161180038

Hilal Ilgaz

CENG471 – Assignment 1

The assignment's aim is colorizing Prokudin-Gorskii photography collection. The collection is very large and photos which are in collection on various themes. Photos contain people, religious architecture, industry, views of different places etc [1]. Photographs look like gray but they have 3 color channels. Those are red, green and blue. These channels' sequence is not classic order. It means channels' order is blue, green and red. Images should be splitted three parts and the parts should be assign the rgb channels. The photo should be colored by stacking the channels [1].

First of all, I wanted to see unaligned results. Because I couldn't imagine how images should be looked like. I splitted images according to height/3. And I assigned bgr channels. When the channels were merged, result images should be unaligned results. According to unaligned results, I had an idea about how images should look.

For aligning images part, I choosed NCC. NCC standardizes the picture being moved, and afterward takes the internal result of that picture and the one it is being contrasted with as a vector. The bigger this worth, the closer the two pictures are, and consequently, the better the match [2]. NCC has its own formula. I applied this formula in my NCC method. I think aligning method more important than NCC's own method. Also I use lots of methods which are already in libraries like numpy. These methods are linspace(), roll(), square(), sum(), sqrt() and mean(). Math functions were used for NCC. I gave two images to NCC method but in aligning method designed to travel all pixels in images. So I also used in this part roll() method and linspace() method. Linspace() function takes x and y coordinates and keeps producing number between this interval. I gave -30 for starting, the method added 1 until 30 for end. Roll() method used for shifting actually. With NCC, I had new variables and I had to match them with the old variables along given axis.

When I aligned images with NCC, I liked the results. I thought results are fine but they can be more than fine. So I cropped images' edges -black borders-. I used matplotlib for plotting. I did not save the images.

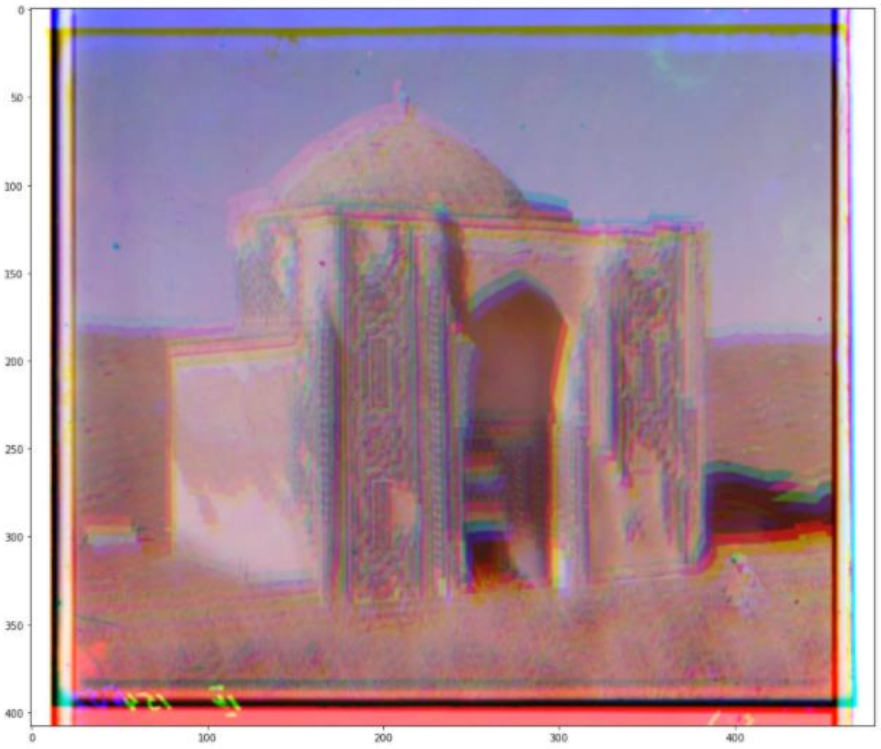
For cropping I chose two coordinates, these were x and y. I choose them randomly. I am not sure, this is the correct way but results are as required. I assumed size of images are MxM when images divided by three. So, I cropped equal amount from top, down, left and right. After cropping, the images looked much better. I added extra aligned results for some images without cropping.

