

Professor_Bear_Image_Analysis_Sepia

March 14, 2017

1 Professor Bear :: Image Analysis :: Sepia Filter

1.1 Professor Bear github

Code for Professor Bear YouTube videos at <https://github.com/nikbearbrown>

1.2 Download Anaconda 4 for Python 2.7

Download Anaconda 4 for Python 2.7 version <https://www.continuum.io/downloads>

Anaconda 4.3.0 includes an easy installation of Python (2.7.13, 3.4.5, 3.5.2, and/or 3.6.0) and updates of over 100 pre-built and tested scientific and analytic Python packages. These packages include NumPy, Pandas, SciPy, Matplotlib, and Jupyter. Over 620 more packages are available.
<https://docs.continuum.io/anaconda/pkg-docs>

1.3 iPython

Go to the directory that has your iPython notebook

At the command line type

jupyter notebook notebookname

ipython notebook notebookname will also work

For example,

jupyter notebook Professor_Bear_Image_Analysis_Loading_Histograms.ipynb

```
In [1]: # Bring in python image analysis libraries
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np
from skimage import color
import skimage.filters as filters
from skimage.transform import hough_circle
from skimage.feature import peak_local_max
from skimage import feature
from skimage import morphology
from skimage.draw import circle_perimeter
from skimage import img_as_float, img_as_ubyte
from skimage import segmentation as seg
from skimage.morphology import watershed
```

```

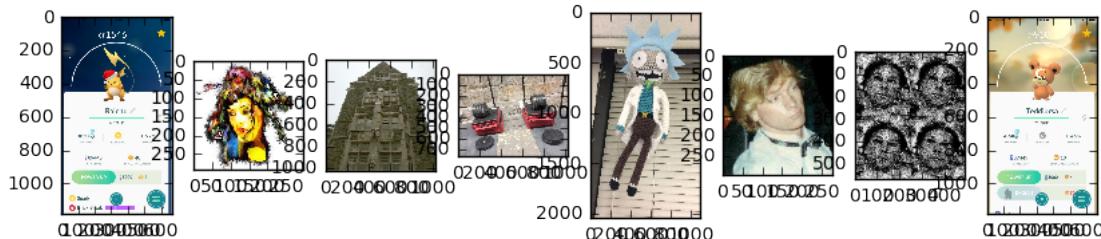
from scipy import ndimage as nd
from scipy.ndimage import convolve
from skimage import feature
import glob # for bulk file import

# Set defaults
plt.rcParams['image.cmap'] = 'gray' # Display grayscale images in...
plt.rcParams['image.interpolation'] = 'none' # Use nearest-neighbour
plt.rcParams['figure.figsize'] = 10, 10

# Import test images
imgpaths = glob.glob("./img/*.jpg") + glob.glob("./img/*.png")
# imgpaths = glob.glob("img/*.jpg") + glob.glob("img/*.png") Windows
# Windows has different relative paths than Mac/Unix
imgset = [mpimg.imread(x) for x in imgpaths]

# Display thumbnails of the images to ensure loading
plt.figure()
for i,img in enumerate(imgset):
    plt.subplot(1, len(imgset), i+1)
    plt.imshow(img, cmap = 'gray')

```



1.4 Sepia

Sepia

```

In [2]: #create a kernel which will be used to the original image.
sepia_filter = np.array([[.393, .769, .189],
                        [.349, .686, .168],
                        [.272, .534, .131]])

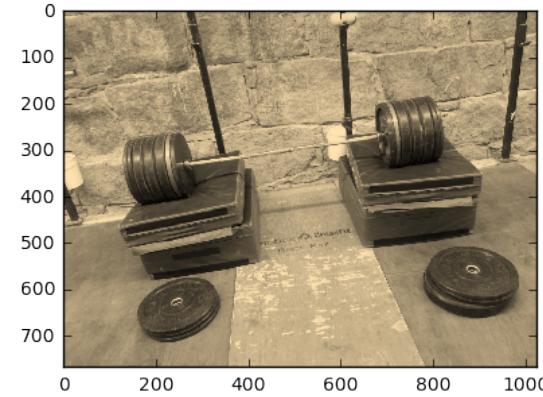
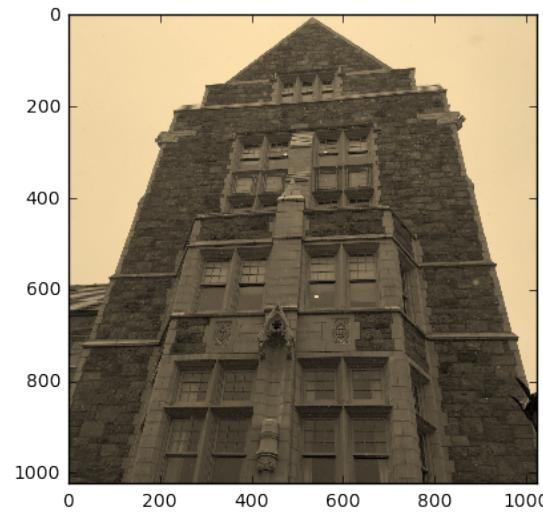
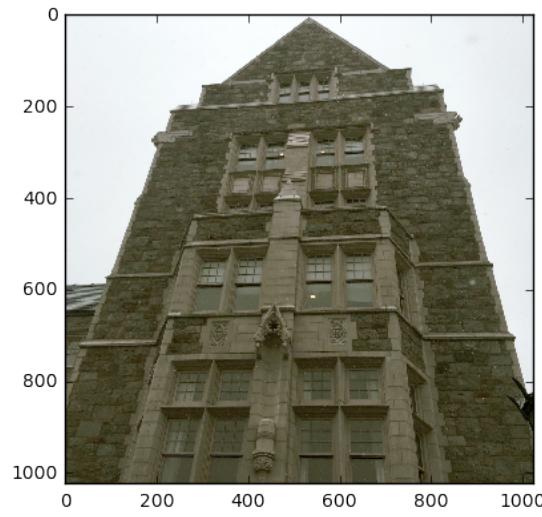
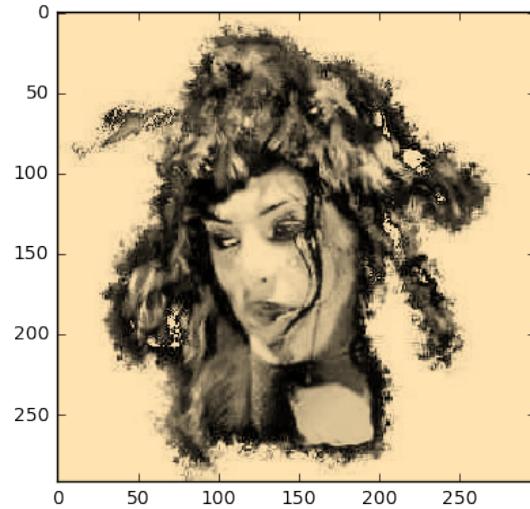
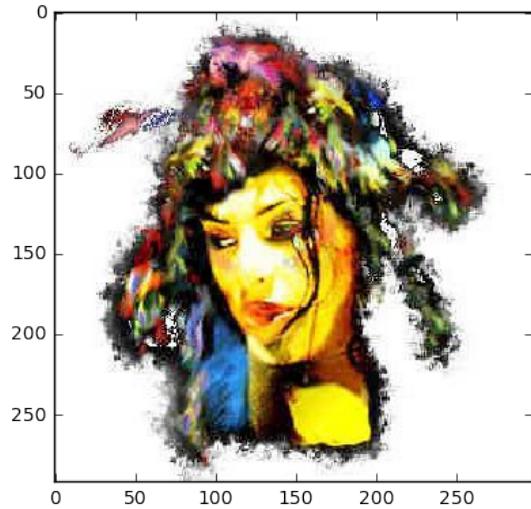
def sepia(image, filter):
    sepia_img = image.dot(filter.T) #Dot product the image and array
    #T is the transpose of the array.
    #Since our filter lines do not have unit sum, so we need to rescale
    sepia_img /= sepia_img.max()
    return sepia_img

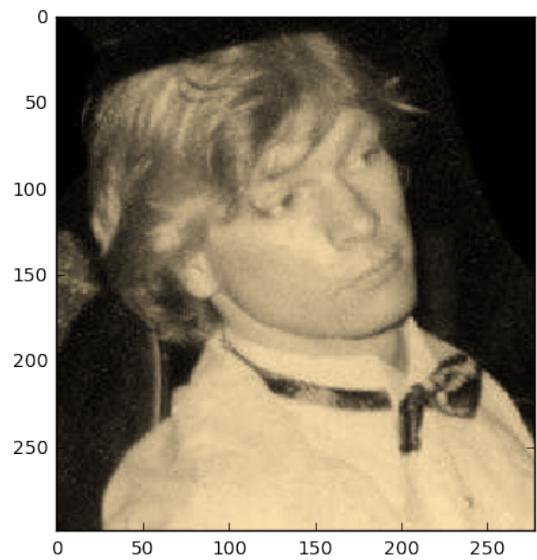
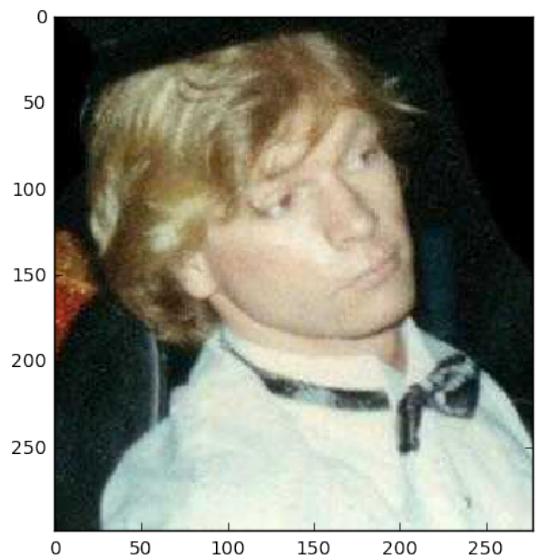
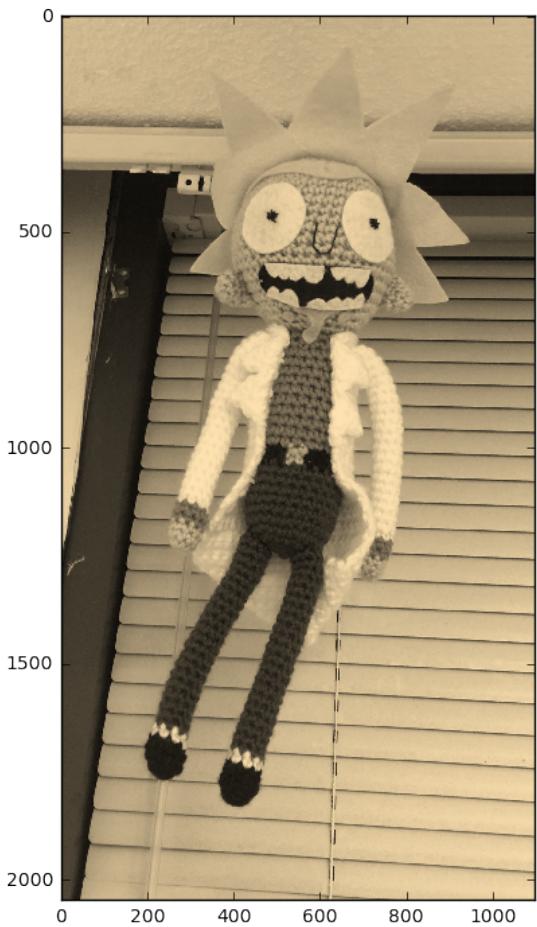
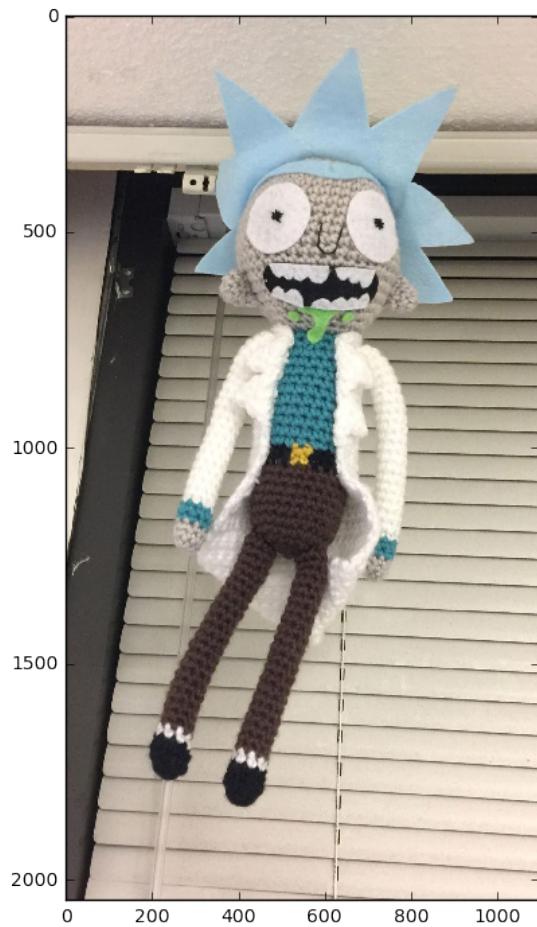
```

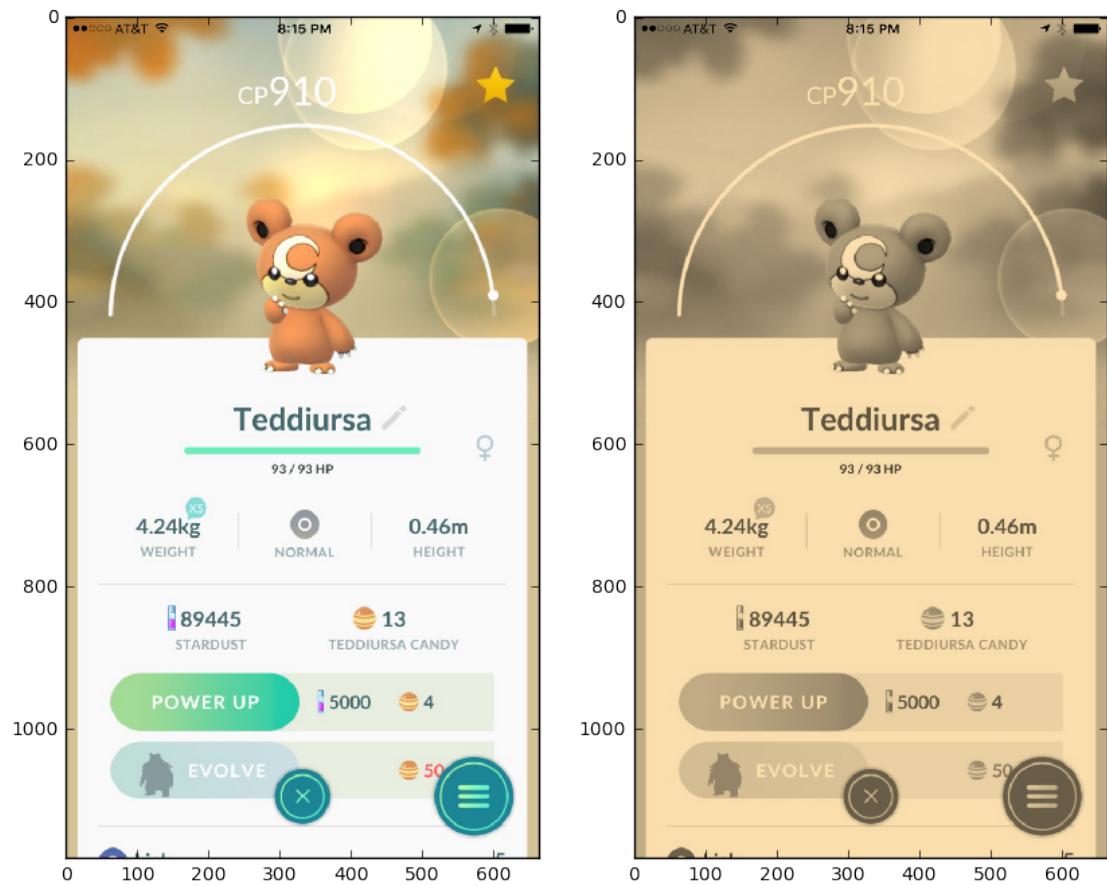
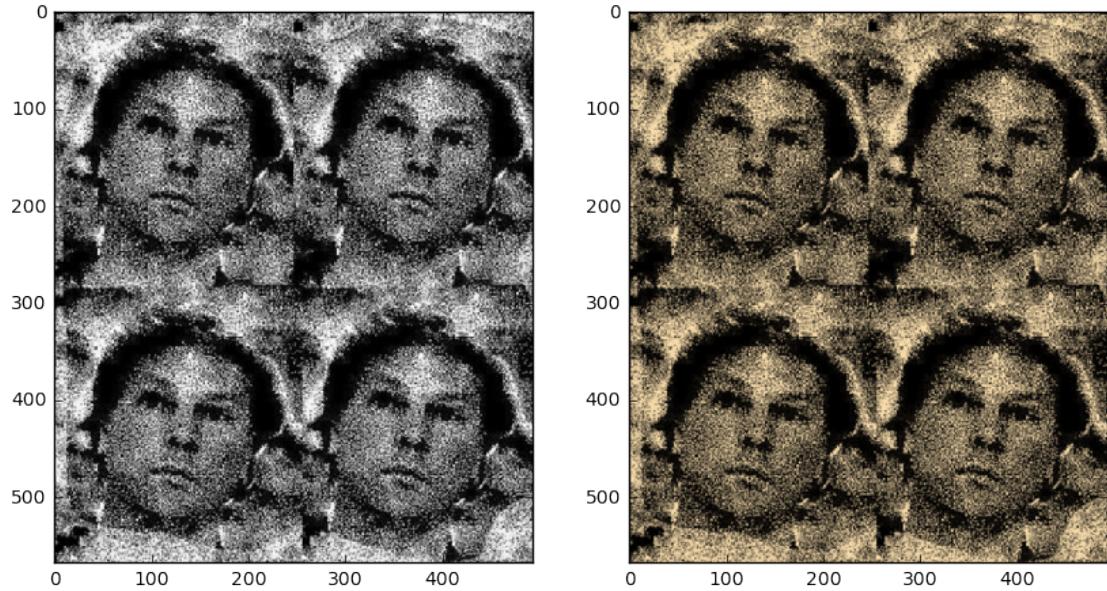
```
In [3]: # Apply to image set
```

```
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,sepia_filter))
```







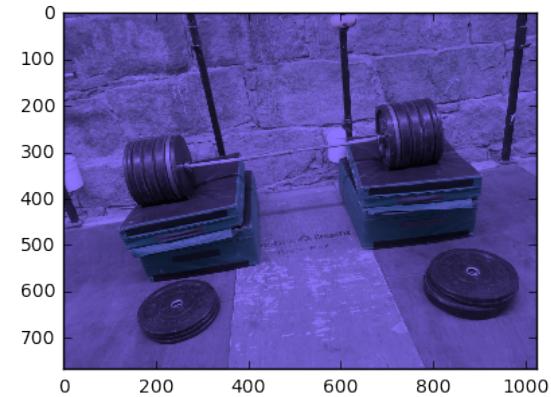
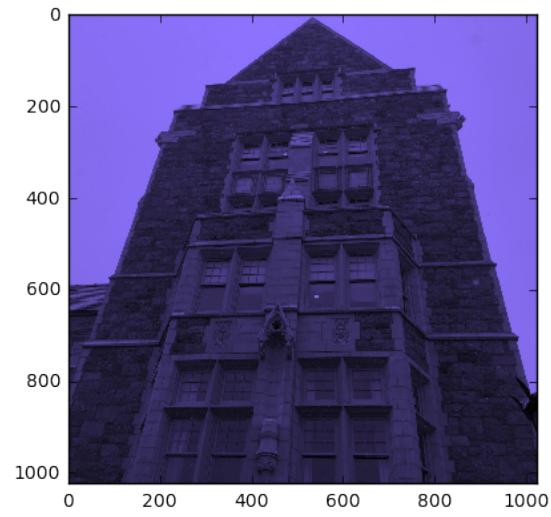
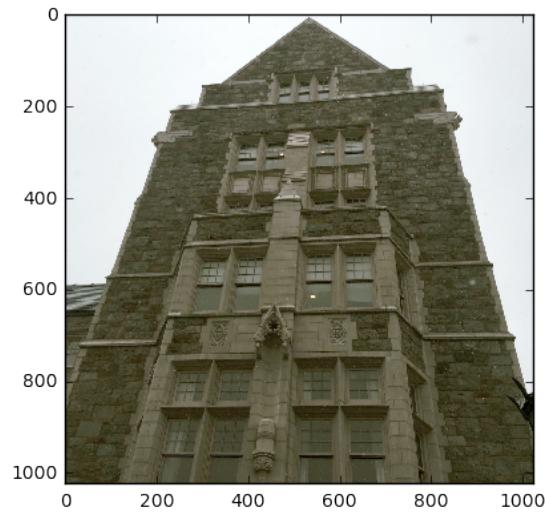
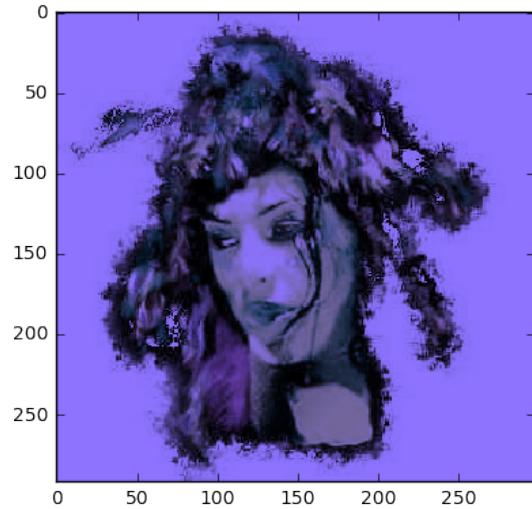
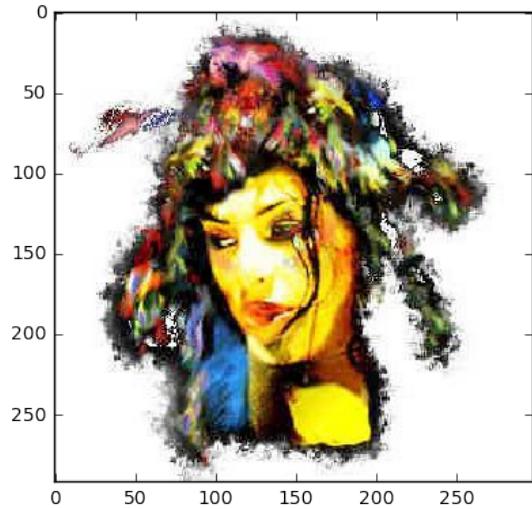


