Ecole Polytechnique de Tunisie

Module : Projet Transversal

MBLLEN : Multi Branch Low Light Enhancement Network

This work is done by : Sofien Resifi & Ahmed Belkhir

Method 1: Efficient cultural heritage image restoration with nonuniform illumination enhancement

Overview

This is the approach done by Marwa JMAL and supervised by M.Wided SOUIDENE and Mr.Rabah ATTIA. this approach was implemented with Matlab.

Our work here is an implementation for Marwa JMAL's approach.

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call dr
ive.mount("/content/drive", force_remount=True).
```

In [ ]:
```
import numpy as np
import pandas as pd
import os
import cv2 as cv
import scipy
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from PIL import Image
import scipy.fft
from PIL import Image
import matplotlib
import colorsys
from matplotlib.colors import hsv_to_rgb
import os
from IPython.display import clear_output
```

In [ ]:

```
def data_distribution(hsv_image):
  S = np.array(1- hsv_image[:,:,1].astype(float)/255)
  V = hsv_image[:,:,2]/255
  overSatV = np.zeros(V.shape)


  underSatV = np.zeros(V.shape)
  sizeMat = V.shape[0]*V.shape[1]

  underSatV = V*((S<=0.1).astype(float))
  overSatV = V*((S>=0.9).astype(float))

  pixelCount_Over, grayLevels_Over=np.histogram(overSatV,bins=256)
  pdfOver = pixelCount_Over / (overSatV.shape[0]*overSatV.shape[1])
  pixelCount_under, grayLevels_under=np.histogram(underSatV,bins=256)
  pdfUnder = pixelCount_under / (underSatV.shape[0]*underSatV.shape[1])

  VpixelCounts, VgrayLevels=np.histogram(V,bins=256)

  pdfV = VpixelCounts / (V.shape[0]*V.shape[1])
  h=5
  L=256
  Z = np.array([0,0,0,0])

  B=np.convolve(pdfV,np.ones(h)/h,mode="full")

  pdfmaV = np.concatenate([B[h-1:B.shape[0]-h+1].T,Z])

  D=np.convolve(pdfOver,np.ones(h)/h,mode="full")
  pdfmaSO=np.concatenate([[0],D[h:D.shape[0]-h+1].T,Z])
  F=np.convolve(pdfUnder,np.ones(h)/h,mode='full')

  pdfmaSU=np.concatenate([[0],F[h:F.shape[0]-h+1].T,Z])
  pdfu = 1/L

  pdfuM = pdfu -  (pdfu< pdfmaSO).astype(int)* pdfmaSO + (pdfu< pdfmaSU).astype
(int) * pdfmaSU

  pdfmod = (pdfmaV>pdfuM).astype(int)*pdfuM + (pdfmaV<pdfuM).astype(int)*pdfmaV
  y=np.sum(pdfuM-pdfmod)

  pdffinal=abs((1-y)*pdfmaV +  y * pdfuM)
  pdffinal=pdffinal.T
  cdfinal=np.round(255*(np.cumsum(pdffinal))+0.5).T

  M=np.round((V*255.0).astype(float)+1)
  outputL=np.zeros((M.shape[0],M.shape[1]))
  for i in range(M.shape[0]):
    for j in range(M.shape[1]):
      outputL[i,j]=cdfinal[int(M[i,j])-1]
  return outputL/255.0
```

In [ ]:
```
def mod_homo_filter(Image):
  M1 = 2*Image.shape[0] +1
  N1 = 2*Image.shape[1] +1
  order = 2
  D0 = 0.5
  alpha = 0.9
  #apply log
  Image = np.log10(1 + Image)
  #Calculate the LPF
  Lpf = lpfilter('btw', M1, N1, D0,order);
  Lpf1 = np.fft.fftshift(Lpf)

  Fl = abs(np.fft.fft2(Image.astype(float),s=(Lpf1.shape[0],Lpf1.shape[1])))

  LPF = (np.fft.ifft2(np.fft.fftshift(Lpf1)*Fl)).real
  Lfilter = np.exp(LPF[0:Image.shape[0], 0:Image.shape[1]]) - 1


  #Calculate the HPF
  Hpf = 1-Lpf
  Fh = np.fft.fft2(Image.astype(float),s=(Hpf.shape[0],Hpf.shape[1])) # Calcula
te the discrete Fourier transform of the Image
  HPF = (np.fft.ifft2(np.fft.fftshift(Hpf)*Fh)).real #multiply the Fourier spec
trum by the HPF and apply the inverse, discrete Fourier transform
  #HPF = HPF(1:size(Image,1), 1:size(Image,2)) # Resize the image to undo paddi
ng
  Hfilter =  np.exp(HPF[0:Image.shape[0], 0:Image.shape[1]]) - 1
  #Sum of two filters
  #optimize the value of beta
  #beta = Golden_Section_Search(Image,alpha ,Lfilter , Hfilter)
  beta = 5.0;
  output_HFCR = alpha * Lfilter + beta * Hfilter

  return output_HFCR
```

In [ ]:
```
def lpfilter(type,M,N,D0,n=1):
  U,V=dftuv(M,N)
  gammaL= 0.05
  gammaH= 2
  D = np.sqrt((U- M/2)**2 + (V-N/2)**2)
  if type=="ideal":
    H=float(D<=D0)
  elif type== 'btw':
    H = 1/(1 + (D/D0)**(2*n))
  elif  type=='gaussian':
    H = (gammaH - gammaL)*(1- exp(-(D**2)/(2*(D0**2))))+ gammaL
  else:
    print("unknown filter type")
  return H
```

In [ ]:
```
def dftuv(M,N):
  u=np.array(range(M))
  v=np.array(range(N))
  idx=np.where(u>M/2)[0][0]
  u[idx-1]=u[idx-1]-M
  idy=np.where(v>N/2)[0][0]
  v[idy-1]=v[idy-1]-N
  V,U=np.meshgrid(v,u)
  return U,V
```

In [ ]:
```
import skimage.measure
def Golden_Section_Search(Image,alpha,Lfilter,Hfilter):
  g=lambda b: alpha * Lfilter + b * Hfilter
  f=lambda b: (abs(np.mean(g(b))-np.mean(Image))/(np.mean(Image)))*skimage.meas
ure.shannon_entropy(g(b))
  blow = 2.5
  bup =4.0
  goldenratio = (np.sqrt(5)-1)/2

  d = goldenratio * (bup - blow)
  x1 = blow + d
  x2 = bup - d
  #eps = 0.000001

  while (abs(x2-x1)> 0.25):
    if (f(x2)> f(x1)):
      bup = x1
      x1=x2
      d=goldenratio * (bup - blow)
      x2 = bup-d
      #f(x2);
    else:
      if (f(x1)> f(x2)):
        blow = x2
        x2 = x1
        d=goldenratio * (bup - blow)
        x1 = blow + d
        #f(x1)
  beta = (blow+bup)/2

  return beta
```

In [ ]:
```
def matr(M,D,J):
  G=np.zeros(M.shape)
  c=0
  for i in range(M.shape[0]):
    for j in range(M.shape[1]):
      a=M[i,j]+D[i,j]+J[i,j]
      if a>255.0:
        c=c+1
        G[i,j]=255.0
      else:
        G[i,j]=a
  return G,c
```

In [ ]:
```
def color_restoration(image,enhanced):
  mu = 0.2
  lamda= 0.2
  T=np.zeros(image.shape,dtype=np.uint32)
  T[:,:,0]= image[:,:,0].astype(int)+image[:,:,1].astype(int) +image[:,:,2].ast
ype(int)

  T[:,:,0][T[:,:,0]>255] = 255
  T[:,:,1]=T[:,:,0]
  T[:,:,2]=T[:,:,0]
  K = image.astype(float)/T.astype(float)
  R = np.exp(lamda*(np.log10(mu * K)))
  restored = R*enhanced
  return restored
```

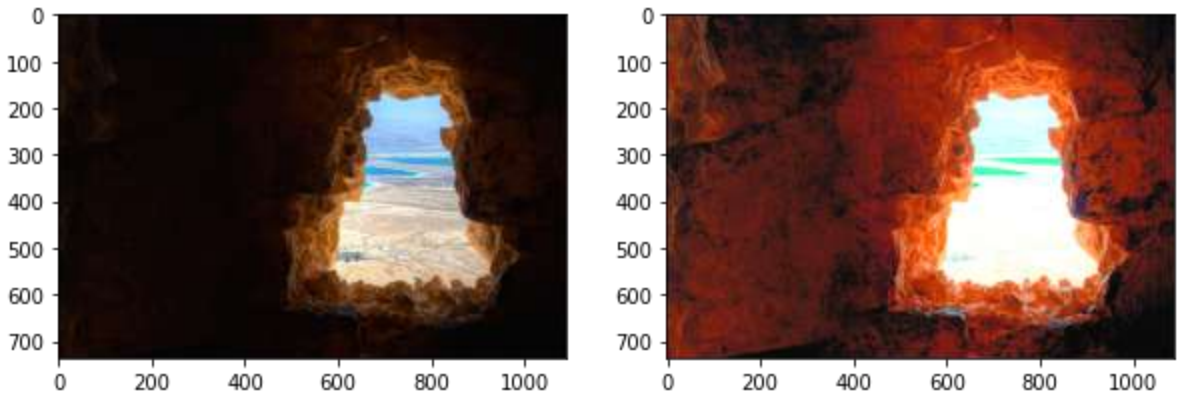
main

In [ ]:
```
def illumination_enhancement_algorithm(input_image):
  hsv_image = cv.cvtColor(input_image, cv.COLOR_RGB2HSV)
  hsv_image=hsv_image.astype(np.uint32)
  outputL = data_distribution (hsv_image)

  hsv_image[:,:,2] = mod_homo_filter(outputL)*255
  outputP = hsv_to_rgb(hsv_image/255)

  output= color_restoration(input_image,outputP)

  return output
```

In [ ]:
```
img = mpimg.imread("/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/cave.jpg")

imaaaa=illumination_enhancement_algorithm(img)
plt.figure(figsize=(10,10))
plt.subplot(121)
plt.imshow(img)
plt.subplot(122)
plt.imshow(imaaaa)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:11: RuntimeWarning: divide by zero encountered in log10
  # This is added back by InteractiveShellApp.init_path()
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

Out[ ]:
```
<matplotlib.image.AxesImage at 0x7f81e3c99da0>
```



Method 2: Multi-branch low-light enhancement network

Import Packages

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

In [ ]:
```
from glob import glob
import numpy as np
import random
import scipy
import os
import cv2 as cv
from keras.layers import Input, Conv2D, Conv2DTranspose, Concatenate
from keras.applications.vgg19 import VGG19
from keras.models import Model
from keras.layers import Input, Conv2D, BatchNormalization, Activation, UpSampl
ing2D, Conv2DTranspose, Reshape, Dropout, concatenate, Concatenate, multiply, a
dd, MaxPooling2D, Lambda, Activation, subtract, Flatten, Dense
from keras.callbacks import ModelCheckpoint, LearningRateScheduler, ReduceLROnP
lateau
from keras.preprocessing.image import ImageDataGenerator
from keras.layers.advanced_activations import LeakyReLU
from keras.optimizers import Adam
from keras.regularizers import l2
import imageio
from keras import backend as K
from keras.models import Model
from keras.utils import plot_model
from scipy import misc
from glob import glob
import tensorflow as tf
import numpy as np
import scipy
import platform
import keras
import os
import random
from PIL import Image, ImageEnhance
from keras import backend as K
import matplotlib.pyplot as plt
random.seed(10)
from google.colab import files
from IPython.display import Image
```

Creating a Dataset Class

Here we create a Dataset class to manage the communication with our dataset.

This class contain 3 functions:

imread_color: This funtion reads an RGB image from the dataset.

–

imwrite: This function will save an RGB image to a specific path.

–

- The purpuse of this class is to manage the access to the data and create a structured data from images in JPG forma.

The output of this class is:

- input_imgs :which contains images
- gt : contains the target images.

In [ ]:

```
class Dataloader():
    def __init__(self, dataset_name, crop_shape=(256, 256)):
        self.dataset_name = dataset_name
        self.crop_shape = crop_shape
#This function will read images in rgb forma
    def imread_color(self, path):
        img = cv.imread(path, cv.IMREAD_COLOR | cv.IMREAD_ANYDEPTH)
        img=img/255.0
        b, g, r = cv.split(img)
        img_rgb = cv.merge([r, g, b])
        return img_rgb
#This images will save an image to a specific path
    def imwrite(self, path, img):
        r, g, b = cv.split(img)
        img_rgb = cv.merge([b, g, r])
        cv.imwrite(path, img_rgb)
#This function will load the images from the folder in the google drive and cre
ate a structured data
    def load_data(self, batch_size=16):
        path = glob('/content/drive/MyDrive/New Drive/Projet transversal/Transv
ersal_dataset/our_dataset/train/*.jpg')
        path=path[:100]
        self.n_batches = int(len(path) / batch_size)
        while 1:
            random.shuffle(path)
            for i in range(self.n_batches - 1):
                batch_path = path[i * batch_size:(i + 1) * batch_size]
                input_imgs = np.empty((batch_size, self.crop_shape[0], self.cro
p_shape[1], 6), dtype="float32")
                gt = np.empty((batch_size, self.crop_shape[0], self.crop_shape
[1], 3), dtype="float32")

                number = 0
                for img_B_path in batch_path:
                    img_B = self.imread_color(img_B_path)
                    path_mid = os.path.split(img_B_path)
                    path_A_1 = path_mid[0] + '_' + self.dataset_name
                    path_A = os.path.join(path_A_1, path_mid[1])
                    img_A = self.imread_color(path_A)

                    crop_img_A = cv.resize(img_A, (self.crop_shape[0], self.cro
p_shape[1]))
                    crop_img_B = cv.resize(img_B, (self.crop_shape[0], self.cro
p_shape[1]))

                    #Data augmentation
                    if np.random.randint(2, size=1)[0] == 1:  # random flip
                        crop_img_A = np.flipud(crop_img_A)
                        crop_img_B = np.flipud(crop_img_B)
                    if np.random.randint(2, size=1)[0] == 1:
                        crop_img_A = np.fliplr(crop_img_A)
                        crop_img_B = np.fliplr(crop_img_B)
                    if np.random.randint(2, size=1)[0] == 1:  # random transpos
e
                        crop_img_A = np.transpose(crop_img_A, (1, 0, 2))
                        crop_img_B = np.transpose(crop_img_B, (1, 0, 2))

                    input_imgs[number, :, :, :] = np.concatenate([crop_img_A, c
rop_img_B], axis=-1)
                    gt[number, :, :, :] = crop_img_B
```

```
                    number += 1
                yield input_imgs, gt
```

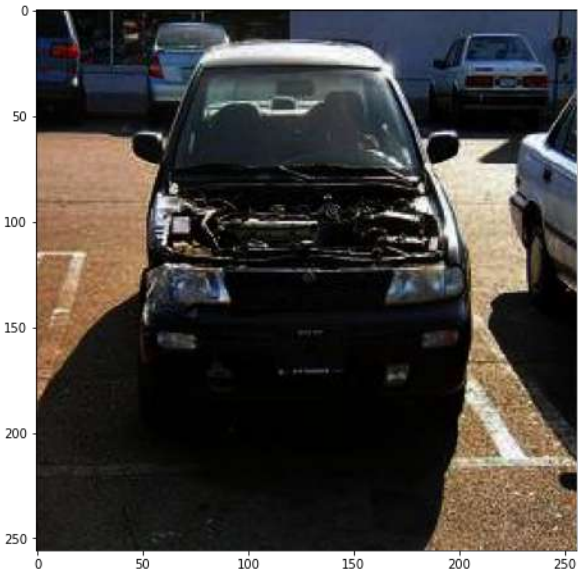
Here we are going to see an example in ourdata

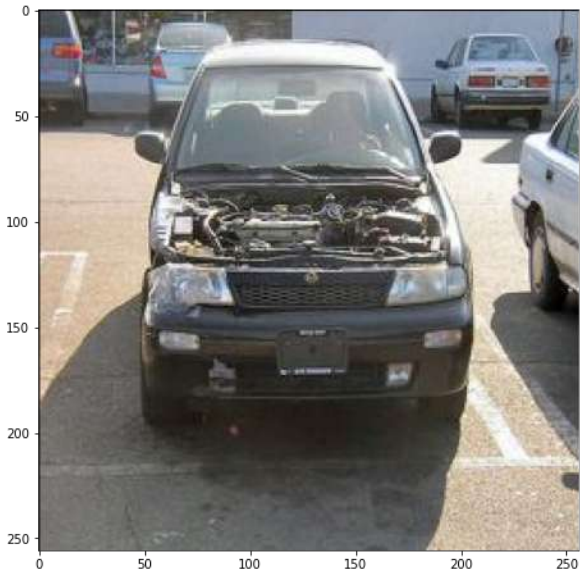
In [ ]:
```
dataset_name ='dark'
img_rows = 256
img_cols = 256
img_channels = 3
crop_shape = (img_rows, img_cols, img_channels)
input_shape = (img_rows, img_cols, img_channels*2)
dataloader1=Dataloader(dataset_name,crop_shape=(img_rows, img_cols))
dataL=dataloader1.load_data(batch_size=16)
```

In [ ]:
```
vis_imag=next(dataL)
```

In [ ]:
```
plt.figure(figsize=(20,8))
plt.subplot(121)
plt.imshow(vis_imag[0][0][:,:,:3])
plt.subplot(122)
plt.imshow(vis_imag[1][0])
```

Out[ ]:
```
<matplotlib.image.AxesImage at 0x7f85e3ec4eb8>
```





Defining our Losses

We defined these losses based on the paper.

In [ ]:
```
def bright_mae(y_true, y_pred):
    return K.mean(K.abs(y_pred[:,:,:,:3] - y_true[:,:,:,:3]))
```

In [ ]:
```
def bright_mse(y_true, y_pred):
    return K.mean((y_pred[:,:,:,:3] - y_true[:,:,:,:3])**2)
```

In [ ]:
```
def bright_AB(y_true, y_pred):
            return K.abs(K.mean(y_true[:,:,:,:3])-K.mean(y_pred[:,:,:,:3]))
```

In [ ]:
```
def log10(x):
    numerator = K.log(x)
    denominator = K.log(K.constant(10, dtype=numerator.dtype))
    return numerator / denominator
```

In [ ]:
```
def bright_psnr(y_true, y_pred):
    mse = K.mean((K.abs(y_pred[:,:,:,:3] - y_true[:,:,:,:3])) ** 2)
    max_num = 1.0
    psnr = 10 * log10(max_num ** 2 / mse)
    return psnr
```

In [ ]:
```
def tf_ssim(img1,img2,max_val=1,filter_size=11,filter_sigma=1.5,K1=0.01,K2=0.0
3):
  return tf.image.ssim(img1,img2,max_val,filter_size,filter_sigma,K1,K2)
```

In [ ]:
```
def bright_SSIM(y_true, y_pred):
    SSIM_loss = tf_ssim(tf.expand_dims(y_pred[:,:,:,0], -1), tf.expand_dims(y_t
rue[:,:,:,0], -1))+tf_ssim(tf.expand_dims(y_pred[:,:,:,1], -1), tf.expand_dims
(y_true[:,:,:,1], -1)) + tf_ssim(tf.expand_dims(y_pred[:,:,:,2], -1), tf.expand
_dims(y_true[:,:,:,2], -1))
    return SSIM_loss/3
```

In [ ]:
```
def psnr_cau(y_true, y_pred):
    mse = np.mean((np.abs(y_pred - y_true)) ** 2)
    max_num = 1.0
    psnr = 10 * np.log10(max_num ** 2 / mse)
    return psnr
```

In [ ]:
```
def save_model(model, name, epoch, batch_i):
    modelname = './Res_models/' + str(epoch) + '_' + str(batch_i) + name + '.h
5'
    model.save_weights(modelname)
```

In [ ]:
```
def imread_color(path):
    img = cv.imread(path, cv.IMREAD_COLOR | cv.IMREAD_ANYDEPTH) / 255.
    b, g, r = cv.split(img)
    img_rgb = cv.merge([r, g, b])
    return img_rgb
```

In [ ]:
```
def imwrite(path, img):
    r, g, b = cv.split(img*255)
    img_rgb = cv.merge([b, g, r])
    cv.imwrite(path, img_rgb)
```

In [ ]:
```
def range_scale(x):
    return x * 2 - 1.
```

Bulding the Model Architecture

In [ ]:
```
def build_vgg():
    vgg_model = VGG19(include_top=False, weights='imagenet')
    vgg_model.trainable = False
    return Model(inputs=vgg_model.input, outputs=vgg_model.get_layer('block3_co
nv4').output)
```