Consequently, my idea is images which are not very complicated give good results. Also image metrics are very important for aligning. Without NCC, images looked like shifted. It means they can not stack properly. Also, with black borders aligned images are similar to unaligned images.

Unaligned Results Without Cropping

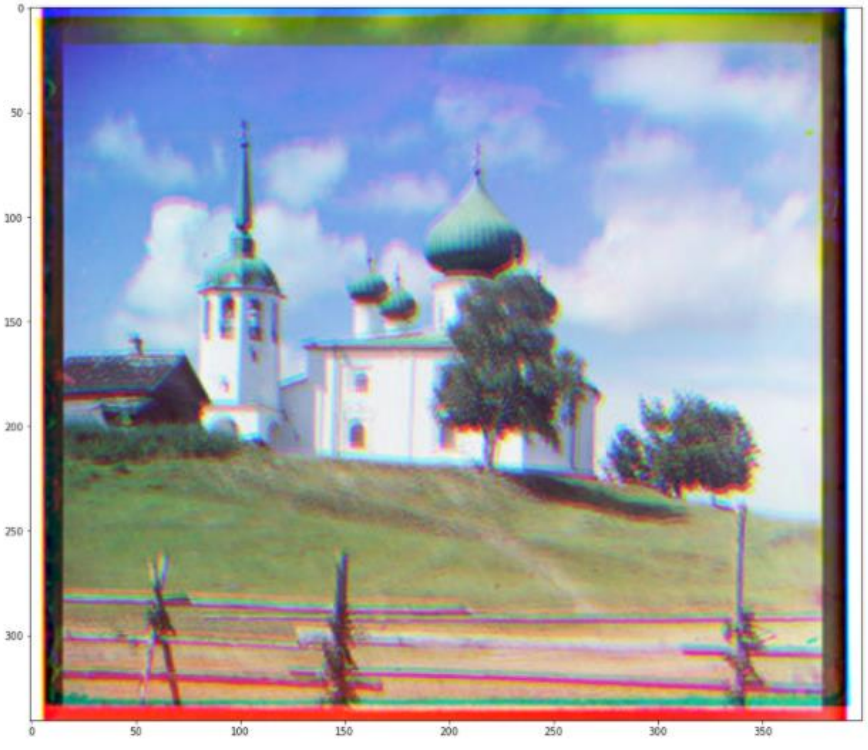

0
50
100
150
200
250
300
0
50
100
150
200
250
300
350
400


0
50
100
150
200
250
300
0
50
100
150
200
250
300
350


0
50
100
150
200
250
300
350
400
0
100
200
300
400


0
50
100
150
200
250
300
350
400
0
100
200
300
400


0
50
100
150
200