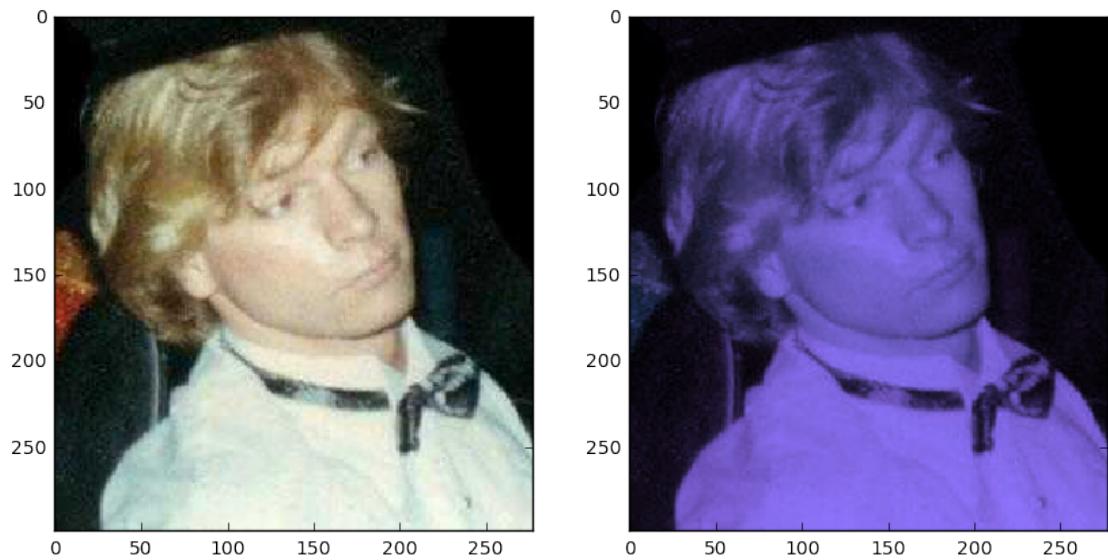
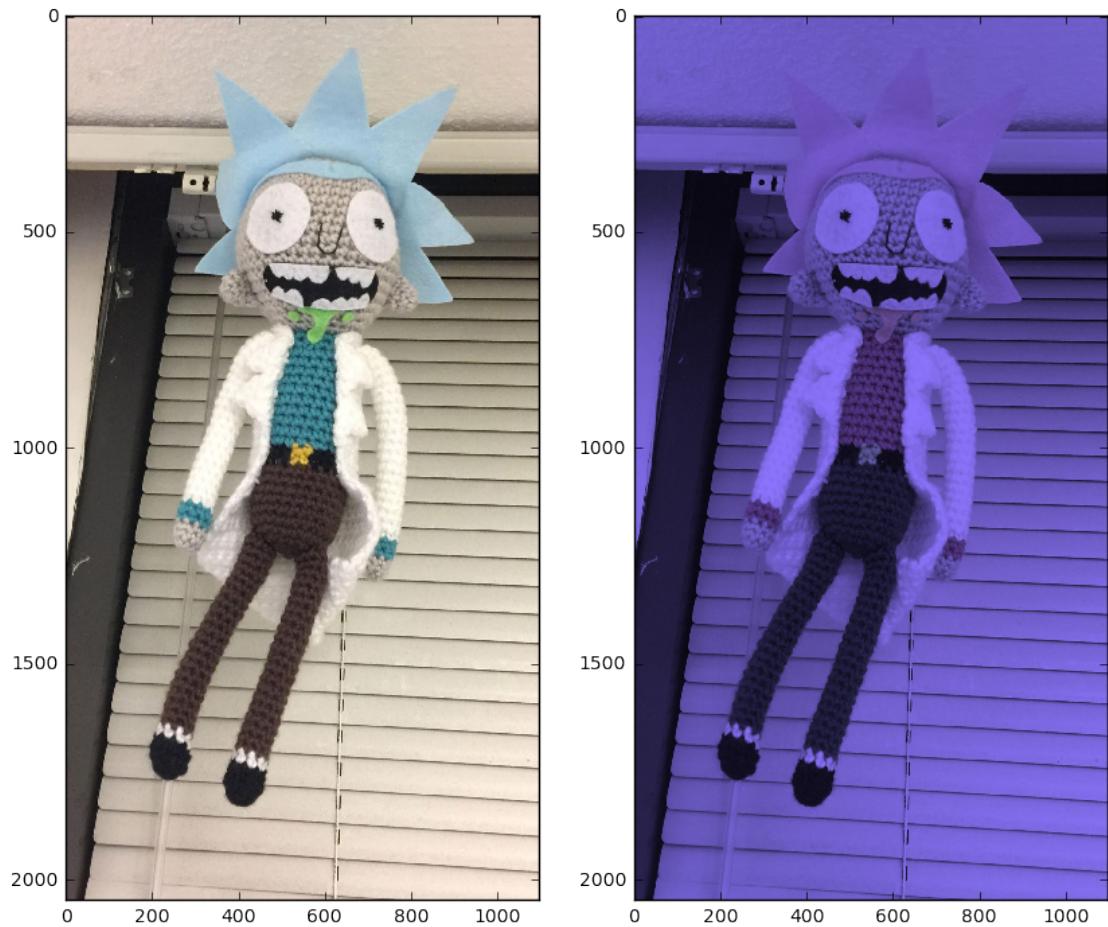
```
In [4]: purple_filter = np.array([[.03, .769, .189],
                                 [.349, .36, .1],
                                 [.55, .534, .7]])
```

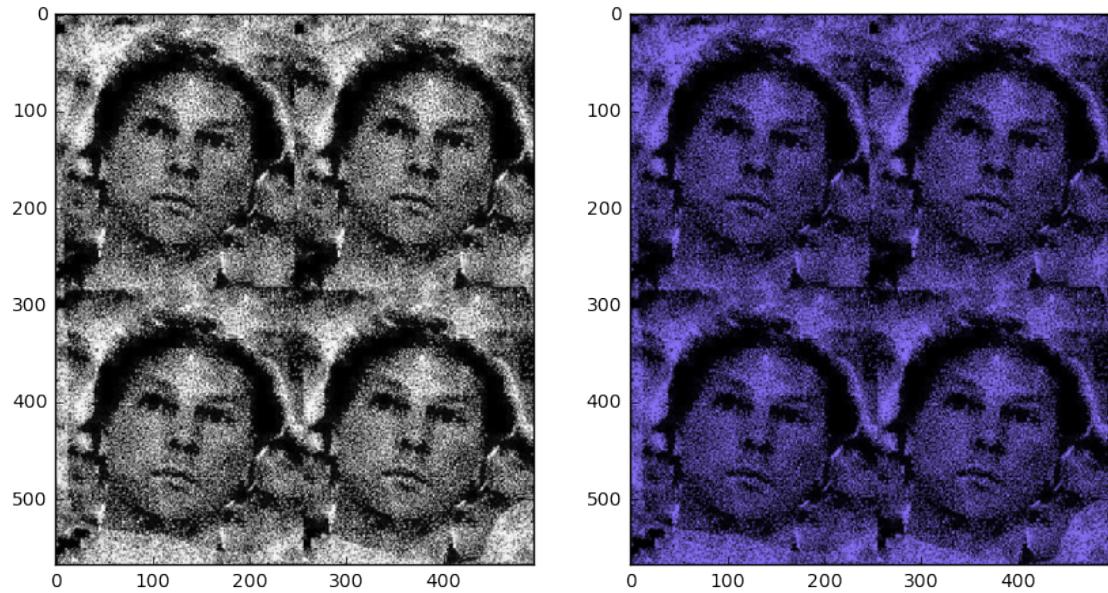
Apply to image set

```
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,purple_filter))
```





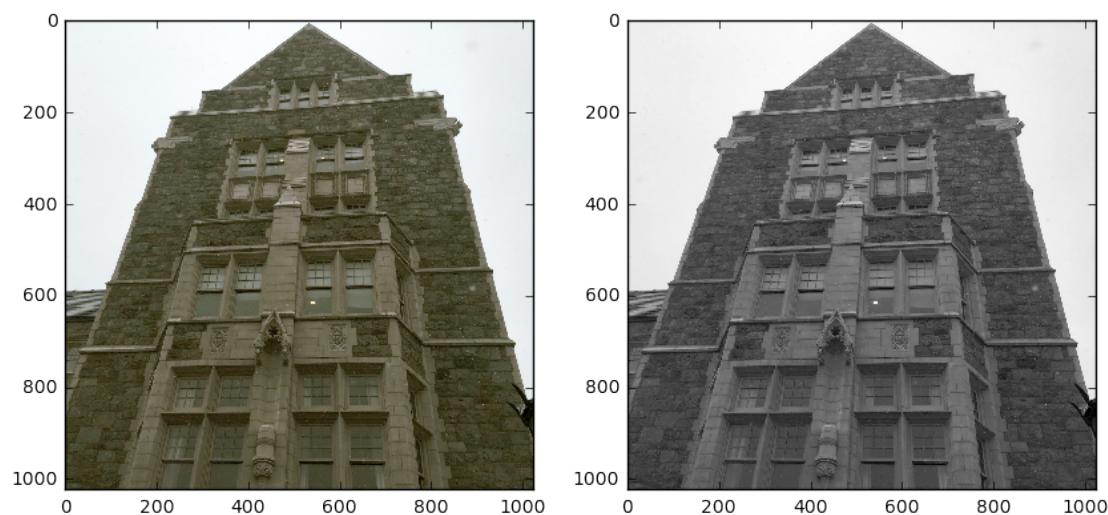
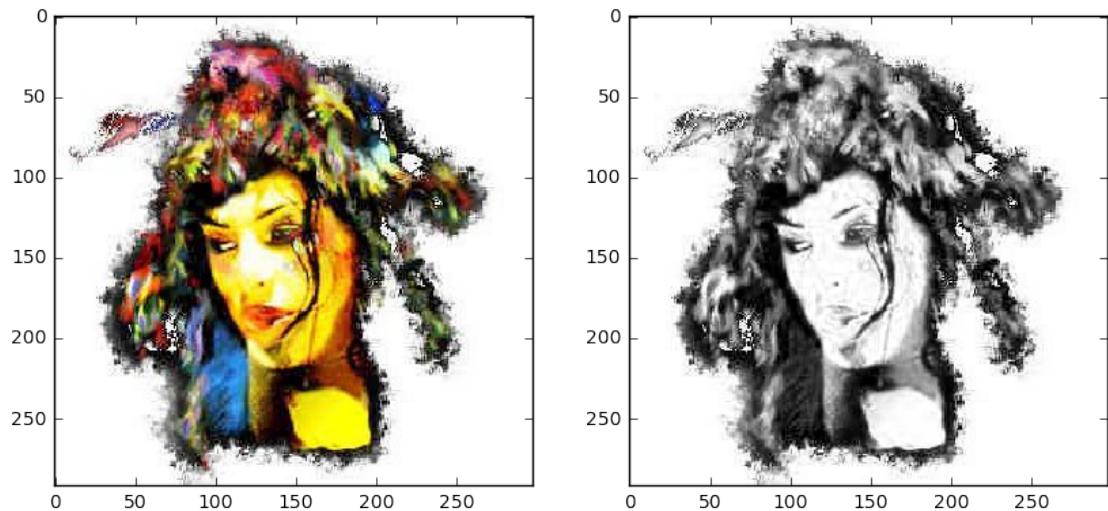


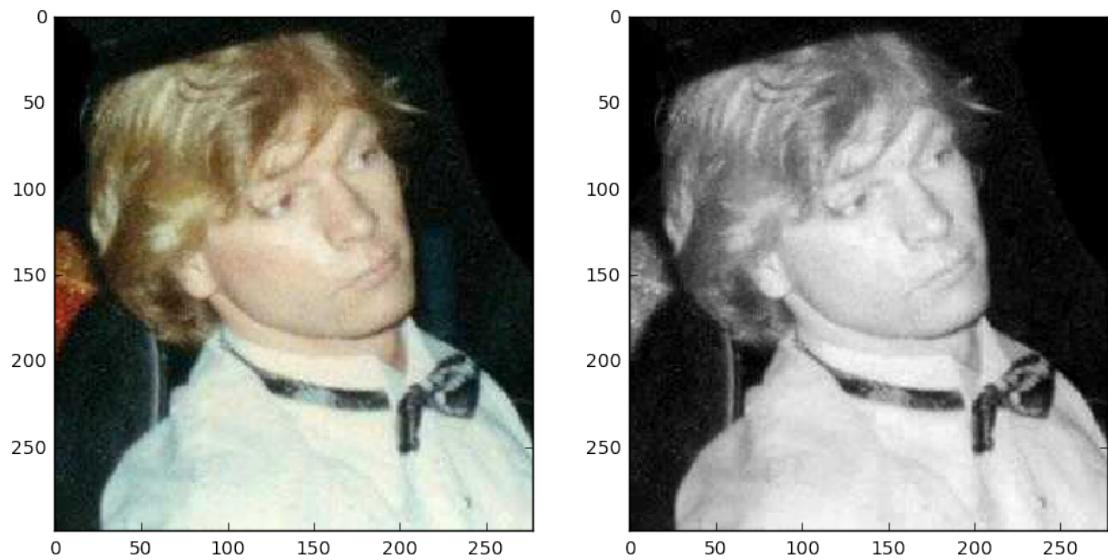
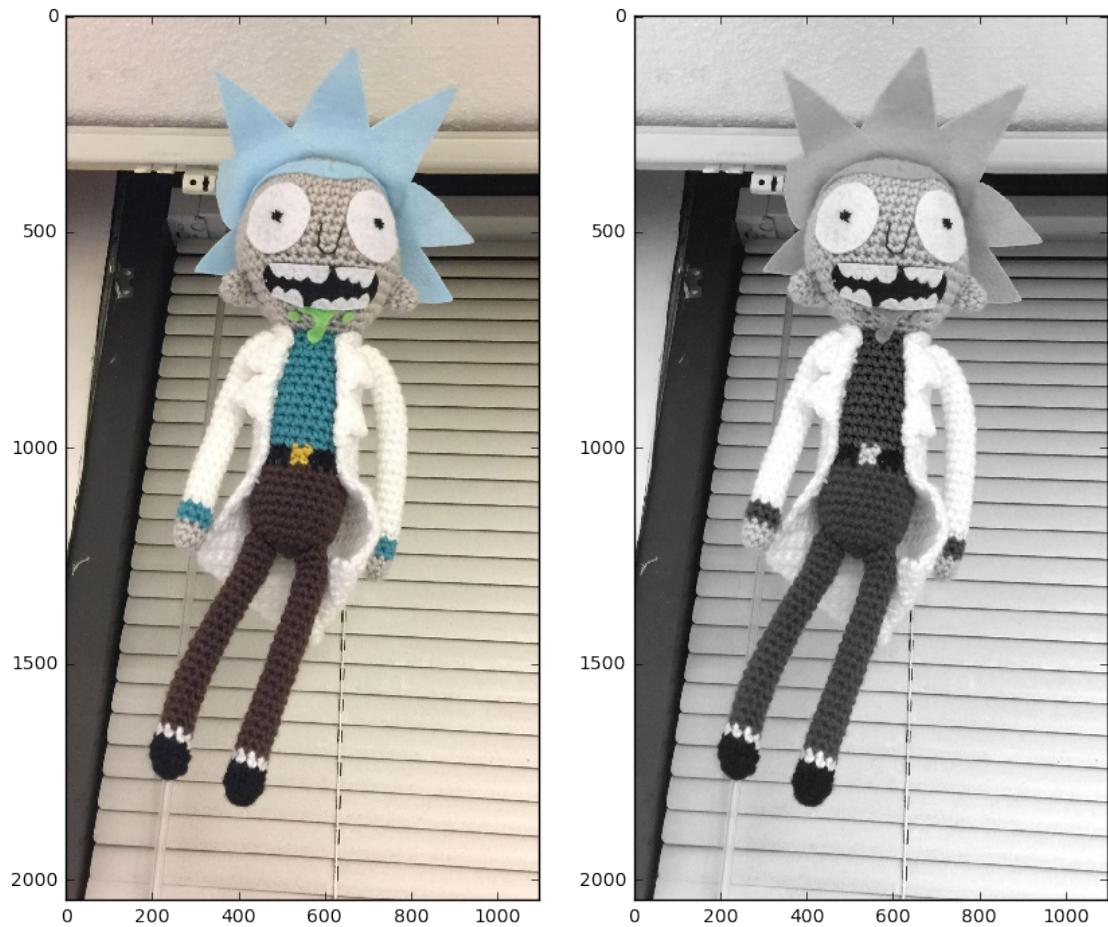


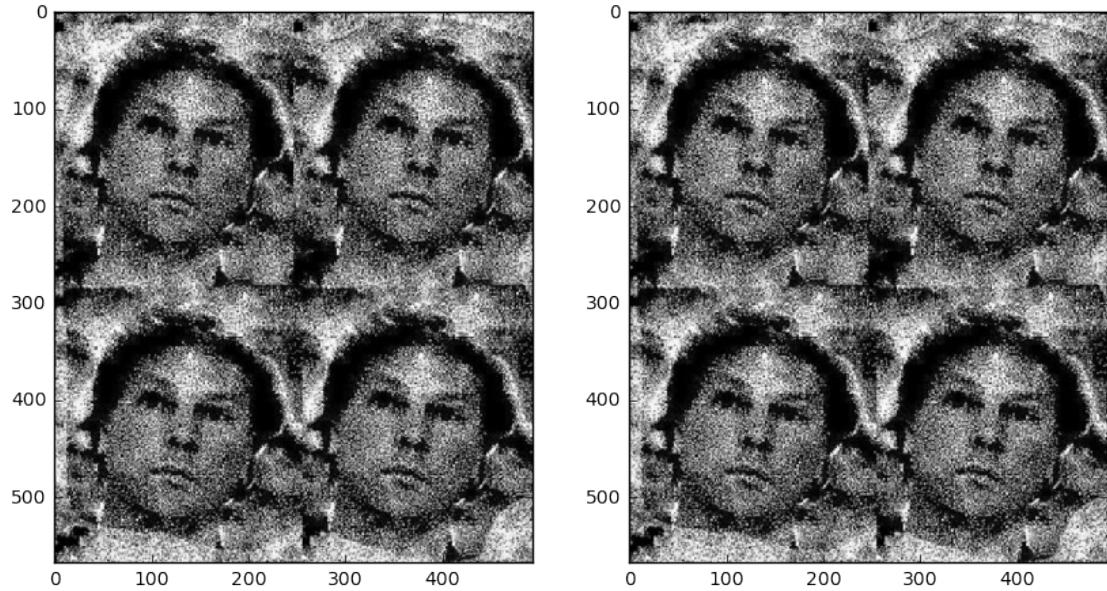
```
In [5]: filter_a = np.array([[1.0, 0.0, 0.0],
                           [1.0, 0.0, 0.0],
                           [1.0, 0.0, 0.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_a))
```





