```python
import numpy as np
import pandas as pd
import cv2
from matplotlib import pyplot as plt
from pylab import imread
from skimage.color import rgb2gray
```

```python
def imshows(ImageData, LabelData, rows, cols, gridType = False):
    # Convert ImageData and LabelData to List
    ImageArray = list(ImageData)
    LabelArray = list(LabelData)
    if(rows == 1 & cols == 1):
        fig = plt.figure(figsize=(20,20))
    else:
        fig = plt.figure(figsize=(cols*8,rows*5))

    for i in range(1, cols * rows + 1):
        fig.add_subplot(rows, cols, i)
        image = ImageArray[i - 1]
        if (len(image.shape) < 3):
            plt.imshow(image, plt.cm.gray)
            plt.grid(gridType)
        else:
            plt.imshow(image)
            plt.grid(gridType)
        plt.title(LabelArray[i - 1])
    plt.show()
```

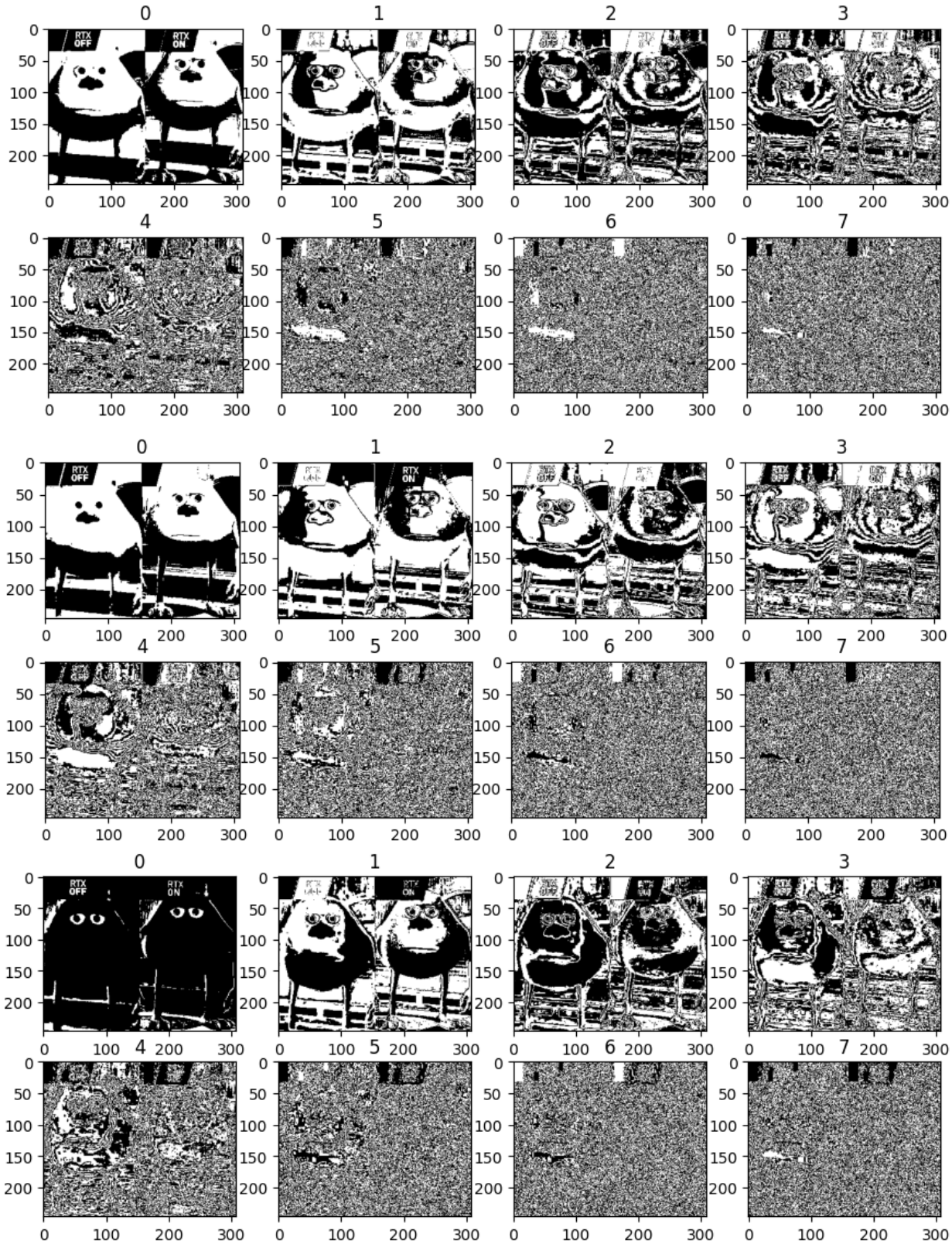
```python
def ShowOneImage(IM):
    imshows([IM], ["Image"], 1, 1)
def ShowTwoImages(IM1, IM2):
