 3.1, you can see how the histogram of the custom filter matches the trends of the histogram from Instagram's filter. The histogram for our custom filter appears much more 'jagged'. Images such as the ones in Fig 3.3 show how the custom filter was similar to, but not perfect in replicating Instagram's filter.

In an attempt to understand how my results differed in comparison to the Instagram filter, I ran some basic image analysis. One of the most revealing graphs plotted the grey values in the image. Fig 4.1 shows a comparison of the distribution of grey values in the results of the custom filter, in comparison to the distribution of grey values in the results from Instagram.

Fig 4.1 - Comparing Grey Values: Custom Filter vs. Instagram's Filter

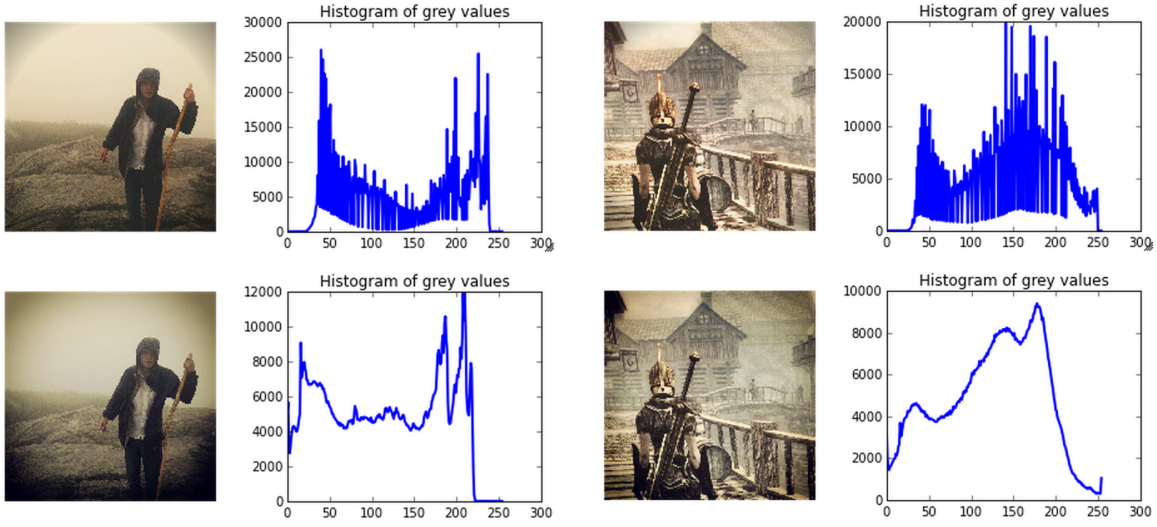
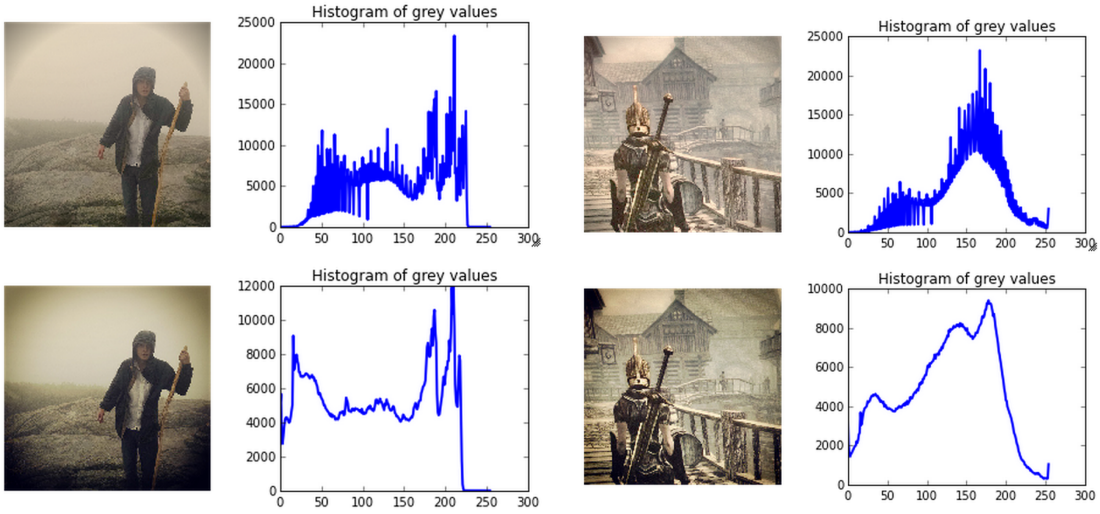


Fig 4.1 shows how the changes between neighboring grey values was much more dramatic the equivalent output from Instagram. I hypothesized that some of this distortion was coming from my implementation of a contrast filter. Fig 3.2 shows the same results, WITHOUT applying a change in contrast. Notice how the differences between certain grey values are less drastic.

Fig 4.2 - Comparing Grey Values: Custom Filter vs. Instagram's Filter - Without Contrast Adjustment



In conclusion, Instagram filters are more complicated than what meets the eye. In an attempt to replicate the Instagram filter EARLYBIRD, I deconstructed Instagram's EARLYBIRD filter, analyzed their output data, and worked to develop my own custom algorithm for implementing a similar filter. While my results were similar to those of Instagram's filter, the output files showed a few key differences and a hidden complexity that would require more than a rudimentary understanding of Python and more than a few dozen hours to implement.

References

[1] Mayne, Melanie. "How to Make Instagram Filters in Photoshop: Earlybird." Photodoto.com. N.p., 7 Jan. 2013. Web. <http://photodoto.com/how-to-make-instagram-filters-photoshop-earlybird/>.

[2]Schlegel, Daniel R. "Image Manipulation". http://danielschlegel.org. Web. <http://danielschlegel.org/pages/teaching/111/lecture6.html>.

[3] Module: exposure. Web. <http://scikit-image.org/docs/dev/api/skimage.exposure.html>.

[4] Gaille, Brandon. "10 Most Popular Instagram Photo Filters". brandongaille.com. 23 Jun. 2013. Web. <http://brandongaille.com/10-most-popular-instagram-photo-filters>.