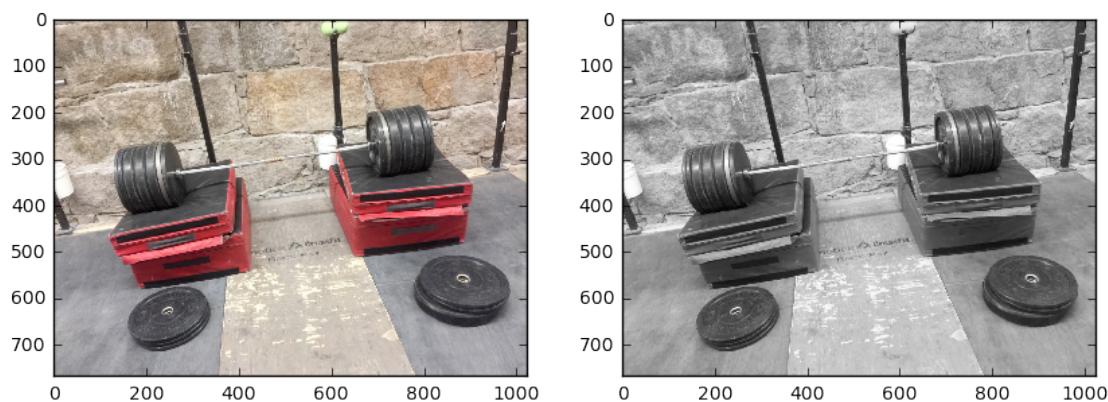
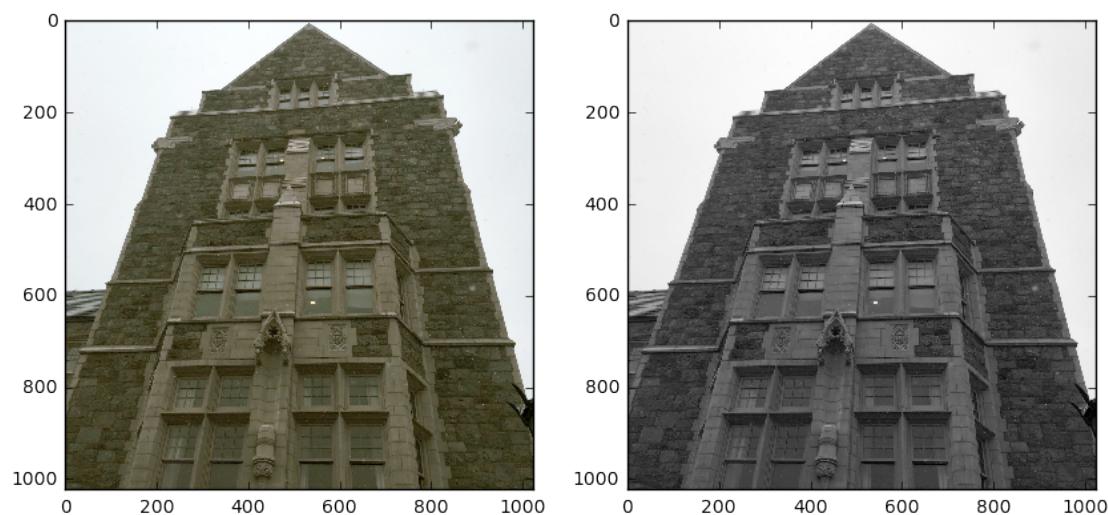
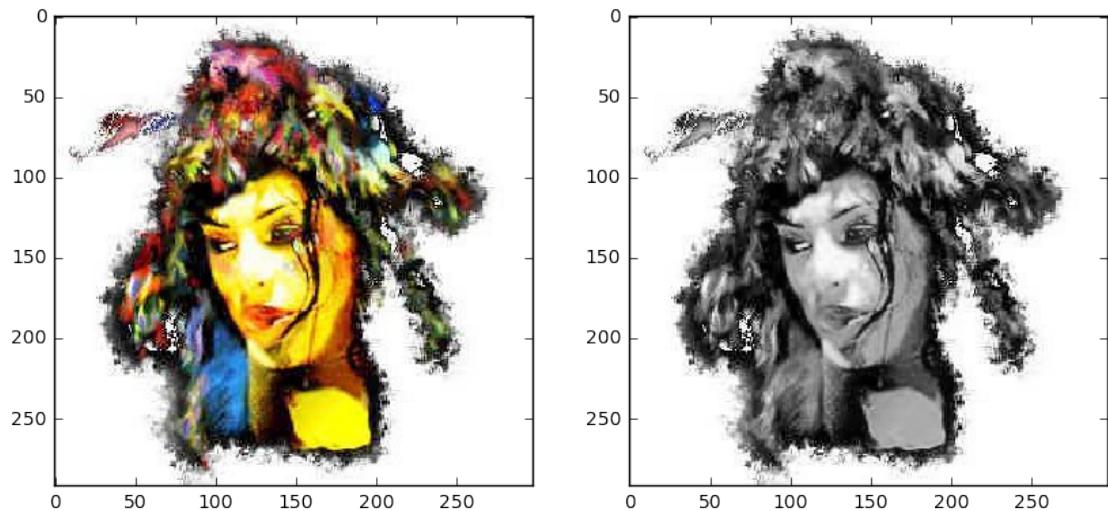


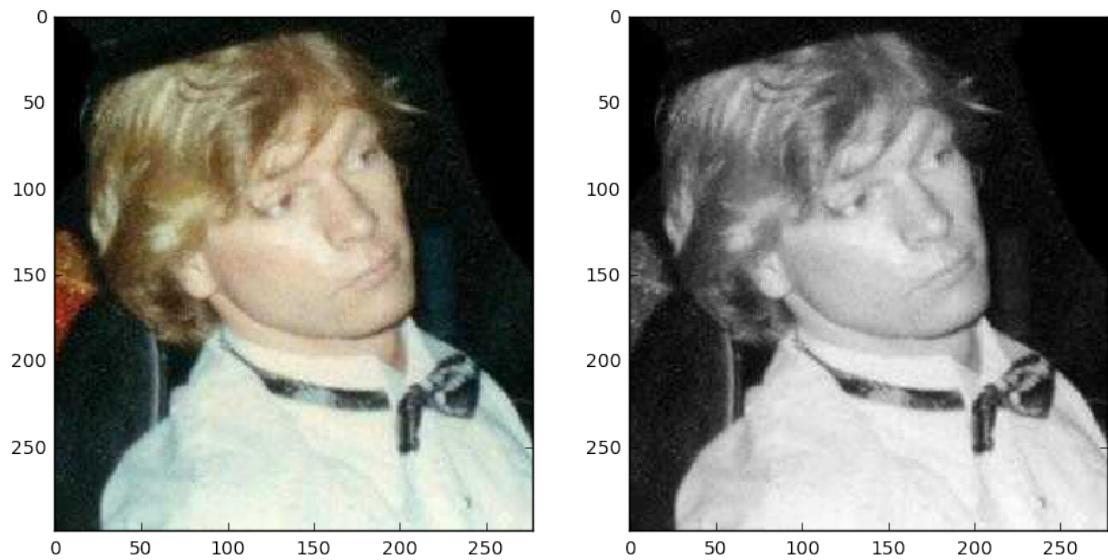
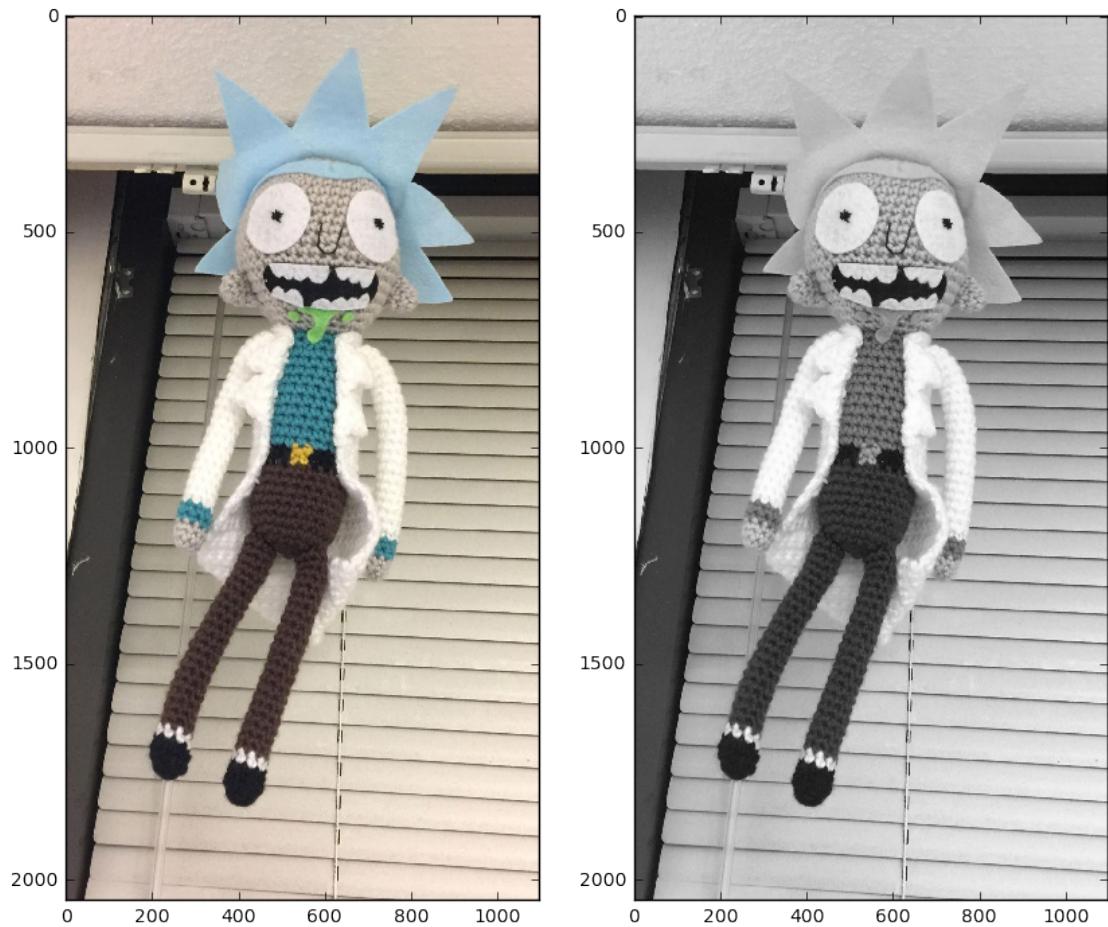


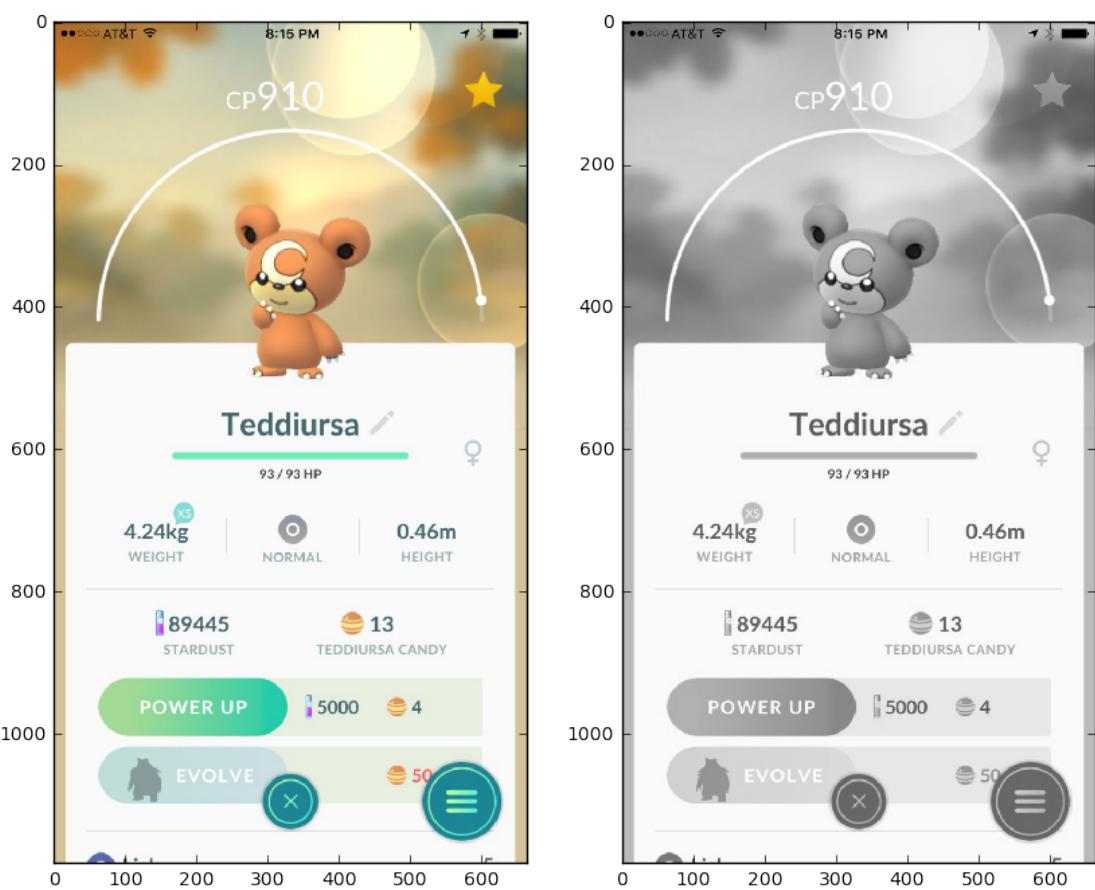
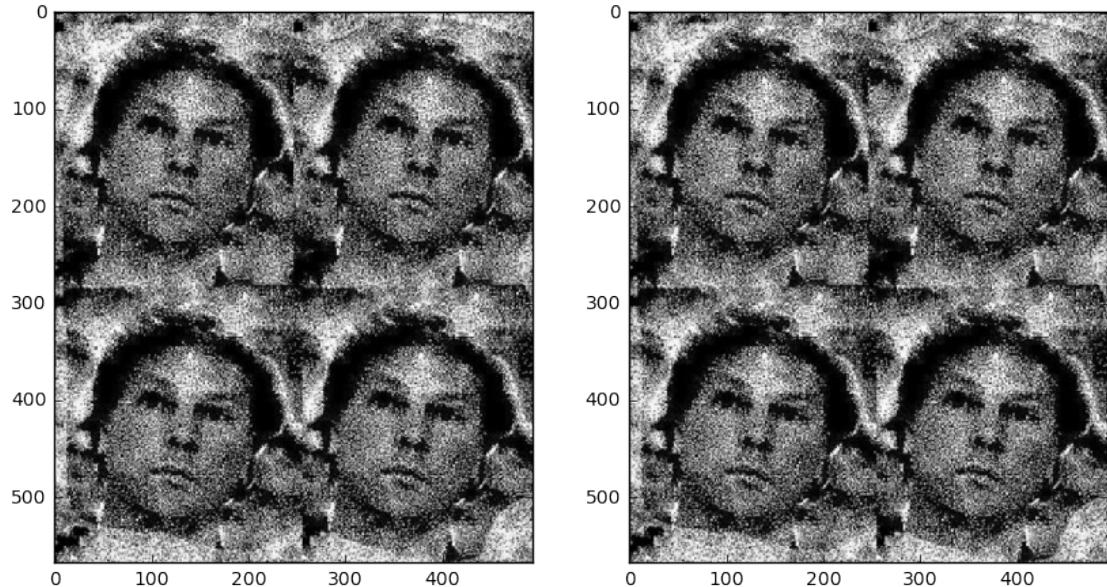
```
In [6]: filter_all = np.array([[1.0, 1.0, 1.0],
                             [1.0, 1.0, 1.0],
                             [1.0, 1.0, 1.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_all))
```



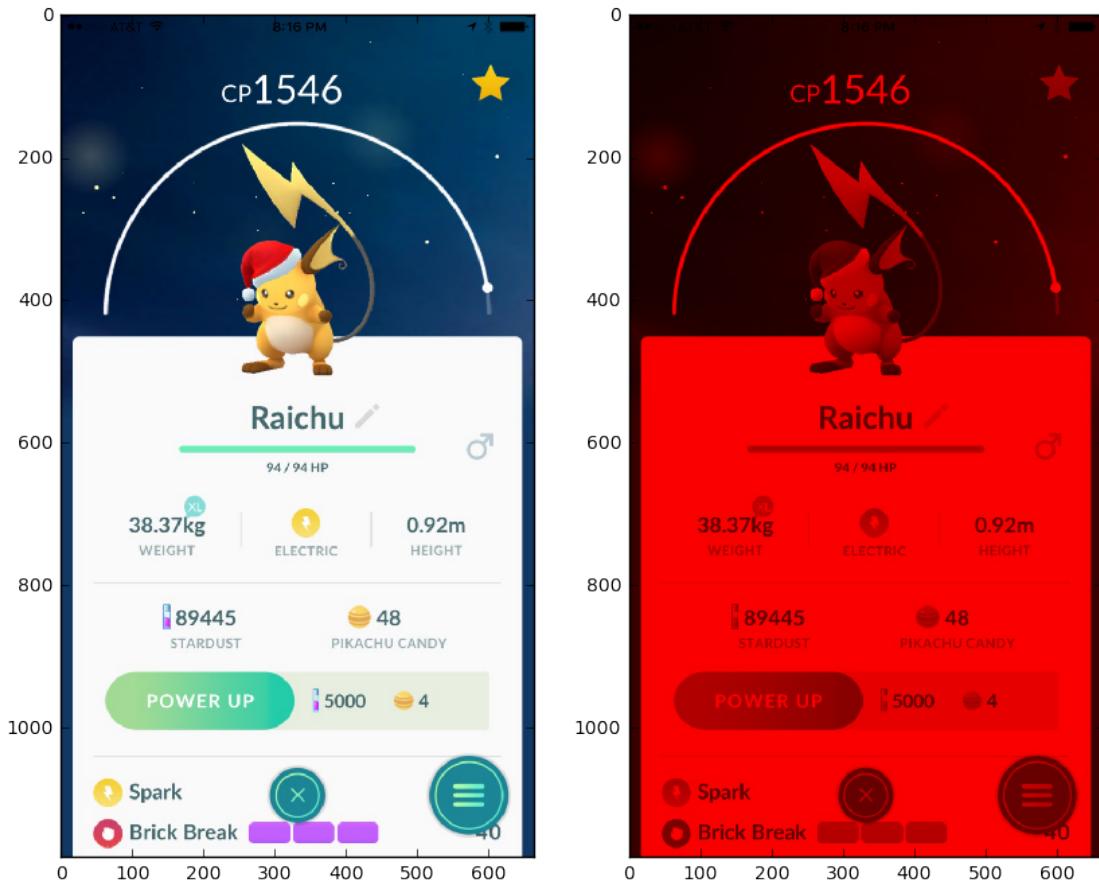


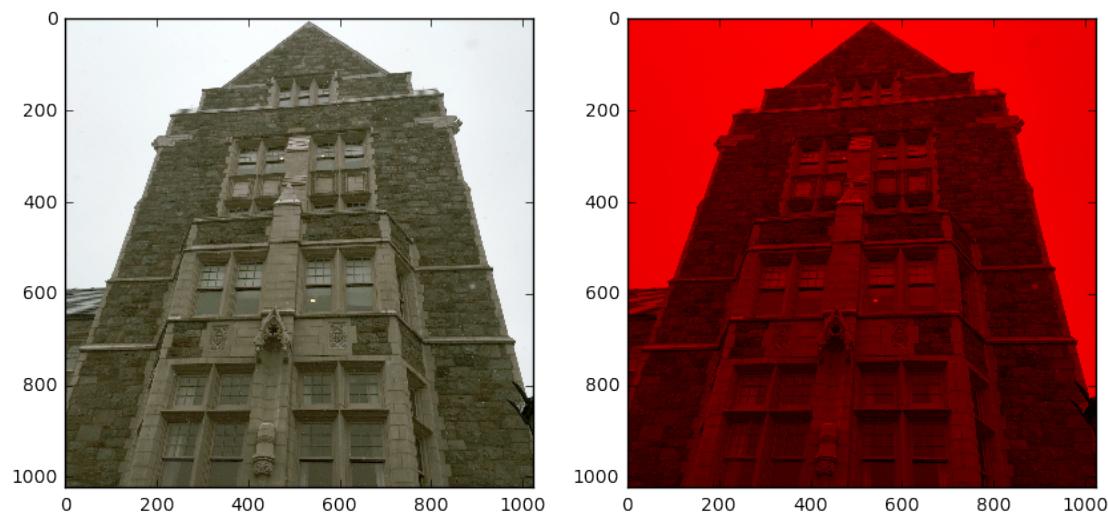
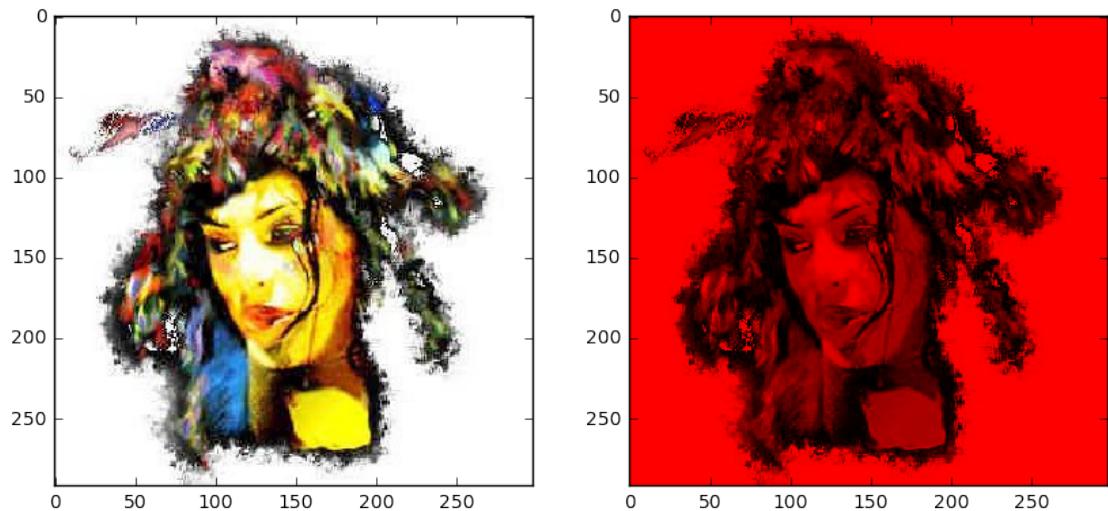