    imshows([IM1, IM2], ["Image 1","Image 2"], 1, 2)
def ShowThreeImages(IM1, IM2, IM3):
    imshows([IM1, IM2, IM3], ["Image 1","Image 2", "Image 3"], 1, 3)
def ShowListImages(listImage, row, col):
    listCaption = []
    for i in range(len(listImage)):
        listCaption.append(str(i))
    imshows(listImage,listCaption,row,col)
# Read Image
image_color = imread("Sample02/mike.jpg")
# Convert Image into Gray
image_gray = cv2.cvtColor(image_color, cv2.COLOR_RGB2GRAY)
# Convert Image into HSV color spaces
image_hsv = cv2.cvtColor(image_color, cv2.COLOR_BGR2HSV)
```

```python
# Show Information of Image
def ShowImageInfo(IM):
    Image = IM.copy()
    Width = Image.shape[1]
    Height = Image.shape[0]
    Channel = len(Image.shape)
    print("Width: ", Width, " Height: ", Height, " Channel: ", Channel)
    if(Channel == 2):
        print("This is Gray Image.")
        print("Min Intensity: ", Image.min(), " Max Intensity: ", Image.max())
    else:
        print("This is Color Image.")
        print("Red - Min Intensity: ", Image[:,:,0].min(), " Max Intensity: ", Image[:,:,0].max())
        print("Green - Min Intensity: ", Image[:,:,1].min(), " Max Intensity: ", Image[:,:,1].max())
        print("Blue - Min Intensity: ", Image[:,:,2].min(), " Max Intensity: ", Image[:,:,2].max())
        print('\n')

ShowColorImageInfo(image_color)
```

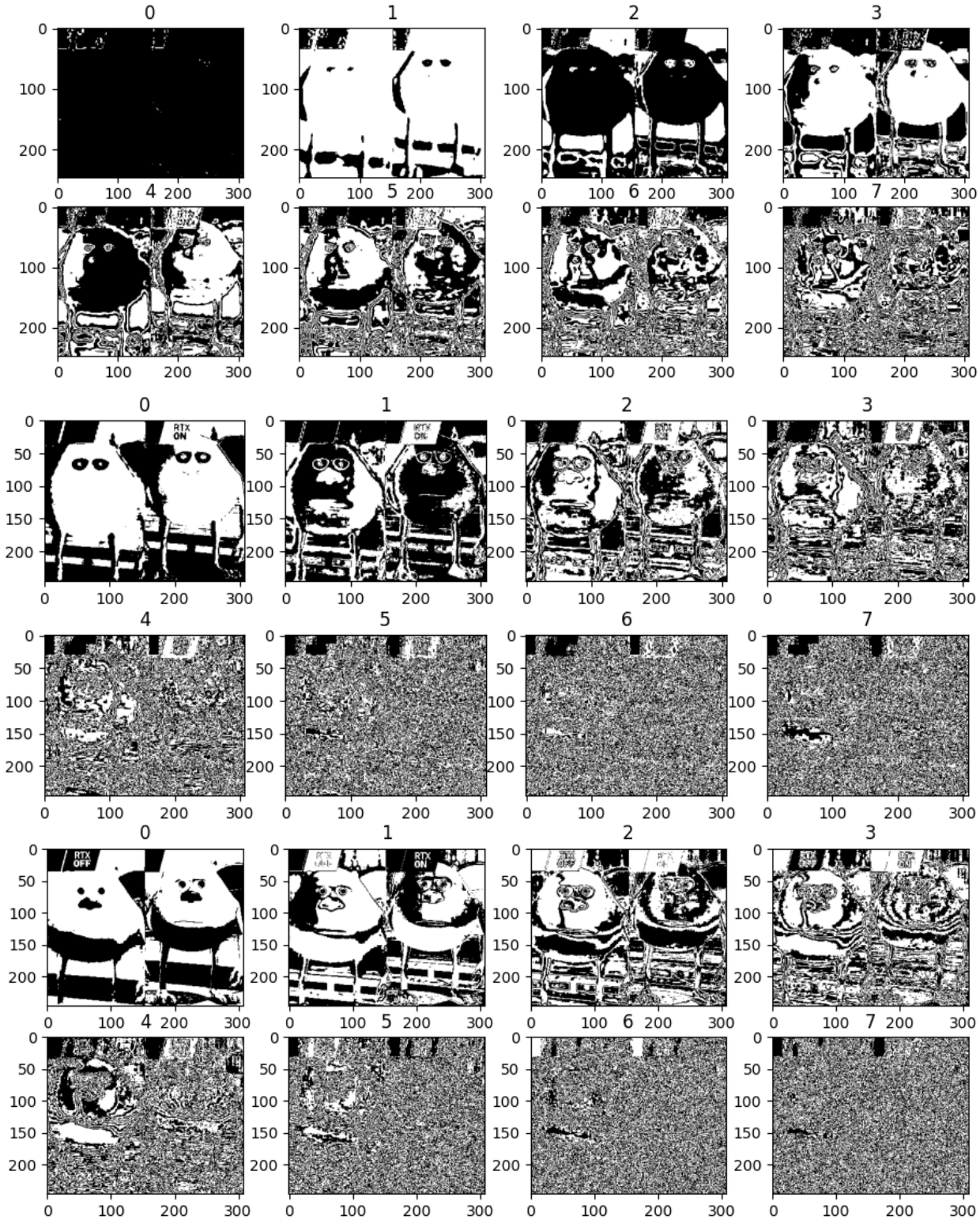
```
Width:  308  Height:  246  Channel:  3
This is Color Image.
Red - Min Intensity:  0  Max Intensity:  255
Green - Min Intensity:  0  Max Intensity:  255
Blue - Min Intensity:  0  Max Intensity:  255
```

```python
def intToBitArray(img) :
    row, col = img.shape
    list = []
    for i in range(row):
        for j in range(col):
            list.append (np.binary_repr( img[i][j], width=8 ) )
    return list
def bitplane(bitImgVal , img1D ):
    bitList = [ int( i[bitImgVal] ) for i in img1D]
    return bitList
def GetBitImage(index, image2D):
    ImageIn1D = intToBitArray(image2D)