In [ ]:
```
def build_mbllen(input_shape):

    def EM(input, kernal_size, channel):
        conv_1 = Conv2D(channel, (3, 3), activation='relu', padding='same', dat
a_format='channels_last')(input)
        conv_2 = Conv2D(channel, (kernal_size, kernal_size), activation='relu',
padding='valid', data_format='channels_last')(conv_1)
        conv_3 = Conv2D(channel*2, (kernal_size, kernal_size),activation='rel
u', padding='valid', data_format='channels_last')(conv_2)
        conv_4 = Conv2D(channel*4, (kernal_size, kernal_size), activation='rel
u', padding='valid', data_format='channels_last')(conv_3)
        conv_5 = Conv2DTranspose(channel*2, (kernal_size, kernal_size), activat
ion='relu', padding='valid', data_format='channels_last')(conv_4)
        conv_6 = Conv2DTranspose(channel, (kernal_size, kernal_size), activatio
n='relu', padding='valid', data_format='channels_last')(conv_5)
        res = Conv2DTranspose(3, (kernal_size, kernal_size), activation='relu',
padding='valid', data_format='channels_last')(conv_6)
        return res

    inputs = Input(shape=input_shape)
    FEM = Conv2D(32, (3, 3), activation='relu', padding='same', data_format='ch
annels_last')(inputs)
    EM_com = EM(FEM, 5, 8)

    for j in range(3):
        for i in range(0, 3):
            FEM = Conv2D(32, (3, 3), activation='relu', padding='same', data_fo
rmat='channels_last')(FEM)
            EM1 = EM(FEM, 5, 8)
            EM_com = Concatenate(axis=3)([EM_com, EM1])

    outputs = Conv2D(3, (1, 1), activation='relu', padding='same', data_format
='channels_last')(EM_com) #choosing the best weights
    return Model(inputs, outputs)
```

Custom Loss

In [ ]:
```
def my_loss(y_true, y_pred):
    MAE_loss = K.mean(K.abs(y_pred[:,:,:,:3] - y_true))
    SSIM_loss = tf_ssim(tf.expand_dims(y_pred[:, :, :, 0], -1),tf.expand_dims(y
_true[:, :, :, 0], -1)) + tf_ssim(
        tf.expand_dims(y_pred[:, :, :, 1], -1), tf.expand_dims(y_true[:, :, :,
1], -1)) + tf_ssim(
        tf.expand_dims(y_pred[:, :, :, 2], -1), tf.expand_dims(y_true[:, :, :,
2], -1))
    VGG_loss = K.mean(K.abs(y_pred[:, :, :, 3:19] - y_pred[:, :, :, 19:35]))

    percent = 0.4
    index = int(256 * 256 * percent - 1)
    gray1 = 0.39 * y_pred[:, :, :, 0] + 0.5 * y_pred[:, :, :, 1] + 0.11 * y_pre
d[:, :, :, 2]
    gray = tf.reshape(gray1, [-1, 256 * 256])
    gray_sort = tf.nn.top_k(-gray, 256 * 256)[0]
    yu = gray_sort[:, index]
    yu = tf.expand_dims(tf.expand_dims(yu, -1), -1)
    mask = tf.cast(gray1 <= yu,dtype="float")
    mask1 = tf.expand_dims(mask, -1)
    mask = tf.concat([mask1, mask1, mask1], -1)

    low_fake_clean = tf.multiply(mask, y_pred[:, :, :, :3])
    high_fake_clean = tf.multiply(1 - mask, y_pred[:, :, :, :3])
    low_clean = tf.multiply(mask, y_true[:, :, :, :])
    high_clean = tf.multiply(1 - mask, y_true[:, :, :, :])
    Region_loss = K.mean(K.abs(low_fake_clean - low_clean) * 4 + K.abs(high_fak
e_clean - high_clean))

    loss = MAE_loss + VGG_loss/3. + 3 - SSIM_loss + Region_loss
    return loss
```

In [ ]:
```
if not os.path.isdir('./val_images'):
    os.makedirs('./val_images')
if not os.path.isdir('./logs'):
    os.makedirs('./logs')
if not os.path.isdir('./models'):
    os.makedirs('./models')
```

In [ ]:
```
def f1(x):
    return x[:, :, :, :3]

def f2(x):
    return x[:, :, :, 3:]

def f3(x):
    return tf.reshape(x,[-1, 256, 256, 16])
```

In [ ]:
```
img_rows = 256
img_cols = 256
img_channels = 3
crop_shape = (img_rows, img_cols, img_channels)
input_shape = (img_rows, img_cols, img_channels*2)
dataset_name = 'dark'
data_loader = Dataloader(dataset_name=dataset_name, crop_shape=(img_rows, img_c
ols))
```

In [ ]:

```
# Build the network
mbllen = build_mbllen(crop_shape)
# mbllen.load_weights('./1_dark2_color_identity_param.h5')

Input_MBLLEN = Input(shape=input_shape)
img_A = Lambda(f1)(Input_MBLLEN)
img_B = Lambda(f2)(Input_MBLLEN)

# VGG19 feature, content loss
vgg = build_vgg()
vgg.trainable = False

fake_B = mbllen(img_A)
vgg_fake = Lambda(range_scale)(fake_B)
fake_features = vgg(vgg_fake)
fake_features = Lambda(f3)(fake_features)

img_B_vgg = Lambda(range_scale)(img_B)
imgb_features = vgg(img_B_vgg)
imgb_features = Lambda(f3)(imgb_features)

output_com = concatenate([fake_B, fake_features, imgb_features], axis=3)

opt = Adam(lr=1*1e-03, beta_1=0.9, beta_2=0.999, epsilon=1e-08)
combined = Model(inputs=Input_MBLLEN, outputs=output_com)
combined.compile(loss=my_loss,
                 metrics=[bright_mae, bright_mse, bright_psnr, bright_SSIM, bri
ght_AB],
                 optimizer=opt)

combined.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applicatio
ns/vgg19/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5
80142336/80134624 [==============================] - 1s 0us/step
Model: "model_2"
__________________________________________________________________________________
_________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================
==================
input_2 (InputLayer)            [(None, 256, 256, 6) 0
__________________________________________________________________________________
_________________
lambda (Lambda)                 (None, 256, 256, 3)  0           input_2[0][0]
__________________________________________________________________________________
_________________
model (Functional)              (None, 256, 256, 3)  450171      lambda[0][0]
__________________________________________________________________________________
_________________
lambda_1 (Lambda)               (None, 256, 256, 3)  0           input_2[0][0]
__________________________________________________________________________________
_________________
lambda_2 (Lambda)               (None, 256, 256, 3)  0           model[0][0]
__________________________________________________________________________________
_________________
lambda_4 (Lambda)               (None, 256, 256, 3)  0           lambda_1[0][0]
__________________________________________________________________________________
_________________
model_1 (Functional)            (None, None, None, 2 2325568     lambda_2[0][0]
                                                                 lambda_4[0][0]
__________________________________________________________________________________
_________________
lambda_3 (Lambda)               (None, 256, 256, 16) 0           model_1[0][0]
__________________________________________________________________________________
_________________
lambda_5 (Lambda)               (None, 256, 256, 16) 0           model_1[1][0]
__________________________________________________________________________________
_________________
concatenate_9 (Concatenate)     (None, 256, 256, 35) 0           model[0][0]
                                                                 lambda_3[0][0]
                                                                 lambda_5[0][0]
==================================================================================
==================
Total params: 2,775,739
Trainable params: 450,171
Non-trainable params: 2,325,568
__________________________________________________________________________________
_________________
```

In [ ]:
```python
def scheduler(epoch):
    lr = K.eval(combined.optimizer.lr)
    print("LR =", lr)
    lr = lr * 0.99
    return lr
```

Training the Model

In [ ]:

```
change_lr = LearningRateScheduler(scheduler)
tbCallBack = keras.callbacks.TensorBoard(log_dir='./logs', histogram_freq=0, wr
ite_graph=True, write_images=False,
                                         embeddings_freq=0, embeddings_layer_na
mes=None, embeddings_metadata=None)
nanstop = keras.callbacks.TerminateOnNaN()
reducelearate = keras.callbacks.ReduceLROnPlateau(monitor='loss', factor=0.5, p
atience=2, min_lr=1e-10)
earlystop = keras.callbacks.EarlyStopping(monitor='loss', min_delta=3, patience
=0, verbose=0, mode='min')

batch_size = 16
step_epoch = 20
num_epoch=10
combined.fit_generator(
        data_loader.load_data(batch_size),
        steps_per_epoch=step_epoch,
        epochs=num_epoch,
        callbacks=[tbCallBack, change_lr, nanstop, reducelearate])
print('Done!')
modelname = './models/' + str(num_epoch) + '_' + dataset_name + '_base.h5'
mbllen.save_weights(modelname)
```

```
/usr/local/lib/python3.6/dist-packages/tensorflow/python/keras/engine/training.py:1844: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '

Epoch 1/10
LR = 0.001
20/20 [==============================] - 65s 3s/step - loss: 5.6890 - bright_mae: 0.3459 - bright_mse: 0.1875 - bright_psnr: 7.5176 - bright_SSIM: 0.2020 - bright_AB: 0.2918
Epoch 2/10
LR = 0.0009900001
20/20 [==============================] - 34s 2s/step - loss: 4.3270 - bright_mae: 0.2345 - bright_mse: 0.1125 - bright_psnr: 9.5228 - bright_SSIM: 0.4082 - bright_AB: 0.1986
Epoch 3/10
LR = 0.0009801001
20/20 [==============================] - 35s 2s/step - loss: 3.1868 - bright_mae: 0.1777 - bright_mse: 0.0677 - bright_psnr: 12.2121 - bright_SSIM: 0.5925 - bright_AB: 0.1295
Epoch 4/10
LR = 0.0009702991
20/20 [==============================] - 34s 2s/step - loss: 2.1746 - bright_mae: 0.0863 - bright_mse: 0.0157 - bright_psnr: 18.1463 - bright_SSIM: 0.7715 - bright_AB: 0.0330
Epoch 5/10
LR = 0.00096059614
20/20 [==============================] - 35s 2s/step - loss: 1.7548 - bright_mae: 0.0715 - bright_mse: 0.0100 - bright_psnr: 20.0947 - bright_SSIM: 0.8115 - bright_AB: 0.0277
Epoch 6/10
LR = 0.0009509902
20/20 [==============================] - 34s 2s/step - loss: 1.5376 - bright_mae: 0.0616 - bright_mse: 0.0074 - bright_psnr: 21.3223 - bright_SSIM: 0.8335 - bright_AB: 0.0170
Epoch 7/10
LR = 0.0009414803
20/20 [==============================] - 34s 2s/step - loss: 1.4651 - bright_mae: 0.0612 - bright_mse: 0.0072 - bright_psnr: 21.4777 - bright_SSIM: 0.8415 - bright_AB: 0.0209
Epoch 8/10
LR = 0.0009320655
20/20 [==============================] - 34s 2s/step - loss: 1.3859 - bright_mae: 0.0611 - bright_mse: 0.0068 - bright_psnr: 21.7074 - bright_SSIM: 0.8513 - bright_AB: 0.0268
Epoch 9/10
LR = 0.0009227448
20/20 [==============================] - 35s 2s/step - loss: 1.2987 - bright_mae: 0.0540 - bright_mse: 0.0055 - bright_psnr: 22.6144 - bright_SSIM: 0.8593 - bright_AB: 0.0171
Epoch 10/10
LR = 0.0009135174
20/20 [==============================] - 35s 2s/step - loss: 1.2651 - bright_mae: 0.0538 - bright_mse: 0.0053 - bright_psnr: 22.8422 - bright_SSIM: 0.8656 - bright_AB: 0.0145
Done!
```

Importing the model

In [ ]:
```
mbllen = build_mbllen((None, None, 3))
mbllen.load_weights("/content/models/10_dark_base.h5")
opt = keras.optimizers.Adam(lr=2 * 1e-04, beta_1=0.9, beta_2=0.999, epsilon=1e-08)
mbllen.compile(loss='mse', optimizer=opt)
```

In [ ]:
```
original_image=imread_color("/content/141750613_457076548644933_2676900420131963288_n.png")
original_image = original_image[np.newaxis, :]

def predict(image):
  out_pred = mbllen.predict(image)
  return out_pred[0]

mbllen_enhanced_image=predict(original_image)

plt.figure(figsize=(20,10))
plt.subplot(122)
plt.title("Model output")
plt.imshow(mbllen_enhanced_image)
plt.subplot(121)
plt.title("Original Image")
plt.imshow(original_image[0])
```

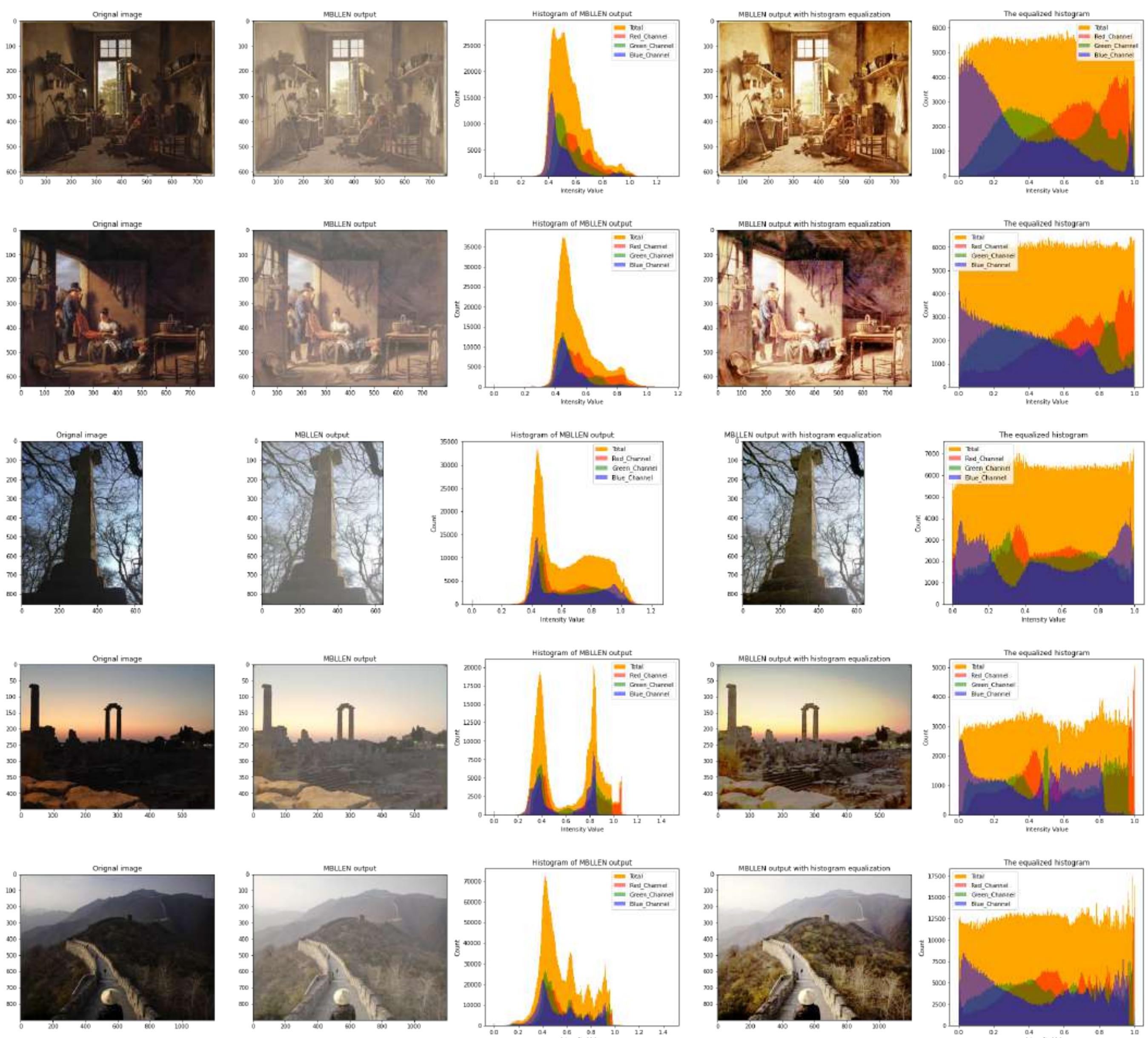
Contrast problem

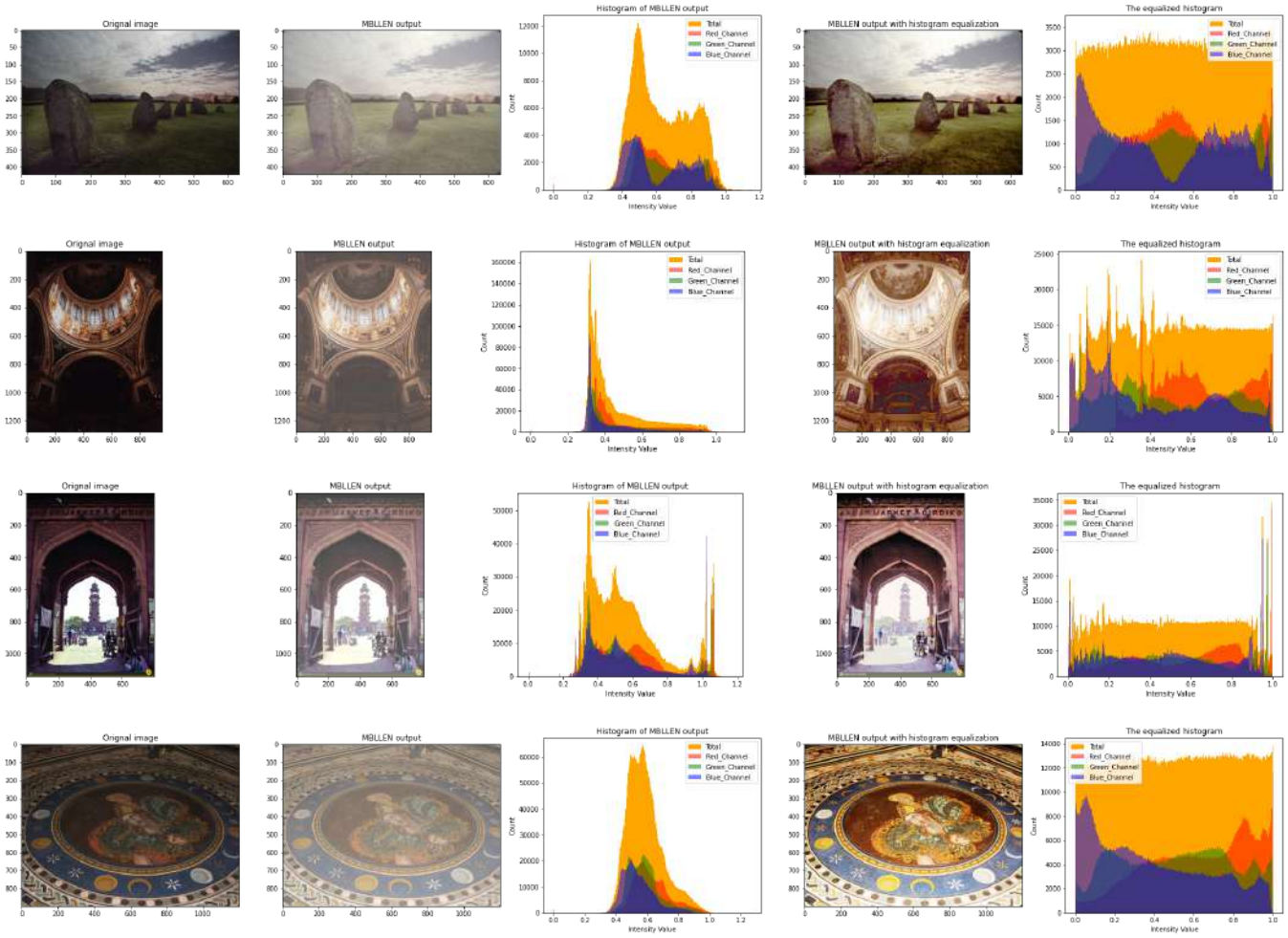
In [ ]:
```
import skimage.exposure
```

In [ ]:

```
contrast_path=glob("/content/drive/MyDrive/New Drive/Projet transversal/Low_contrast/*.jpg")

for image_index in range(len(contrast_path)):
  original_image_path = contrast_path[image_index]
  original_image=imread_color(original_image_path)
  mbllen_enhanced_image = predict(original_image[np.newaxis, :])
  post_processed_image = skimage.exposure.equalize_hist(mbllen_enhanced_image)


  plt.figure(figsize=(35,5))
  plt.subplot(151)
  plt.title("Orignal image")
  plt.imshow(original_image)
  plt.subplot(152)
  plt.title("MBLLEN output")
  plt.imshow(mbllen_enhanced_image)
  plt.subplot(153)
  plt.title("Histogram of MBLLEN output")

  _ = plt.hist(mbllen_enhanced_image.ravel(), bins = 256, color = 'orange', )
  _ = plt.hist(mbllen_enhanced_image[:, :, 0].ravel(), bins = 256, color = 'red', alpha = 0.5)
  _ = plt.hist(mbllen_enhanced_image[:, :, 1].ravel(), bins = 256, color = 'Green', alpha = 0.5)
  _ = plt.hist(mbllen_enhanced_image[:, :, 2].ravel(), bins = 256, color = 'Blue', alpha = 0.5)
  _ = plt.xlabel('Intensity Value')
  _ = plt.ylabel('Count')
  _ = plt.legend(['Total', 'Red_Channel', 'Green_Channel', 'Blue_Channel'])

  plt.subplot(154)
  plt.title("MBLLEN output with histogram equalization")
  plt.imshow(post_processed_image)
  plt.subplot(155)
  plt.title("The equalized histogram")
  _ = plt.hist(post_processed_image.ravel(), bins = 256, color = 'orange', )
  _ = plt.hist(post_processed_image[:, :, 0].ravel(), bins = 256, color = 'red', alpha = 0.5)
  _ = plt.hist(post_processed_image[:, :, 1].ravel(), bins = 256, color = 'Green', alpha = 0.5)
  _ = plt.hist(post_processed_image[:, :, 2].ravel(), bins = 256, color = 'Blue', alpha = 0.5)
  _ = plt.xlabel('Intensity Value')
  _ = plt.ylabel('Count')
  _ = plt.legend(['Total', 'Red_Channel', 'Green_Channel', 'Blue_Channel'])
```

```
/usr/local/lib/python3.6/dist-packages/skimage/exposure/exposure.py:181: UserWar
ning: This might be a color image. The histogram will be computed on the flatten
ed image. You can instead apply this function to each color channel.
  hist, bin_centers = histogram(image, nbins)
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



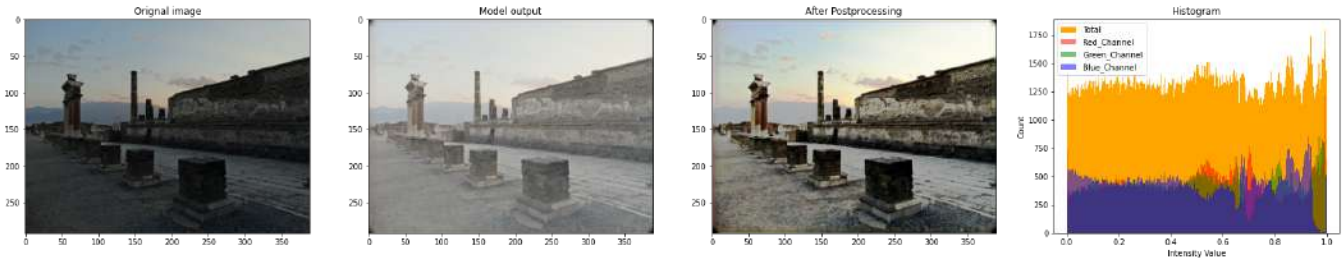


Trying on the e-Heritage dataset

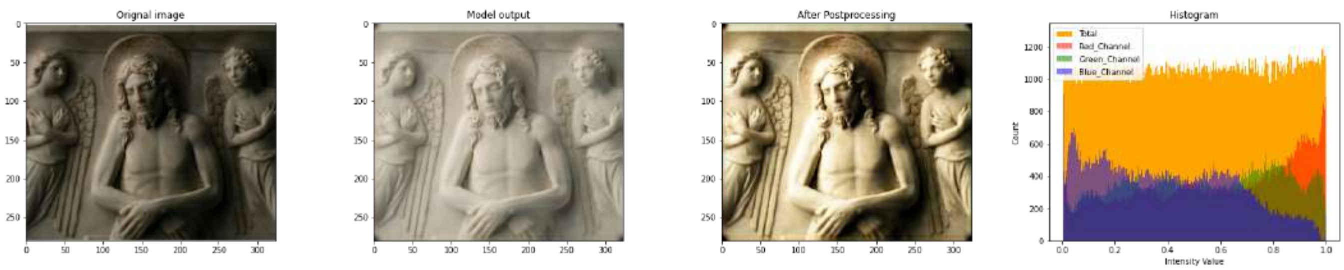
In [ ]:

```
path_test=glob("/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/*.jpg")

for image_index in range(len(path_test)):
  original_image_path = path_test[image_index]
  original_image=imread_color(original_image_path)
  mbllen_enhanced_image = predict(original_image[np.newaxis, :])
  post_processed_image = skimage.exposure.equalize_hist(mbllen_enhanced_image)

  plt.figure(figsize=(30,5))
  plt.subplot(141)
  plt.title("Orignal image")
  plt.imshow(original_image)
  plt.subplot(142)
  plt.title("Model output")
  plt.imshow(mbllen_enhanced_image)
  plt.subplot(143)
  plt.title("After Postprocessing")
  plt.imshow(post_processed_image)

  plt.subplot(144)
  plt.title("Histogram")

  _ = plt.hist(post_processed_image.ravel(), bins = 256, color = 'orange', )
  _ = plt.hist(post_processed_image[:, :, 0].ravel(), bins = 256, color = 'red', alpha = 0.5)
  _ = plt.hist(post_processed_image[:, :, 1].ravel(), bins = 256, color = 'Green', alpha = 0.5)
  _ = plt.hist(post_processed_image[:, :, 2].ravel(), bins = 256, color = 'Blue', alpha = 0.5)
  _ = plt.xlabel('Intensity Value')
  _ = plt.ylabel('Count')
  _ = plt.legend(['Total', 'Red_Channel', 'Green_Channel', 'Blue_Channel'])
  plt.show()
```

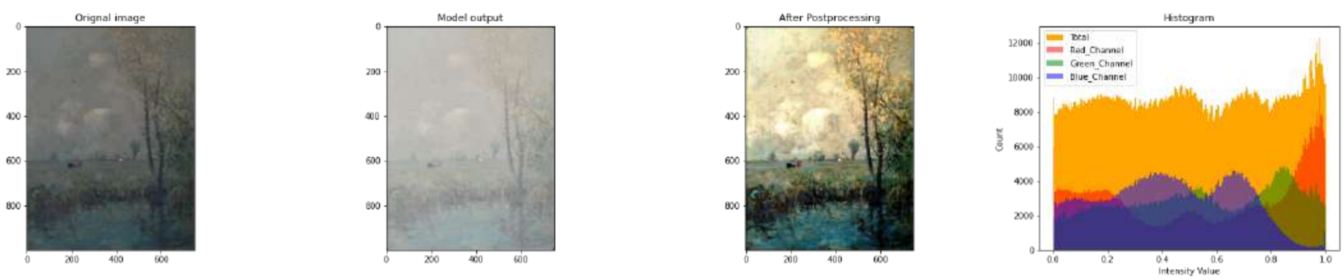
```
/usr/local/lib/python3.6/dist-packages/skimage/exposure/exposure.py:181: UserWarning: This might be a color image. The histogram will be computed on the flattened image. You can instead apply this function to each color channel.
  hist, bin_centers = histogram(image, nbins)
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



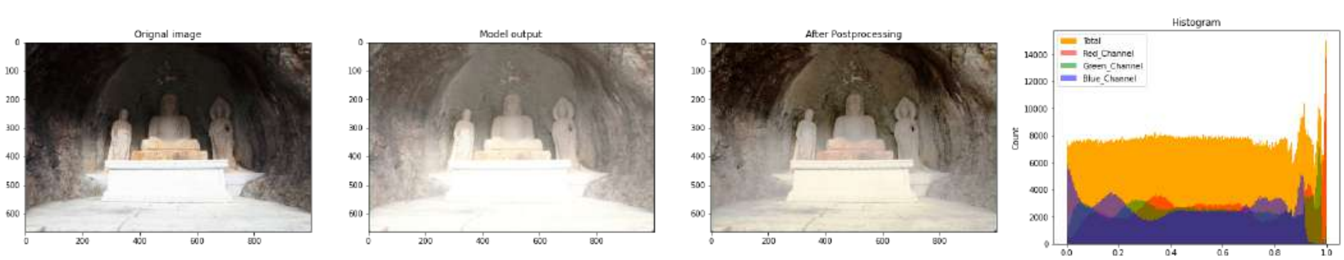

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



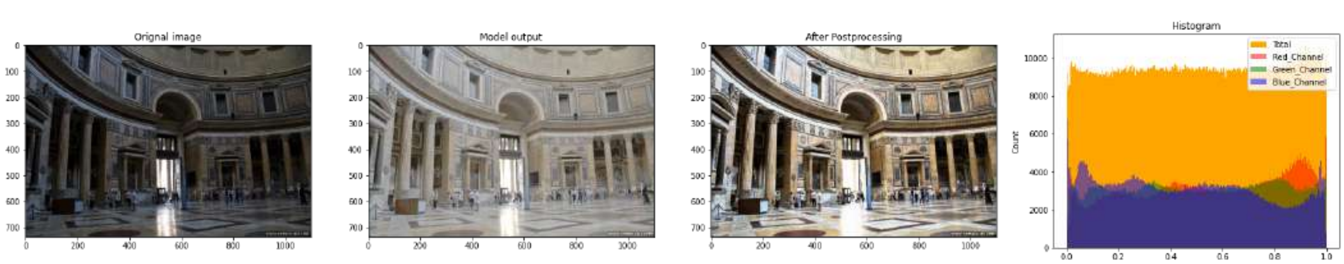

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



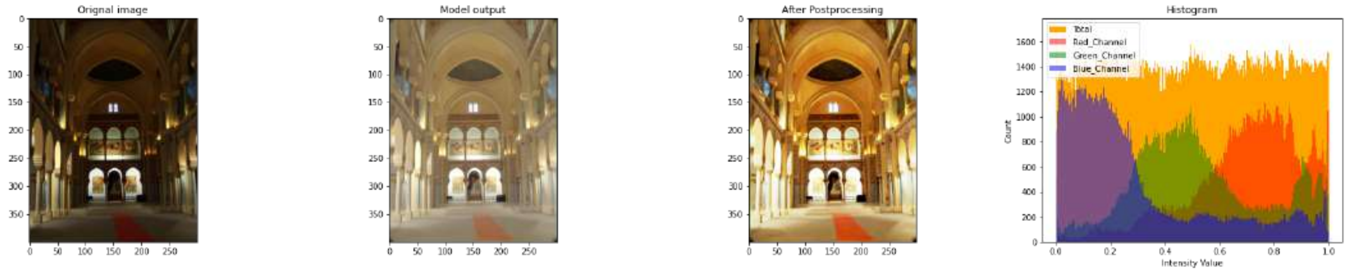

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```




```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

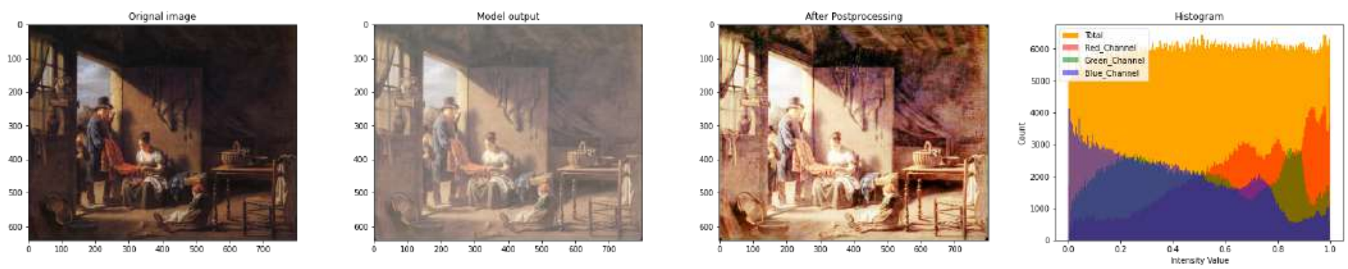


```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



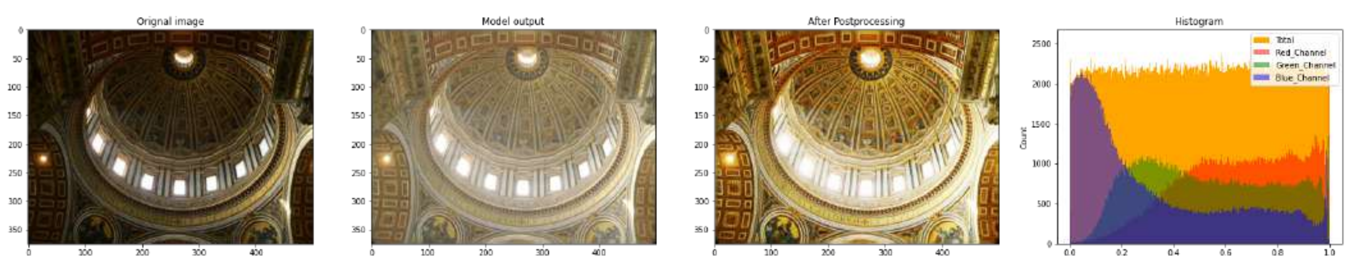

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```


```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

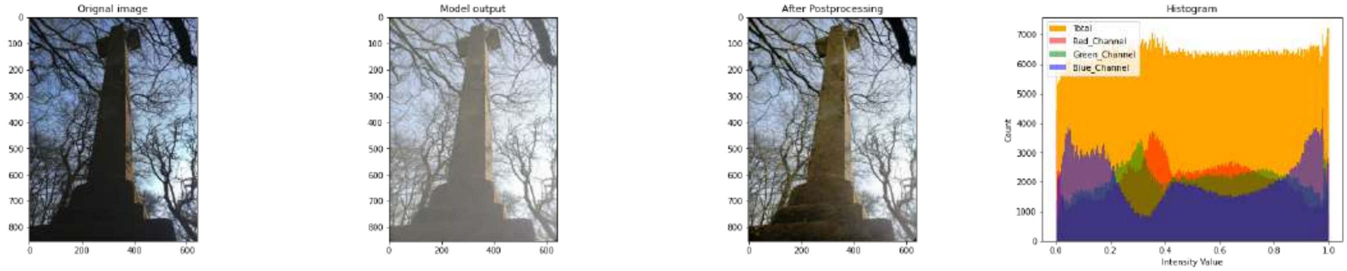



```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

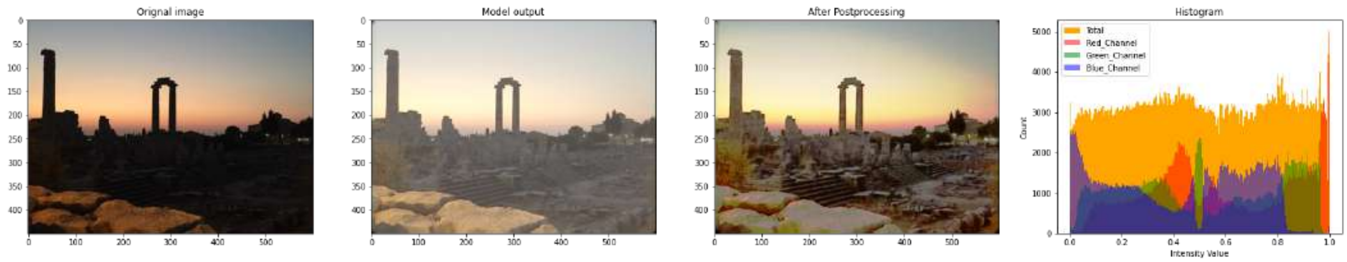

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



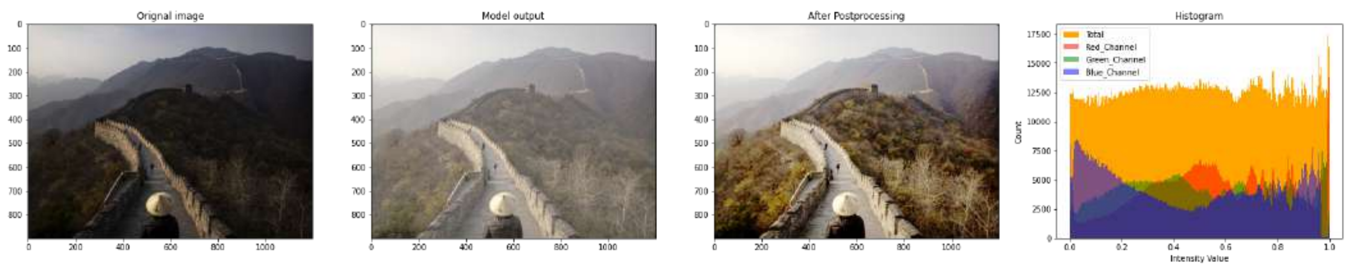

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



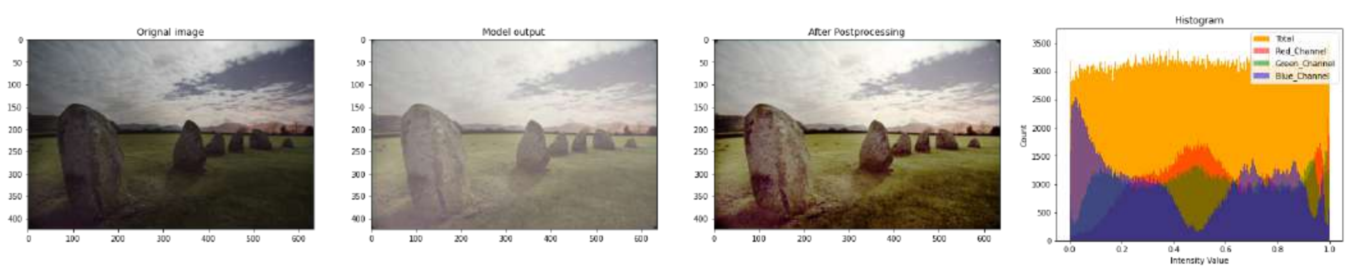
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



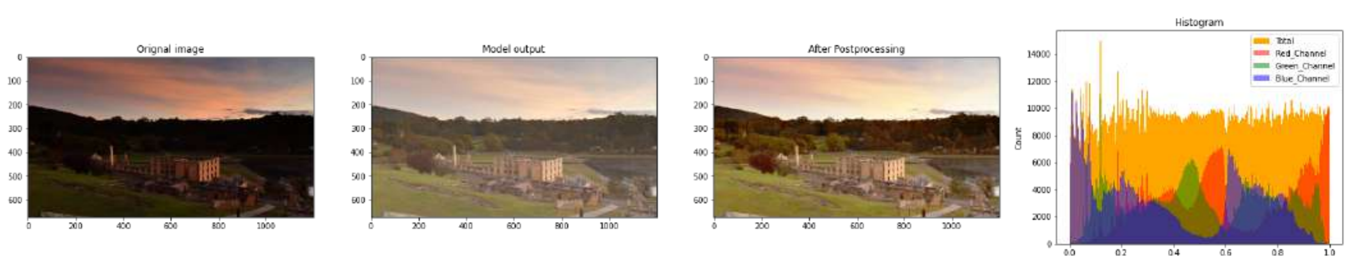
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



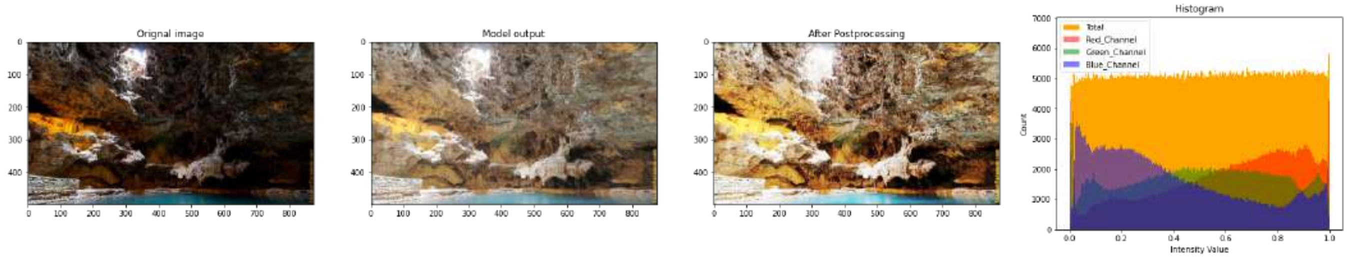
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



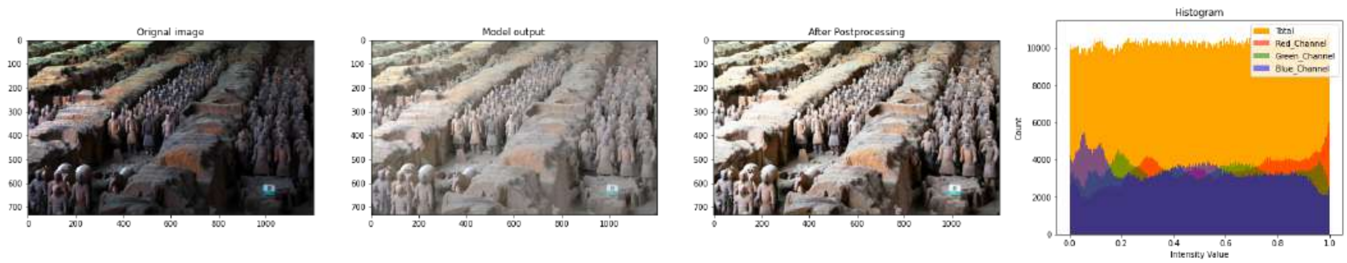
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



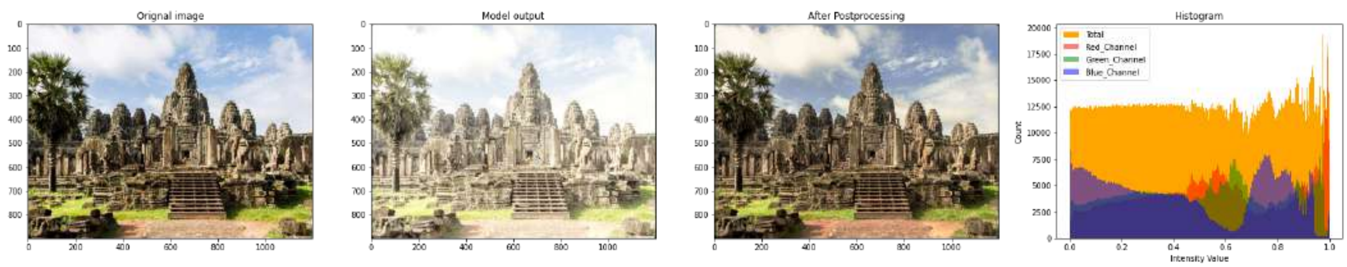
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