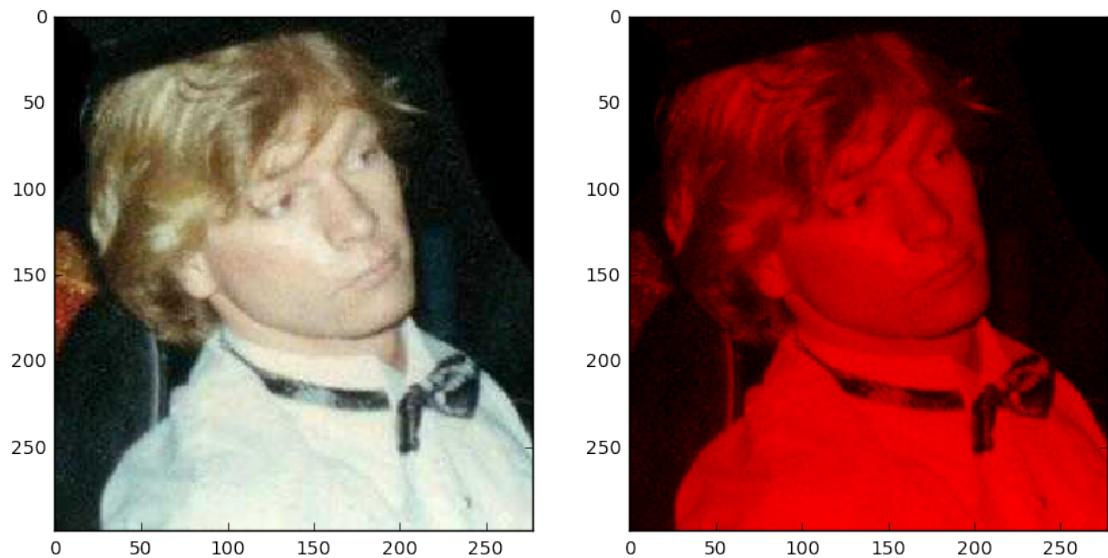
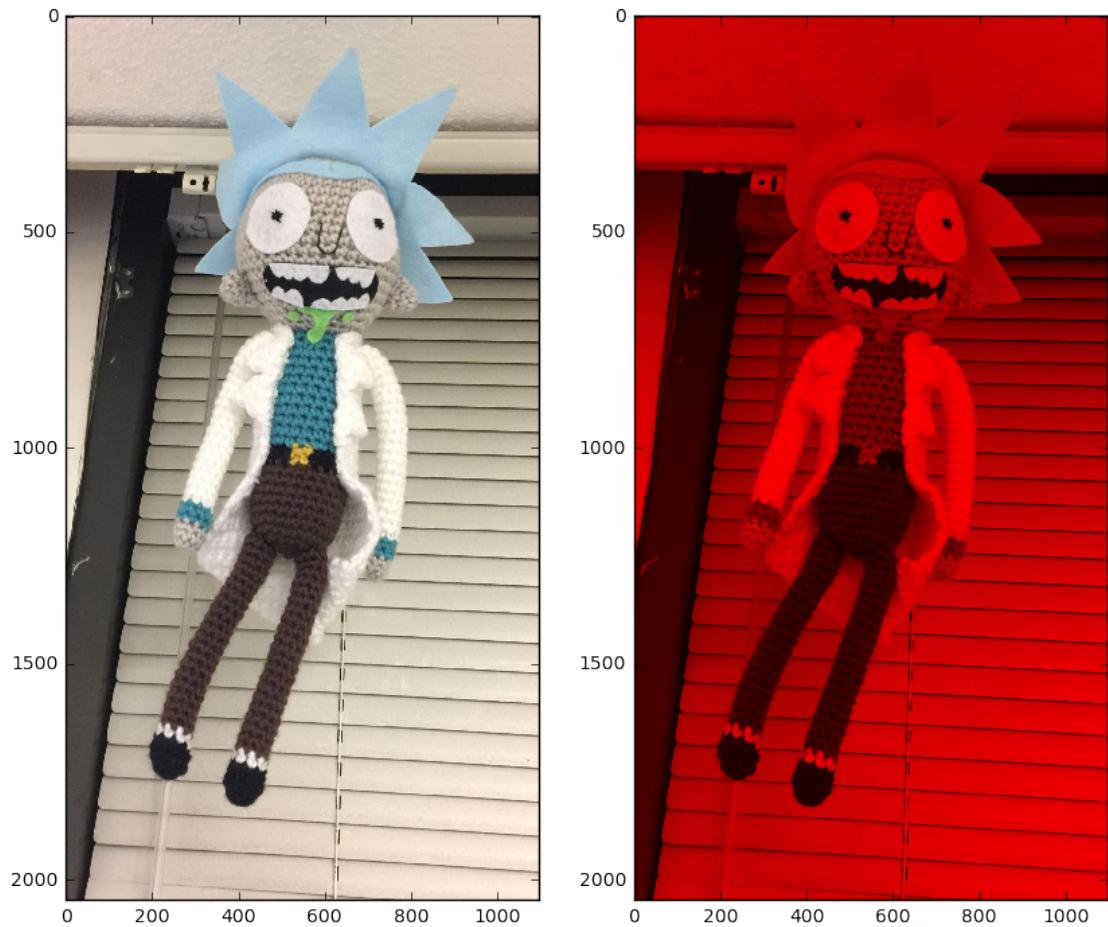


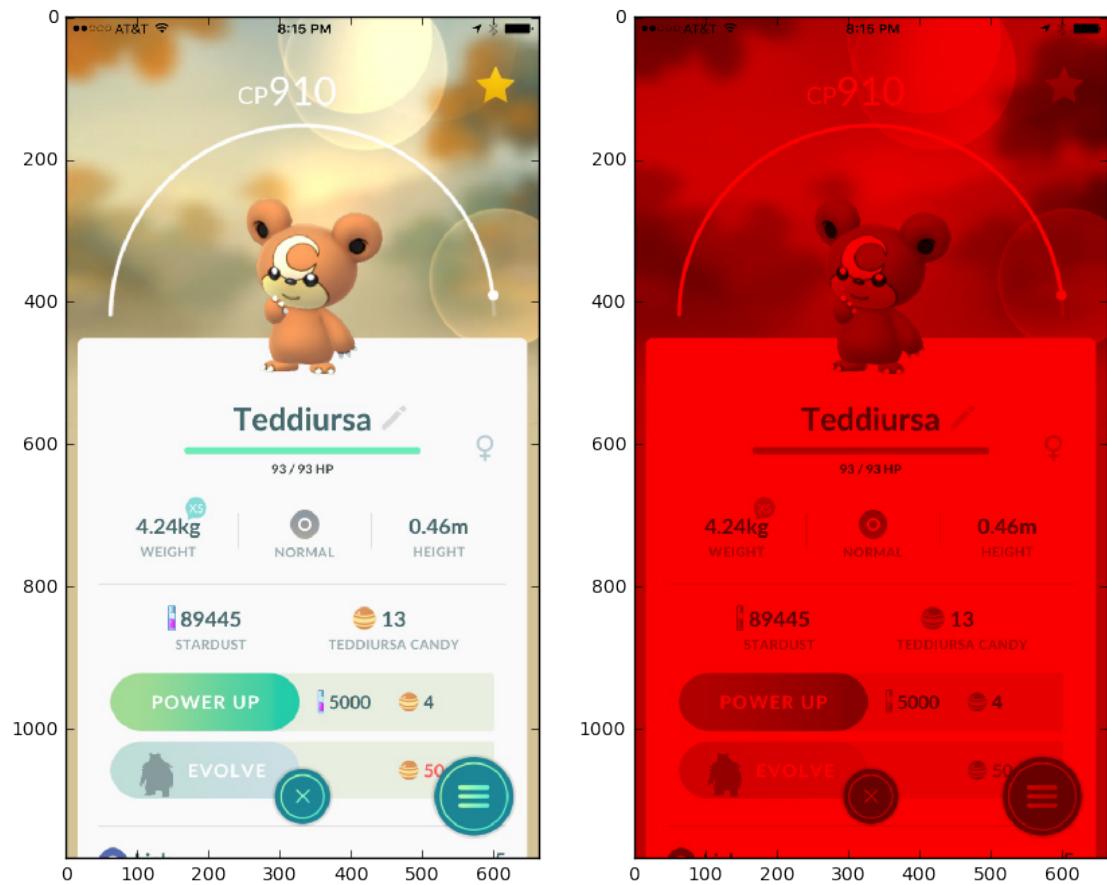
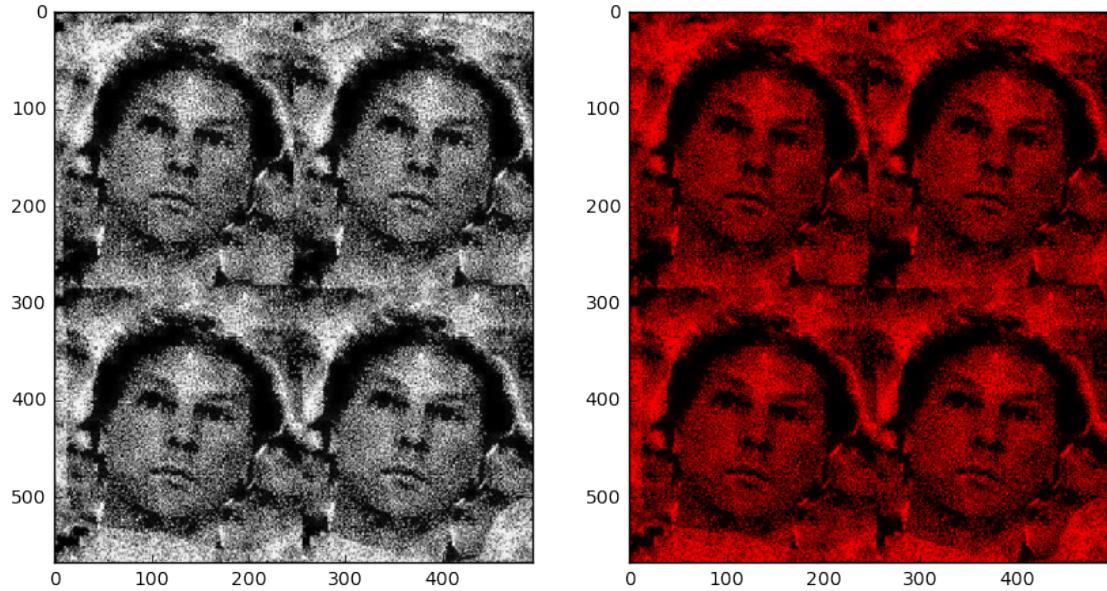
```
In [7]: filter_b = np.array([[1.0, 1.0, 1.0],
                           [0.0, 0.0, 0.0],
                           [0.0, 0.0, 0.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_b))
```





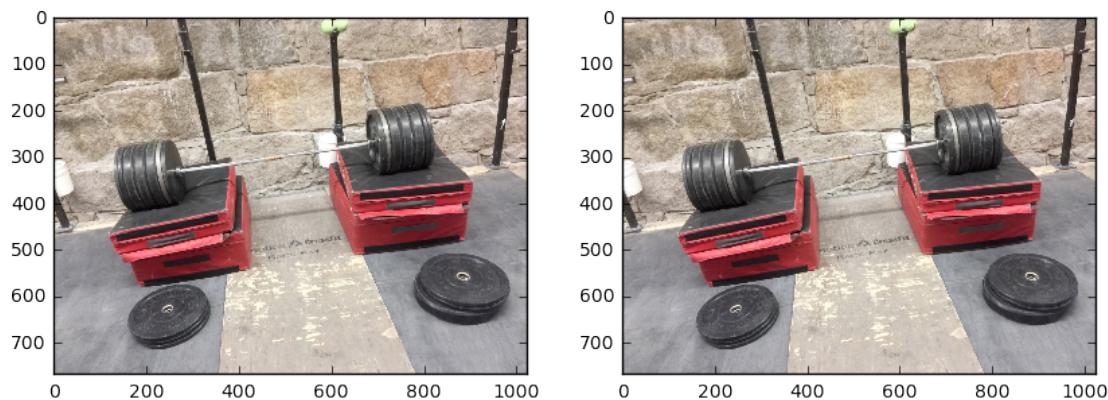
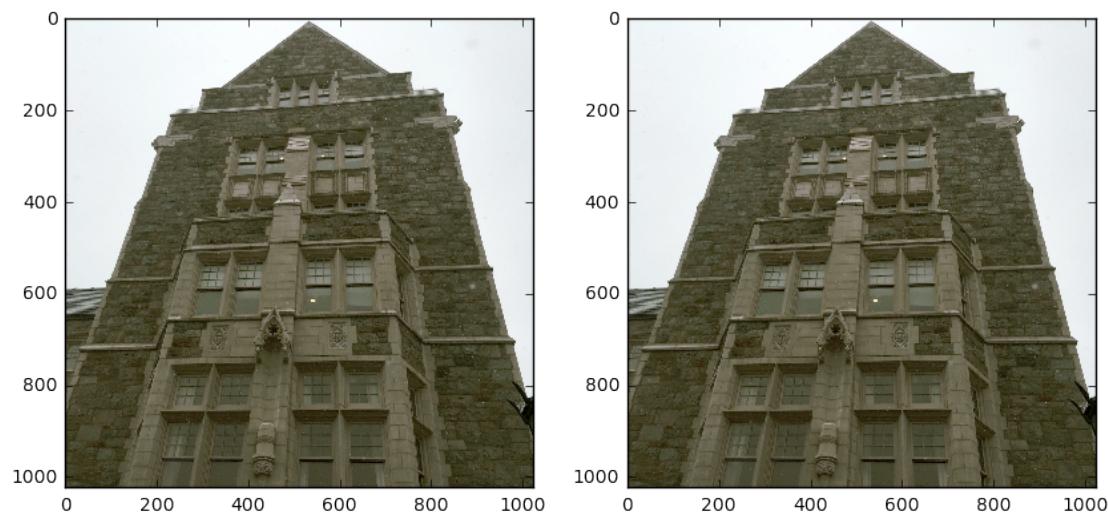
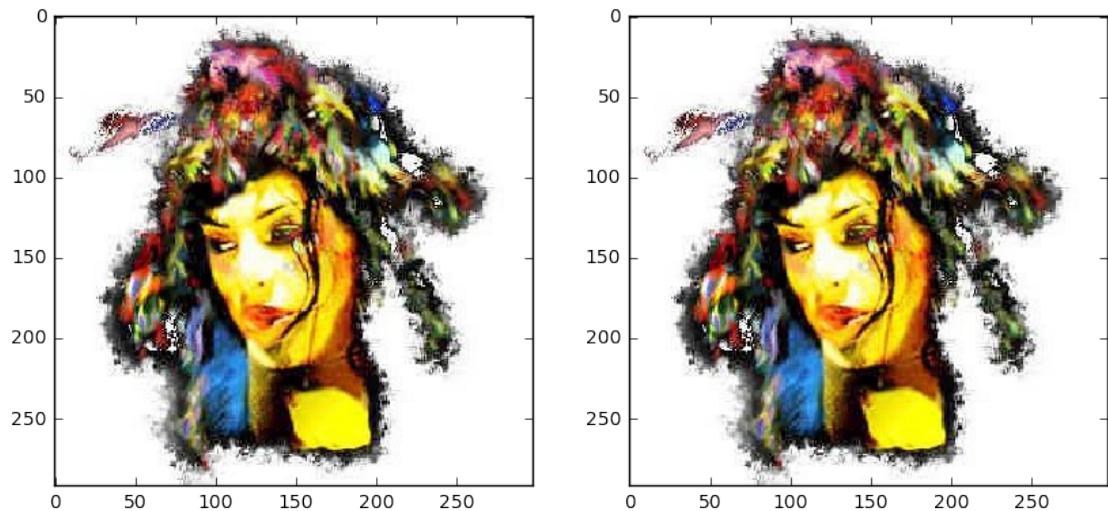


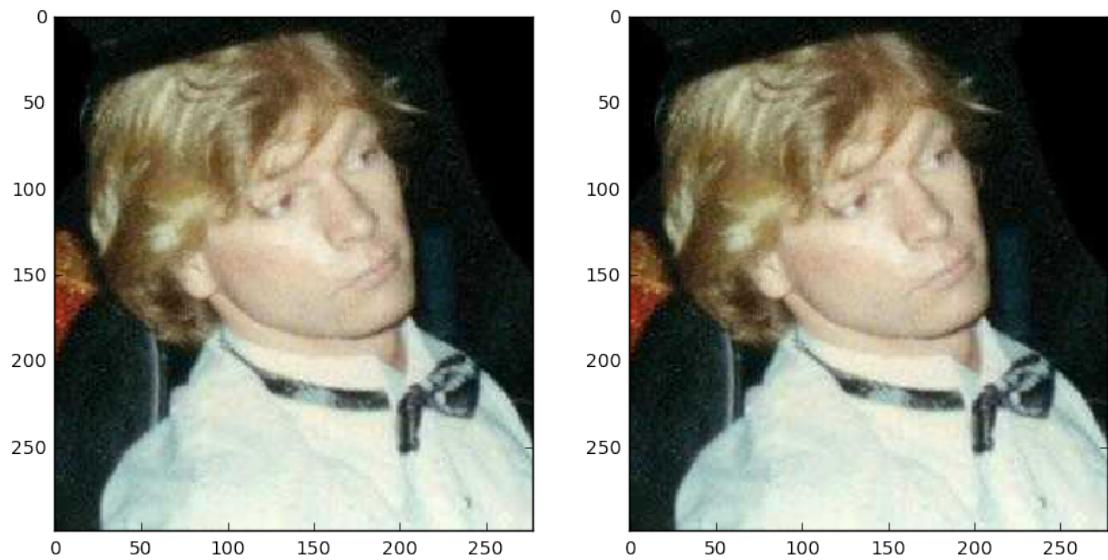
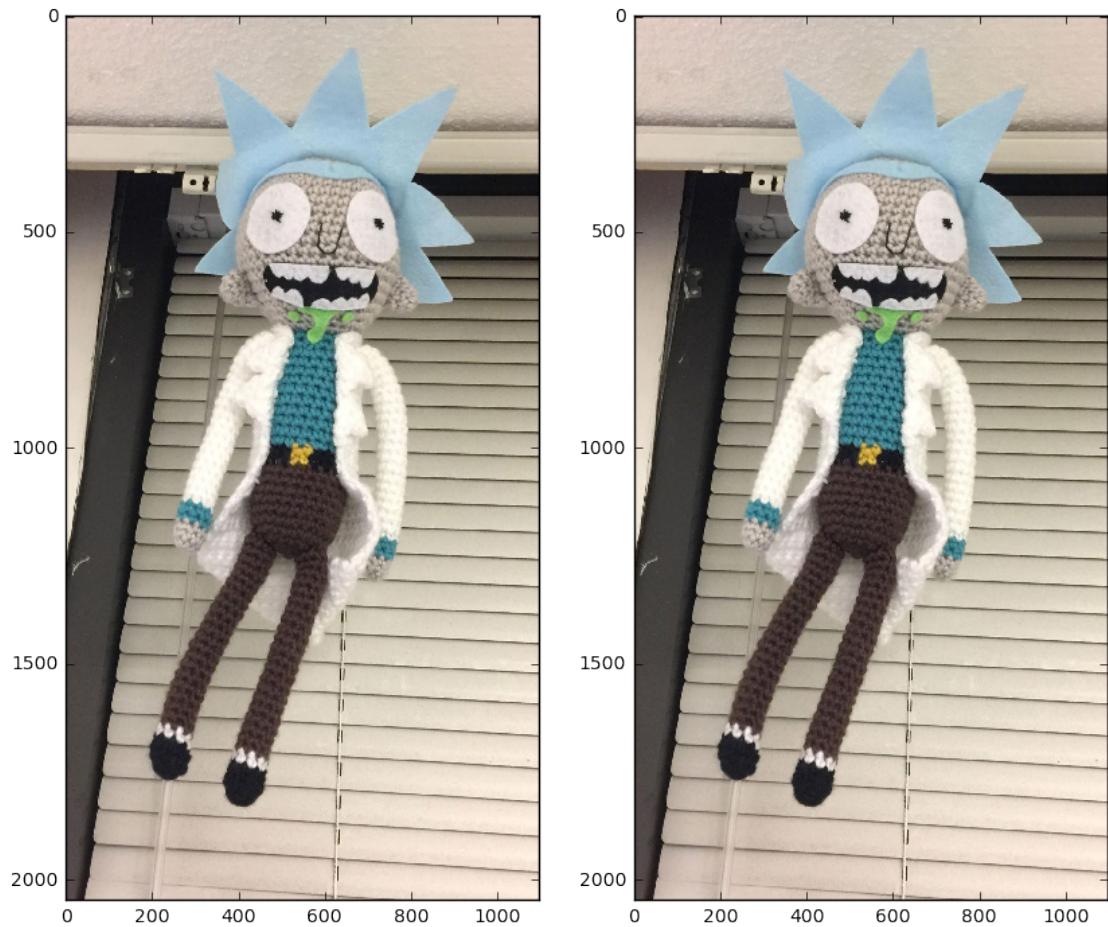