250
300
0
50
100
150
200
250
300
350

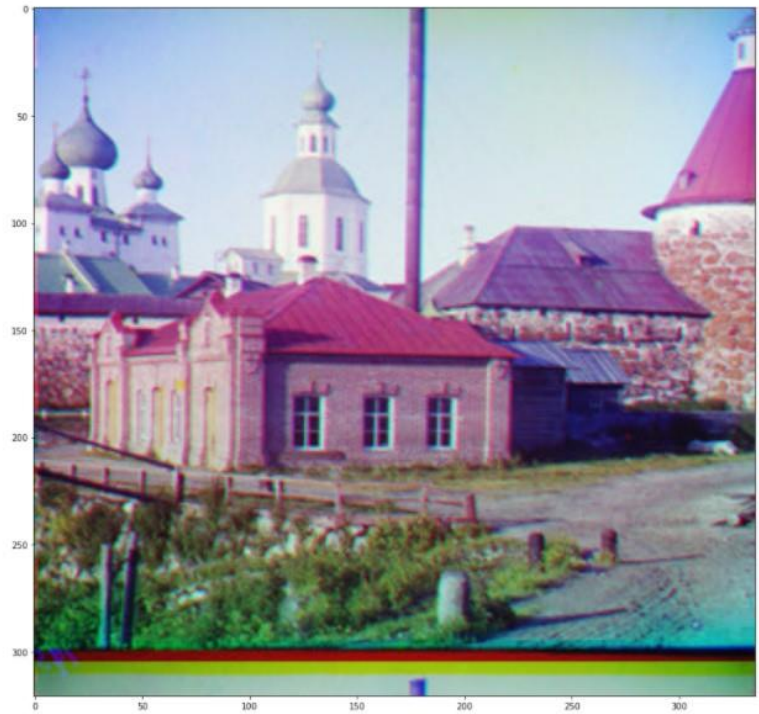

0
50
100
150
200
250
300
0
50
100
150
200
250
300
350

Aligned Results Using NCC With Cropping

```
GreenNCC: [-14, 2]
RedNCC: [-28, 4]

<matplotlib.image.AxesImage at 0x1deda2e2ca0>

<Figure size 432x288 with 0 Axes>
```


```
GreenNCC: [-16, 0]
RedNCC: [-30, 0]

<matplotlib.image.AxesImage at 0x1deda5f5e80>

<Figure size 432x288 with 0 Axes>
```

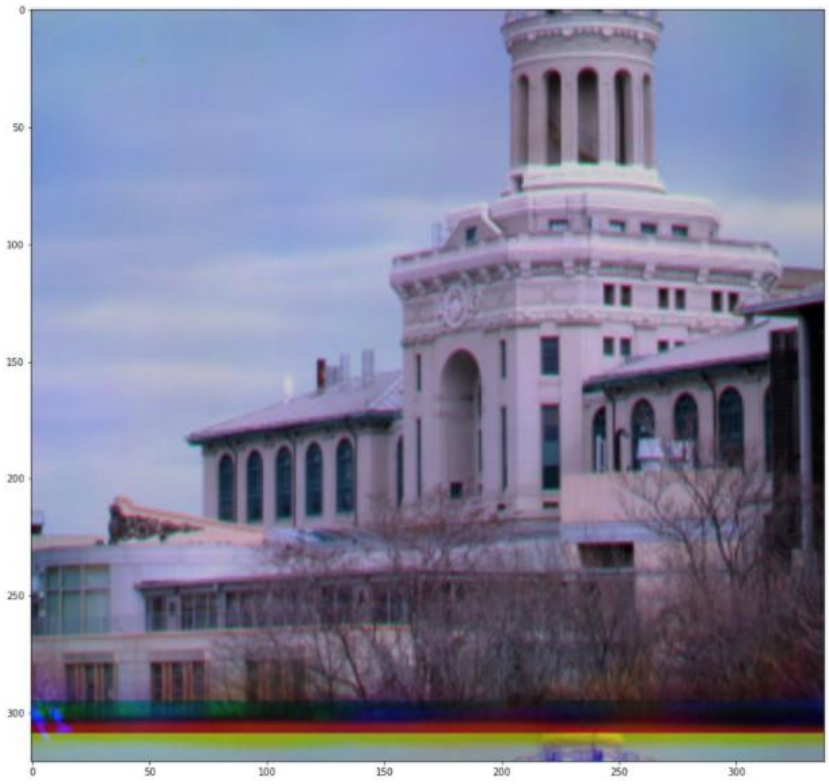
```
GreenNCC: [-16, 0]
RedNCC: [-26, 2]

<matplotlib.image.AxesImage at 0x1dedaf8f2b0>

<Figure size 432x288 with 0 Axes>
```