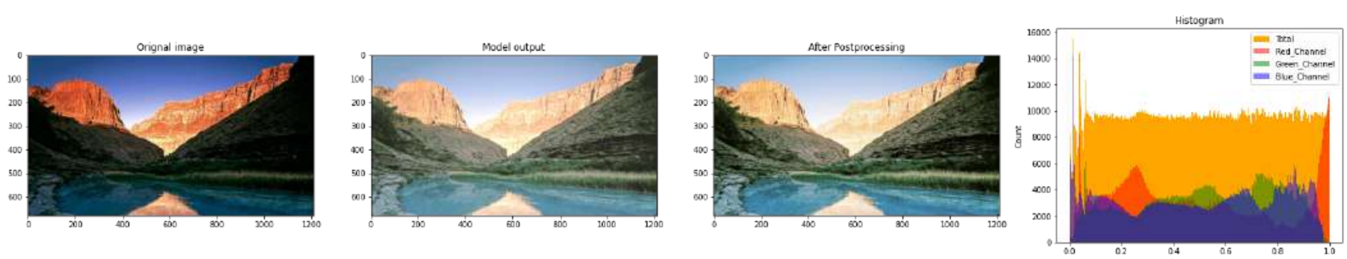

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



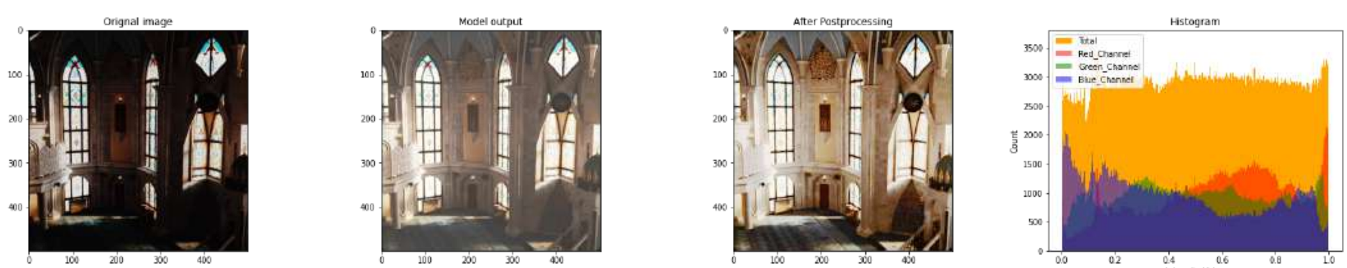

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



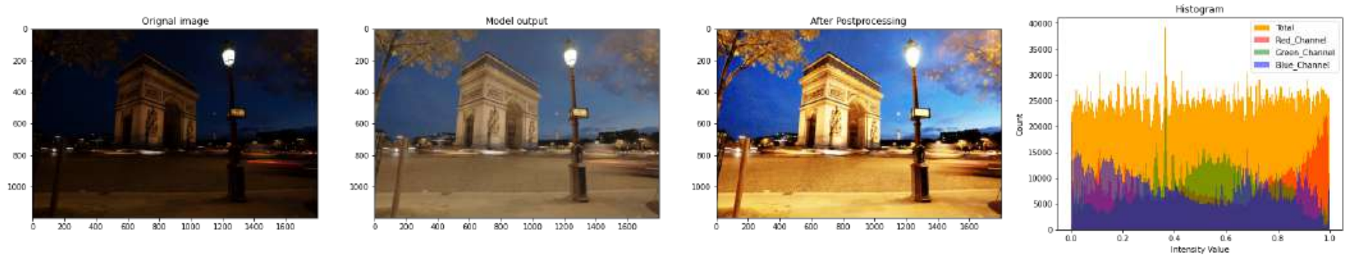

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



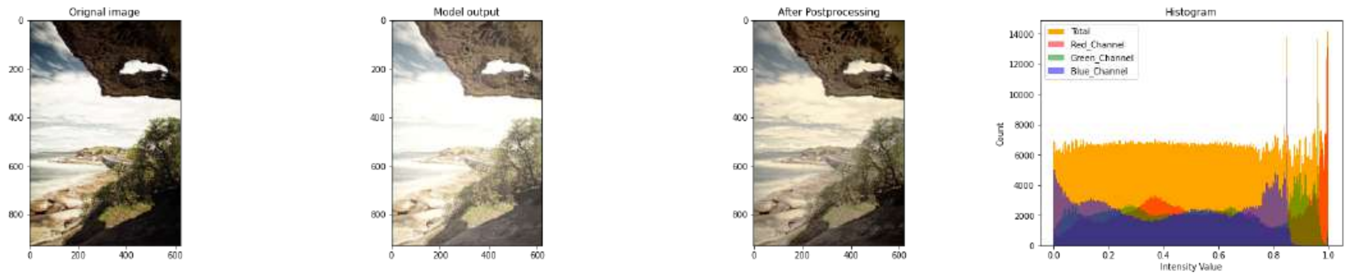

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



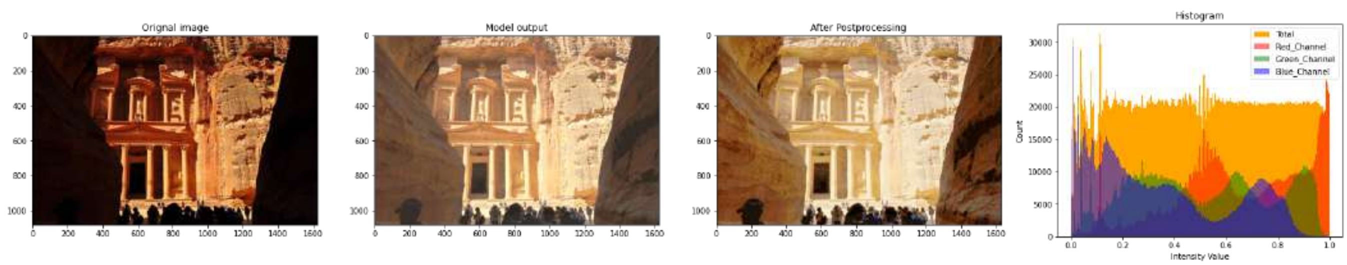

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



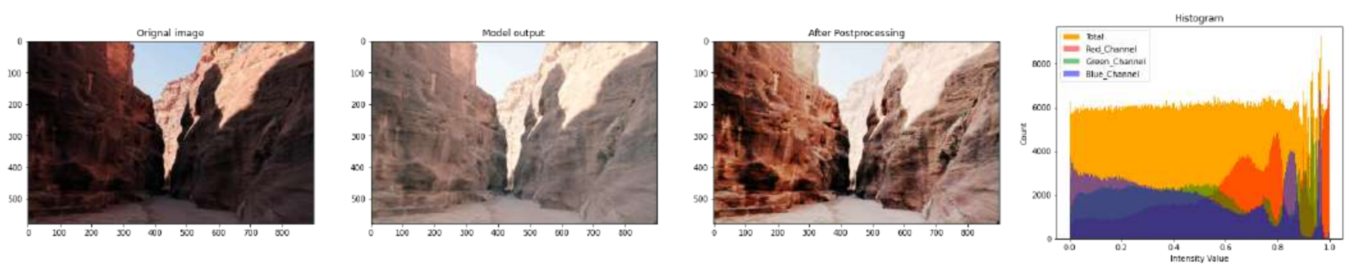
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



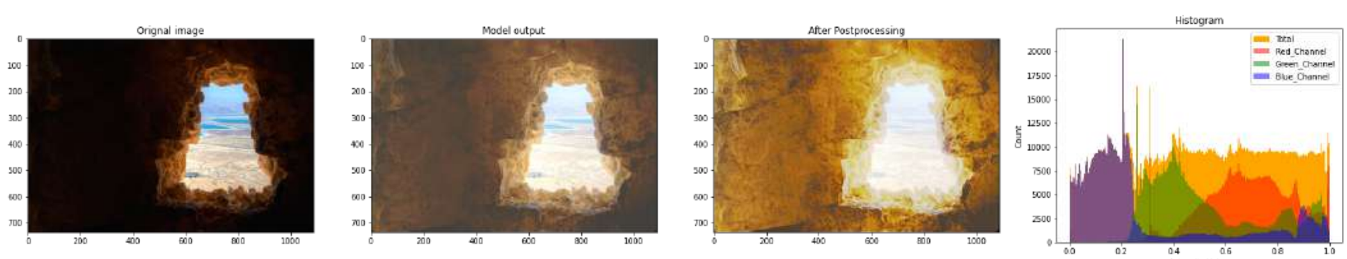
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



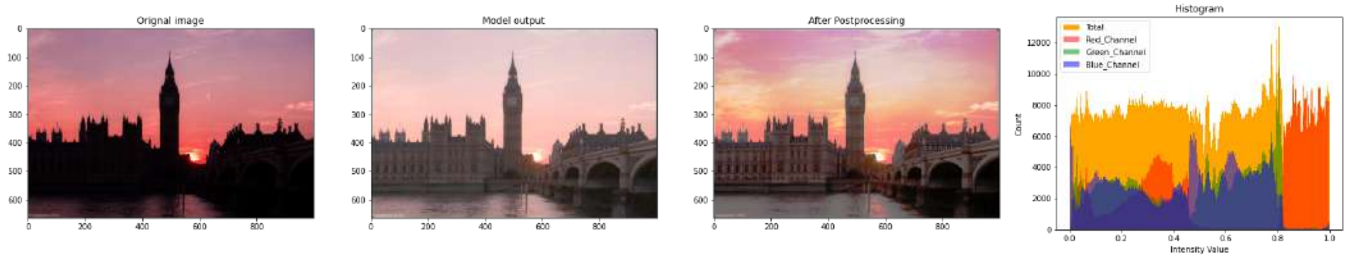
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



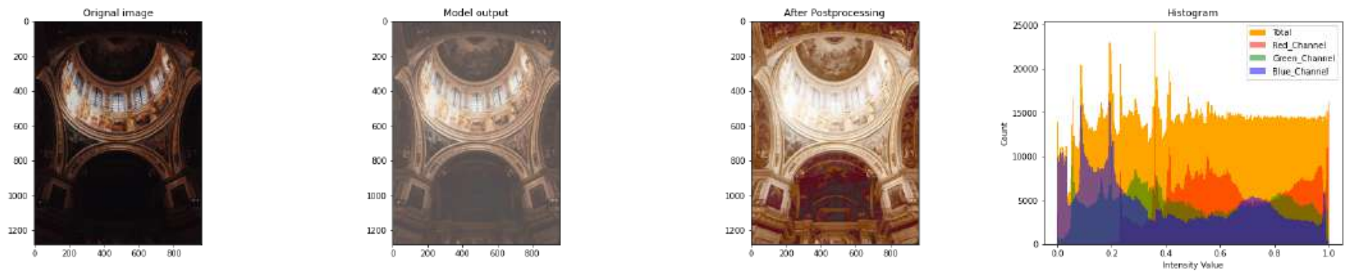
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



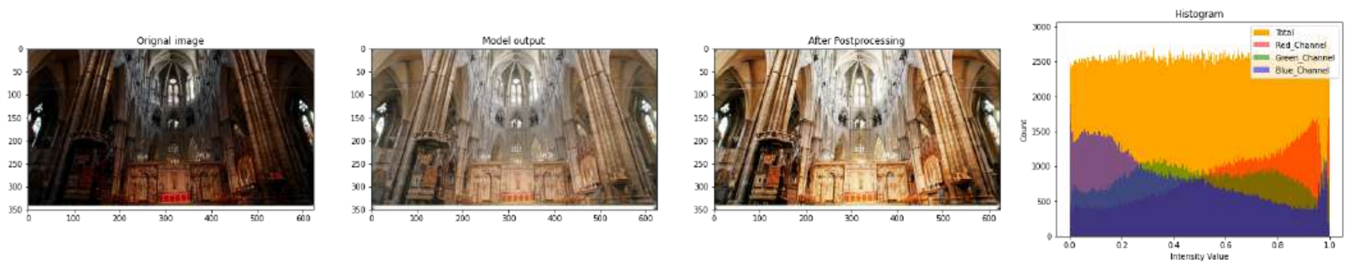
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



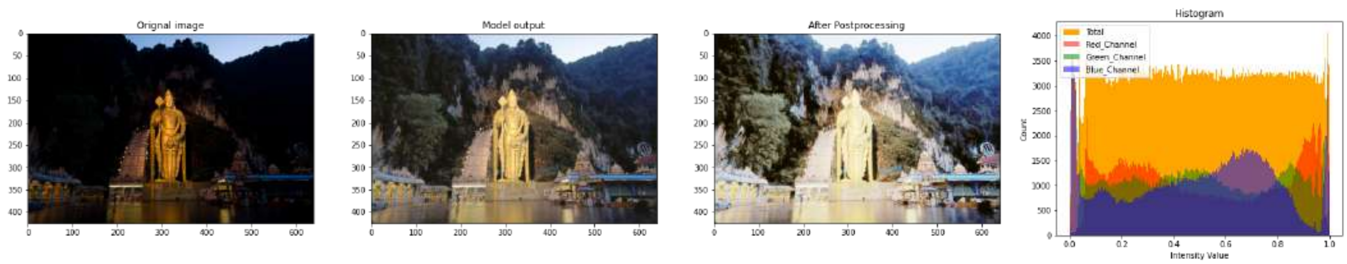
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



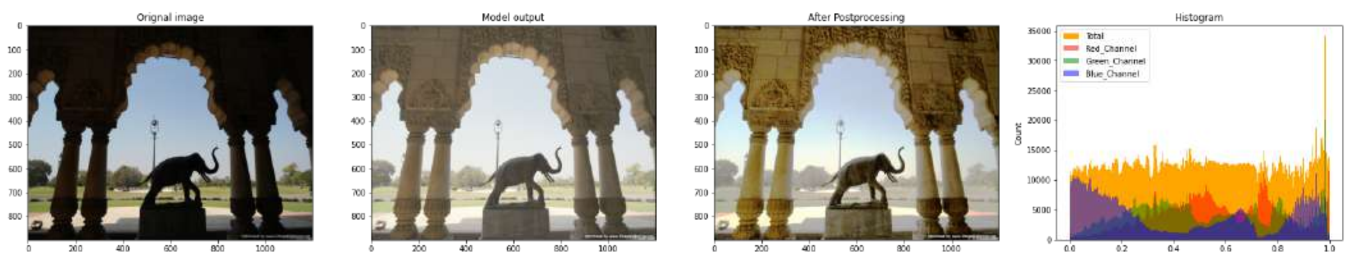
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



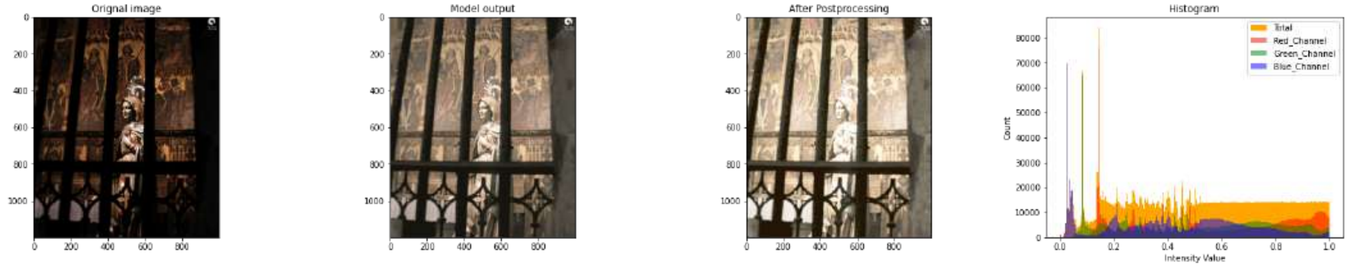
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



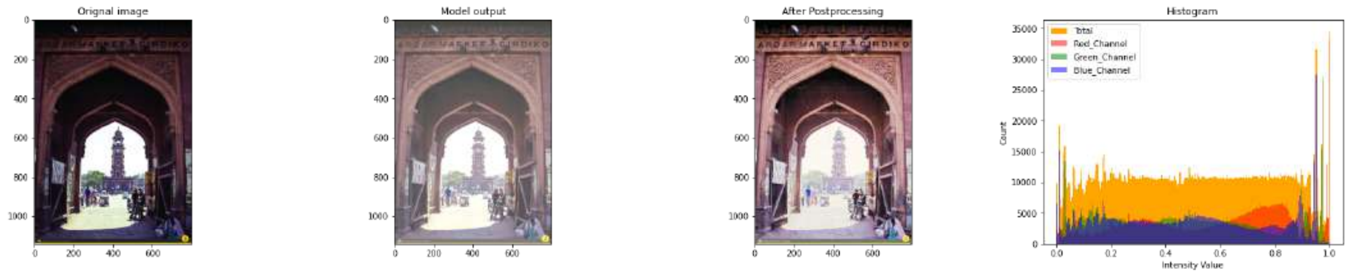
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



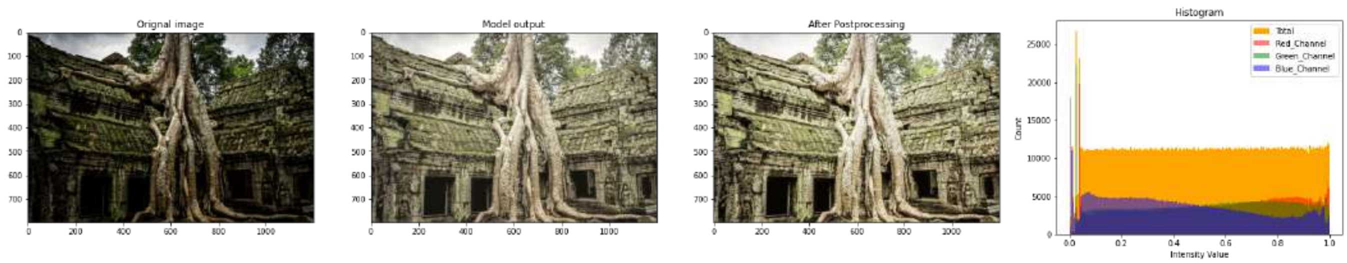
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



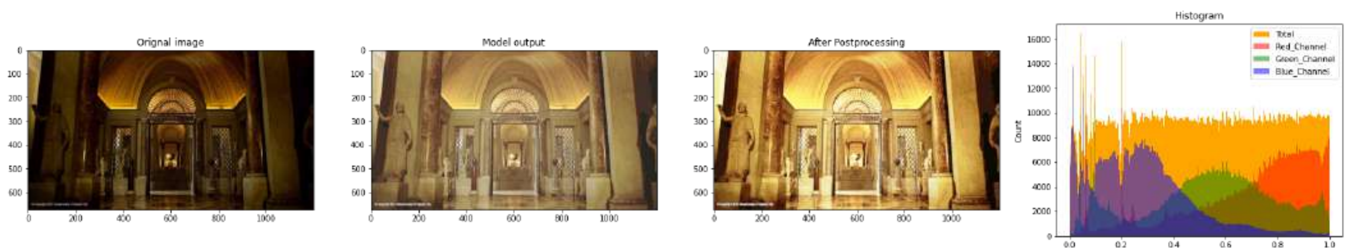
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



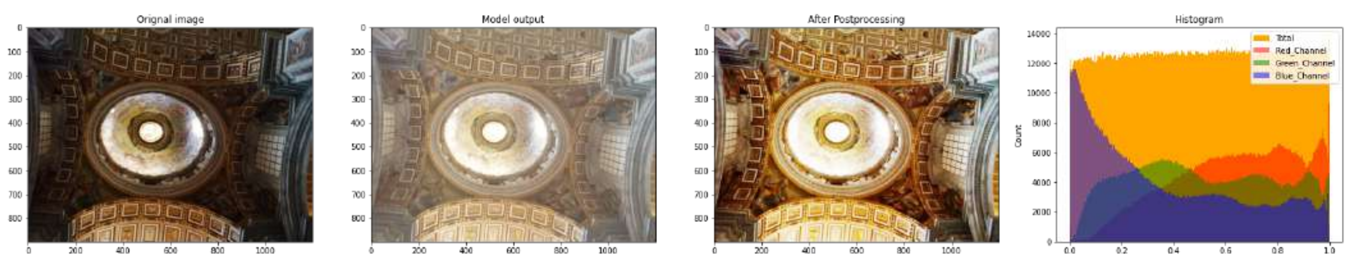
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



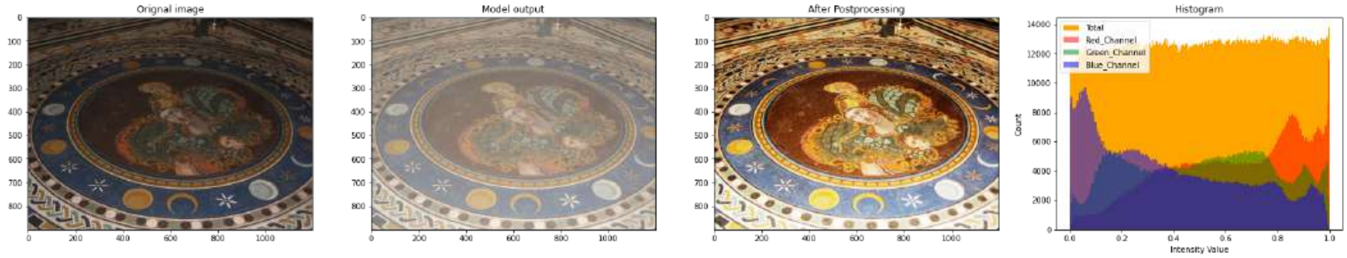
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