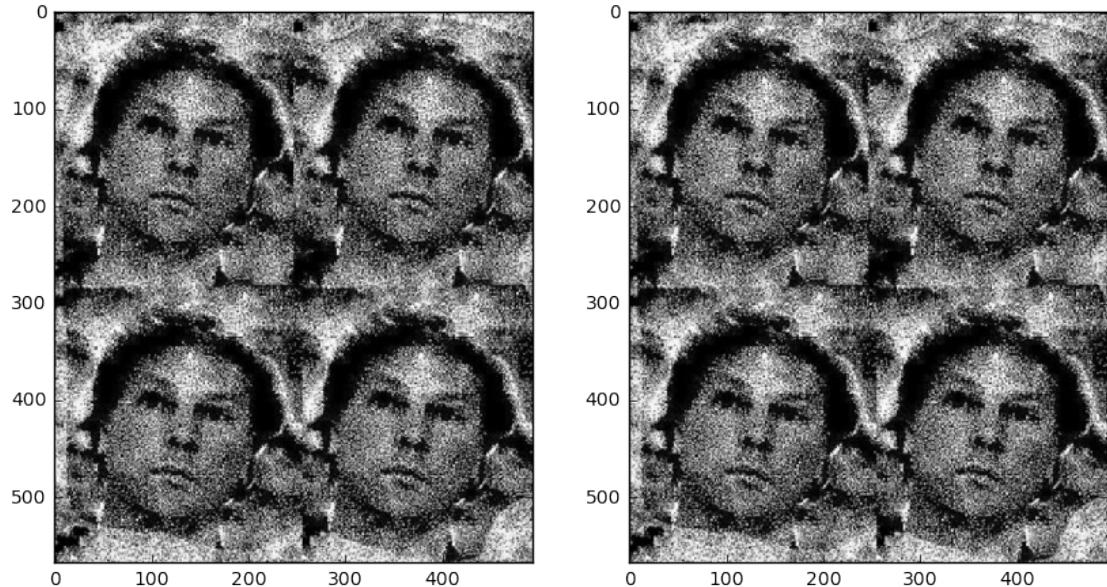
```
In [8]: filter_c = np.array([[1.0, 0.0, 0.0],
                           [0.0, 1.0, 0.0],
                           [0.0, 0.0, 1.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_c))
```





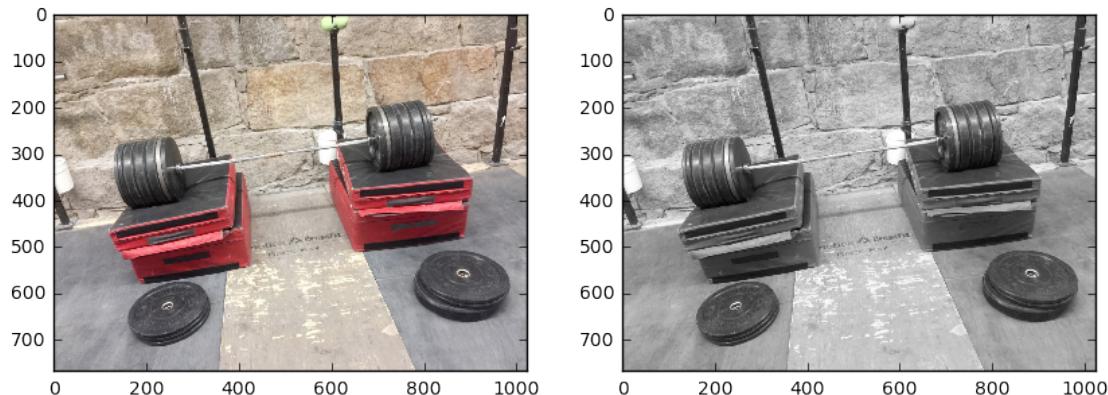
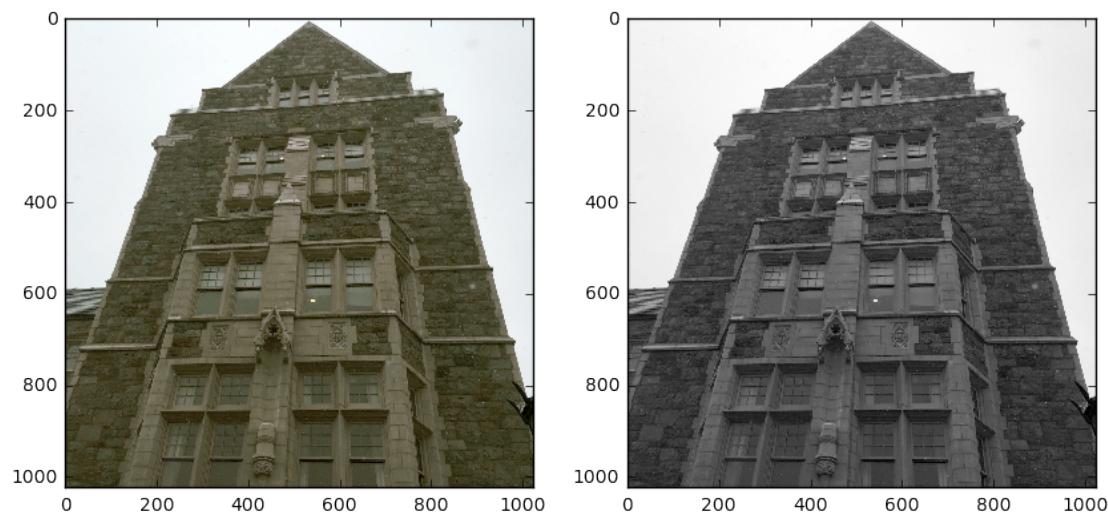
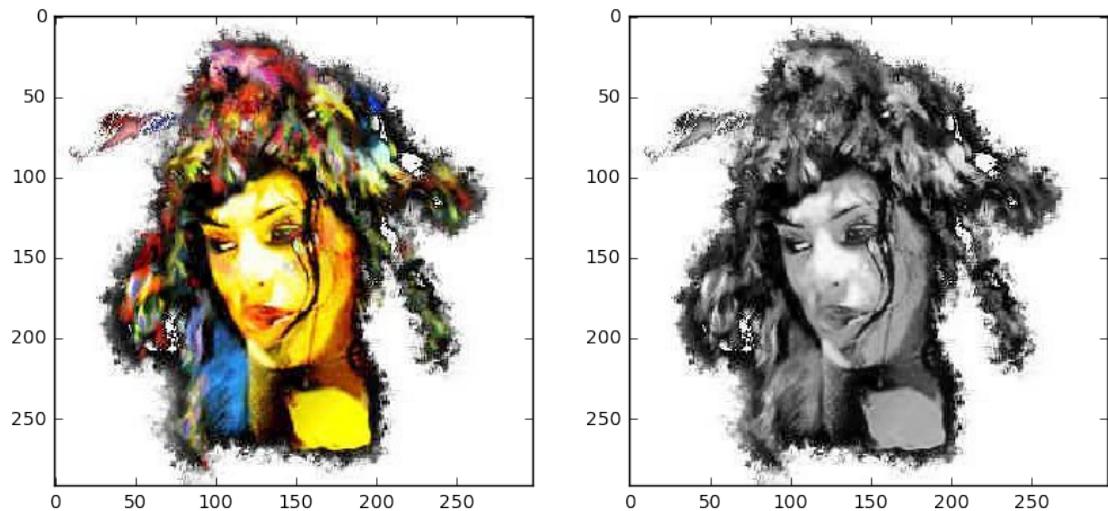


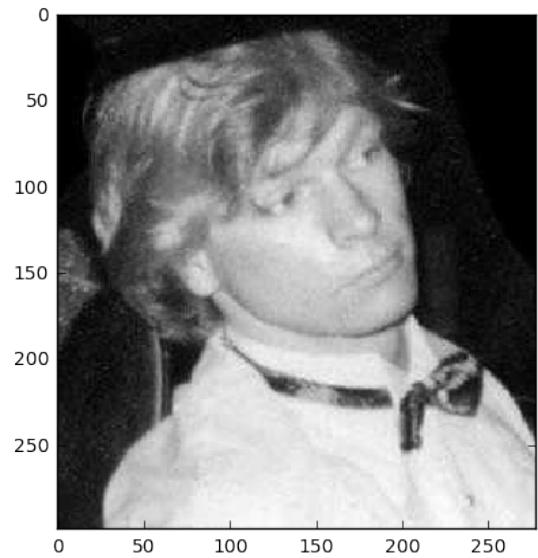
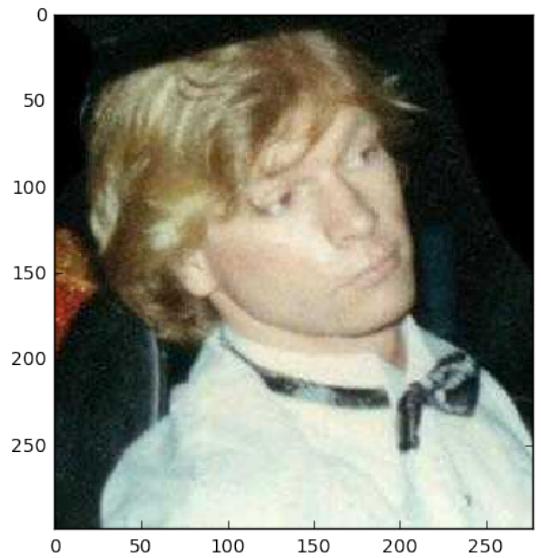
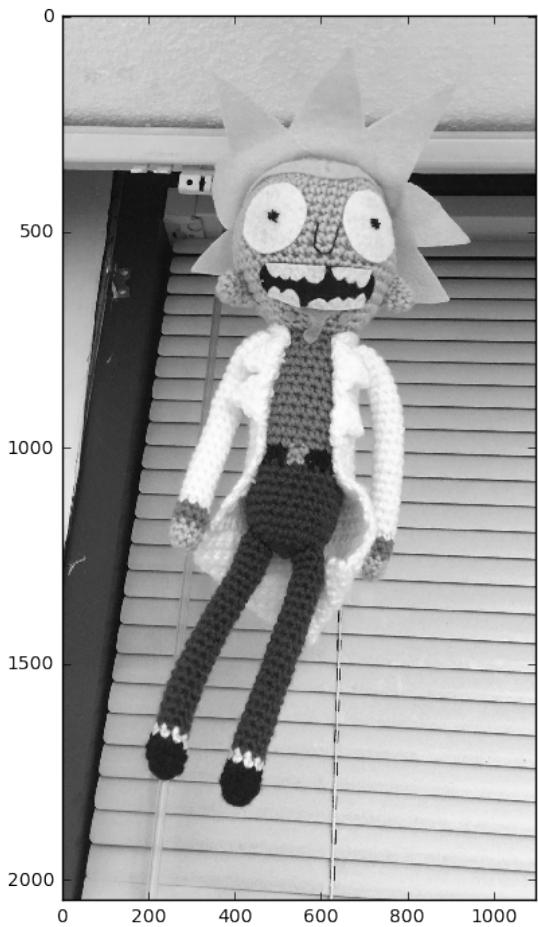
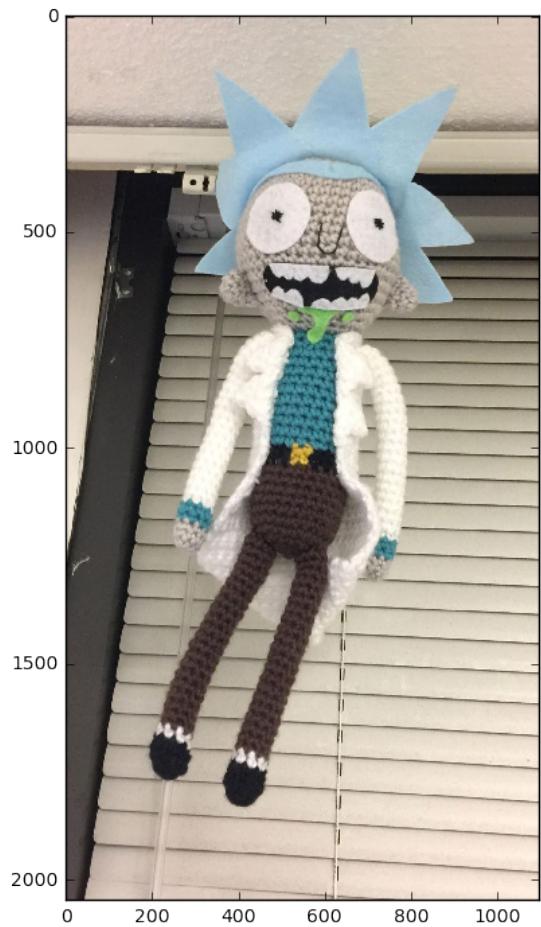


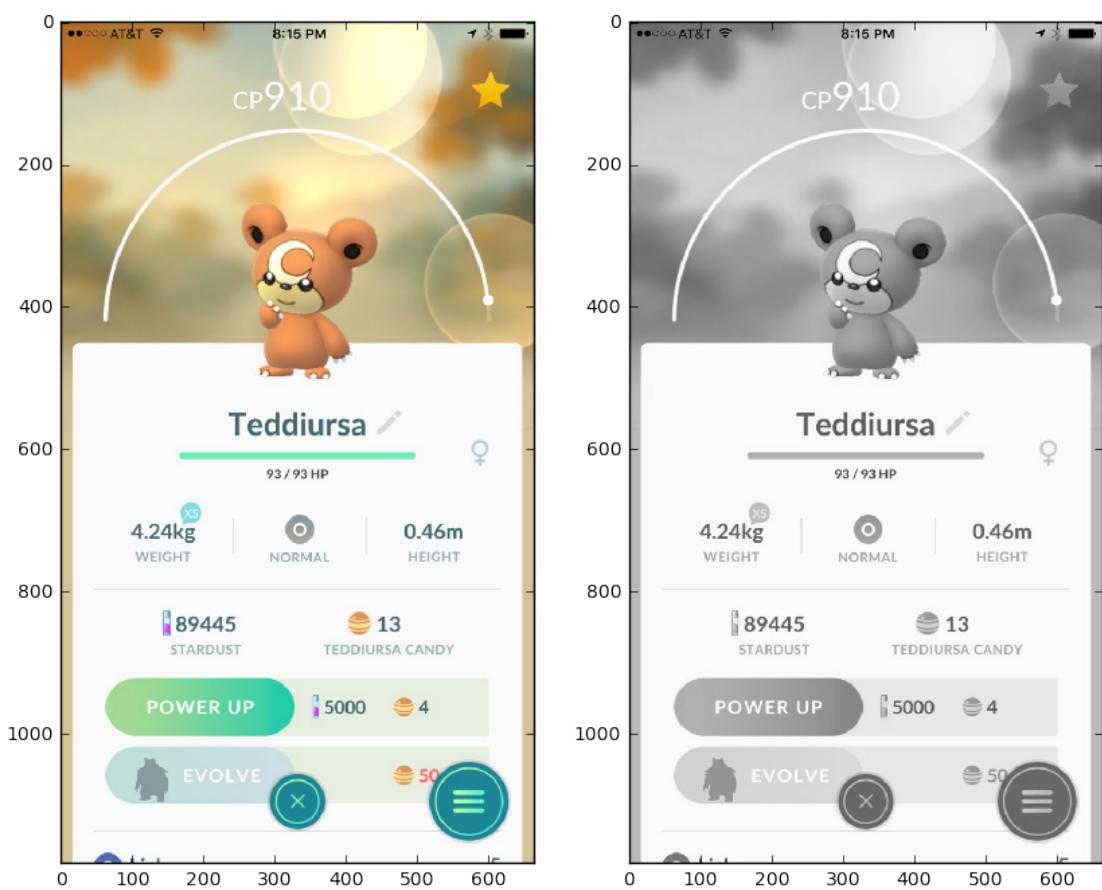
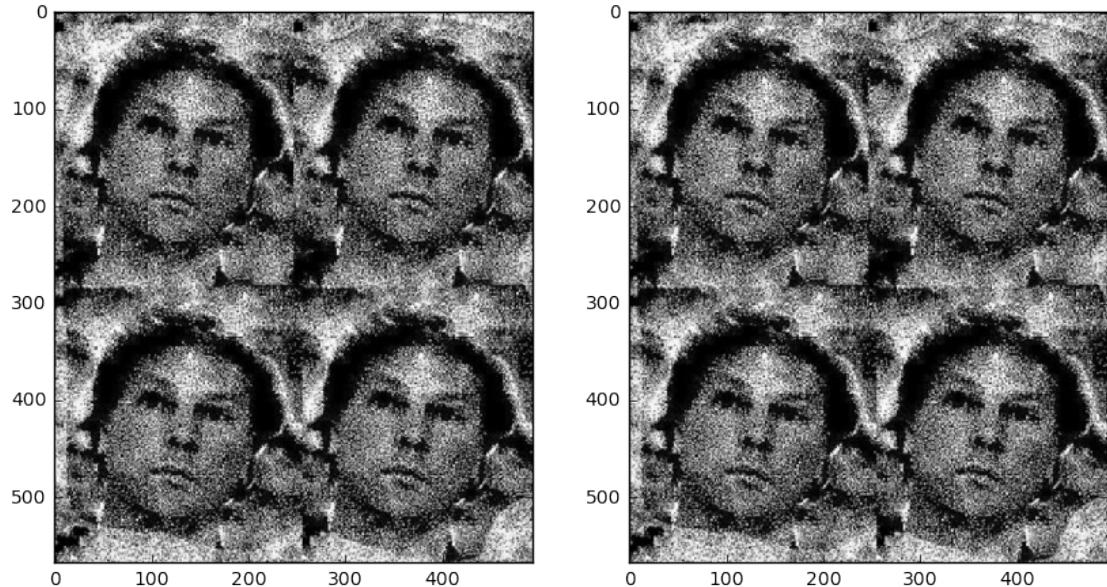
```
In [9]: filter_d = np.array([[0.5, 0.5, 0.5],
                           [0.5, 0.5, 0.5],
                           [0.5, 0.5, 0.5]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_d))
```





