レポート提出票

科目名: 情報工学実験3

実験課題名: 課題4 画像変換

実施日: 2021年 4月 22日

学籍番号: 4619055

氏名: 辰川力駆

共同実験者: ＿＿＿＿＿＿ ＿＿＿＿＿＿

＿＿＿＿＿＿ ＿＿＿＿＿＿

＿＿＿＿＿＿ ＿＿＿＿＿＿

＿＿＿＿＿＿ ＿＿＿＿＿＿

1　実験の要旨

実験環境を整備した上で、Jupyther notebook で Python 言語とそのライブラリの使い方を理解し、画像処理の基礎を学習する。

2　実験の目的

画像変換の処理を題材に、画像処理のプログラミングと評価を通じて、画像処理の基本的な考え方を理解することを目的とする。

第 1 回目の実験では、濃淡変換の実装と実験を通じて、Python 言語とライブラリの使用方法、画像処理の基本を理解することを目的とする。

3　課題 1

3.1　実験方法

次のようなプログラムを作成する。ただし、指定の領域は、2 点 (64,25),(192,220) を対角とする矩形である。

1. 画像ファイル data/Mandrill.bmp を読み込み画像 im1 とする。
2. 画像 im1 の指定の領域をグレースケールにした画像 im2 を作成する。
3. 画像 im1 の指定の領域を上下反転した画像 im3 を作成する。
4. 画像 im1、im2、im3 にタイトルを付けて、横に並べて表示する。

3.2　実験結果

作成したプログラムは付録のソースコード 1 に載せた。このプログラムを実行すると、以下のようになった。

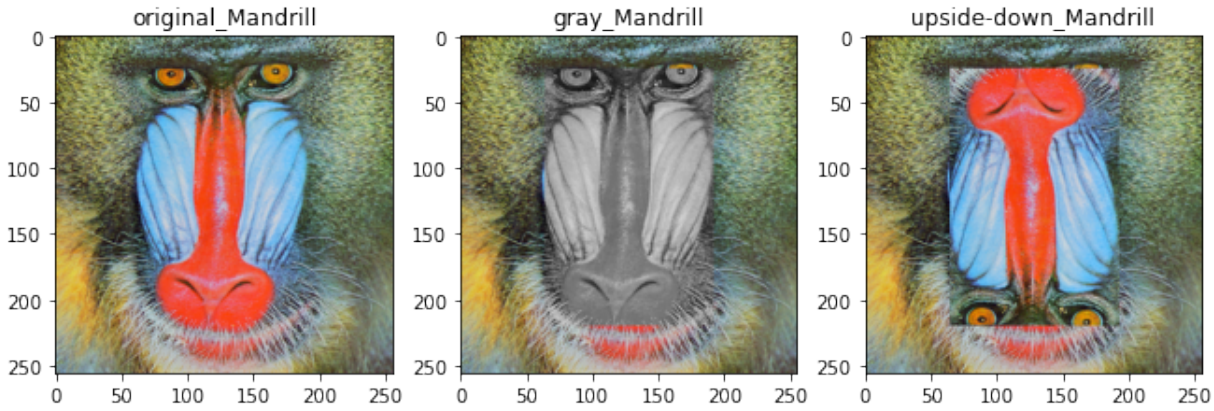

図 1: 課題 1 の実行結果

3.3 検討・考察

im1 の一部を反転させるのは、容易であったが (upside-down_Mandrill)、im1 の一部をグレースケールにするには、一点一点を変更していく (プログラム 11 行目から 13 行目) ので、高速に実行するのは難しいと考える。

4 課題 2

4.1 実験方法

自分自身で撮影した画像 f に対して、以下を実行する。

1. ガンマ値 γ を変化させながら、画像 f にガンマ補正を施した結果を求める。
2. ガンマ補正によるコントラスト向上の効果を確認する。そして、見た目が良くなるガンマ値を選択する。
3. 2 つ以上のガンマ値に対して補正結果を図示する。