    Imagebit = np.array( bitplane(index, ImageIn1D) )
    Imagebit = np.reshape(Imagebit, image2D.shape)
    return Imagebit
def GetAllBitImage(image2D):
    image2D_Bit = list()
    for i in range(8):
        image2D_Bit.append(GetBitImage(i, image2D))
    return image2D_Bit
# Get 8 Mask Image corresponding to 8 bit of RBG Image and HSV Image
for i in range(3):
    image2D_Bit = GetAllBitImage(image_color[:,:,i])
    ShowListImages(image2D_Bit, 2, 4)
for i in range(3):
    image2D_Bit = GetAllBitImage(image_hsv[:,:,i])
    ShowListImages(image2D_Bit, 2, 4)
```

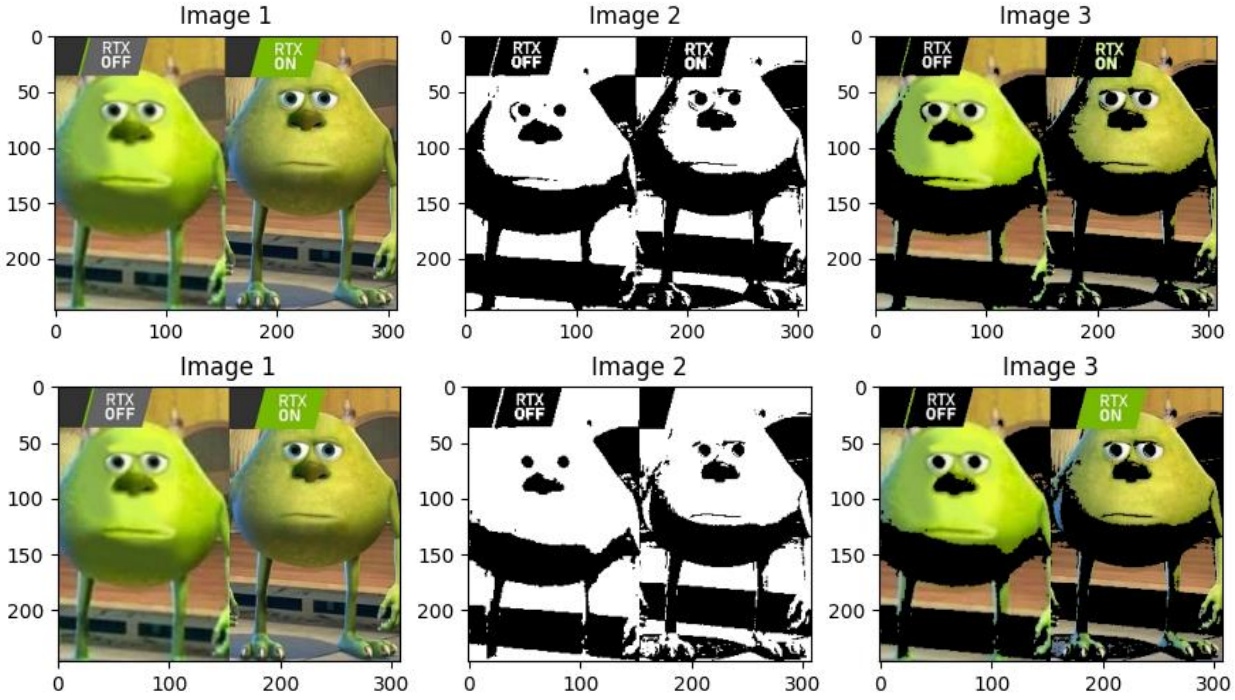
RGB



HSV

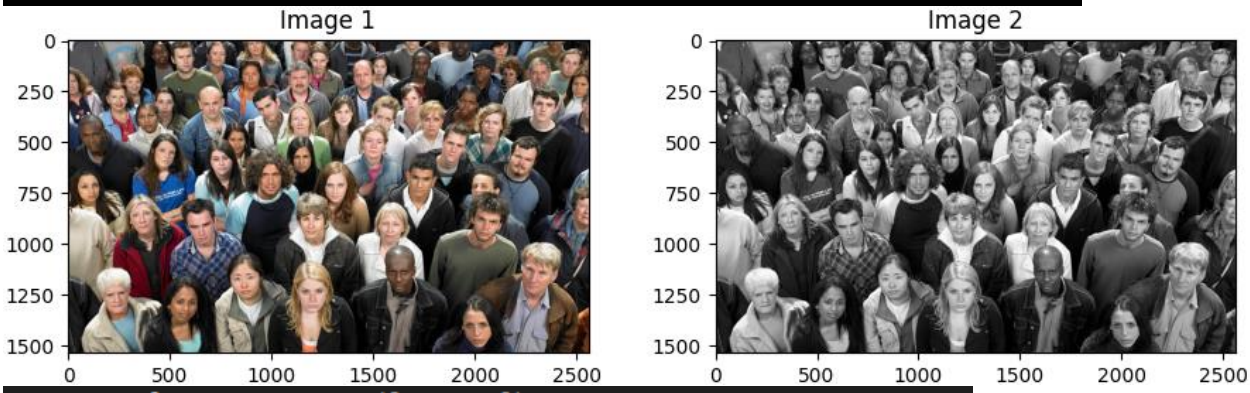


```python
def SegmentColorImageByMask(IM, Mask):
    Mask = Mask.astype(np.uint8)
    result = cv2.bitwise_and(IM, IM, mask = Mask)
    return result

image2D_Bit = GetAllBitImage(image_color[:,:,0])
Mask01 = image2D_Bit[0]
Mask01_segment = SegmentColorImageByMask(image_color, Mask01)