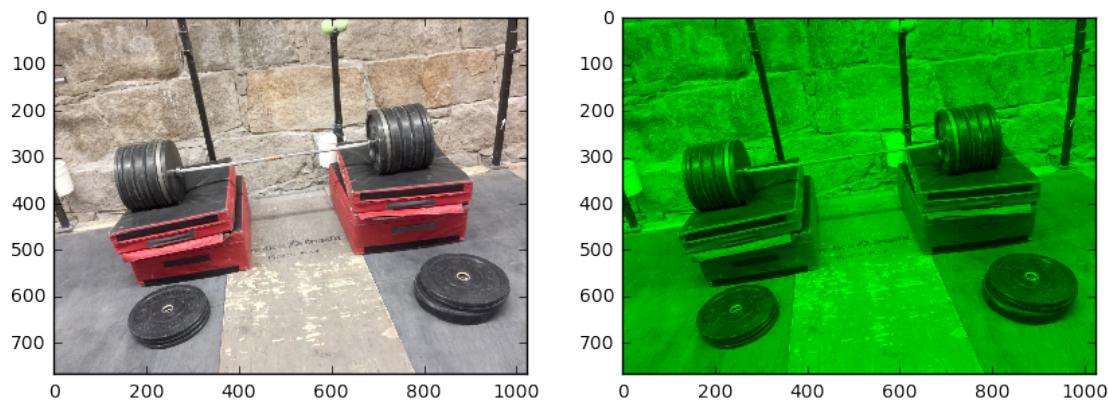
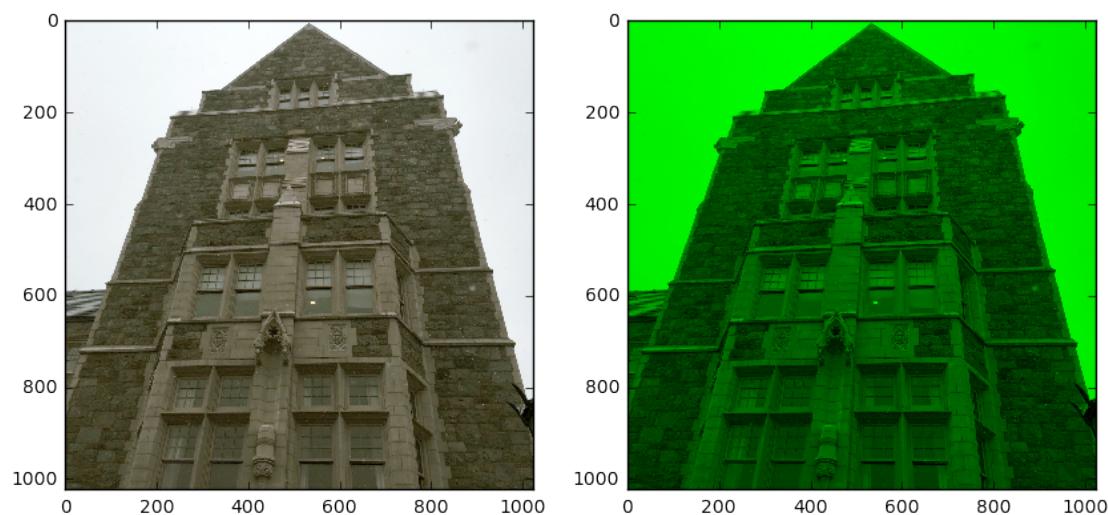
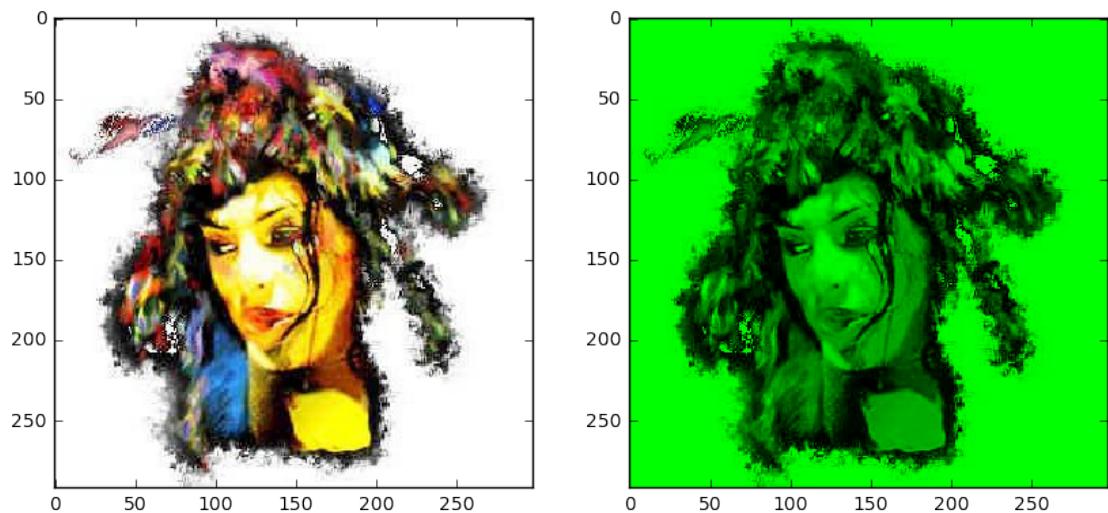


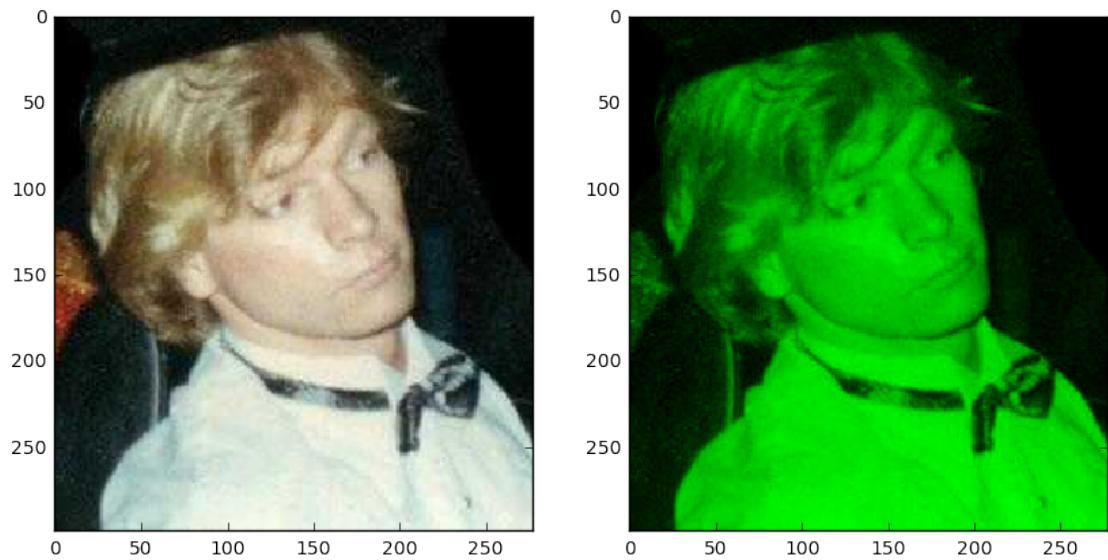
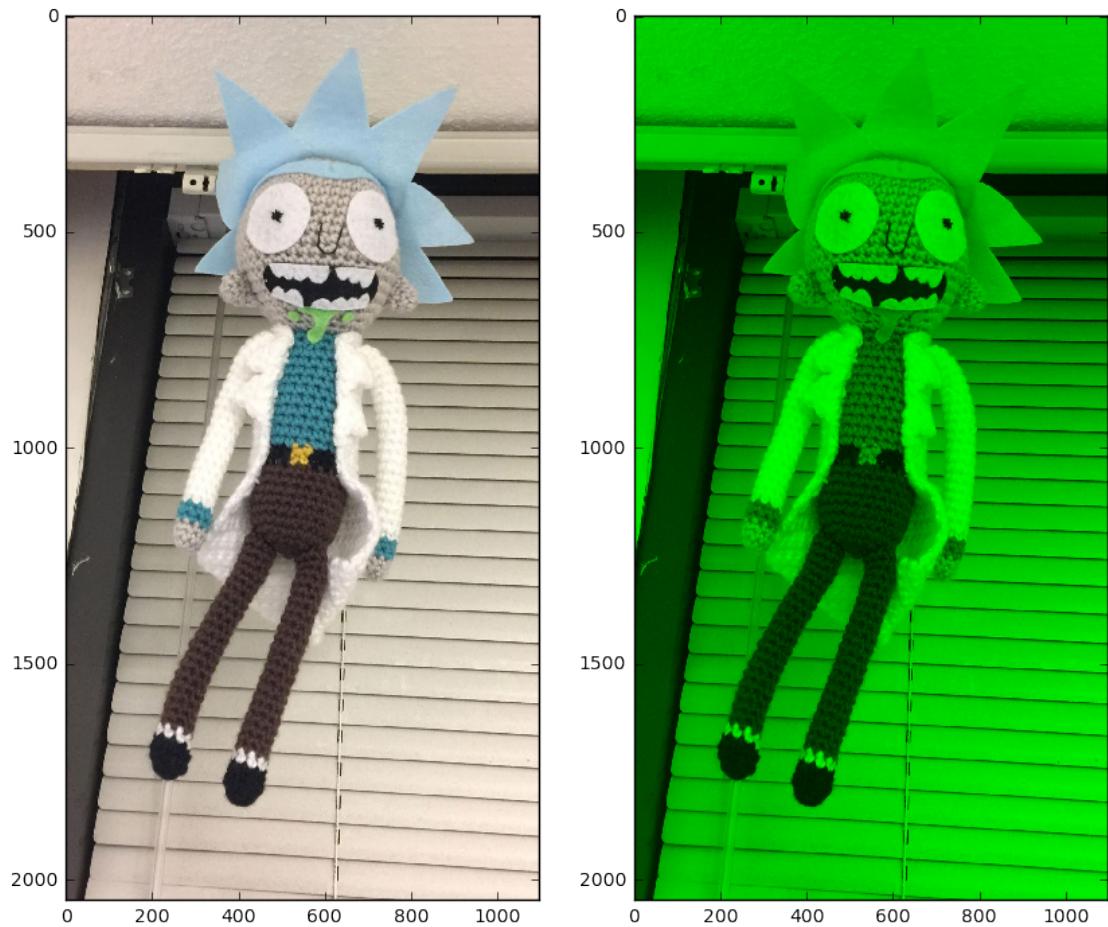


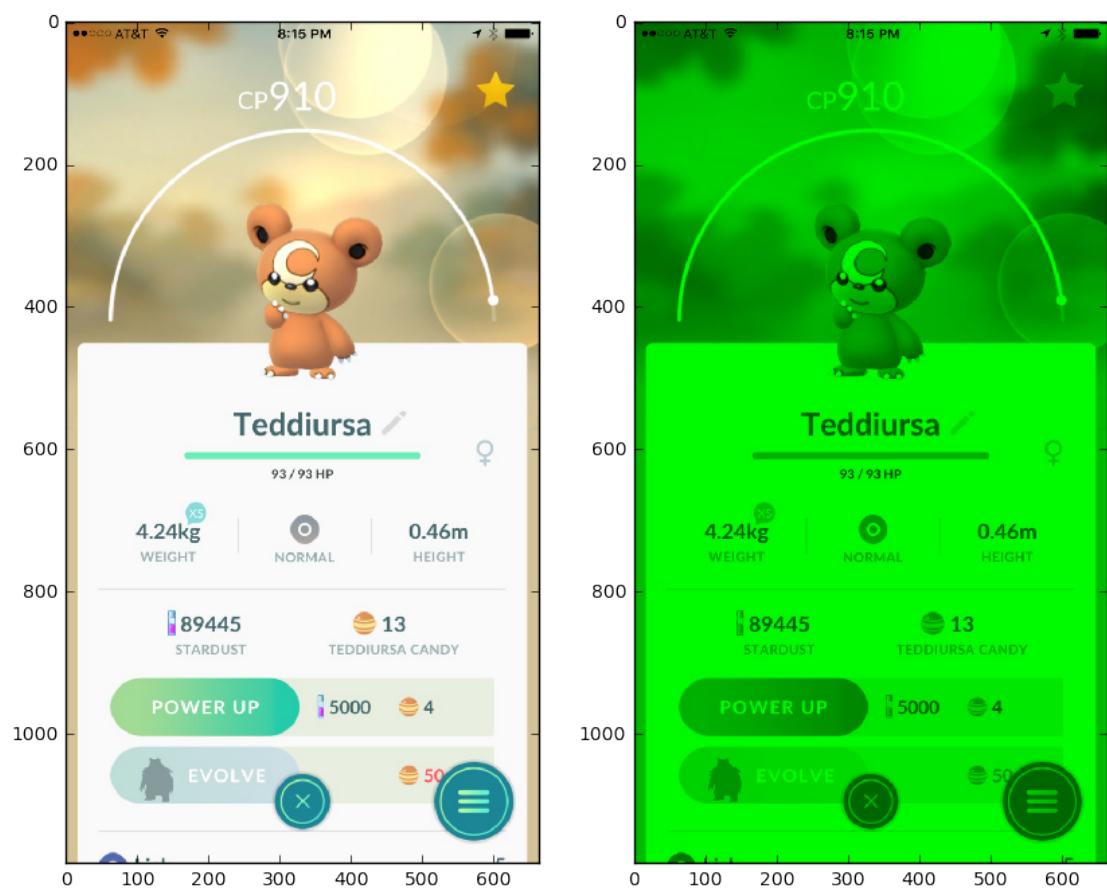
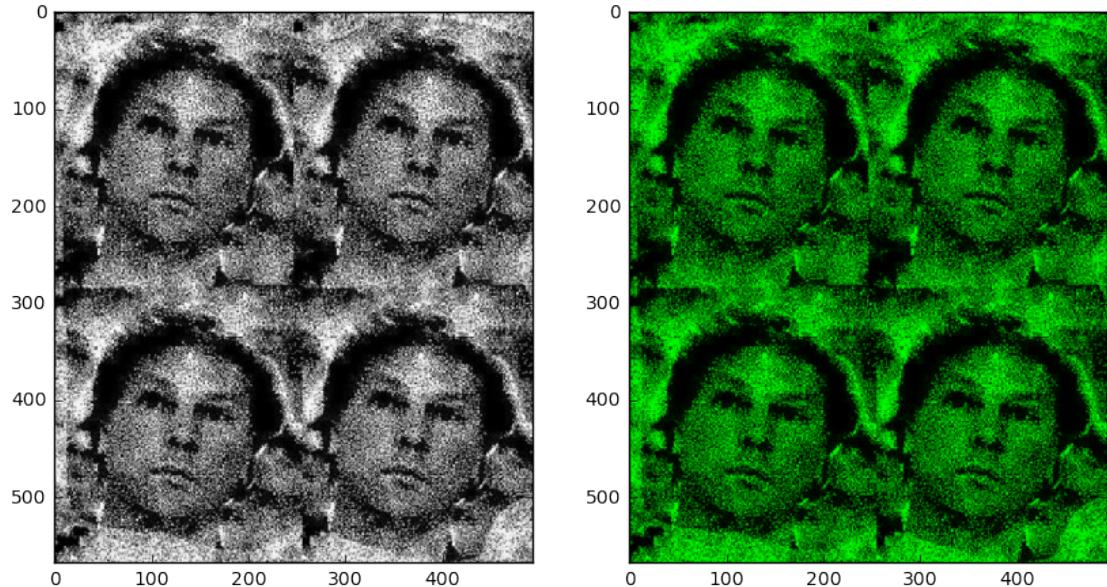
```
In [10]: filter_e = np.array([[0.0, 0.0, 0.0],
                           [1.0, 1.0, 1.0],
                           [0.0, 0.0, 0.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_e))
```





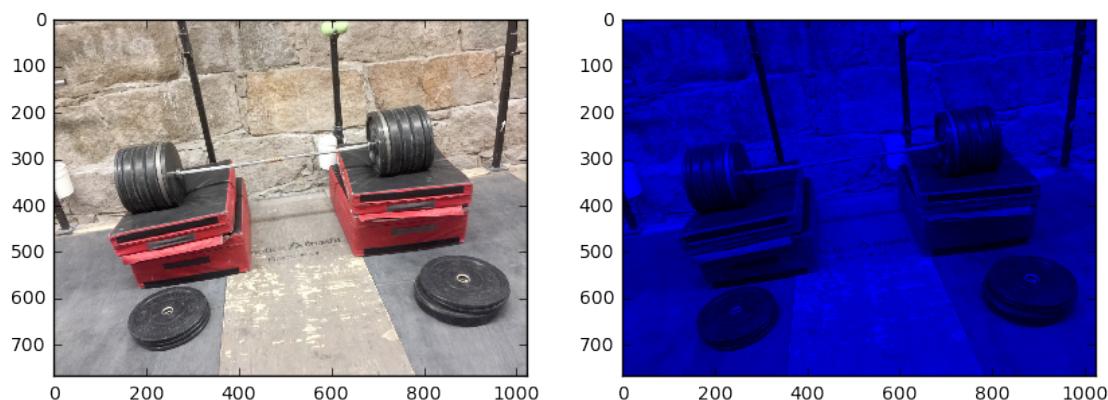
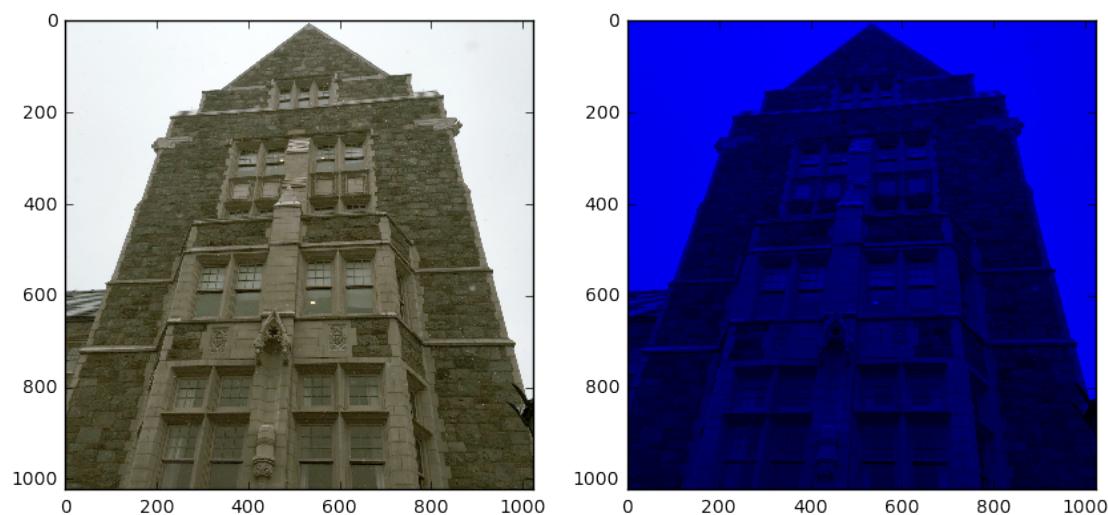
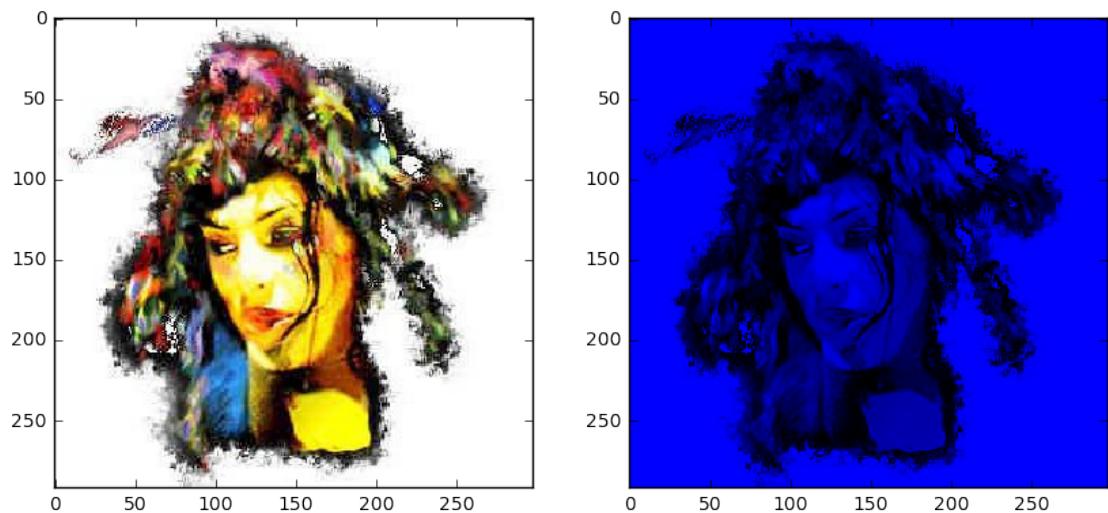