4.2 実験結果

ガンマ値 γ を変化させながら、画像 f にガンマ補正を施した結果を下記に示す。

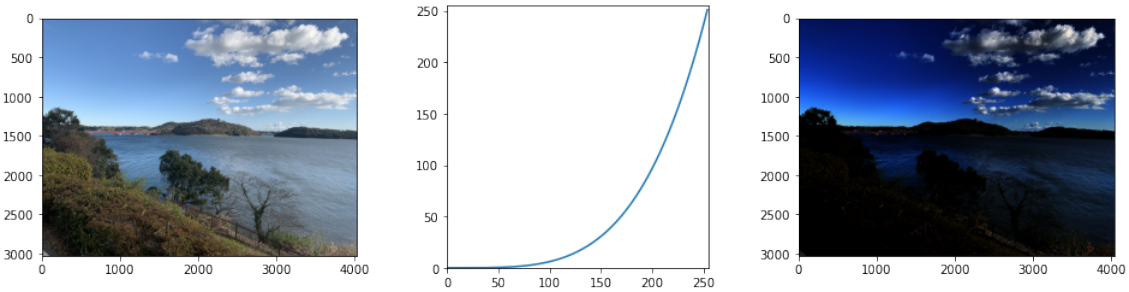

図 2: $\gamma = \frac{1}{4}$ の場合

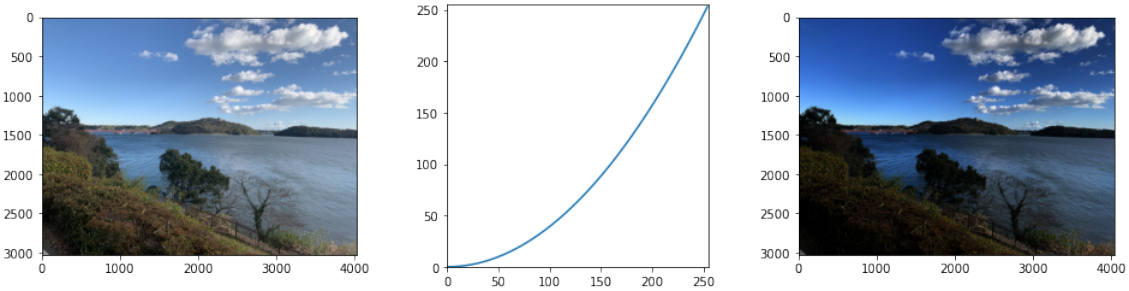

図 3: $\gamma = \frac{1}{2}$ の場合

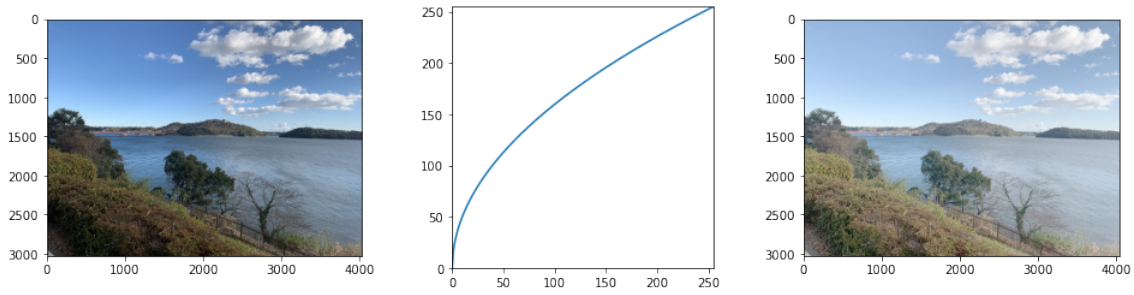

図 4: $\gamma = 2$ の場合

また、作成するために用いたプログラムは付録のソースコード 2 に載せた。

4.3 検討・考察

実験の結果より、γ の値を 1 より小さくすることで全体が暗くなり、γ の値を 1 より大きくすることで全体が明るい画像となった。

今回の画像に関しては元が明るい風景なので、γ の値を図 4 のように明るくするのではなく、図 2,3 のように暗くするほうが見た目が良いと考える。γ の値を小さくしすぎると海の色が真っ黒になってあまり見えなくなるので主観的に一番見た目が良いと思うものは $\gamma = \frac{1}{2}$(図 3) である。