ShowThreeImages(image_color, Mask01, Mask01_segment)

image2D_Bit = GetAllBitImage(image_hsv[:,:,2])
Mask02 = image2D_Bit[0]
Mask02_segment = SegmentColorImageByMask(image_color, Mask02)
ShowThreeImages(image_color, Mask02, Mask02_segment)
```



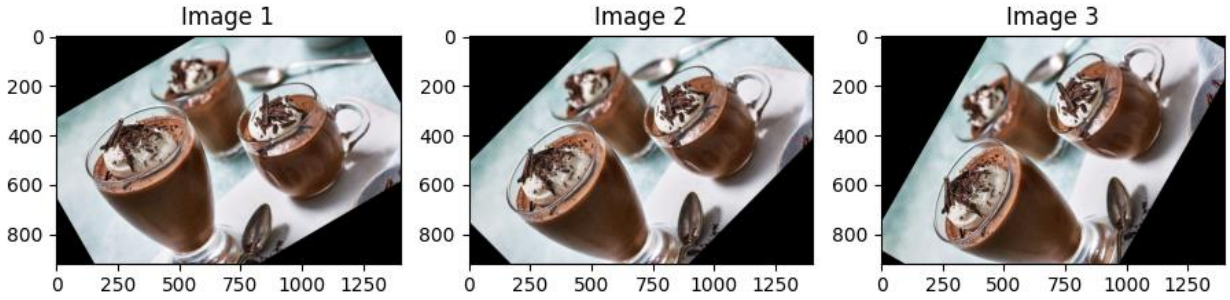
```
174     def InputColorImage():
175         name = input("Input Color Image File Name and its extension: Sample02/")
176         name = "Sample02/" + name
177         image_color = imread(name)
178         image_gray = cv2.cvtColor(image_color, cv2.COLOR_RGB2GRAY)
179         ShowTwoImages(image_color, image_gray)
180
181     InputColorImage()
```

```
Input Color Image File Name and its extension: Sample02/
Input Color Image File Name and its extension: Sample02/people.jpg
```

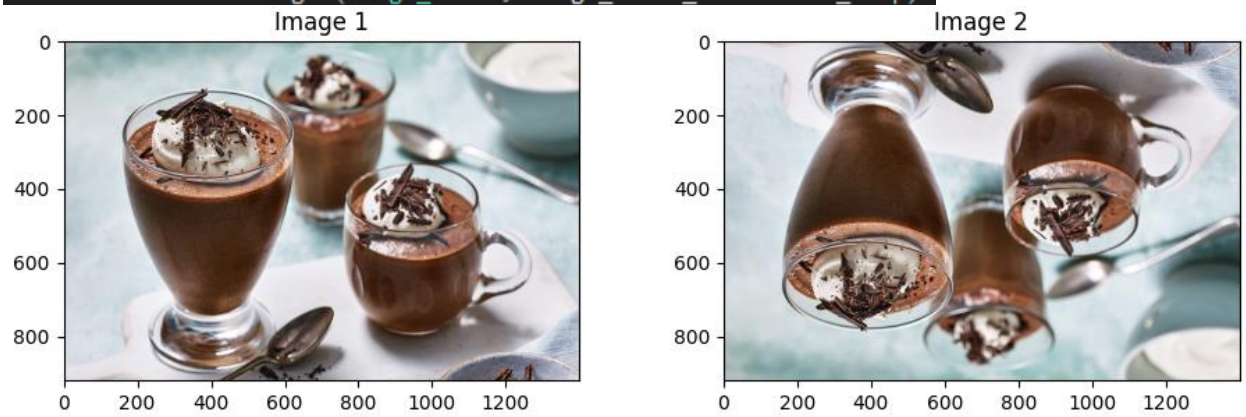


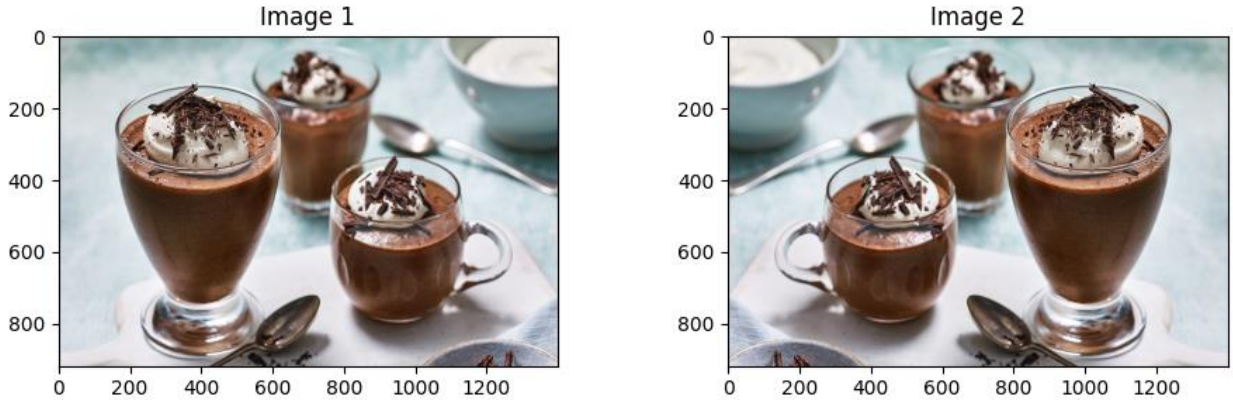

```
175     lower = np.array([175,0,0])
176     upper = np.array([255,255,255])
177     Mask = cv2.inRange(image_color, lower, upper)
178