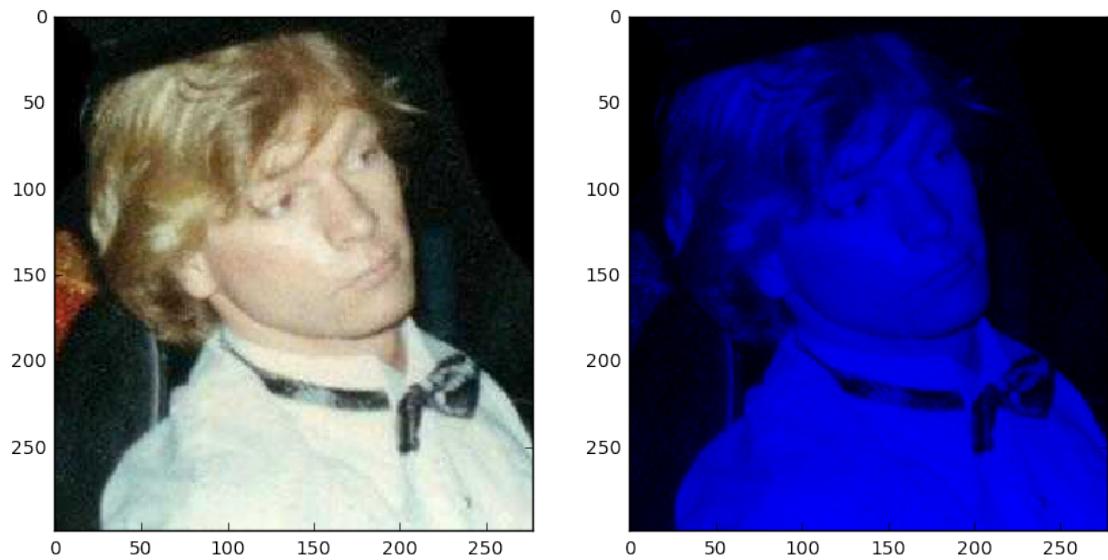
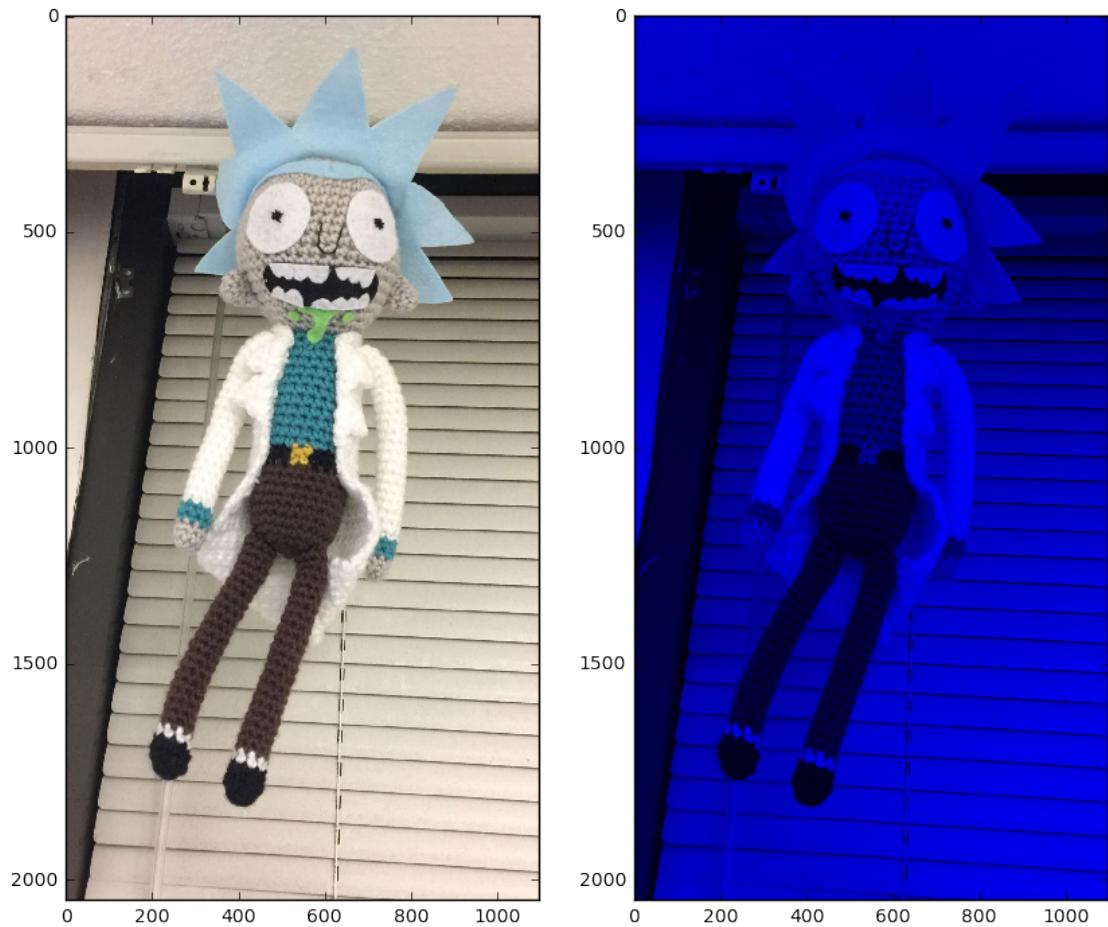


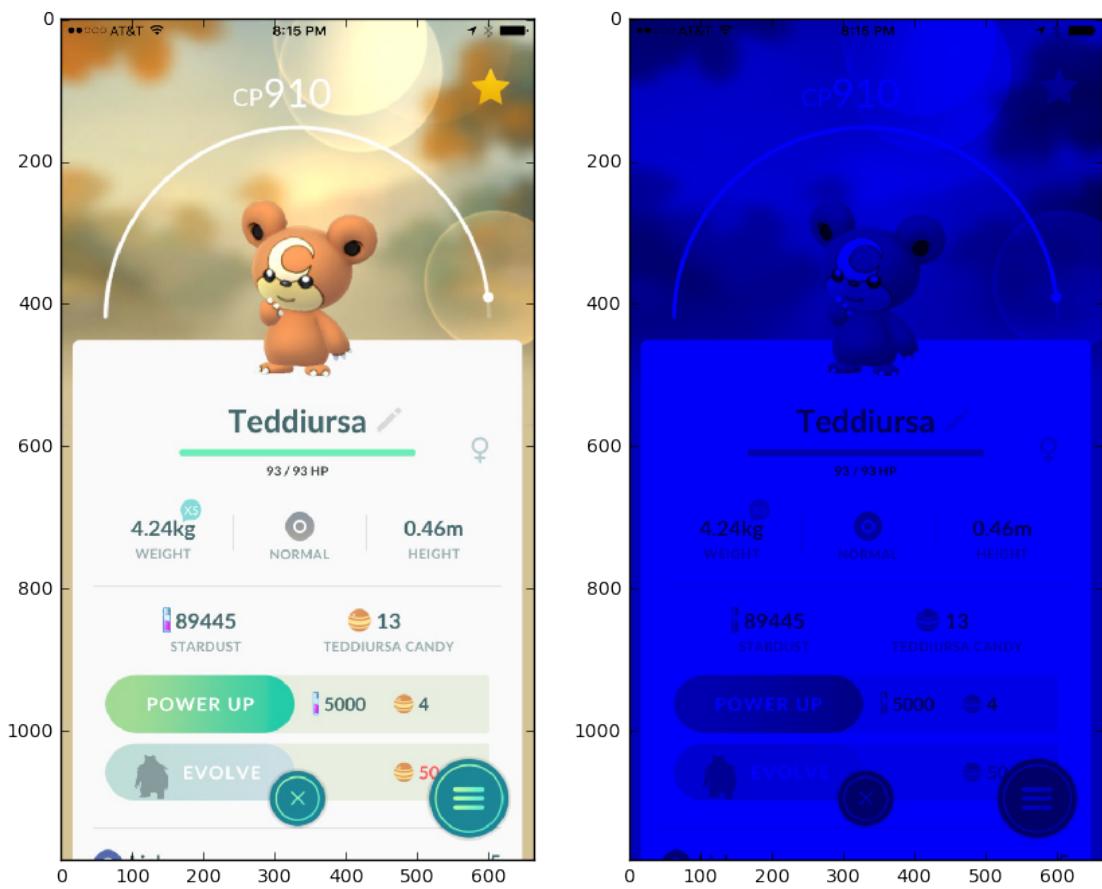
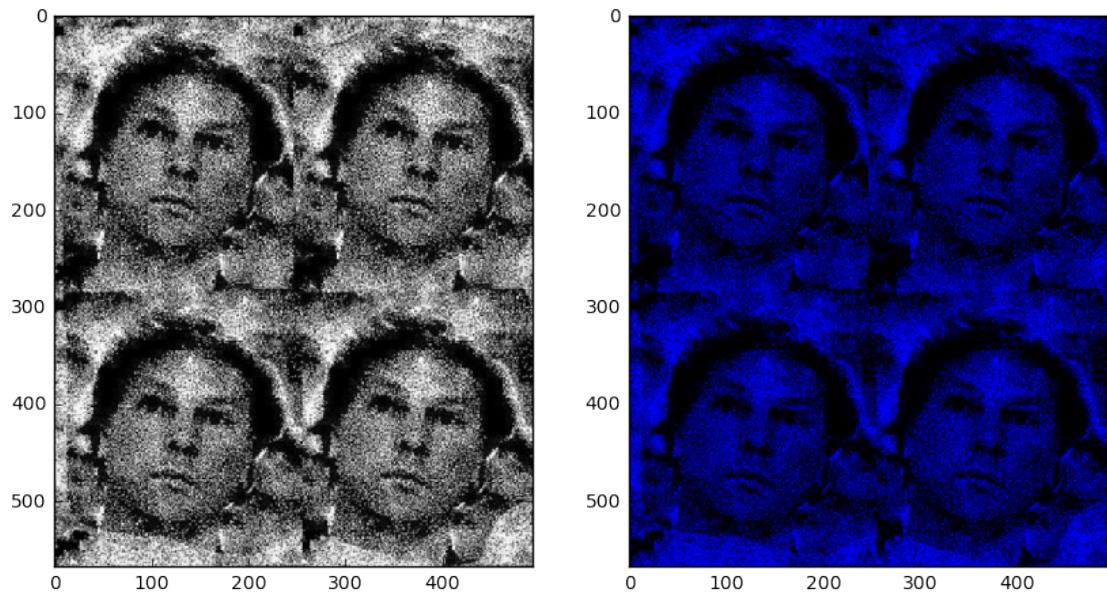
```
In [11]: filter_f = np.array([[0.0, 0.0, 0.0],
                           [0.0, 0.0, 0.0],
                           [1.0, 1.0, 1.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_f))
```





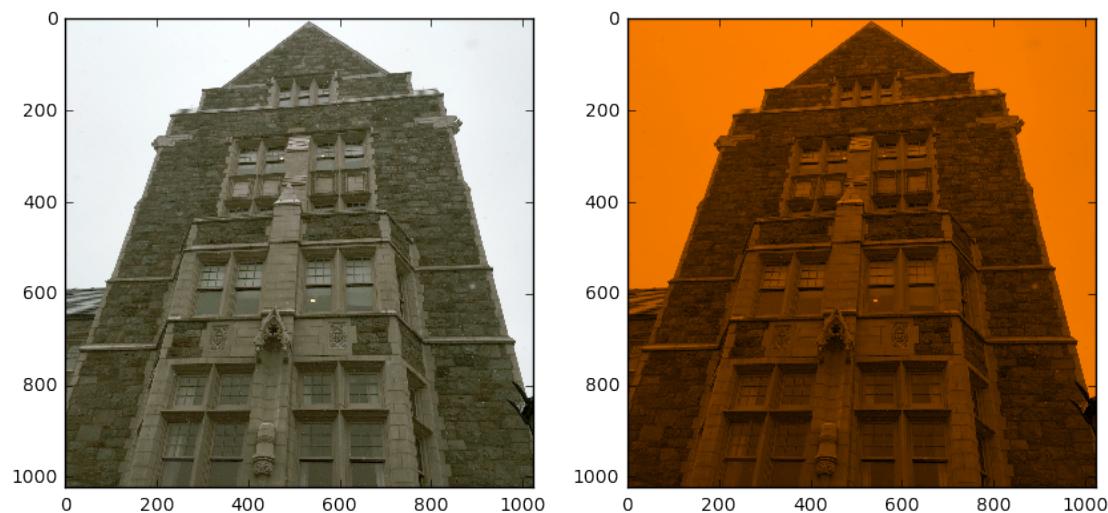
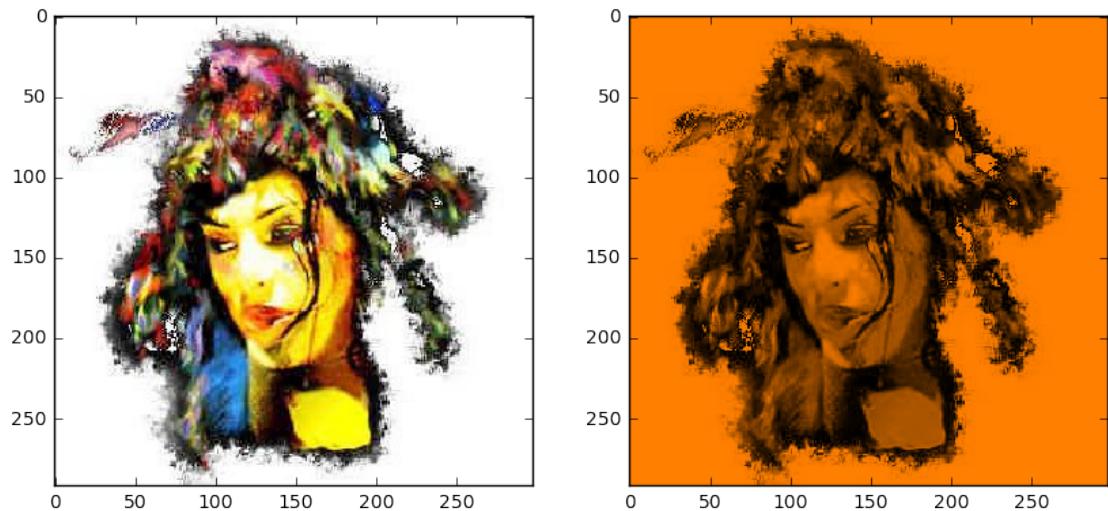


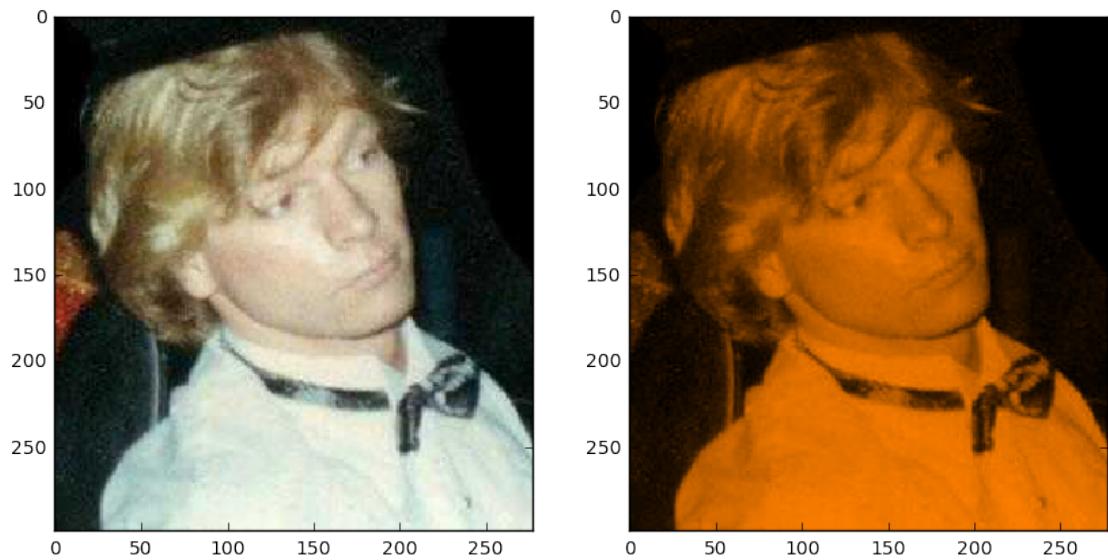
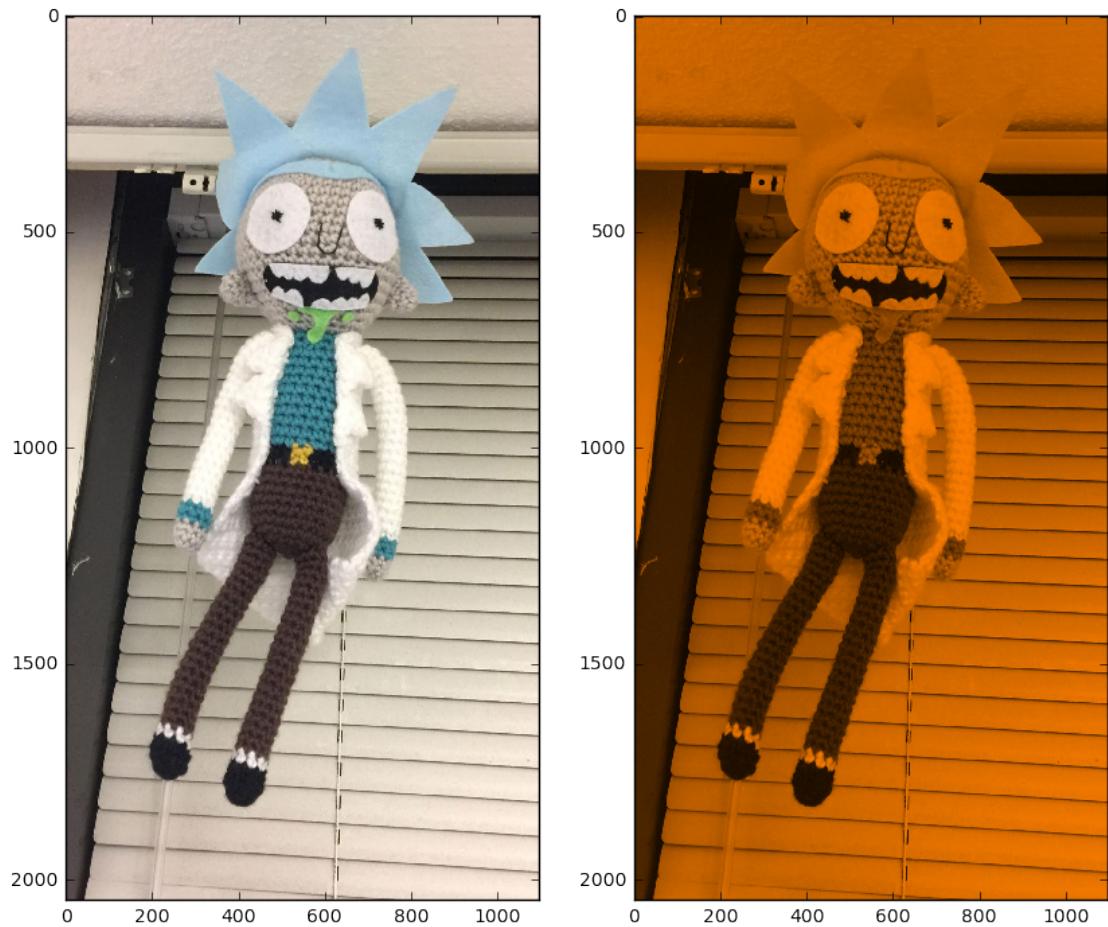