図 3 は、元の画像でぼんやりしている雲をくっきりしているので見た目が良くなっている。

5 課題 3

5.1 実験方法

指定した 2 枚の画像 f、f' について以下の課題に取り組む。ここで、画像 f' は未知の階調変換関数 $g(x)$ による濃淡変換によって生成される画像、すなわち、$f'(i,j) = g(f(i,j))$ であるとする。

1. 画像 f, f' を、それぞれ画像ファイル exercise/himeji_ex3.jpg,exercise/himeji_ex3_G?.png から読み込み、画像の幅・高さを求める。(”G?” には自分のグループの数字)
2. 画像 $f(i,j)$, $f'(i,j)$ の対応する画素値の散布図を作成する。
3. 階調変換関数 $g(x)$ を推測し、数式として表す。
4. 3 でで推測した階調変換関数 $g(x)$ によって画像 f から画像 f' が生成されること確かめる。具体的には画像 $f'(i,j)$ と階調変換関数によって画像 f から生成された画像 $g(f(i,j))$ の平均二乗誤差 (MSE: Mean Squared Error)

$$\mathrm{MSE} = \frac{1}{WH} \sum_{j=0}^{H-1} \sum_{i=0}^{W-1} \left(f'(i,j) - g(f(i,j)) \right)^2$$

が 1 以下であることを確認する。ただし、画素値 $f'(i,j)$, $f(i,j)$ の最大値、最小値は 255、0 とする。また、カラー画像の場合の MSE は以下の式で表される。

$$\mathrm{MSE} = \frac{1}{|C|WH} \sum_{c \in C} \sum_{j=0}^{H-1} \sum_{i=0}^{W-1} \left(f'(i,j,c) - g(f(i,j,c)) \right)^2$$