179     def SegmentColorImageByMask(IM, Mask):
180         Mask = Mask.astype(np.uint8)
181         result = cv2.bitwise_and(IM, IM, mask = Mask)
182         return result
183     Mask_segment = SegmentColorImageByMask(image_color, Mask)
184     ShowThreeImages(image_color, Mask, Mask_segment)
```

```
215     # Read Image
216     image_color = imread("Sample02/choco.jpg")
217     # Convert Image into Gray
218     image_gray = cv2.cvtColor(image_color, cv2.COLOR_RGB2GRAY)
219     # Convert Image into HSV color spaces
220     image_hsv = cv2.cvtColor(image_color, cv2.COLOR_BGR2HSV)
225    from skimage.transform import rotate
226    image_color_rotate_30 = rotate(image_color, 30)
227    image_color_rotate_45 = rotate(image_color, 45)
228    image_color_rotate_60 = rotate(image_color, 60)
229    ShowThreeImages(image_color_rotate_30, image_color_rotate_45, image_color_rotate_60)
```

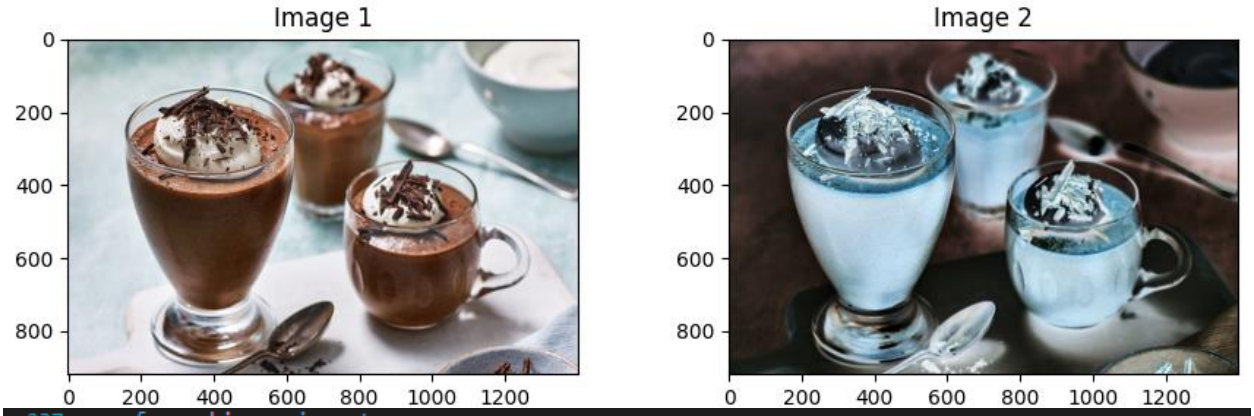



```
230     image_color_vertical_flip = image_color[::-1, :]