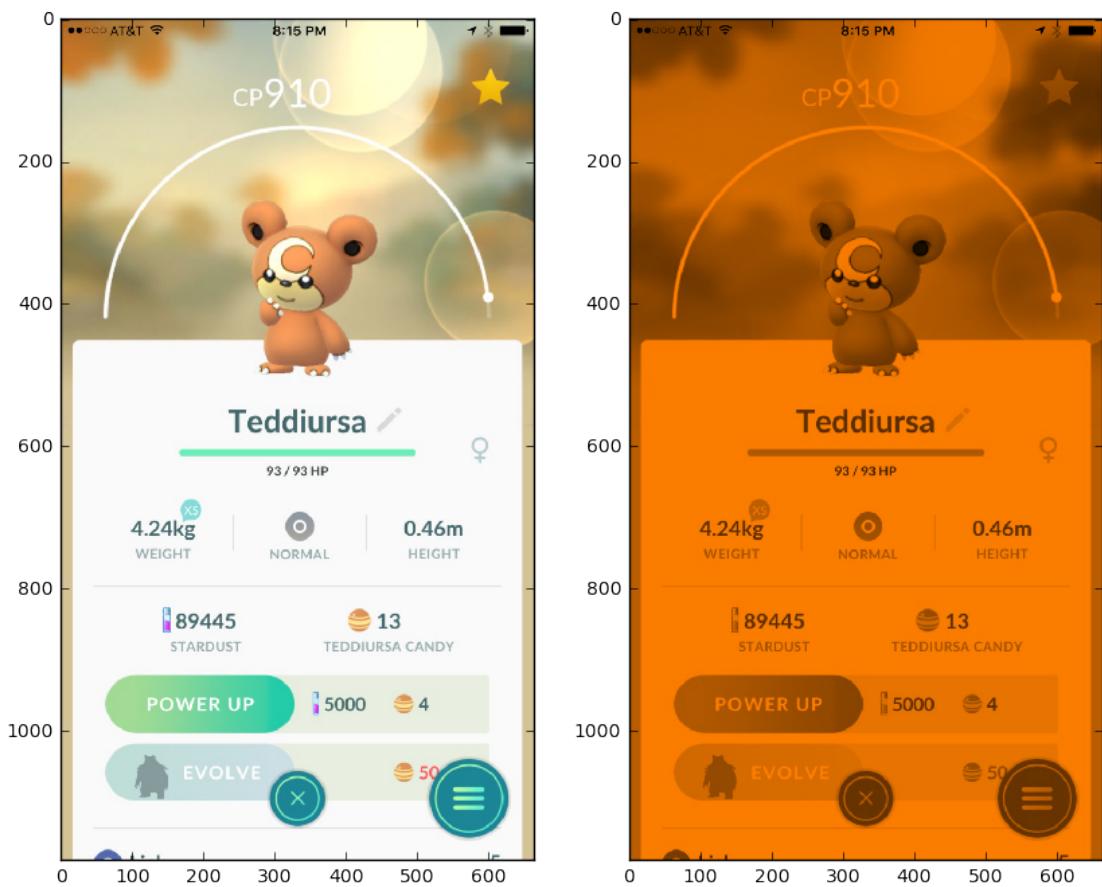
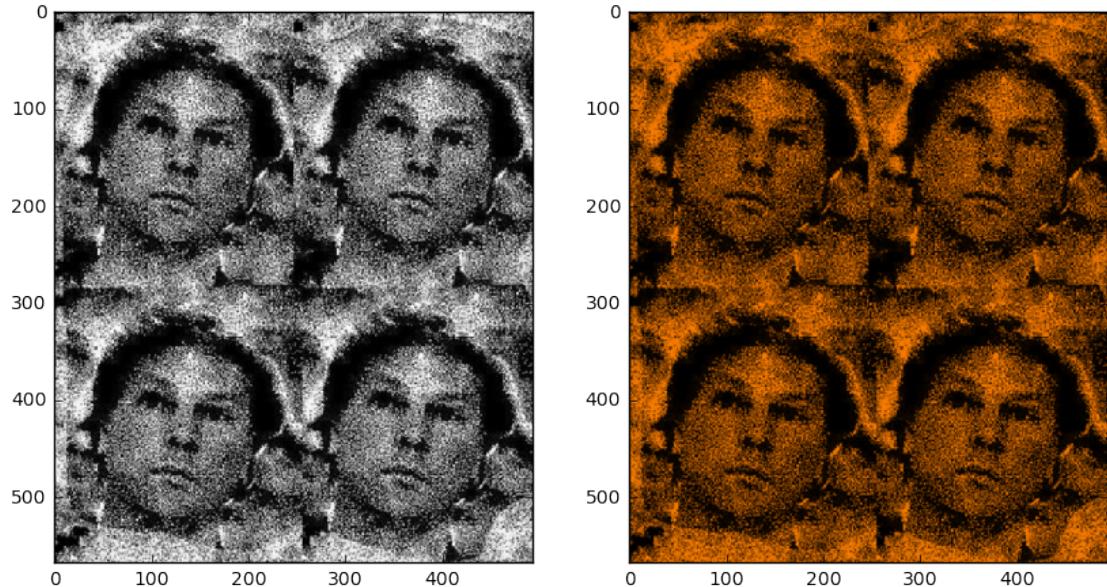
```
In [12]: filter_g = np.array([[1.0, 1.0, 1.0],
                           [0.5, 0.5, 0.5],
                           [0.0, 0.0, 0.0]])

# Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia(img,filter_g))
```



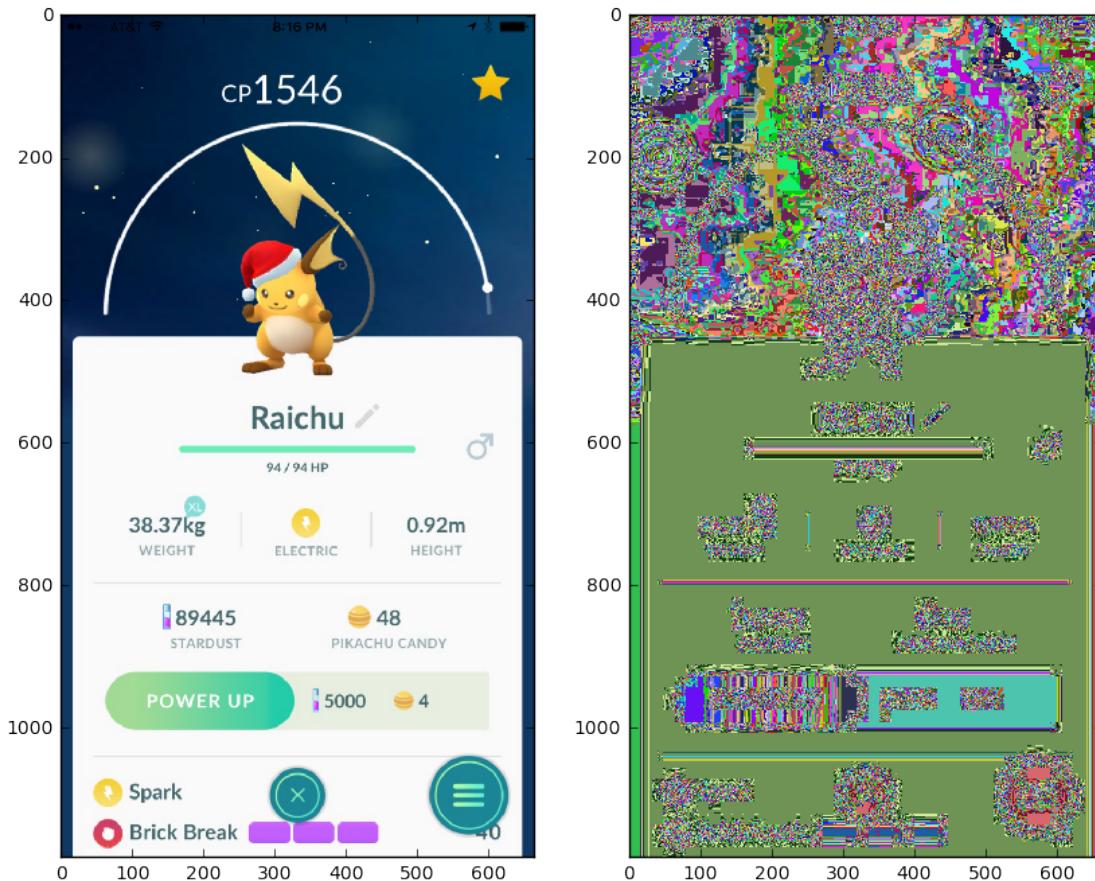


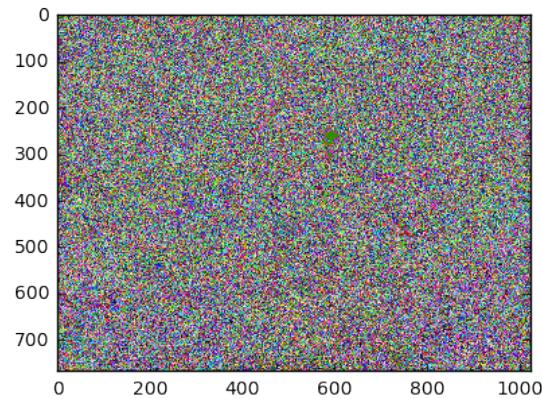
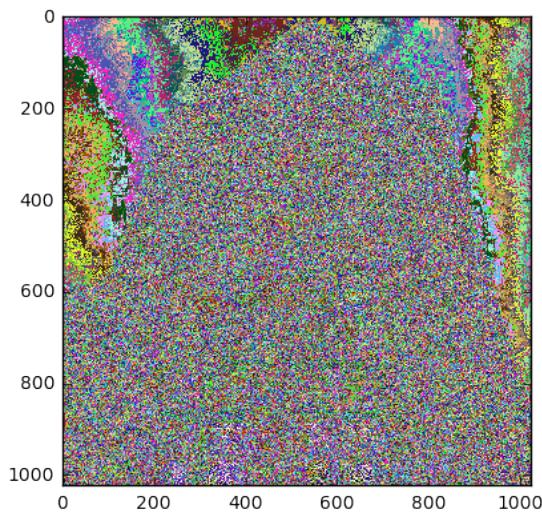
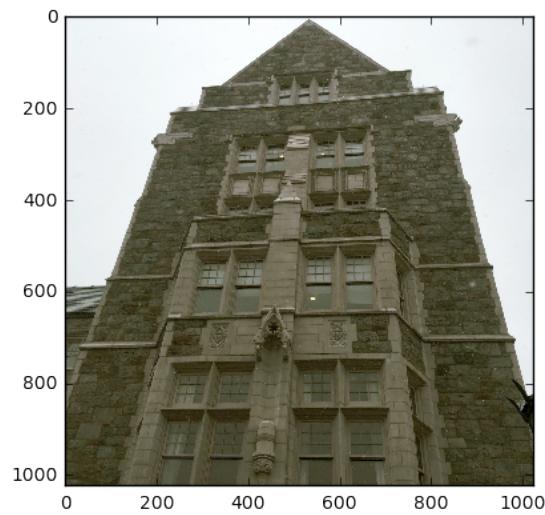
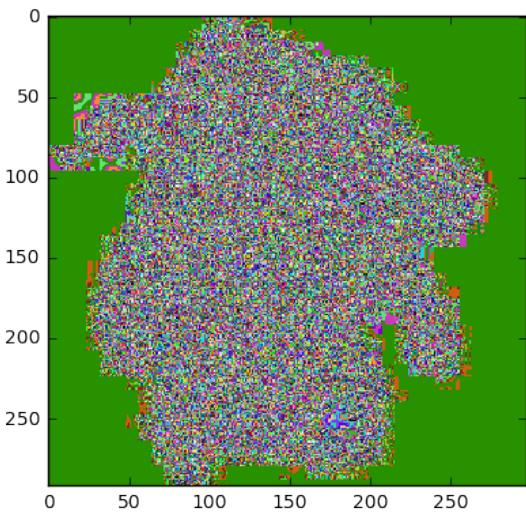
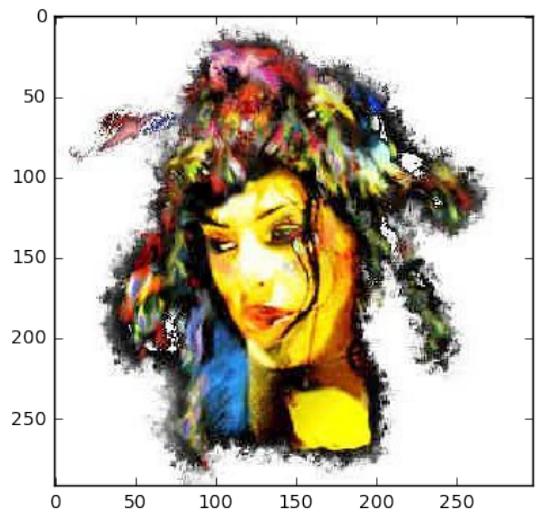


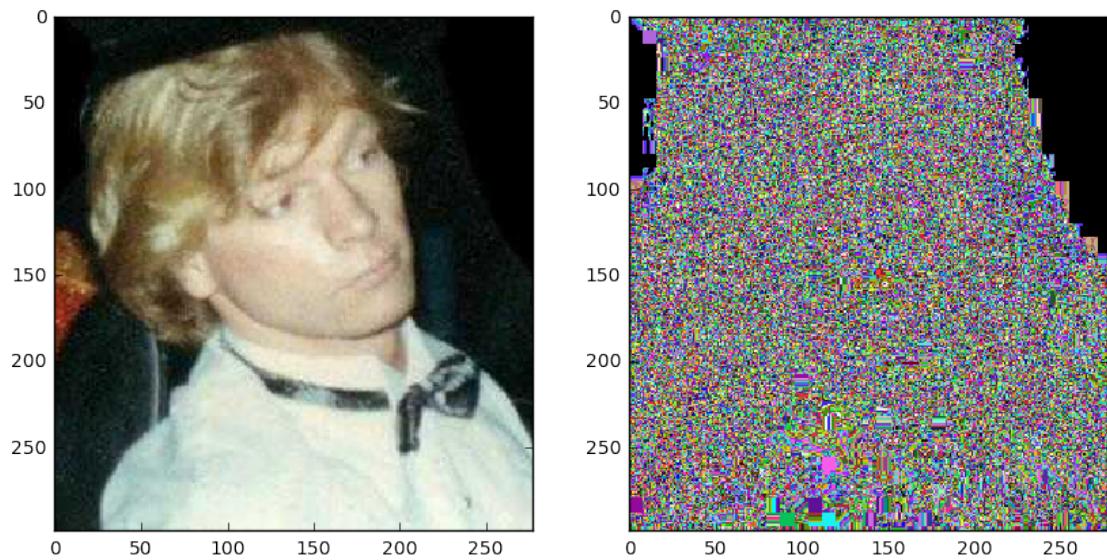
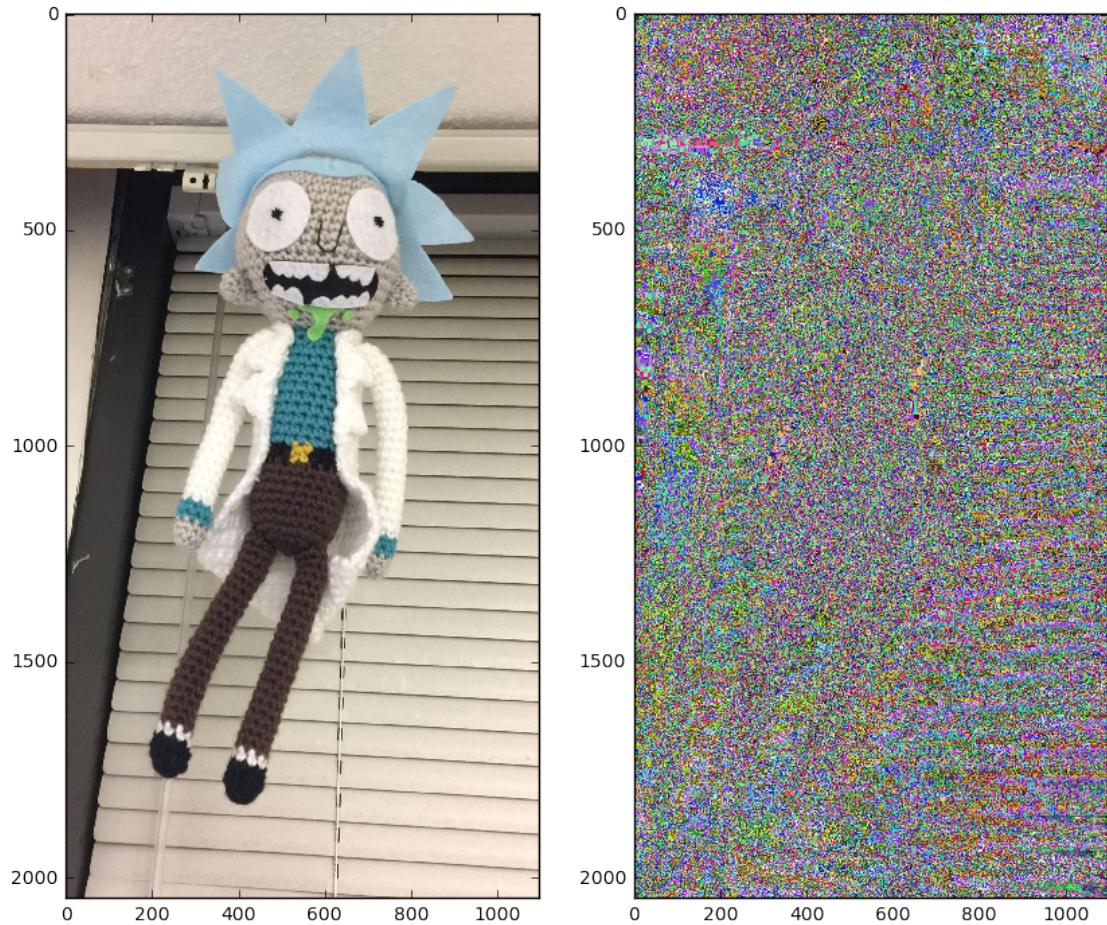


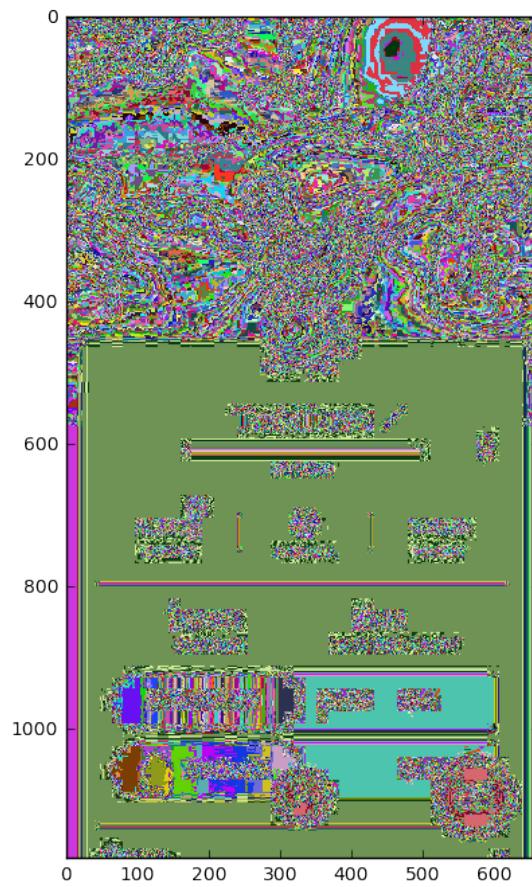
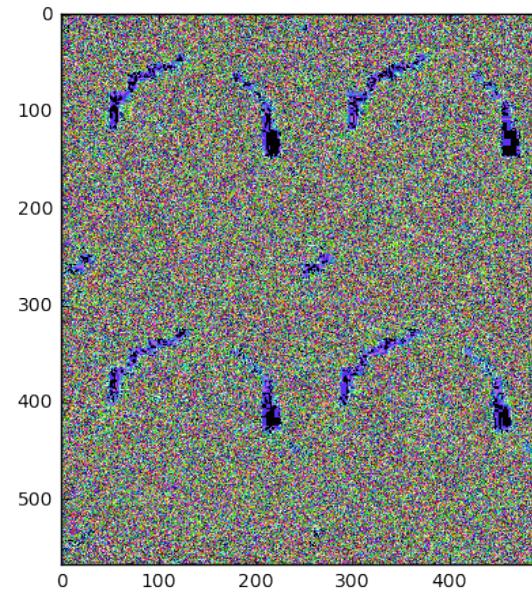
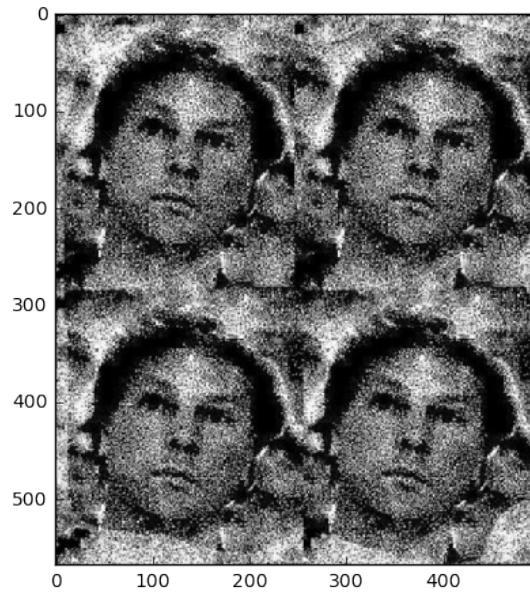
```
In [13]: def sepia_bad(image, filter):
    sepia_img = image.dot(filter.T) #Dot product the image and array
    #T is the transpose of the array.
    #Since our filter lines do not have unit sum, so we need to rescale
    # sepia_img /= sepia_img.max()
    return sepia_img
```

```
In [14]: # Apply to image set
for i,img in enumerate(imgset):
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.imshow(img)
    plt.subplot(1, 2, 2)
    plt.imshow(sepia_bad(img,sepia_filter))
```









2 Python Tutorials

Python 101 Beginning Python http://www.rexx.com/~dkuhlman/python_101/python_101.html

The Official Python Tutorial - <http://www.python.org/doc/current/tut/tut.html>

The Python Quick Reference - <http://rgruet.free.fr/PQR2.3.html>

YouTube Python Tutorials

Google Python Class - <http://www.youtube.com/watch?v=tKTZoB2Vjuk>

Python Fundamentals Training – Classes <http://www.youtube.com/watch?v=rKzZEtxIX14>

Python 2.7 Tutorial Derek Banas - http://www.youtube.com/watch?v=UQi-L-_chcc

Python Programming Tutorial thenewboston - <http://www.youtube.com/watch?v=4Mf0h3HphEA>

3 Evaluation

Install Anaconda 4 for Python 2.7 and get this notebook to run with a set of your images.