Automatic White Balance algorithm

Video Engineering Activity

Paola Sanchez Pinedo, Pablo Gonzalez Carrizo, and Francisco Javier Bleda Molina

1

March 19, 2016

ABSTRACT

A robust automatic white balance algorithm is proposed in this paper. It will assume that in a well balanced image, the average of all the colors is a neutral gray. Comparing the average color with the gray color we will be able to estimate the color temperature of the light source. This method, commonly named as Gray World Method, has some problems when the difference between the average color of the image and grey is due to the objects that appear in the image, and not to the light. Analyzing the histogram we will be able to detect these cases and don't apply the gray world method to them. The test results show that the proposed algorithm can provide a good perceive effect and has the advantage of easy realization, low complexity and robust convergence.

Key words. white balance, gray world, algorithm, improvement

1. Introduction

The main aim of every white balance algorithm is always the same, we try to get the colors in our images as accurate as posible. Sometimes, when we take some shots and we examine them, images come out with a yellow, blue or orange look to them. This is due to the light source in the place where the image has been taken. However, when we see the real scene we don't see this color. That's because our eyes (actually, our brain) are doing the hard work in order that we perceive a well balanced image.

To solve this problem, usually in cameras we can find some presets that will adjust white balance for different situations as cloudy day, sunny day, fluorescent light or tungsten light.

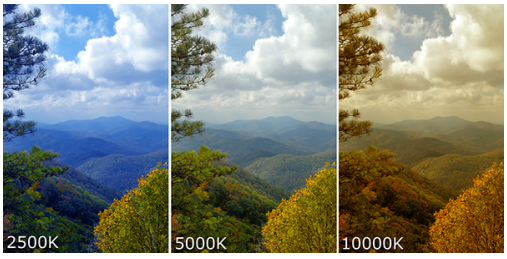

Fig. 1: Example of different presets for white balance.

What we are trying to achieve, is an algorithm that works in every situation, an automatic white balance algorithm. The difficult part now is that we must detect light conditions in order to decide which corrections apply to the image.

Our method is based on the idea that the average colour of a well balanced image is gray[2]. As we work with the three chromatic components: Red, Green and Blue, that means that the sum of all the pixels of the channels R,G and B must be the same. Assuming this, it will be easy to find a correction factor from the average values of the pixels of each channel.

However, this method has a problem. It may wrongly detect as bad balanced, images where an specific color predominates because it is the color of the objects that appear in the image. That's why we will sepparate the algorithm in three parts.

1. Detection of the predominant color of the image.
2. Detection of images with color predominance due to the objects that appear in them
3. Gray Wold Method

Fig. 2: Example of bad balanced image by a gray world algorithm.

2. Detection of the predominant color of the image

In order to detect which color predominates in the image we will only need to sum the values of all the pixels in the red, green and blue channels. The returned color will be the corresponding to the maximum value among the results of the three sums.

```
function [color] = imageColorDetection(input)
    input=double(input);
    R=input(:,:,1);
    G=input(:,:,2);
    B=input(:,:,3);
    sums=[sum(R(:)) sum(G(:)) sum(B(:))];
    color=find(sums==max(sums));
end
```

3. Detection of images with color predominance due to the objects that appear in them

As said before, we are able to differentiate between pictures with a bad white balance, and pictures like a blue sky where a colour predominates but the white balance its correct. In order to achieve this, we are using the histogram of the predominant color of the image, obtained in the previous section.

```
I=imread(ruta);
[m,n,c]=size(I);
type=imageColorDetection(I);
h=histogram(I,type);
```

Depending on the kind of image we can have two different kind of histograms.

1. If we have a predominance of a color in the image due to the color of the objects of the image, we will clearly see a pick in the histogram of this channel, an only absolut maximum in the graph The position of this maximum value defines the color of the object.
2. If this color predominance is due to the light (meaning we have a bad balanced image) the different colors that we see on the image will be the result of the action of the light over different objects with different colors. So if we see the histogram of the predominant color it won't have a section with much more amplitud than the others. It will be more or less flat.

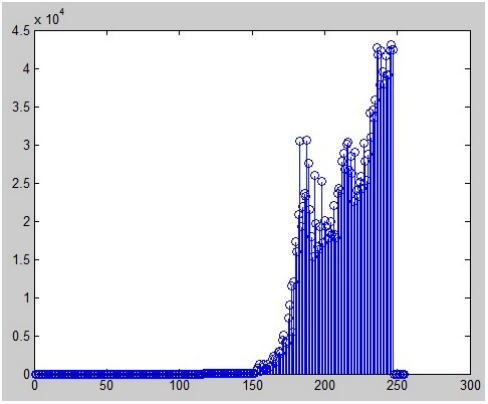

Fig. 3: Histogram of an image with some objects of the same color.

