Report

1. Gray world algorithm implementation:

The Gray World assumption is based on the observation that on average, the colors of a scene tend to gather around a neutral gray point when averaged over a large enough area.

First, we need to get the total sum of each color plane (R, G, B). Compute the average of each color plane and get the gray value of the image which is assumed to be the neutral gray over the image. After this, we can get the scales of each plane to adjust the color constancy.

```
// do gray world
for(int y=0; y<height; ++y) {
    for(int x=0; x<width; ++x) {
        int offset = y * real_row_width + x * channels;
        sumB += (int)data[offset + 0];
        sumG += (int)data[offset + 1];
        sumR += (int)data[offset + 2];
    }
}

avgB = sumB / (height * width);
avgG = sumG / (height * width);
avgR = sumR / (height * width);
gray = (avgR + avgG + avgB) / 3.0;
scaleB = (gray / avgB) + sb_bias;
scaleG = (gray / avgG) + sg_bias;
scaleR = (gray / avgR) + sr_bias;
```

Finally, multiply the scales with original data to get the correction output.

Notice that, I add the bias here to let the handmade adjustment available.

Discussion:

I notice that this algorithm does not perform well on test image 2 and 3, the reason is the scenes above dominated by a single color, namely, the scenes have relatively smooth color changes in most place of the image. Such that the lighting conditions significantly deviate from a neutral color.

Original

Gray world

MaxRGB

From the results above, the sand and the sky are flatten in color change, such that the average of each color plane deviates from the neutral gray value.

2. MaxRGB algorithm implementation:

The MaxRGB algorithm assumes that the brightest color (highest intensity color) across the R, G, B planes corresponds to a surface represent the illuminant color. After determining the illuminant color, each pixel's value then divided by this value which is aimed to normalize the colors in the image to compensate for the effects of varying lighting conditions.

First, we need to find the maximum pixel value of each color planes and find out the global maximum value. After this, we can get the scales for adjustments.

```
// do max RGB
for(int y=0; y<height; ++y) {
    for(int x=0; x<width; ++x) {
        int offset = y * real_row_width + x * channels;
        maxB = max(maxB, (int)data[offset + 0]);
        maxG = max(maxG, (int)data[offset + 1]);
        maxR = max(maxR, (int)data[offset + 2]);
        for(int k=0; k<3; ++k) {
            global_max = max(global_max, (int)data[offset + k]);
        }
    }
}

scaleB = ((float)maxB / global_max) + sb_bias;
scaleG = ((float)maxG / global_max) + sg_bias;
scaleR = ((float)maxR / global_max) + sr_bias;
```

Finally, divide the data by scales to get the correction output.

Discussion:

If the maximum value of the image saturated, namely, the original image has the same or closed maximum values on R, G and B planes. It may not work well under this situation.

Original

Gray world

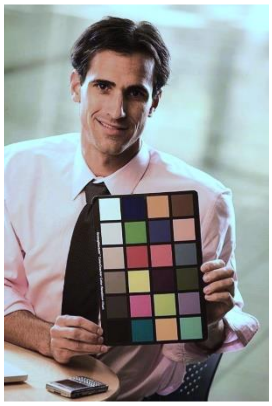
MaxRGB

The maximum value of this image on R, G and B planes are all 255, which makes no change on the scales. The MaxRGB method fails on this test image.

3. Enhancement attempts:

For test image 1:

The image after doing color constancy suffers from low light level condition. So, I use brightness and contrast adjustment mechanism to enhance the image with contrast ratio sets to 1.6 and brightness bias sets to +10.

For test image 2:

The condition in this image is like the test image 1. So, I choose the same strategy to adjust this image with the contrast ratio set to 1.5 and brightness bias sets to +10.


For test image 3:



For this image, I try two methods to do enhancements. The middle one is using power law transform with the gamma ratio sets to 0.8, which can improve the low light part details. The third one I use the bilateral filter (k=5, σ_s=5, σ_i=5) at first then adjust the brightness (+15) and contrast (1.25).

For test image 4:


For this image, I notice that the target image provide by TA seems to be warm color because of the wooden reflection of the light. Hence, I further adjust the scales used in the gray world implementation, which is slightly decreasing the blue and green components. In this case, I adjust the blue scale with -0.2 and green scale with -0.18 while red one remains unchanged.