Creating a dataset of Recolored Images and Original Images for training the Model

In [1]:

```python
import numpy as np
import cv2
```

Color Transfer Algorithm

- Recoloring is done by transerring the color properties of one image to other
- Input a **Source** and **Target** image. The color space of **Source** image is transferred to the color space of **Target** image.

In [2]:

```python
def color_transfer(source, target):
    # convert color space from BGR to L*a*b color space
    ## L* for the lightness from black to white, a* from green to red, and b* from blue
    # note - OpenCV expects a 32bit float rather than 64bit
    source = cv2.cvtColor(source, cv2.COLOR_BGR2LAB).astype("float32")
    target = cv2.cvtColor(target, cv2.COLOR_BGR2LAB).astype("float32")

    # compute color stats for both images
    (lMeanSrc, lStdSrc, aMeanSrc, aStdSrc, bMeanSrc, bStdSrc) = image_stats(source)
    (lMeanTar, lStdTar, aMeanTar, aStdTar, bMeanTar, bStdTar) = image_stats(target)

    # split the color space
    (l, a, b) = cv2.split(target)

    # substarct the means from target image
    l -= lMeanTar
    a -= aMeanTar
    b -= bMeanTar

    # scale by the standard deviation
    l = (lStdTar/lStdSrc)*l
    a = (aStdTar/aStdSrc)*a
    b = (bStdTar/bStdSrc)*b

    # add the source mean
    l += lMeanSrc
    a += aMeanSrc
    b += bMeanSrc

    # clipping the pixels between 0 and 255(0 denotes black and 255 denotes white)
    l = np.clip(l, 0, 255)
    a = np.clip(a, 0, 255)
    b = np.clip(b, 0, 255)

    # merge the channels
    transfer = cv2.merge([l, a, b])

    # converting back to BGR
    transfer = cv2.cvtColor(transfer.astype("uint8"), cv2.COLOR_LAB2BGR)

    return transfer
```

In [3]:

```python
def image_stats(image):
    # compute mean and standard deviation of each channel
    (l, a, b) = cv2.split(image)
    (lMean, lStd) = (l.mean(), l.std())
    (aMean, aStd) = (a.mean(), a.std())
    (bMean, bStd) = (b.mean(), b.std())

    return (lMean, lStd, aMean, aStd, bMean, bStd)
```

In [4]:

```python
def show_image(title, image, width=720):
    r = width/float(image.shape[1])
    dim = (width, int(image.shape[0]*r))
    resized = cv2.resize(image, dim, interpolation=cv2.INTER_AREA)

    cv2.imshow(title, resized)
```

- Viewing a sample image to demonstrate the result of recoloring

In [6]:

```python
source = cv2.imread("dataset/source/source (1).jpg")
target = cv2.imread("dataset/target/target (1).jpg")


# transfer of color
transfer = color_transfer(source, target)

# display of image
show_image("Source", source)
show_image("Target", target)
show_image("Transfer", transfer)
cv2.waitKey(0)
```

Out[6]:

```
-1
```

- Applying Color transfer to some of the images(including both indoor and outdoor images) taken from VOC PASCAL 2012 dataset

In [5]:

```
from os import listdir
from os.path import isfile, join
import numpy
import cv2
import os

mypath1='dataset/source/'
mypath2='dataset/target/'

onlyfiles1 = [ f for f in listdir(mypath1) if isfile(join(mypath1,f)) ]
onlyfiles2 = [ f for f in listdir(mypath2) if isfile(join(mypath2,f)) ]

print(len(onlyfiles1))
print(len(onlyfiles2))
images1 = numpy.empty(len(onlyfiles1), dtype=object)
images2 = numpy.empty(len(onlyfiles2), dtype=object)
for n in range(0, len(onlyfiles1)):
    images1[n] = cv2.imread( join(mypath1,onlyfiles1[n]) )
    images1[n] = cv2.cvtColor(images1[n], cv2.COLOR_BGR2RGB)
    images1[n]=cv2.resize(images1[n],(500,500))

    images2[n] = cv2.imread( join(mypath2,onlyfiles2[n]) )
    images2[n] = cv2.cvtColor(images2[n], cv2.COLOR_BGR2RGB)
    images2[n]=cv2.resize(images2[n],(500,500))
    # transfer of color
    transfer = color_transfer(images1[n], images2[n])
    #write images in a folder
    path = 'dataset/recolorimg'
    path1='dataset/originalimg'
    cv2.imwrite(os.path.join(path , 'img.{}.jpg'.format(n)),transfer)##for labeling the
    cv2.imwrite(os.path.join(path1 , 'img.{}.jpg'.format(n)),images2[n])#for labeling t


    # display of image
    #show_image("Source", images1[n])
    #show_image("Target",images2[n] )
    #show_image("Transfer", transfer)
    cv2.waitKey(0)

```

```
50
50

C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:22: Runtime
Warning: divide by zero encountered in float_scalars
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:23: Runtime
Warning: divide by zero encountered in float_scalars
```

Creating the training dataset

- By applying the above recoloring algorithm we create a dataset of images containing both recolored and original images

In [7]:

```
from os import listdir
from os.path import isfile, join
import numpy
import cv2
import os

mypath1='dataset/source/'
mypath2='dataset/target/'

onlyfiles1 = [ f for f in listdir(mypath1) if isfile(join(mypath1,f)) ]
onlyfiles2 = [ f for f in listdir(mypath2) if isfile(join(mypath2,f)) ]

print(len(onlyfiles1))
print(len(onlyfiles2))
images1 = numpy.empty(len(onlyfiles1), dtype=object)
images2 = numpy.empty(len(onlyfiles2), dtype=object)
for n in range(0, len(onlyfiles1)):
    images1[n] = cv2.imread( join(mypath1,onlyfiles1[n]) )
    images1[n] = cv2.cvtColor(images1[n], cv2.COLOR_BGR2RGB)
    images1[n]=cv2.resize(images1[n],(500,500))

    images2[n] = cv2.imread( join(mypath2,onlyfiles2[n]) )
    images2[n] = cv2.cvtColor(images2[n], cv2.COLOR_BGR2RGB)
    images2[n]=cv2.resize(images2[n],(500,500))
    # transfer of color
    transfer = color_transfer(images1[n], images2[n])
    #write images in a folder
    path2='dataset/trainingset'
    cv2.imwrite(os.path.join(path2 , 'img.{}.jpg'.format(n)),transfer)##for labeling th
    cv2.imwrite(os.path.join(path2 , 'pic.{}.jpg'.format(n)),images2[n])#for labeling t

    #path = 'dataset/transfer'
    #path1='dataset/original'
    #cv2.imwrite(os.path.join(path , 'recolor.{}.jpg'.format(n)),transfer)##for labelin
    #cv2.imwrite(os.path.join(path1 , 'original_color.{}.jpg'.format(n)),images2[n])#fo

    # display of image
    #show_image("Source", images1[n])
    #show_image("Target",images2[n] )
    #show_image("Transfer", transfer)
    #cv2.waitKey(0)

```

```
50
50

C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:22: Runtime
Warning: divide by zero encountered in float_scalars
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:23: Runtime
Warning: divide by zero encountered in float_scalars
```