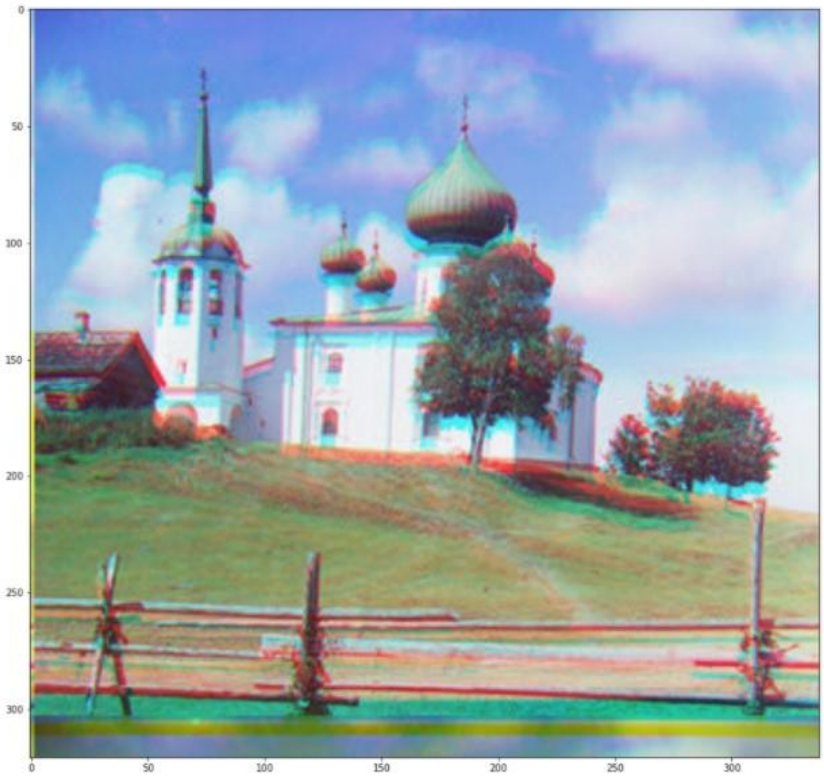

```
GreenNCC: [-18, 4]
RedNCC: [-28, 6]

<matplotlib.image.AxesImage at 0x1dedcd36370>

<Figure size 432x288 with 0 Axes>
```

```
GreenNCC: [-16, 4]
RedNCC: [-26, 8]

<matplotlib.image.AxesImage at 0x1dedece3970>

<Figure size 432x288 with 0 Axes>
```

```
GreenNCC: [-12, 0]
RedNCC: [-26, 0]

<matplotlib.image.AxesImage at 0x1dedf9478e0>

<Figure size 432x288 with 0 Axes>
```



```
GreenNCC: [-18, 2]
RedNCC: [-30, 2]

<matplotlib.image.AxesImage at 0x1dedfa92a00>

<Figure size 432x288 with 0 Axes>
```




```
GreenNCC: [-14, 6]
RedNCC: [-26, 8]

<matplotlib.image.AxesImage at 0x1dee1883520>

<Figure size 432x288 with 0 Axes>
```

Aligned Results Using NCC Without Cropping

```
GreenNCC: [2, 0]
RedNCC: [4, -2]
```

```
<matplotlib.image.AxesImage at 0x2c8427f5520>
```

```
<Figure size 432x288 with 0 Axes>
```


```
GreenNCC: [6, 0]
RedNCC: [12, 0]
```

```
<matplotlib.image.AxesImage at 0x2c842518b80>
```

```
<Figure size 432x288 with 0 Axes>
```

I did not add all aligned images without cropping.

References

1) Internet: Anonim. (nd) *About the Prokudin-Gorskii Collection.* Web: https://www.loc.gov/pictures/collection/prok/

2)Internet: Chen E.(nd) *Image Alignment.* Web: http://www.emileechen.com/projects/image alignment/