231     ShowTwoImages(image_color, image_color_vertical_flip)
232     image_color_horizontal_flip = image_color[:, ::-1]
233     ShowTwoImages(image_color, image_color_horizontal_flip)
```

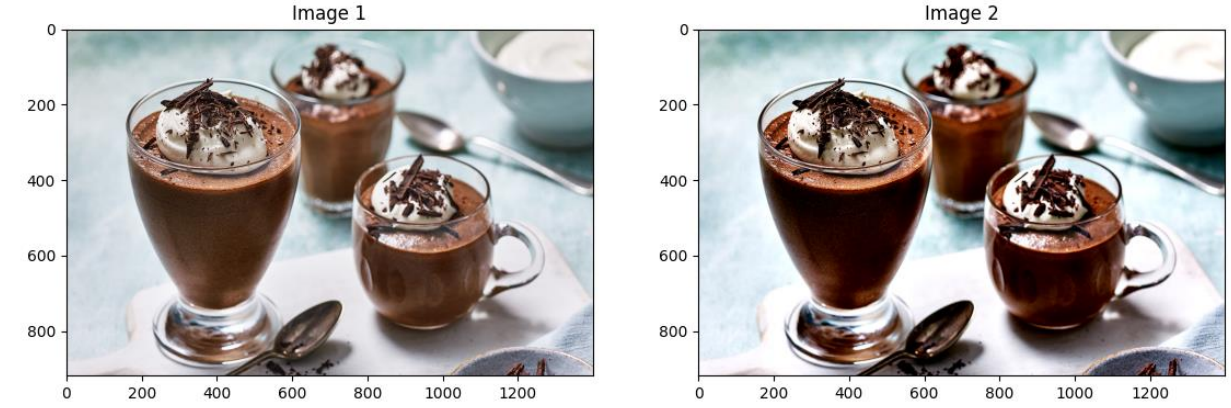




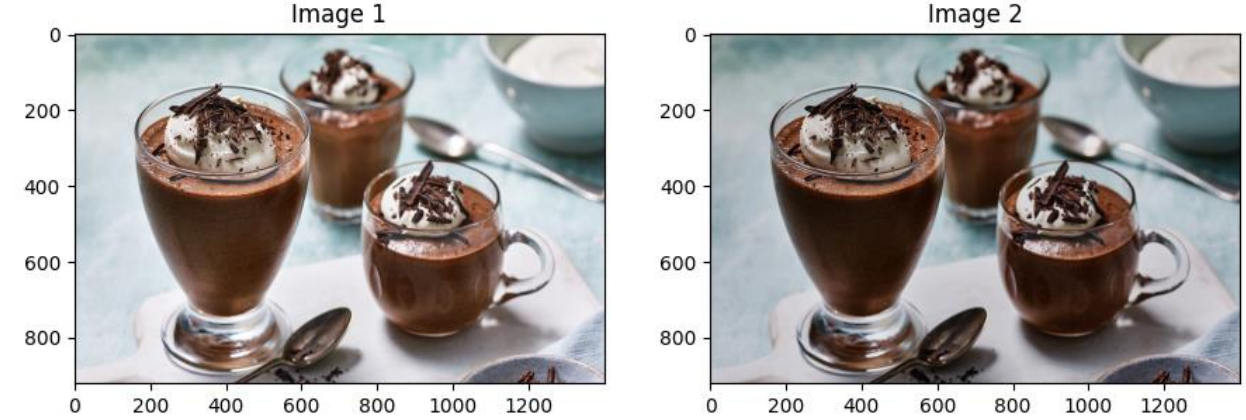
```
234     from skimage import util
235     image_color_inversion = util.invert(image_color)
236     ShowTwoImages(image_color, image_color_inversion)
```




```
237     from skimage import exposure
238     v_min, v_max = np.percentile(image_color, (10, 90))
239     better_contrast_image = exposure.rescale_intensity(image_color, in_range=(v_min, v_max))
240     ShowTwoImages(image_color, better_contrast_image)
```




```
242     adjusted_gamma_image = exposure.adjust_gamma(image_color, gamma=1.3, gain=0.9)
243     ShowTwoImages(image_color, adjusted_gamma_image)
```



```
245     log_correction_image = exposure.adjust_log(image_color)
246     ShowTwoImages(image_color, log_correction_image)
```