ここで，$C = \{R, G, B\}$ はチャンネルを表す。

5. なぜ結果が完全に一致しないか、すなわち MSE $\neq 0$ となる理由を考察する。

5.2 実験結果

1. 自分のグループは G4 なので G4 として画像を読み込むと、画像の幅は 413、高さは 310 となった。

2. 画像 $f(i,j)$、$f'(i,j)$ の画素値の散布図を作成すると、以下のようになった。

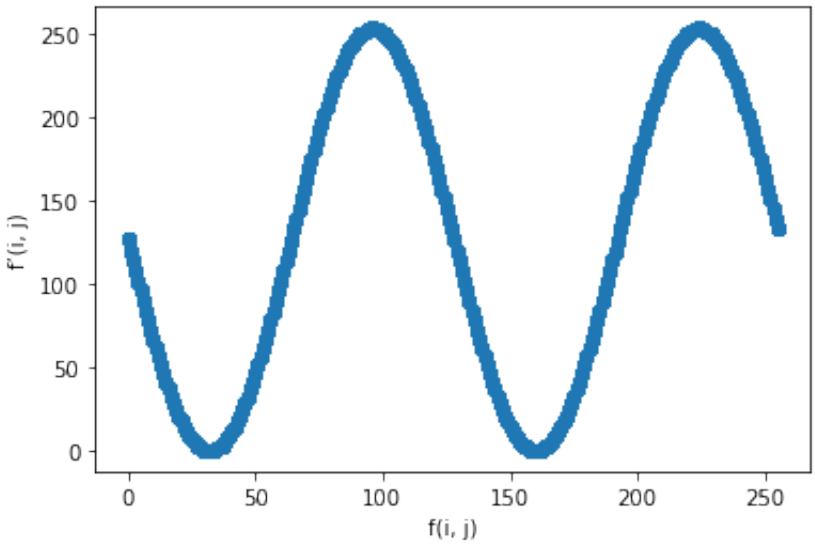

図 5: 画像 $f(i,j)$、$f'(i,j)$ の散布図

3. 推測した階調変換関数 $g(x)$ を以下に示す。

$$g(x) = -127.2 \sin \frac{\pi x}{64} + 126.6$$

4. 推測した階調変換関数 $g(x)$ から画像を生成したものを下に示す。

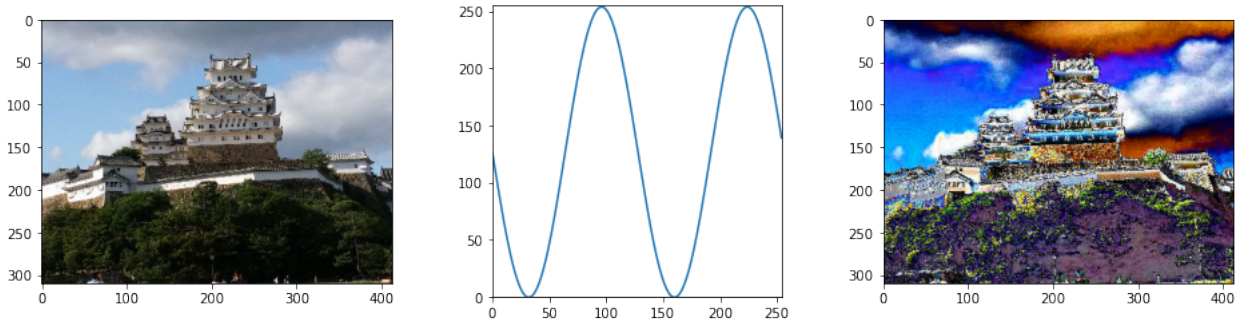

図 6: 推測した $g(x)$ から生成した結果

また、MSE を計算すると、約 0.0979 となって 1 以下であることがわかる。

5.3 検討・考察

推測した階調変換関数を用いて MSE を計算すると、1 以下になったが、0 になることはない。これは、散布図から推測したことによることと、sin によって近似したことが挙げられる。実際には sin 関数を用いられて作られた場合があるが、見た目だけで容易に sin 関数として近似するのが一致しない理由であると考える。

完全一致を目指すならば、散布図で推測するべきではないと考える。

6 まとめ

本実験では、Jupyther notebook で Python 言語とそのライブラリの使い方を理解し、画像処理の基礎を学習することができた。

課題では、画像の濃淡変換を学び、ガンマ補正によって明るさの調整ができることを学んだ。また、散布図から階調変換関数の推定を行った。

A 付録

ソースコード 1: kadai1

```
#4619055
from skimage.io import imread, imsave
import numpy as np
import matplotlib.pyplot as plt
plt.figure(figsize=(11,7))

im1 = imread("data/Mandrill.bmp") # カラー画像として読み込み
im2 = imread("data/Mandrill.bmp")
im3 = imread("data/Mandrill.bmp")

for x in range(64,192):
    for y in range(25,220):
        im2[y,x] = np.dot(im2[y,x],[0.299, 0.587, 0.114])

im3[25:220,64:192]= np.flipud(im3[25:220,64:192])

plt.subplot(1, 3, 1)
plt.imshow(im1)
plt.title("original_Mandrill")
plt.subplot(1, 3, 2)
plt.imshow(im2)
plt.title("gray_Mandrill")
plt.subplot(1, 3, 3)
plt.imshow(im3)
plt.title("upside-down_Mandrill")
plt.show()
```