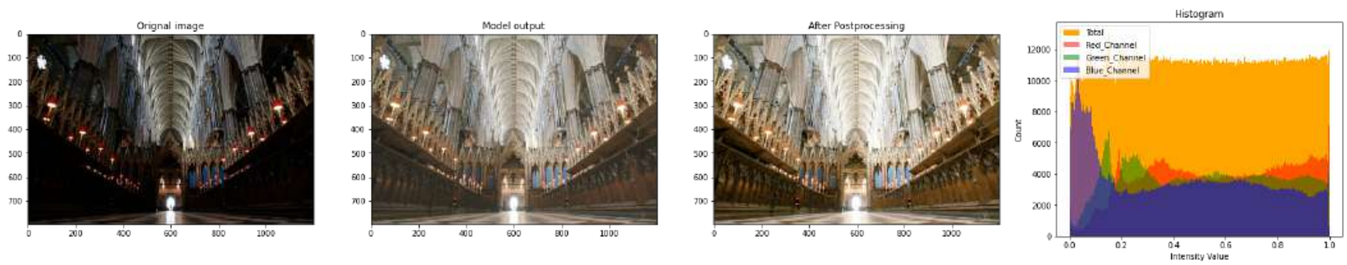


Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

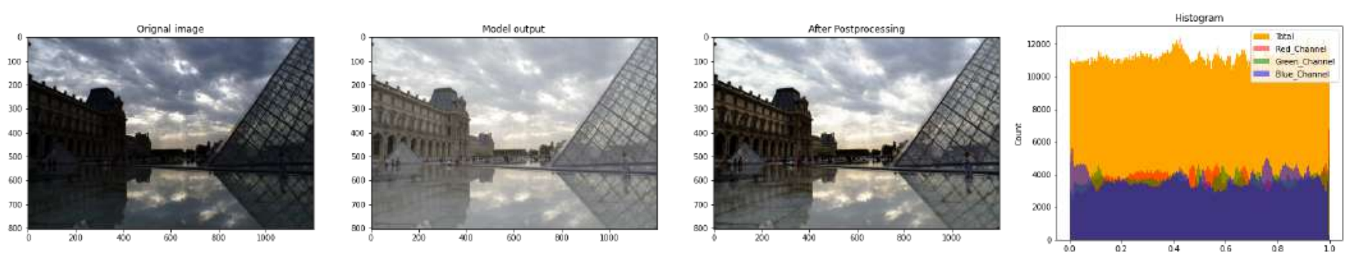


Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

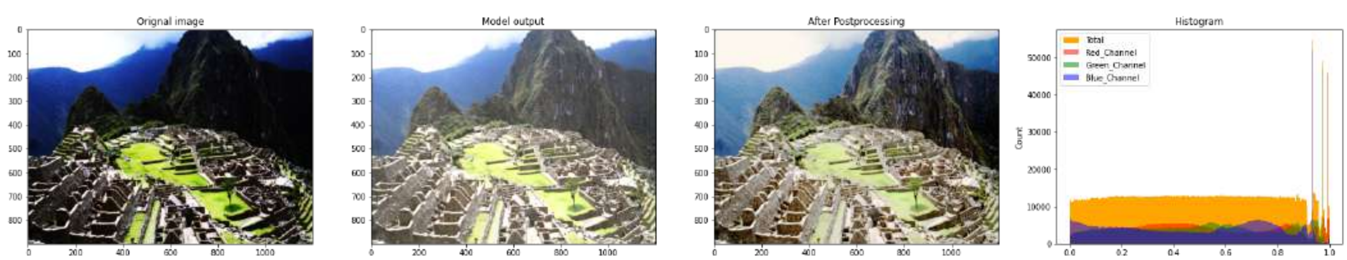
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



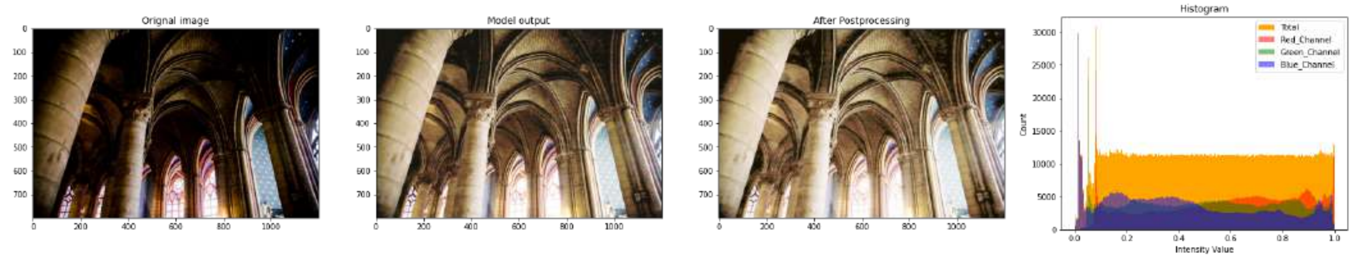
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



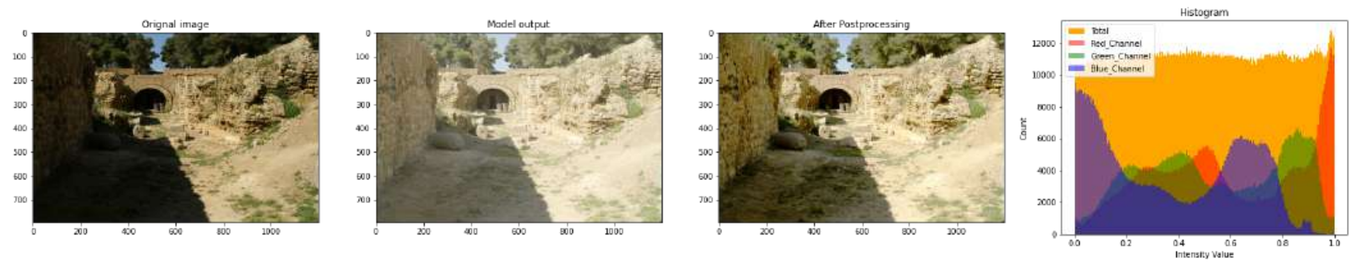
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



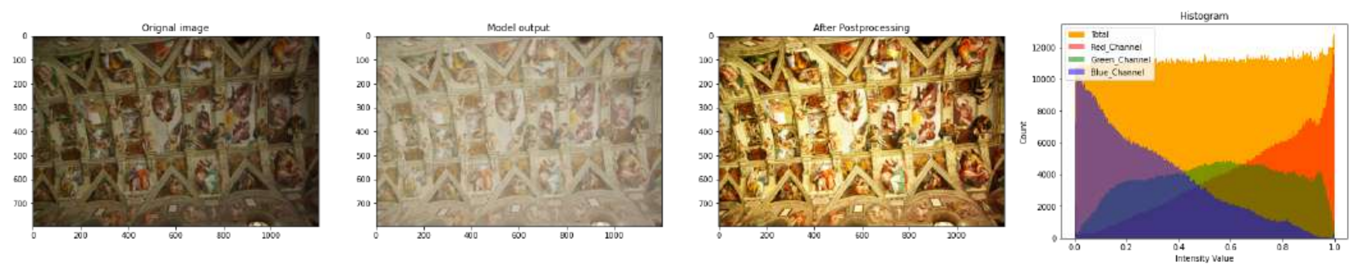
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



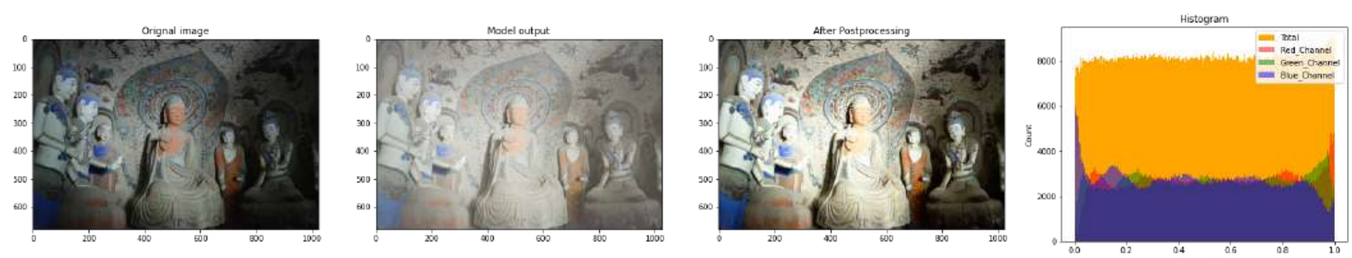
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



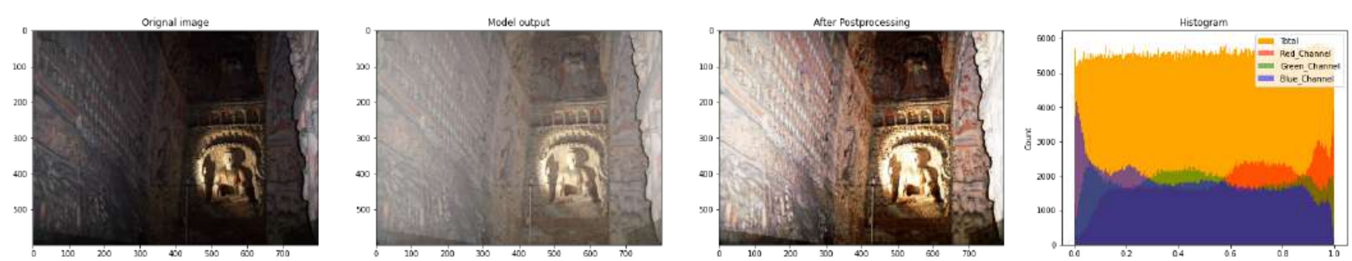
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



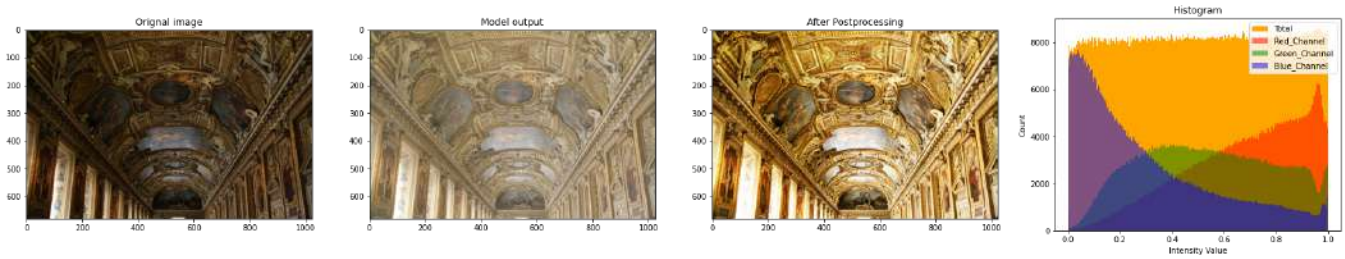
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



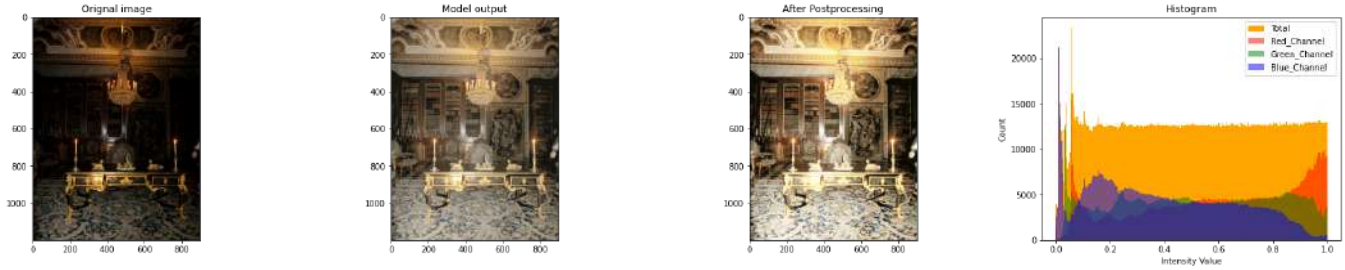
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).



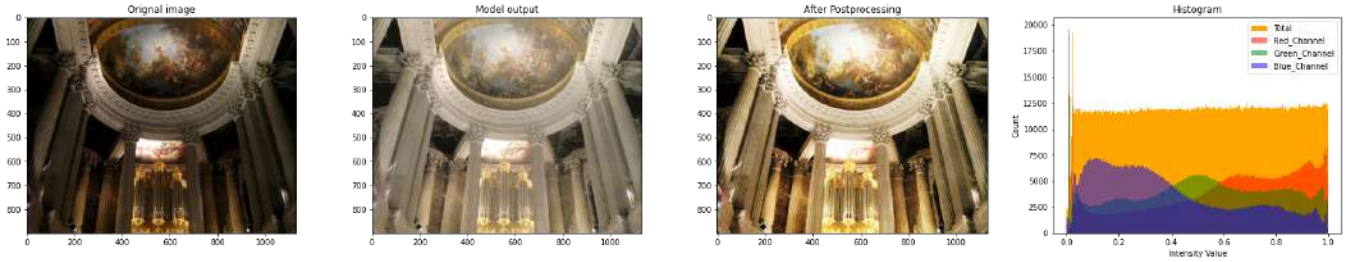
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).




```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



One image testing:

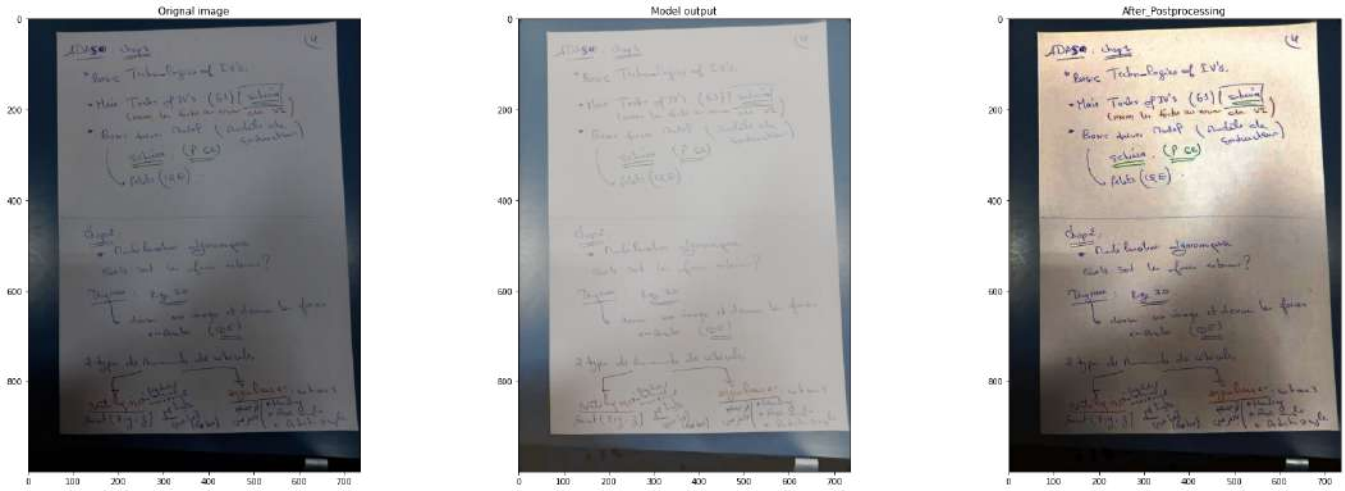
In [ ]:
```
uploaded = files.upload()

original_image_path = list(uploaded.keys())[0]
original_image=imread_color(original_image_path)
mbllen_enhanced_image = predict(original_image[np.newaxis, :])
post_processed_image = skimage.exposure.equalize_hist(mbllen_enhanced_image)

plt.figure(figsize=(30,10))
plt.subplot(131)
plt.title("Orignal image")
plt.imshow(original_image)
plt.subplot(132)
plt.title("Model output")
plt.imshow(mbllen_enhanced_image)
plt.subplot(133)
plt.title("After_Postprocessing")
plt.imshow(post_processed_image)
```

Choose Files No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving 137393249_236965794605361_6500250277301780580_n.jpg to 137393249_23696579
4605361_6500250277301780580_n.jpg

/usr/local/lib/python3.6/dist-packages/skimage/exposure/exposure.py:181: UserWar
ning: This might be a color image. The histogram will be computed on the flatten
ed image. You can instead apply this function to each color channel.
  hist, bin_centers = histogram(image, nbins)
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

Out[ ]: `<matplotlib.image.AxesImage at 0x7f81e3f8a550>`



In [ ]:
```
glob("/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/*.jpg")
```

Out[ ]:
```
['/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/historic_site_rome.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/sculpture.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/painting.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Gunwi_Buddha.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/pantheon_rome.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/cathedral_carthage.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/painting_2.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/painting_3.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/gallery_florence.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/basilica_saint_peter_rome.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/plack.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/apollo.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/china_wall.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/castleriggs_england.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/port_arthur.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/banff_cave_basin_canada.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/terracotta.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/temple_cambodge_2.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/grand_canion.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/inside_mosq_turkey.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/arc_triomphe_paris.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/bare_island_sydney.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Petra_jordan_1.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Petra_jordan_2.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/cave.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/london.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Saint-Petersburg.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Westminster_Abbey.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/batu_caves_Malaysia.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/champaner_temple_india.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/catedral_barcelona_virgin_mary.jpg',
```

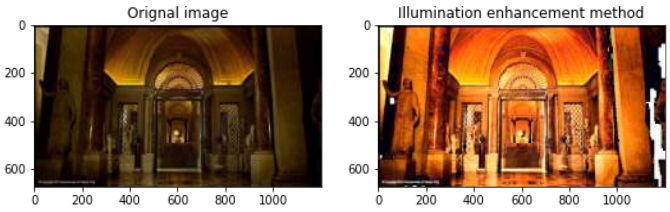
```
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/sadar_indi
a.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/angkor_tem
ple_ cambodge_1.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/vatican_mu
seum_rome.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/dome_marse
ille.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/mosaic_flo
or_vatican.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/graveyard.
jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/inside abb
ey.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/louvre_mus
eum_outside.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/machu-pich
u-peru.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/paris-notr
e_dame.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/roman_site
s_tunisia.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Sistine Ch
apel ceiling.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/mogao_cave
s_chine_2.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/mogao_cave
s_china.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/louvre_mus
eum_inside.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/library_ve
rsailles_palace.jpg',
 '/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/versailles
_palace.jpg']
```

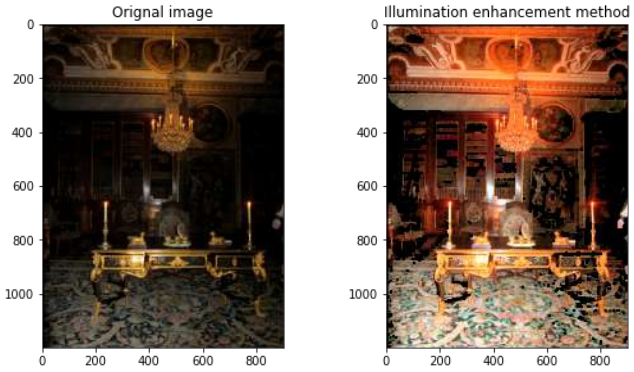
Testing illumination method

In [ ]:

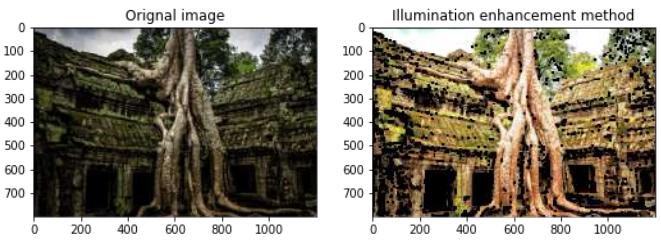
```
path_test=glob("/content/drive/MyDrive/New Drive/Projet transversal/pictures su
itable for illumination method/*.jpg")
random.shuffle(path_test)

for image_index in range(len(path_test)):
  if image_index %2:
    print("The image is called ", path_test[image_index] )
    original_image_path = path_test[image_index]
    original_image=imread_color(original_image_path)

    original_image_2 = mpimg.imread(original_image_path)
    illumination_enhancement_image = illumination_enhancement_algorithm(origina
l_image_2)

    plt.figure(figsize=(20,5))
    plt.subplot(141)
    plt.title("Orignal image")
    plt.imshow(original_image)

    plt.subplot(142)
    plt.title("Illumination enhancement method")
    plt.imshow(illumination_enhancement_image)

    image_index +=1

    original_image_path = path_test[image_index]
    original_image=imread_color(original_image_path)

    original_image_2 = mpimg.imread(original_image_path)
    illumination_enhancement_image = illumination_enhancement_algorithm(origina
l_image_2)

    plt.subplot(143)
    plt.title("Orignal image")
    plt.imshow(original_image)

    plt.subplot(144)
    plt.title("Illumination enhancement method")
    plt.imshow(illumination_enhancement_image)

  plt.show()
```

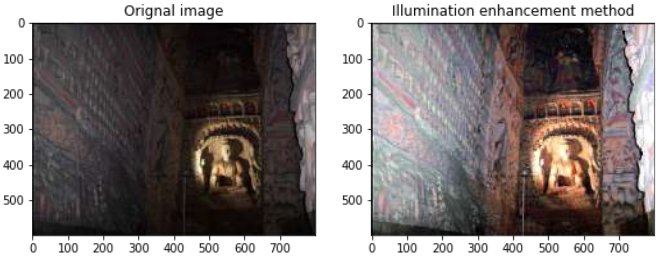
```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/picture
s suitable for illumination method/vatican_museum_rome.jpg

