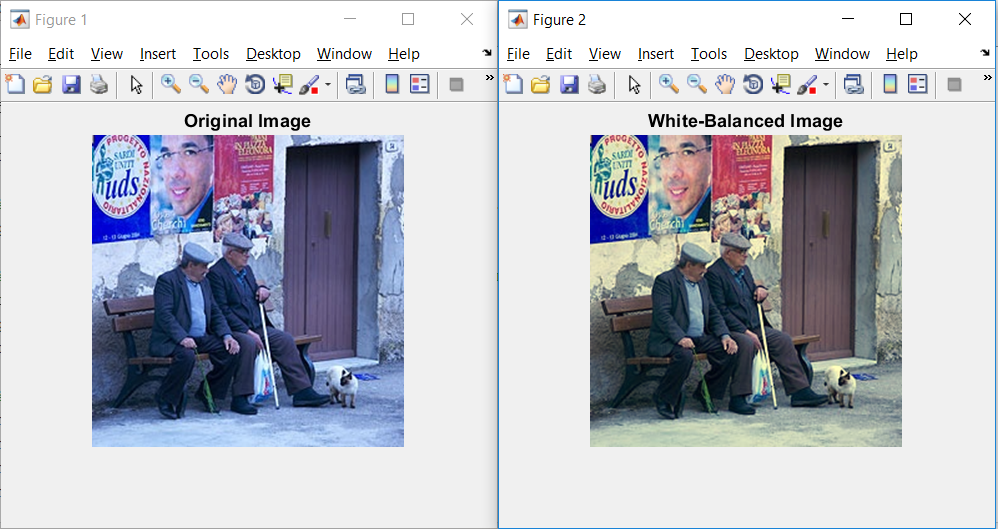
**Thomas Jung/Computer Vision/Hw1**

**Matlab:**



Matlab source code:

imgLoaded = imread('white\_balance\_example.png');

figure

imshow(imgLoaded,'InitialMagnification',100)

title('Original Image')

% Convert to gray so we can get the mean luminance.

grayImage = rgb2gray(imgLoaded);

% Extract the individual red, green, and blue color channels.

redChannel = imgLoaded(:, :, 1);

greenChannel = imgLoaded(:, :, 2);

blueChannel = imgLoaded(:, :, 3);

% Obtaining mean value of each color

meanR = mean2(redChannel);

meanG = mean2(greenChannel);

meanB = mean2(blueChannel);

meanGray = mean2(grayImage);

% Make all channels have the same mean

redChannel = uint8(double(redChannel) \* meanGray / meanR);

greenChannel = uint8(double(greenChannel) \* meanGray / meanG);

blueChannel = uint8(double(blueChannel) \* meanGray / meanB);

% Recombine separate color channels into a single, true color RGB image.

rgbImage = cat(3, redChannel, greenChannel, blueChannel);

figure(2)

imshow(rgbImage)

title('White-Balanced Image')

**Python:**



**Python Source Code:**

# -\*- coding: utf-8 -\*-

import cv2 as cv

import numpy as np

def show(final):

print('display')

cv.imshow('Images', final)

cv.waitKey(0)

cv.destroyAllWindows()

# loading the image

img = cv.imread('C:\Users\PC\Desktop\spring 2018\comp vision\proj 1\white\_balance\_example.png')

# coloring the image

final = cv.cvtColor(img, cv.COLOR\_BGR2LAB)

# obtaining avg color

avg\_a = np.average(final[:, :, 1])

avg\_b = np.average(final[:, :, 2])

for x in range(final.shape[0]):

for y in range(final.shape[1]):

l, a, b = final[x, y, :]

# fix for CV correction

l \*= 100 / 255.0

final[x, y, 1] = a - ((avg\_a - 128) \* (l / 100.0) \* 1.1)

final[x, y, 2] = b - ((avg\_b - 128) \* (l / 100.0) \* 1.1)

final = cv.cvtColor(final, cv.COLOR\_LAB2BGR)

final = np.hstack((img, final))

show(final)

cv.imwrite('result.jpg', final)