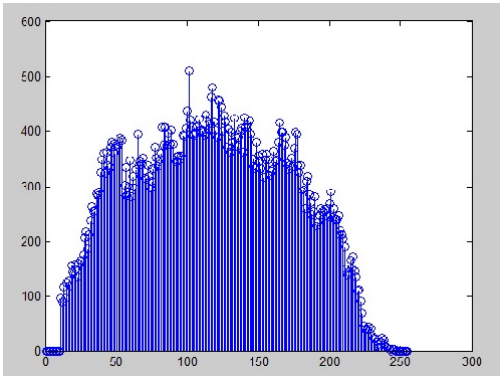

Fig. 4: Histogram of channel of image with bad white balance.

It should be taken into account that in the calculation of the histogram we are not taking into account pixels with value above 248 to avoid problems with burned pixels in the image. In order to detect the case in which we are, we will need a very important math tool, the variance. A histogram that correspond to the first kind (images with objects of the same color) will have such a high variance. However, the variance of the rest of images should not be as high as the first one.

As we have different images, with different kind of pixels, we cannot compare directly variances of different images with different number of pixels, so we will have to divide the variance by the total number of pixels in the image.

```
if( (var(h)/(m*n)) < 20 )
    modificada=1;
    output=grayWorldAdjustment(I);
else
    modificada=0;
    output=I;
end
```

The threshold value 20 has been set experimentally after testing the algorithm with image of both types. If we detect that the image belongs to the first kind of images ((var(h)/(m*n)) >20) we won't do nothing, because the gray world method can't be applied to these images. If we datect that the image belongs to the second kind of images ((var(h)/(m*n))<20) we will apply gray world method adjustment to the image.

4. Gray World method

It's the moment to apply the adjustment. Firstly, we obtain the average value of each of the three channels. After it, we obtain the correction factor. For this, we take a reference channel. In this case, it will be green, because is the component that gives more information of the three. Our eye is more sensible to these component than red or blue. We calculate the factor that relates green component to each one of the others, red and blue, and we multiply this factor by the corresponding channel of the image. We should remember that we are assuming that, in a well balanced image, this average values of the three channels must be equal. This factor that we obtain will allows us to obtain an equal mean value in the three components.

```
function [nueva] = grayWorldAdjustment(image)
    image=double(image);
    [m,n,c]=size(image);
```

```
    R=image(:,:,1);
    G=image(:,:,2);
    B=image(:,:,3);

    Ravg=(1/(m*n))*sum(sum(R));
    Gavg=(1/(m*n))*sum(sum(G));
    Bavg=(1/(m*n))*sum(sum(B));

    alpha=Gavg/Ravg;
    betha=Gavg/Bavg;

    nueva=image;
    nueva(:,:,1)=nueva(:,:,1).*alpha;
    nueva(:,:,2)=nueva(:,:,2);
    nueva(:,:,3)=nueva(:,:,3).*betha;
    nueva=uint8(nueva);
end
```

5. Tests

The algorithm has been tested with different images. We have use a dataset of 19 images containing images of the two groups that have been explained previously. The image on the left is the original image, and the one on the right is the output of our program.

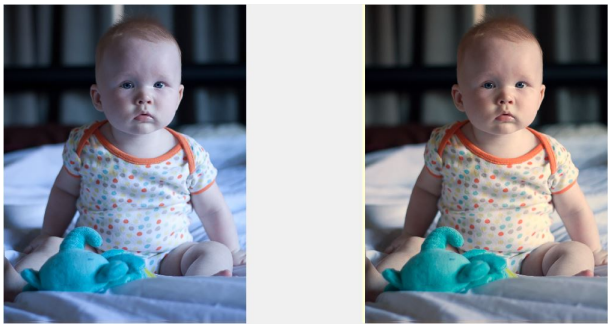

Fig. 5: Example of white balance correction of our algorithm.

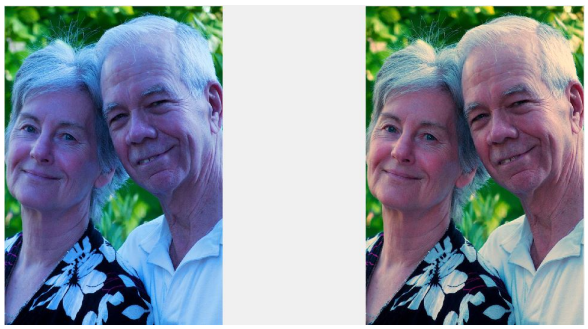

Fig. 6: Example of white balance correction of our algorithm.