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:10: RuntimeWarning:
invalid value encountered in true_divide
  # Remove the CWD from sys.path while we load stuff.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:11: RuntimeWarning:
divide by zero encountered in log10
  # This is added back by InteractiveShellApp.init_path()
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



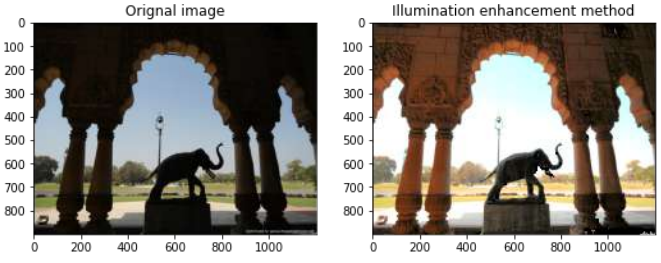


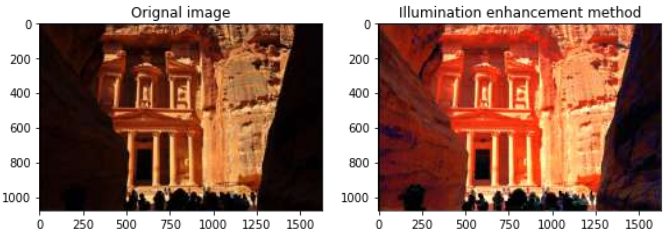

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/picture
s suitable for illumination method/angkor_temple_ cambodge_1.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```





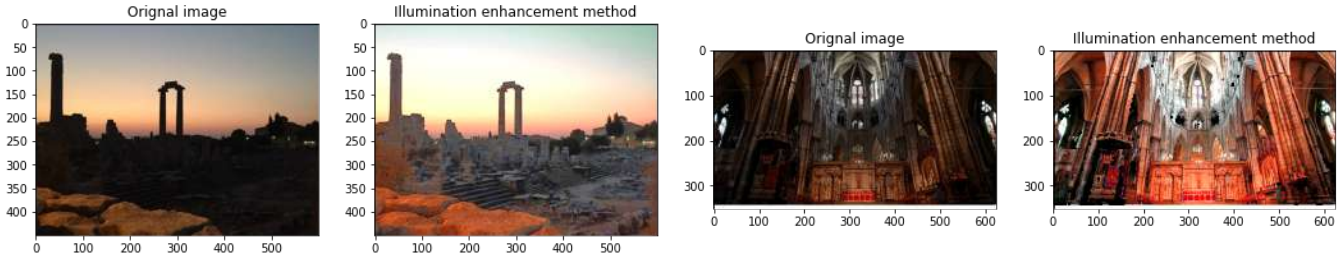

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/picture
s suitable for illumination method/champaner_temple_india.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```






```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/picture
s suitable for illumination method/apollo.jpg
```

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

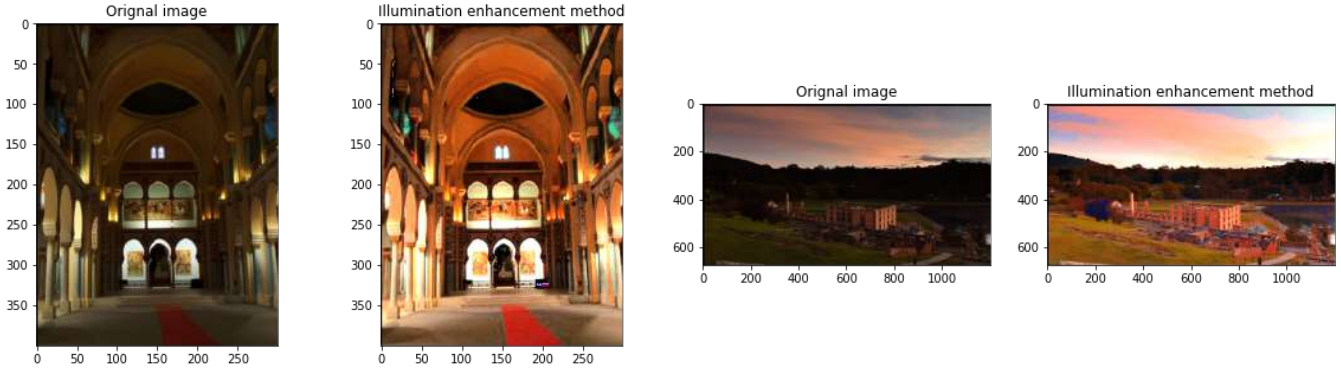


The image is called /content/drive/MyDrive/New Drive/Projet transversal/pictures suitable for illumination method/Sistine Chapel ceiling.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

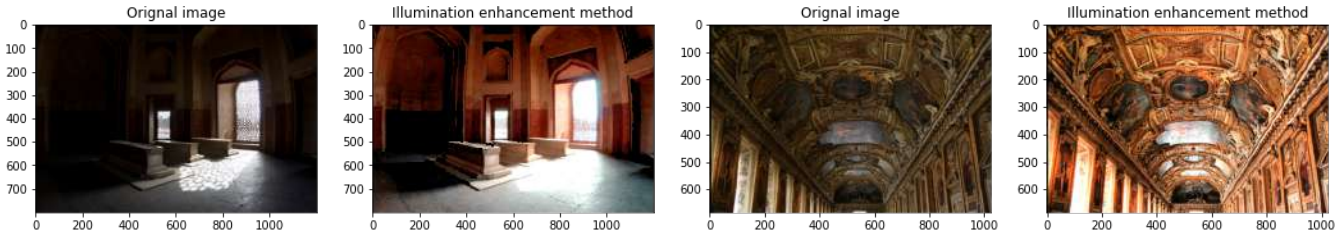
The image is called /content/drive/MyDrive/New Drive/Projet transversal/pictures suitable for illumination method/cathedral_carthage.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



The image is called /content/drive/MyDrive/New Drive/Projet transversal/pictures suitable for illumination method/graveyard.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

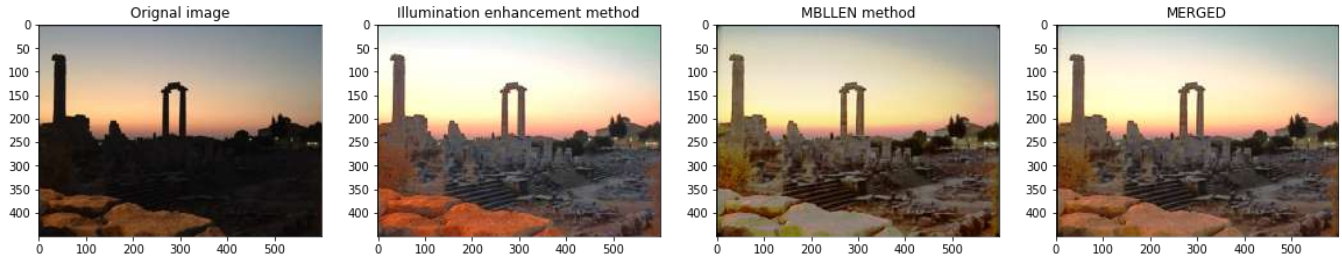


Merging results

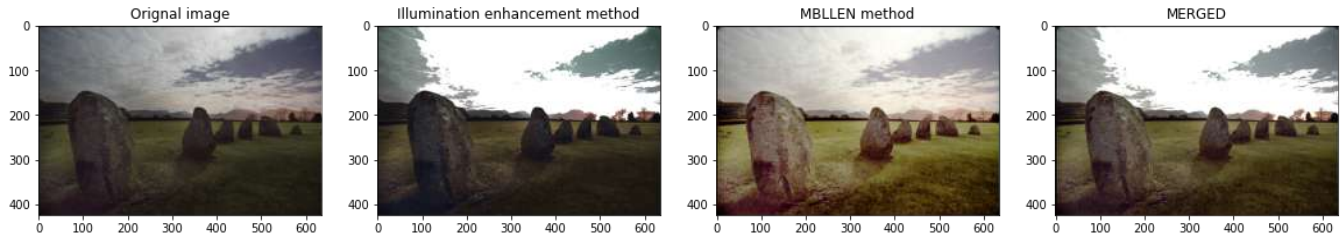
In [ ]:

```
path_test=glob("/content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/*.jpg")
random.shuffle(path_test)

for image_index in range(len(path_test)):
  print("The image is called ", path_test[image_index] )
  original_image_path = path_test[image_index]
  original_image=imread_color(original_image_path)
  mbllen_enhanced_image = predict(original_image[np.newaxis, :])
  post_processed_image = skimage.exposure.equalize_hist(mbllen_enhanced_image)

  original_image_2 = mpimg.imread(original_image_path)
  illumination_enhancement_image = illumination_enhancement_algorithm(original_image_2)

  plt.figure(figsize=(20,5))
  plt.subplot(141)
  plt.title("Orignal image")
  plt.imshow(original_image)

  plt.subplot(142)
  plt.title("Illumination enhancement method")
  plt.imshow(illumination_enhancement_image)

  plt.subplot(143)
  plt.title("MBLLEN method")
  plt.imshow(post_processed_image)

  plt.subplot(144)
  plt.title("MERGED")
  b1, g1, r1 = cv.split(illumination_enhancement_image)
  b2, g2, r2 = cv.split(post_processed_image)
  b12 = (b1+b2)/2
  g12 = (g1+g2)/2
  r12 = (r1+r2)/2
  merged_image = cv.merge((b12,g12,r12))
  plt.imshow(merged_image)

  plt.show()
```

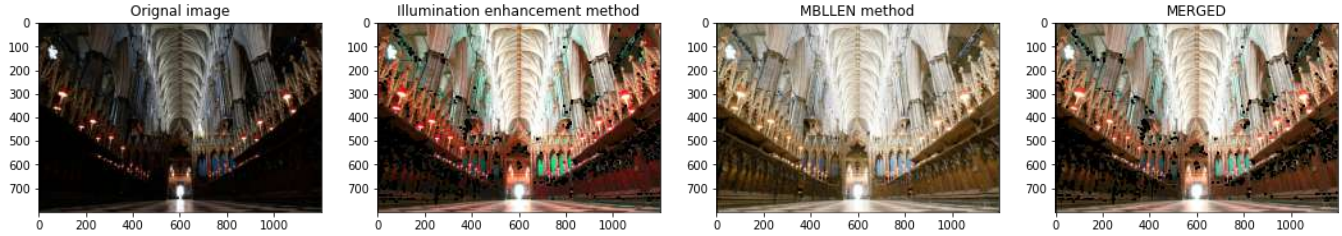
```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/apollo.jpg

/usr/local/lib/python3.6/dist-packages/skimage/exposure/exposure.py:181: UserWar
ning: This might be a color image. The histogram will be computed on the flatten
ed image. You can instead apply this function to each color channel.
  hist, bin_centers = histogram(image, nbins)
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:11: RuntimeWarning:
divide by zero encountered in log10
  # This is added back by InteractiveShellApp.init_path()
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/castlerigg_england.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

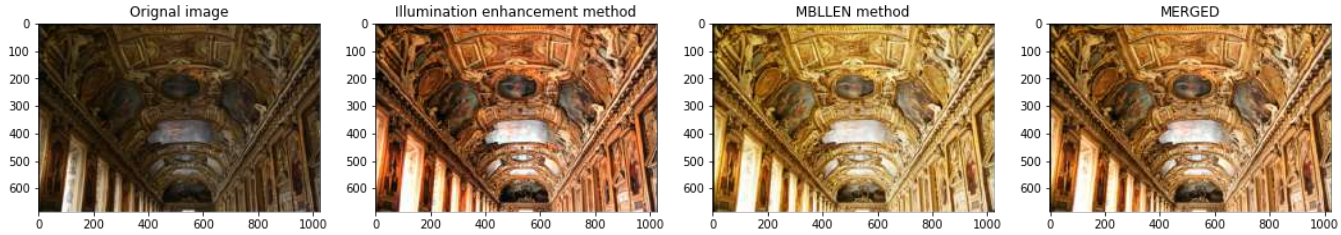



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/inside abbey.jpg

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:10: RuntimeWarning:
invalid value encountered in true_divide
  # Remove the CWD from sys.path while we load stuff.
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

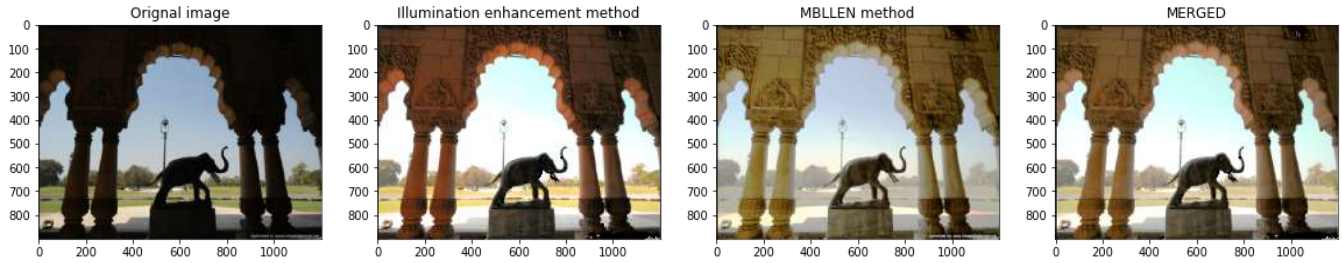



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/louvre_museum_inside.jpg
```

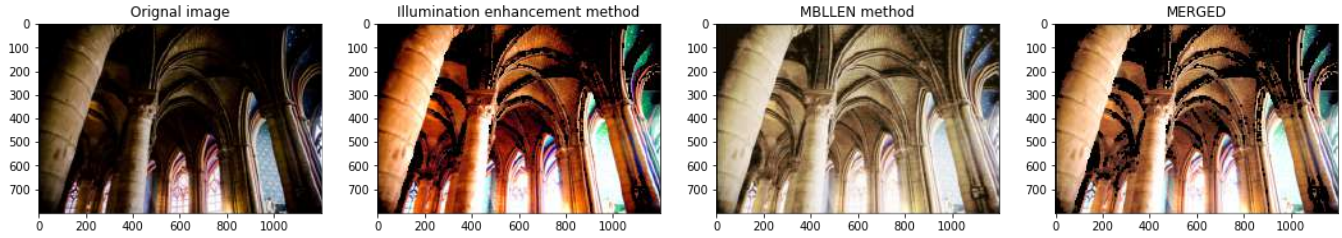
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



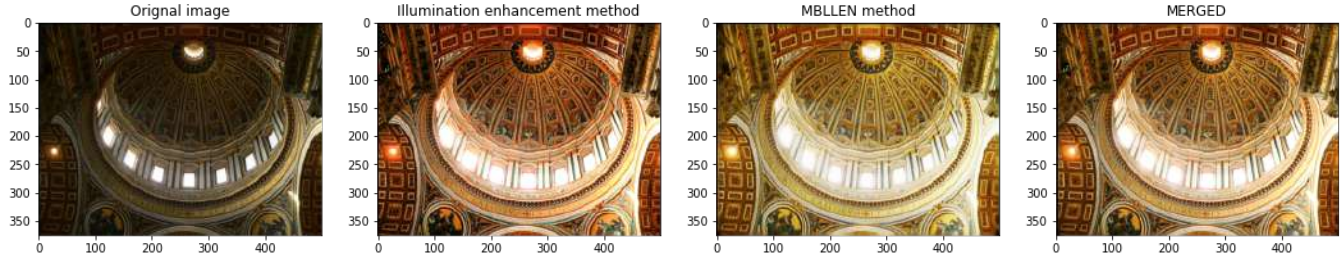

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/champaner_temple_india.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/paris-notre_dame.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

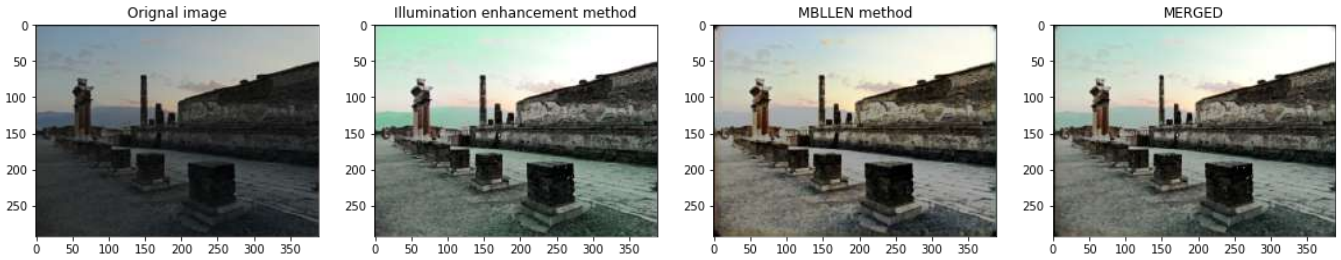



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/basilica_saint_peter_rome.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



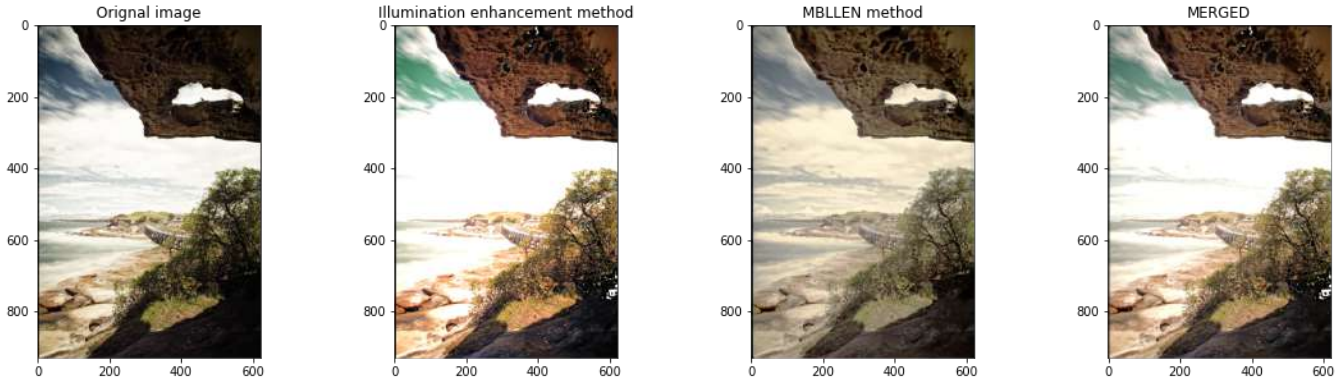
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/historic_site_rome.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



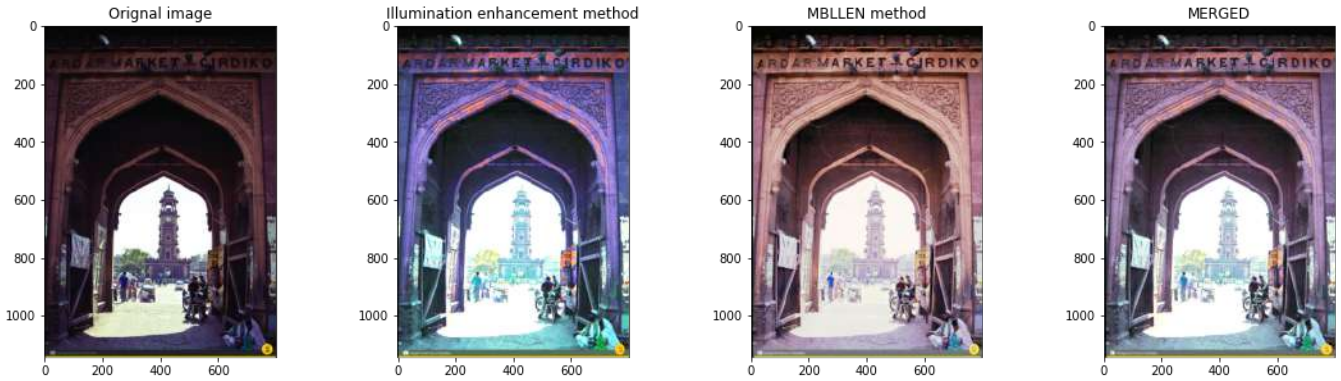
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/bare_island_sydney.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