ソースコード 2: kadai2

```
#4619055
import numpy as np
from skimage.io import imread, imsave
from skimage.color import rgb2gray, gray2rgb
import matplotlib.pyplot as plt
%matplotlib inline

def show_transform(im1, im2, h):
    """ 濃淡変換の入力画像, トーンカーブ, 出力画像を描画
        im1 : 入力画像
        im2 : 出力画像
        h : 階調変換関数
    """
    plt.figure(figsize=(16, 4))
    plt.subplot(1, 3, 1)
    plt.imshow(im1.astype(np.uint8), cmap="gray")

    plt.subplot(1, 3, 2)
    I = np.arange(0, 255)
    plt.xlim(0, 255); plt.ylim(0, 255); plt.gca().set_aspect(1.0)
```

```
        try:
            y = h(I)
            plt.plot(I, y)
        except:
            x = np.vstack([I, I, I])
            y = h(x.T)
            plt.plot(I, y[0,:,0], 'r', label='R')
            plt.plot(I, y[0,:,1], 'g', label='G')
            plt.plot(I, y[0,:,2], 'b', label='B')
            plt.legend()

        plt.subplot(1, 3, 3)
        plt.imshow(im2.astype(np.uint8), cmap="gray")

    def gamma(gam, I): return 255 * (I / 255) ** (1/gam)
    def gamma14(I): return gamma(1/4, I)
    def gamma12(I): return gamma(1/2, I)
    def gamma2(I): return gamma(2, I)

    f = imread("data/kadai2.jpg")

    show_transform(f, gamma14(f), gamma14)
    show_transform(f, gamma12(f), gamma12)
    show_transform(f, gamma2(f), gamma2)
```

ソースコード 3: kadai3

```
    #4619055
    import numpy as np
    from skimage.io import imread, imsave
    from skimage.color import rgb2gray, gray2rgb
    import matplotlib.pyplot as plt
    %matplotlib inline

    def show_transform(im1, im2, h):
        """ 濃淡変換の入力画像, トーンカーブ, 出力画像を描画
            im1 : 入力画像
            im2 : 出力画像
            h : 階調変換関数
        """
        plt.figure(figsize=(16, 4))
        plt.subplot(1, 3, 1)
        plt.imshow(im1.astype(np.uint8), cmap="gray")

        plt.subplot(1, 3, 2)
        I = np.arange(0, 255)
        plt.xlim(0, 255); plt.ylim(0, 255); plt.gca().set_aspect(1.0)
        try:
            y = h(I)
            plt.plot(I, y)
        except:
            x = np.vstack([I, I, I])
            y = h(x.T)
```

```
27              plt.plot(I, y[0,:,0], 'r', label='R')
28              plt.plot(I, y[0,:,1], 'g', label='G')
29              plt.plot(I, y[0,:,2], 'b', label='B')
30              plt.legend()
31
32          plt.subplot(1, 3, 3)
33          plt.imshow(im2.astype(np.uint8), cmap="gray")
34
35      f = imread("exercise/himeji_ex3.jpg")
36      f_prime = imread("exercise/himeji_ex3_G4.png")
37
38      height, width,C =f.shape
39      height_prime, width_prime, _ = f_prime.shape
40      print("f␣:幅 (" + str(width) +"),␣高さ (" + str(height)+ ")")
41      print("f':幅 (" + str(width_prime) + "),␣高さ (" + str(height_prime)+ ")")
42
43      x_point = f.flatten()
44      y_point = f_prime.flatten()
45      plt.scatter(x_point, y_point)
46      plt.xlabel("f(i,␣j)")
47      plt.ylabel("f'(i,␣j)")
48      plt.show()
49
50      def g(x):
51          return 127.2 * −np.sin(x * np.pi / 64) + 126.6
52
53      show_transform(f, g(f), g)
54      mse = np.sum((f_prime − g(f))** 2) / (C * width * height)
55      print("MSE:", mse)
```