Fig. 7: Example of white balance correction of our algorithm.

Fig. 8: Example of white balance correction of our algorithm.

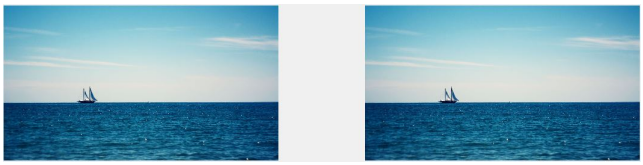

Fig. 9: Example of image that is not detected as bad balanced image.

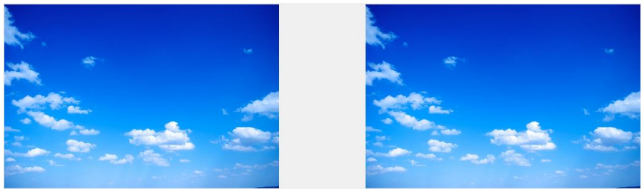

Fig. 10: Example of image that is not detected as bad balanced image.

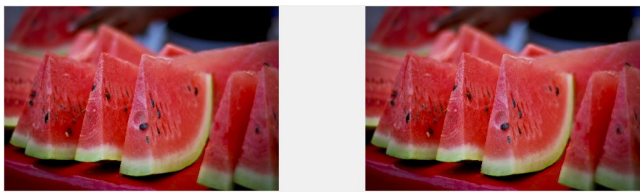

Fig. 11: Example of image that is not detected as bad balanced image.

Fig. 12: Example of image that is not detected as bad balanced image.

6. Team work

The creation of this algorithm can be divided in different stages.

- Preparation of the working plan. At this stage we created a working plan where we defined different aims with their corresponding deadlines in order to organise all the work. In this document, we also created some basic rules that will ensure that each member of the group works correctly.
- Design and test of the first version. During this stage we got information about how should a white balance algorithm work. We created a simple first version based on the gray world method. This first version had several problems with images with objects of the same color. But it obtained very succesful results with the rest of images. We decided to mantain this version and add some code that would be able to detect the cases when our algorith would obtain bad results.
- Design and test of the final system. As we had planned we corrected the errors of the first version of the algorithm.
  All the previous parts were finished in the time established in the working plan. However, we sent the final image on the 11th of March (instead of 9th of March as it was expected). It was because all the members of the group agreed that we should wait until the official deadline in order to have more time to test the code trying to find unexpected errors.
- Results presentation. This task took much more time than expected and it wasn't possible to finish it in the date that we decided in the working plan. However, it could be fnished before the official deadline.

7. Future improvements

In our opinion this algorithm could be improved if we had more time. One problem that we found during the final testing of the code was that the image classification stage did not have a 100% of success rate. We realised that this problems occurred mainly when we have many objects in the image whose color is not red, green, or blue. When the color of the objects was a mix between this primary colors the success rate of our image classification algorithm decreased a lot.

8. Using the algorithm with videos

The most obvious way of using this algorithm with videos is applying it to each one of the frames individually. However, this can produce some undesired effects. The problem is that, considering that scene content is the same, if the current frame is corrected differently that the previous one, an undesired flash effect will appear in the video. If we can detect the scenes changes in the video we will be able to overcame this problem. We will apply the same correction factor to the whole scene. In this way, the effect commented previously won't appear.

In order to achieve the scene change detection, the most easy way is to compare histograms.

9. Conclusions

An automatic white balance algorithm is proposed in this report. It solves a commonly known problem that occurs when using white balance algorithms based on gray world with images where there are many objects with the same color. The detection of these kind of images is based on the histogram variance. The proposed algorithm has a good performance with most images but it could be improved in the future. Maybe, using another color space (like HSV) would be useful and would help us to solve the problems of our algorithm with some images where the main color is not red, green or blue.

Acknowledgements. The idea of classifying images using the histogram was proposed by our Video Engineering teacher Miguel Romá

References

[1] [2010] Jun-yan Huo, Yi-lin Chang, Jing Wang, and Xiao-xia WeiJun-yan Huo, Yi-lin Chang, Jing Wang, and Xiao-xia Wei "Robust Automatic White Balance Algorithm using Gray Color Points in Images"

[2] [2009]Edmund Y. Lam and George S. K. FungEdmund Y. Lam and George S. K. Fung "Automatic White Balancing in Digital Photography"