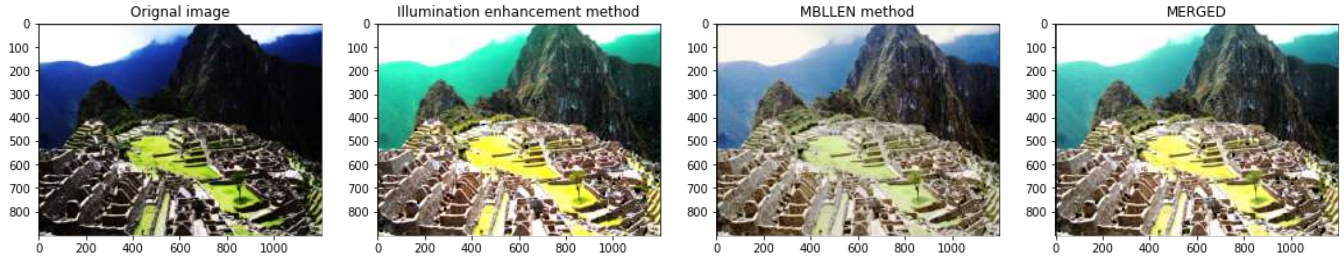
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/sadar_india.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

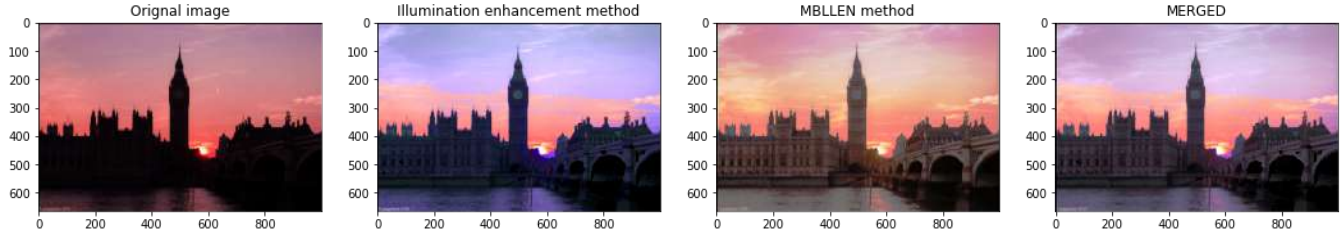


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/machu-pichu-peru.jpg

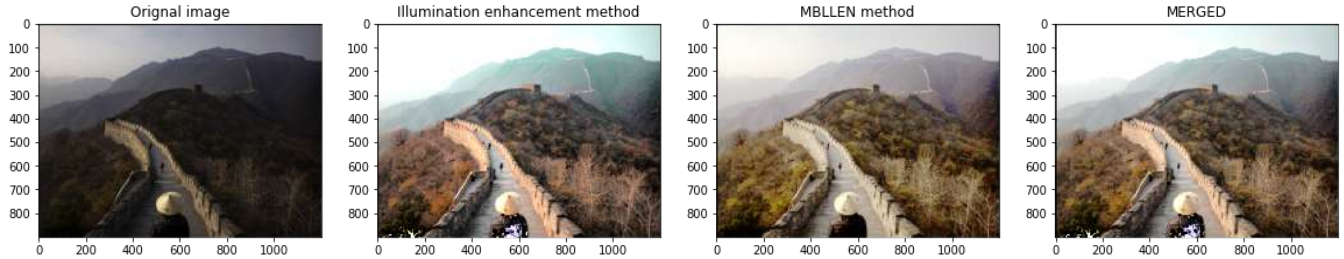
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



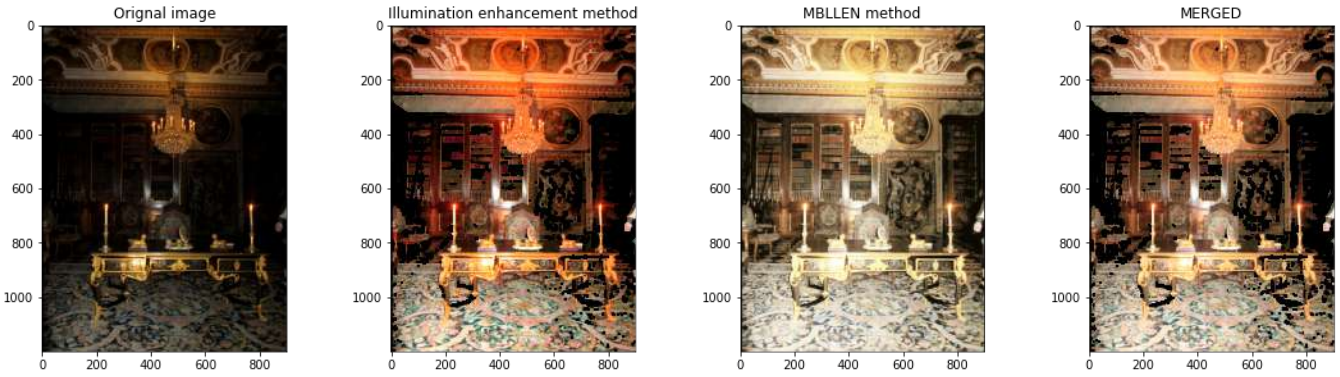

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/london.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/china_wall.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

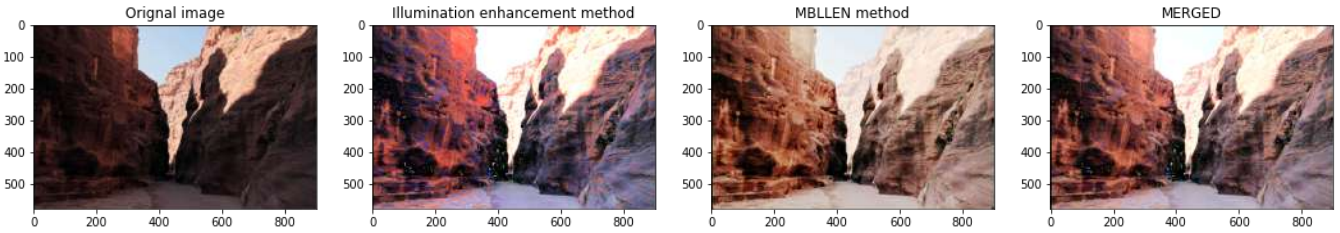



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/library_versailles_palace.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/Petra_jordan_2.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).

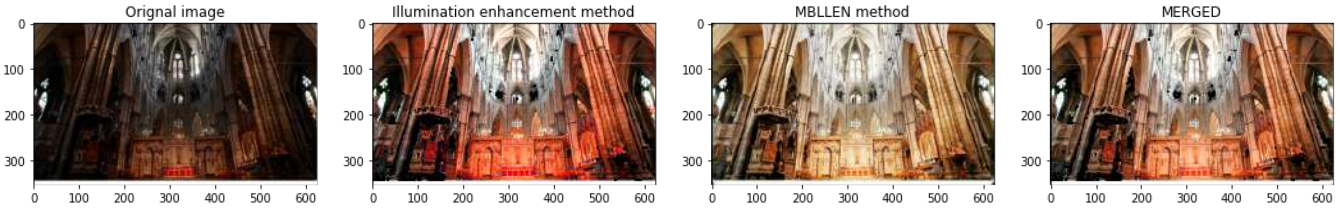


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/painting_3.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).

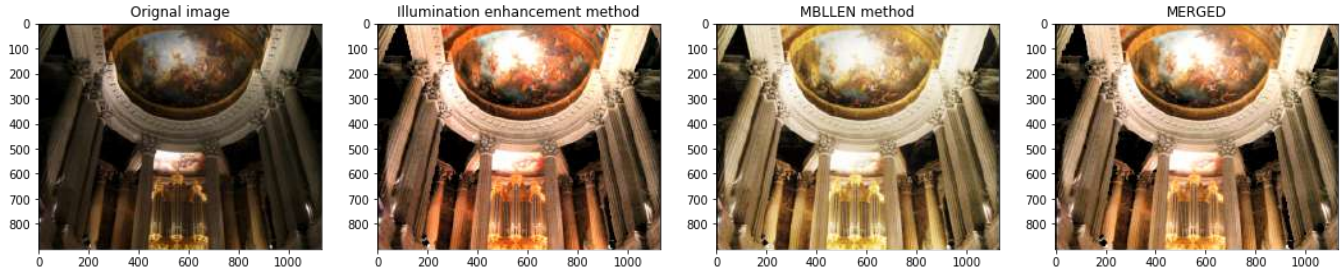
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/Westminster_Abbey.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).

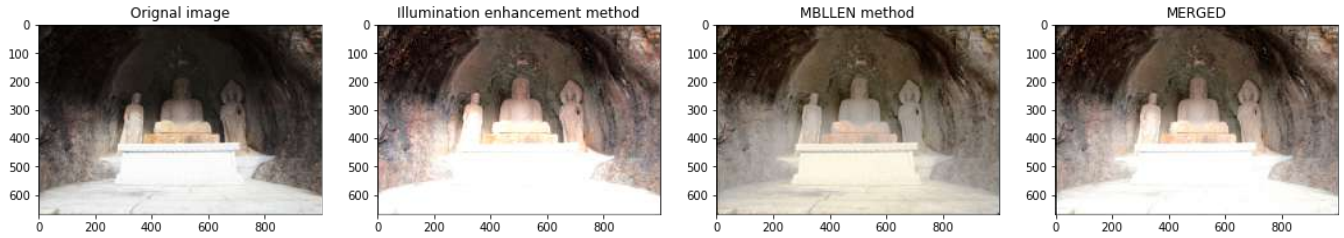


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/versailles_palace.jpg

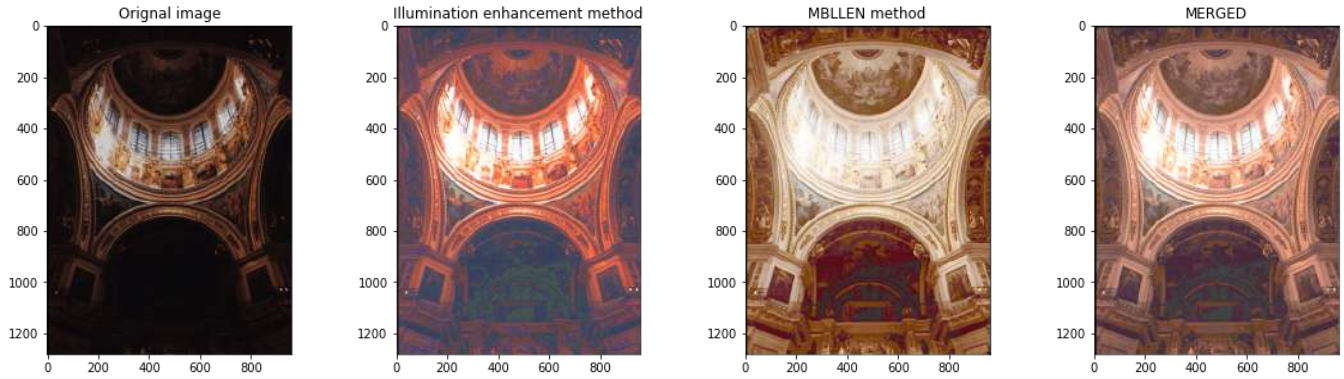
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



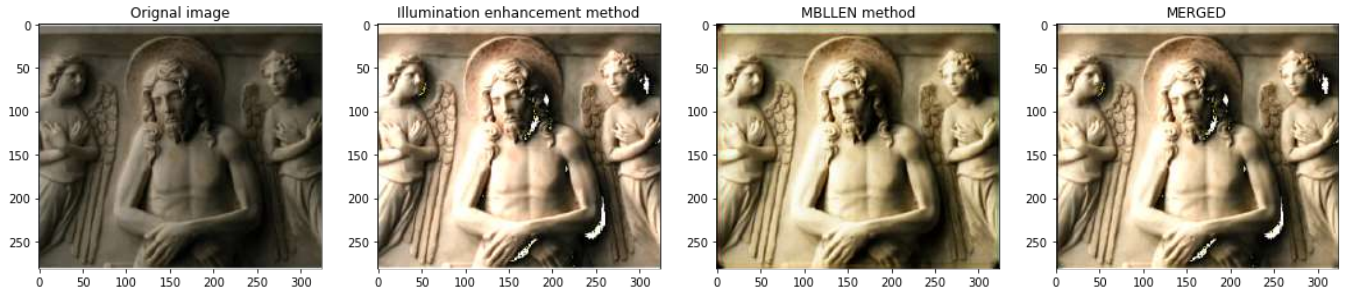

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/Gunwi_Buddha.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/Saint-Petersburg.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

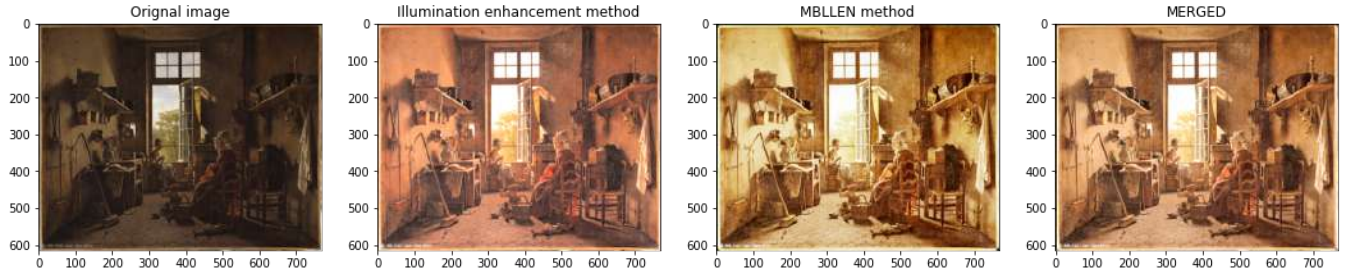



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/sculpture.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



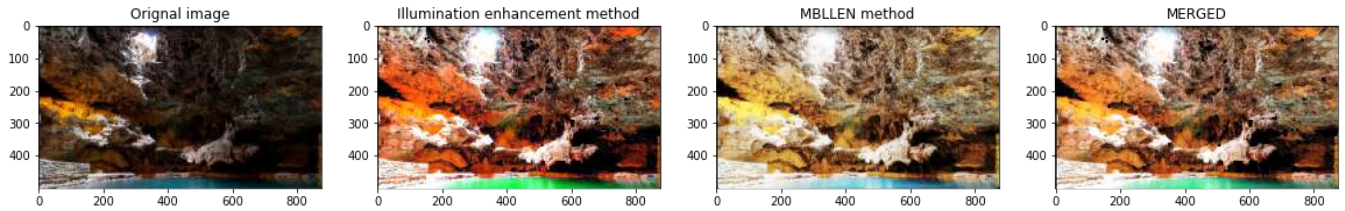
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/painting_2.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).



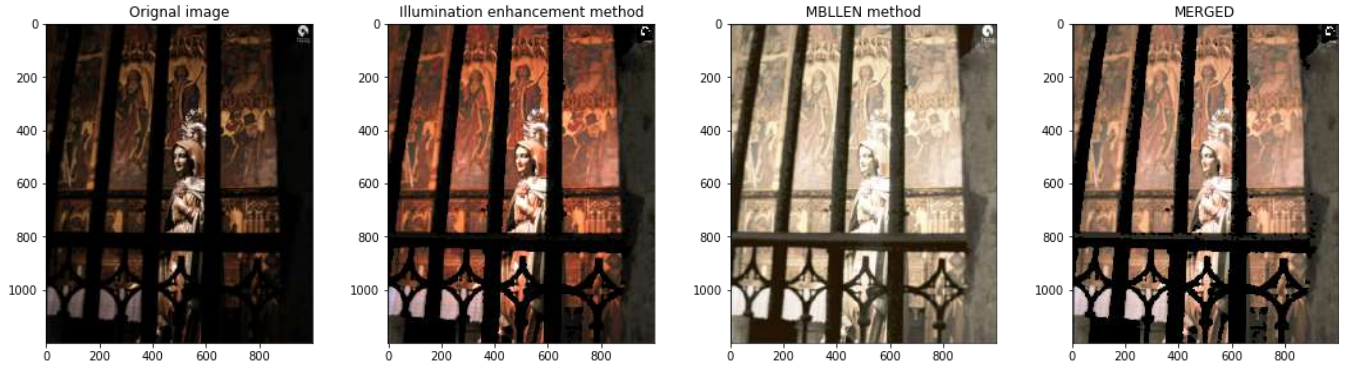
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/banff_cave_basin_canada.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).



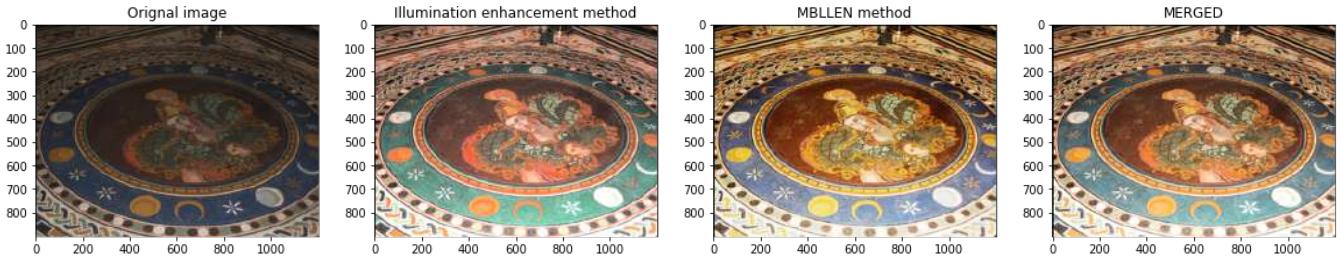
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/catedral_barcelona_virgin_mary.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).



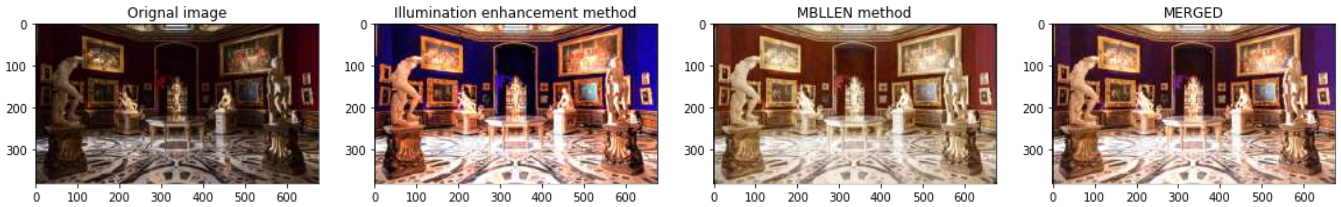
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/mosaic_floor_vatican.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



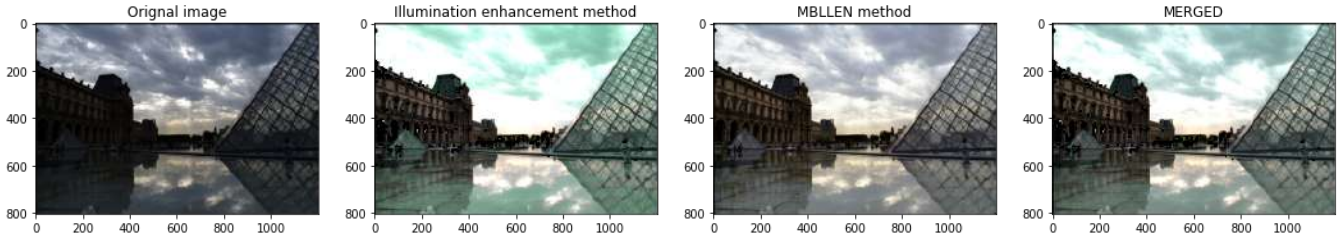
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/gallery_florence.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



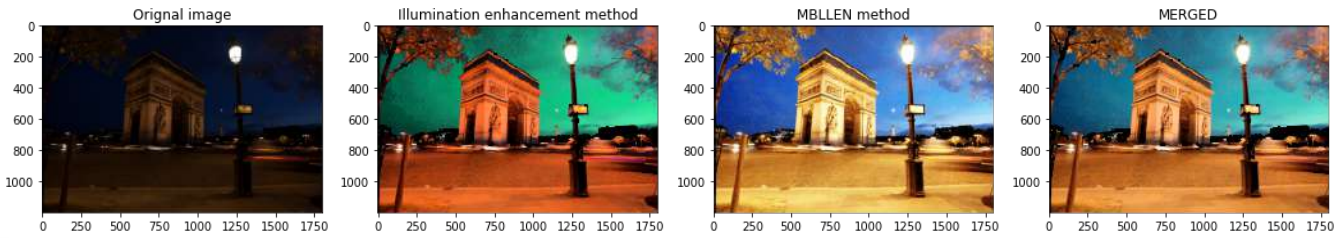
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/louvre_museum_outside.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

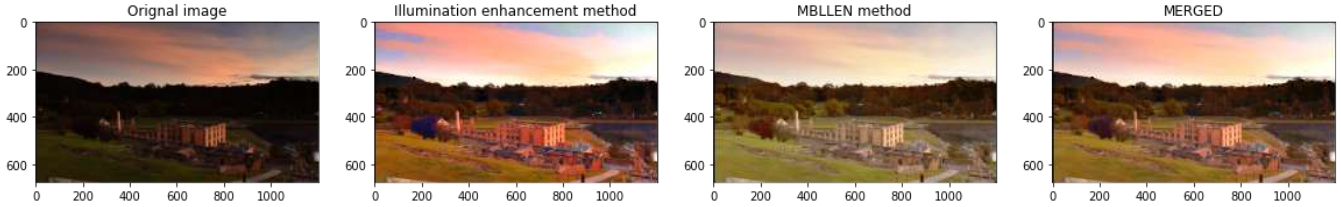


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/arc_triomphe_paris.jpg

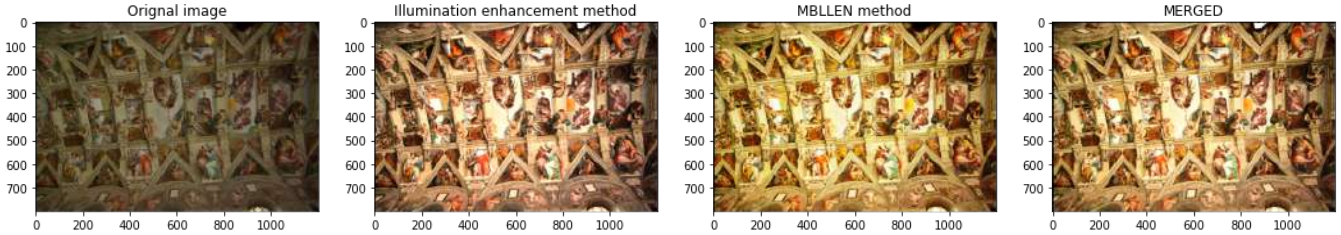
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



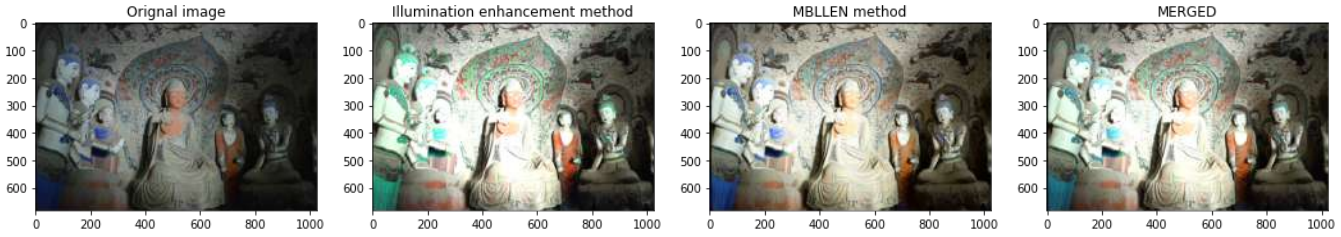
```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/port_arthur.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



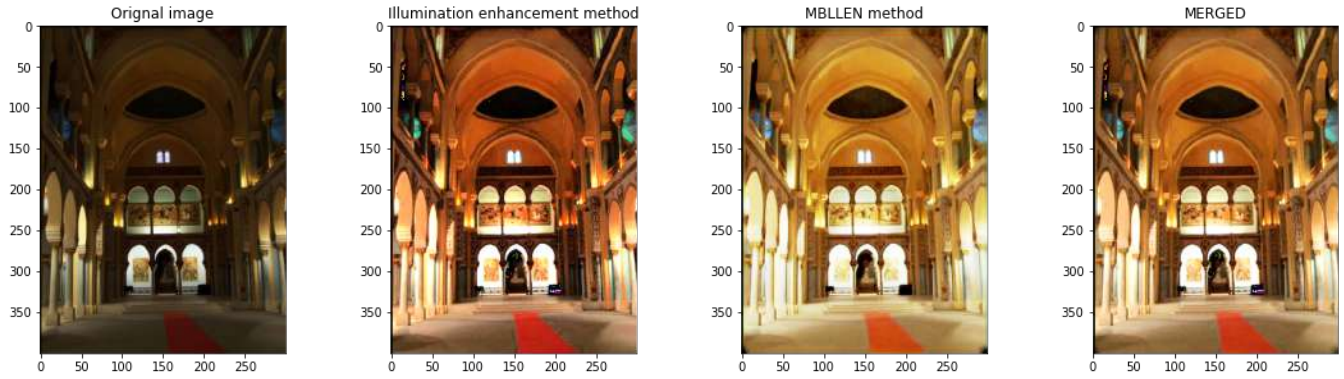

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/Sistine Chapel ceiling.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/mogao_caves_chine_2.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

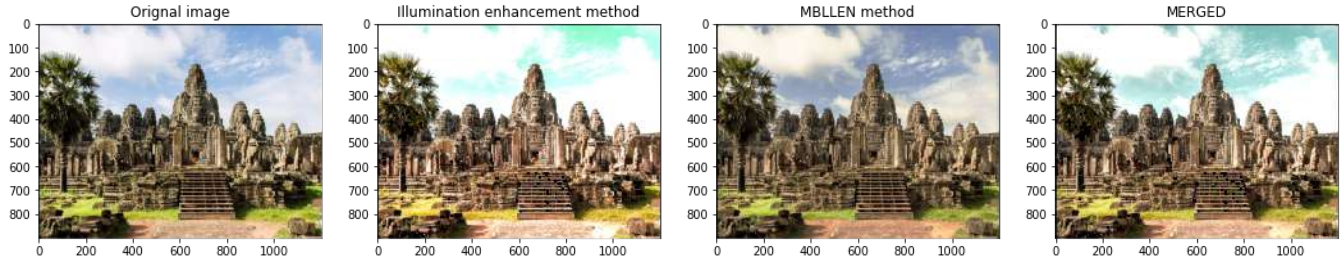



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/cathedral_carthage.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



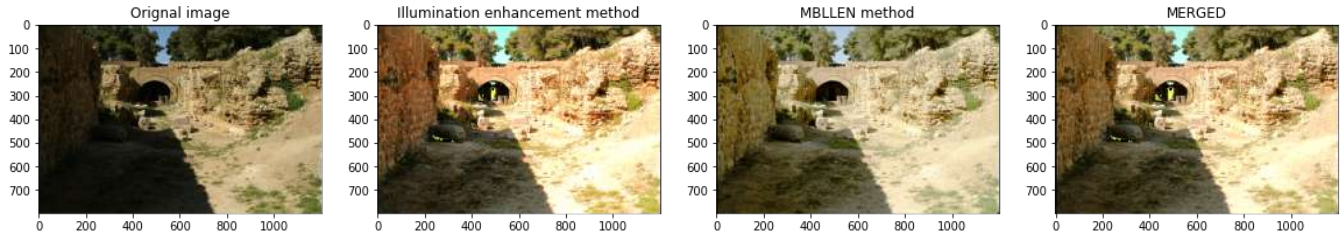
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/temple_cambodge_2.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).



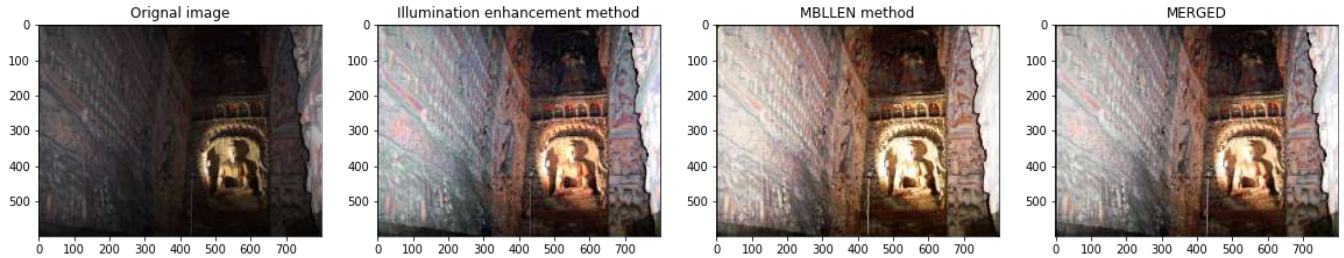
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/roman_sites_tunisia.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).

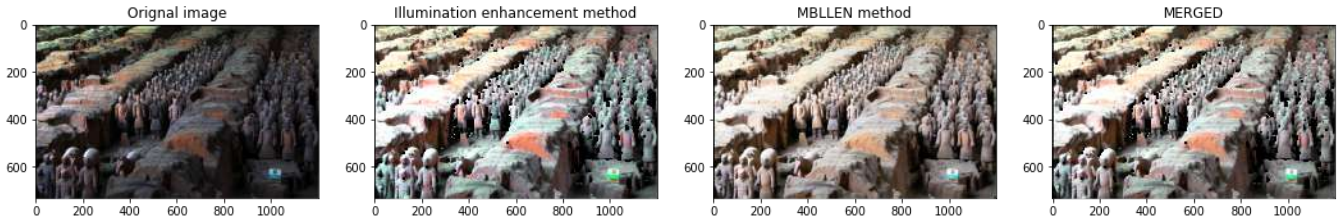


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/mogao_caves_china.jpg

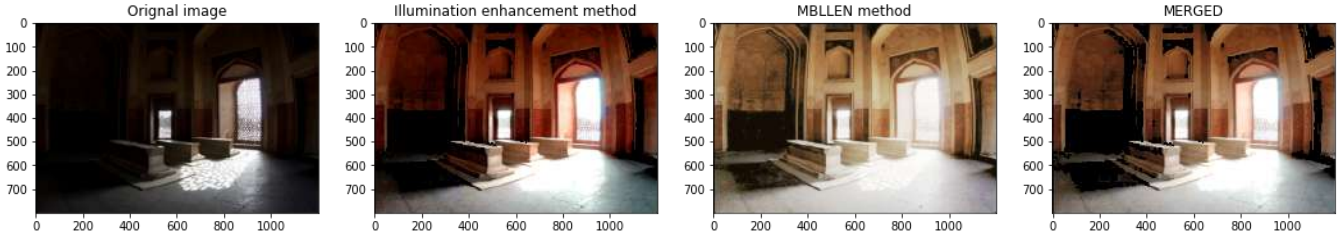
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa ts or [0..255] for integers).



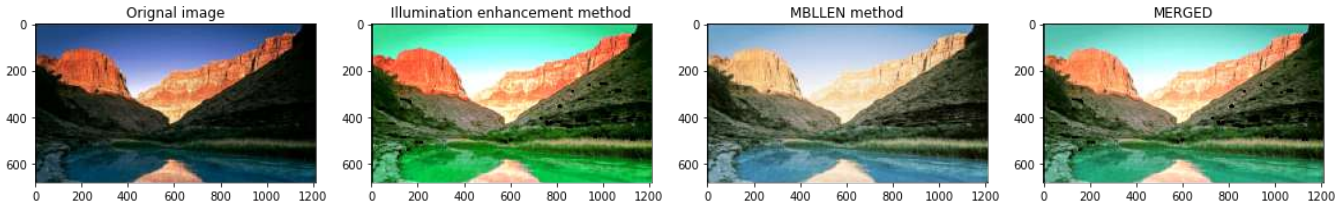
```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/terracotta.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



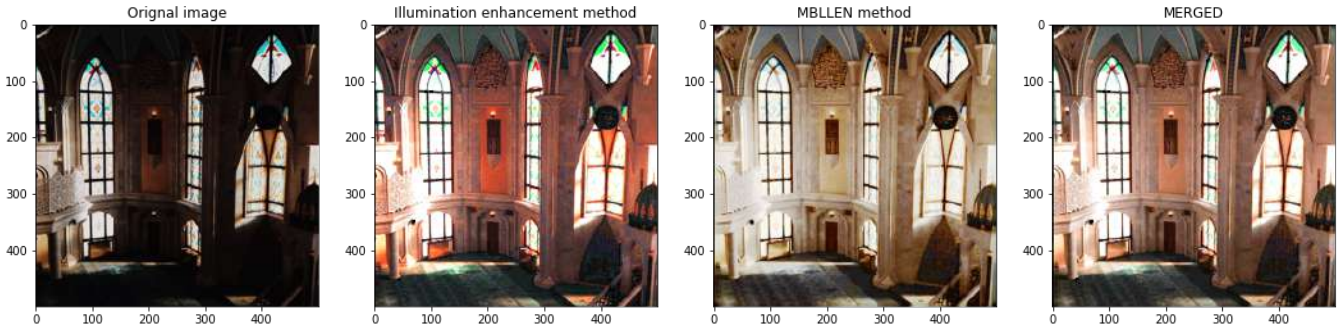

```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/graveyard.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/grand_canion.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```

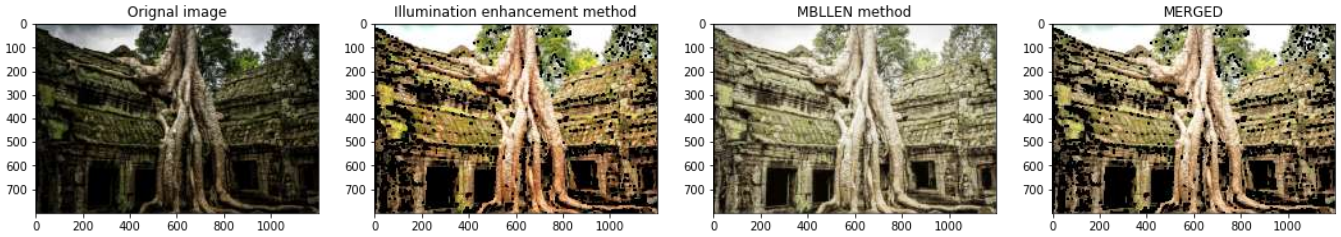



```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/inside_mosq_turkey.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/angkor_temple_ cambodge_1.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

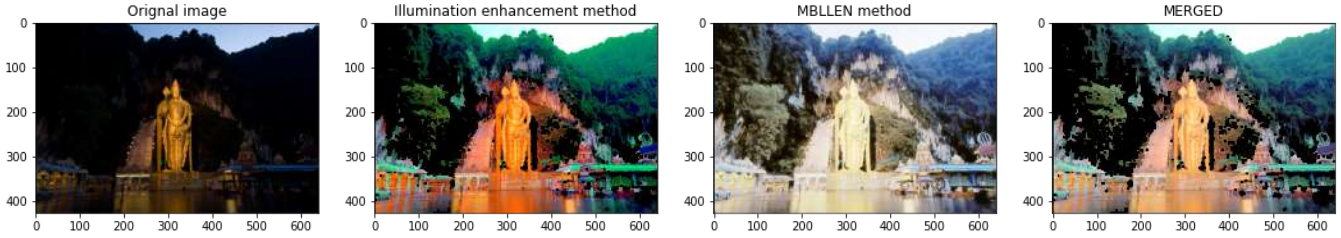


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/plack.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

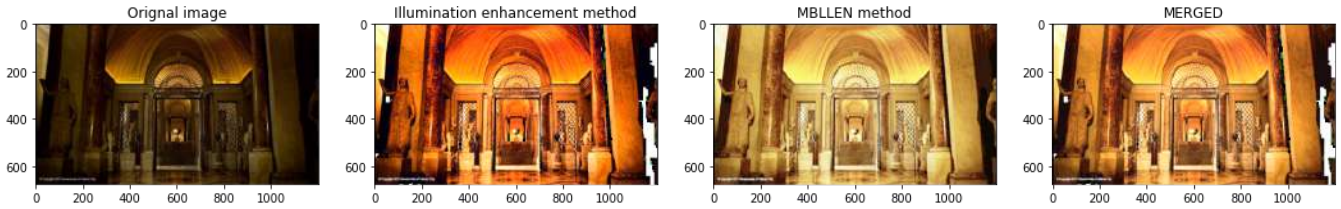
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing _dataset/batu_caves_Malaysia.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



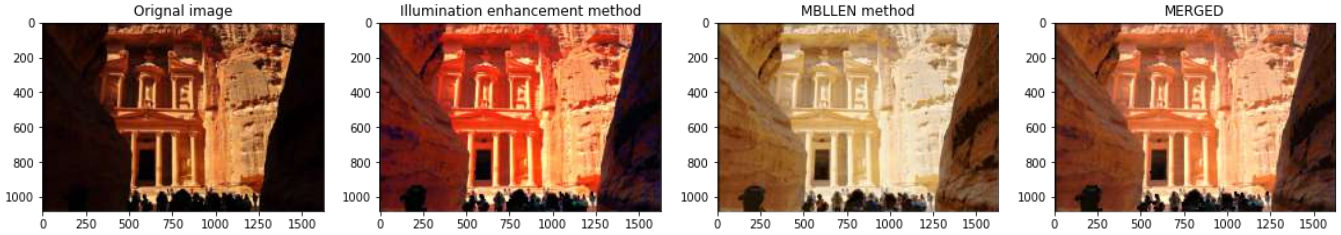
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/vatican_museum_rome.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



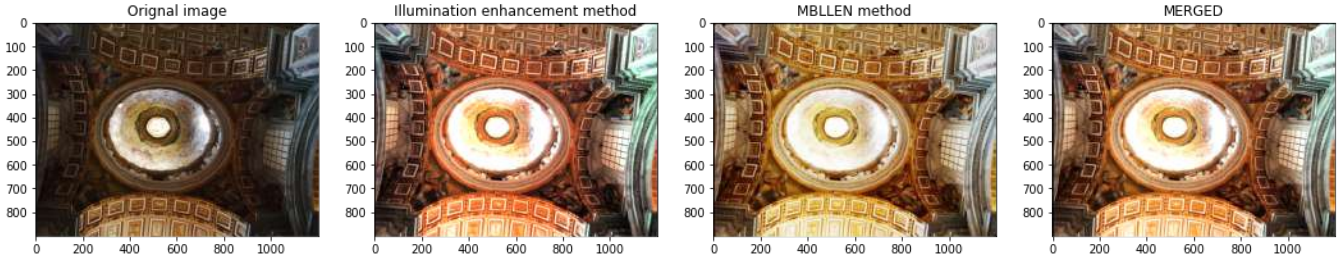
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/Petra_jordan_1.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



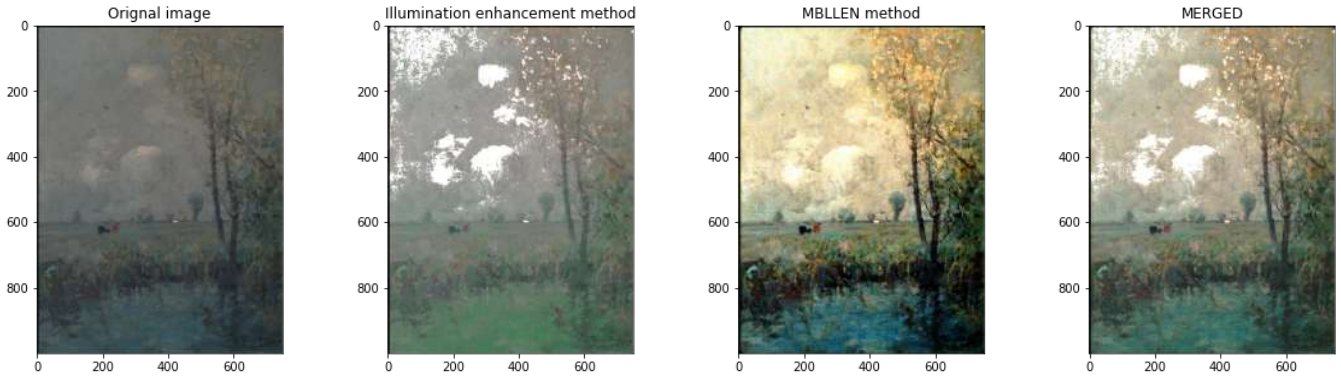
The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/dome_marseille.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).

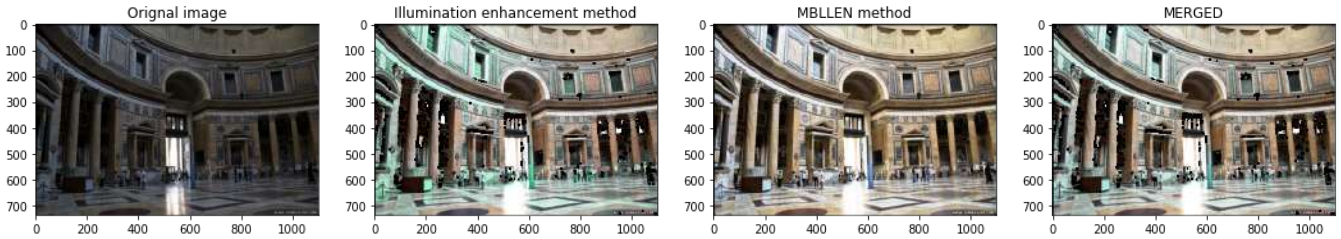


The image is called /content/drive/MyDrive/New Drive/Projet transversal/Testing_dataset/painting.jpg

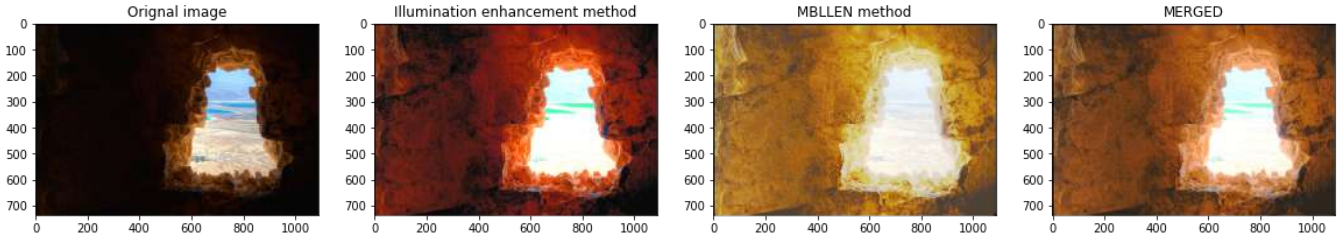
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/pantheon_rome.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```




```
The image is called  /content/drive/MyDrive/New Drive/Projet transversal/Testing
_dataset/cave.jpg

Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
ts or [0..255] for integers).
```



In [ ]: