# Scene Analysis Assisting for AWB Using Binary Decision Trees and Average Image Metrics

K.I. Sofeikov[a,b], I.V. Romanenko[b], I. Yu. Tyukin[a,c], A.N. Gorban[a]
[a] University of Leicester, Leicester, UK
[b] Apical Ltd, Apical Technical Centre, Leicester, UK
[c] Saint-Petersburg State Electrotechnical University, Saint-Petersburg, Russia
Contact e-mails: *sofeykov@gmail.com*, *ilya@apical.co.uk*, *I.Tyukin@le.ac.uk*, *ag153@le.ac.uk*

***Abstract*— We propose a technique for improving Automatic White Balance (AWB) settings in digital cameras on the basis automatic classification of image fragments in pictures. Our approach is based on constructing binary decision trees and using them as decision-making devices for identifying and locating patches of consistent texture in an image, such as grass, sky etc. We demonstrate with examples that this approach can be applied successfully to enhance color reproduction of images in challenging light conditions. Furthermore, due to low levels of false-positives, the method can be used in combination with any other AWB algorithms that do not rely on color clues obtained from the inference and analysis of content in images taken.**

## I. INTRODUCTION

Consider a situation when someone takes photos of landscapes in strong light on a sunny day in the middle of summer. Frames containing large patches of green grass or blue sky or both are not unusual in these circumstances. Every grass stem reflects sun rays, and since the relative amount of grass is large, such reflection gives rise to an additional source of light in the scene. This light, however, can easily be confused with reflection from a gray object under artificial light in standard $(R/G, B/G)$ color metrics. Such confusion decreases reliability and correctness of Automatic White Balance (AWB) decisions in these situations. Furthermore, it suggests that using mere raw $(R/G,B/G)$ data is hardly sufficient for producing correct AWB decisions. Similar scenario occurs in presence of large areas of blue sky in images. Even though true color representation may be distorted in such images it is nevertheless desirable to be able to correctly represent true colors of objects in these scenes using AWB settings.

In what follows we present a technique for improving performance of AWB algorithms which is based on detecting grass or sky patches in image frames. The paper is organized as follows. Section II describes our grass/sky detection and classification engine, Section III contains experimental results, and Section IV demonstrates application of the technique to AWB setting. Section V concludes the paper.

## II. CLASSIFICATION ENGINE

### *A. Clusters configuration*

Any machine learning task starts with forming and analysing feature spaces and training sets that are available for observation. In our task the variables available for direct observation have been restricted to: 1) average $R/G$ value, 2) average $B/G$ value, 3) variance of $R/G$ value, 4) variance of $B/G$ value, 5) Illumination of scene (lux value) 6) Intensity variation. The reason for choosing this particular set of parameters for enhancing AWB settings is that we were aiming at embedding developed algorithms in the available existing image processing hardware pipeline in which these 6 characteristics of image patches were readily available.

Every photo was split into $15 \times 15$ pieces and for every such piece we formed a corresponding $6D$ parameter/feature vector. Each training set was built from about 100 photos, and the total number of training $6D$ points was about 22500. Training sets were organized as follows. The training set was split into two subsets: positive (with grass featuring in the images) and negative (no grass objects in the images). Images in the positive training set contained large amount of different textures of grass and foliage in different light conditions. The size occupied by grass/foliage patches varied from one image to another. The negative training set contained photos with different objects and colors and, at the same time, did not contain any grass/foliage or sharp green real world objects.

Fig. 1 shows examples of possible $2D$ projections of clusters of the original $6D$ feature vectors. As we can see from these pictures relationships between individual components of feature vectors corresponding to different clusters is rather complicated, with large overlapping areas in relevant $2D$ projections.

### *B. Techniques*

In order to solve the classification problem we focused primarily on binary decision trees, mainly due to simplicity of their implementation. Performance of the binary tree classifier was compared with that of a standard linear classifier (Fisher linear discriminant [1]). Posterior analysis revealed that binary decision trees were more advantageous in our task.

*1) Fisher's linear discriminant:* Out first try was to employ Fisher's linear discriminant [1] in the space of points $(R/G, B/G)$ in order to determine presence of color cues, e.g. patches of grass, in a scene. As a by-product of using Ficher's discriminant one can reduce dimensionality of the classifier for this task. The main idea of this approach is

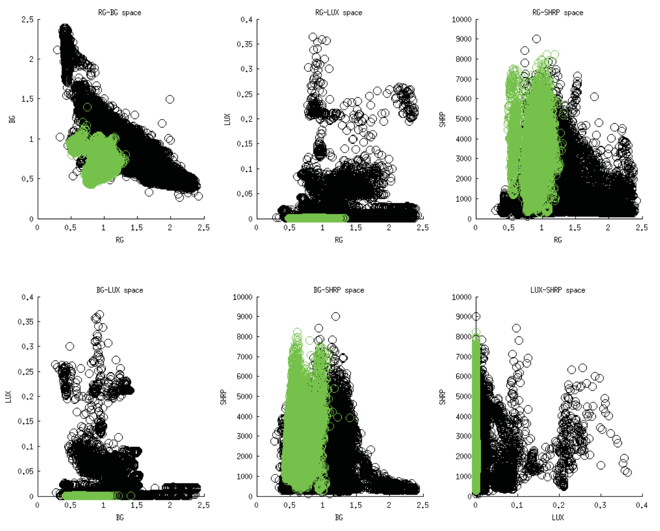


Fig. 1: Representation of the dataset on the planes of pairs of attributes. Green circles correspond to patches of grass, black circles mark patches of images without grass.

to project data points onto a line so that separation of points from different clusters to this line is maximized in some sense. Since points are being projected on the line, dimension of the classifier is reduced. Formally, the approach is summarised as follows. Let $N$ will be number of points in each class in the training set, and $x = (R/G, B/G)$, and $\mathbf{w}$ be a normal to a discriminating hyper-plane. The vector $\mathbf{w}$ determines the line to which the datum will be projected. This vector is chosen so that variance of the projections of points from different clusters is maximal and, at the same time, variance of the projections of points within the same class is minimal. The solution of this problem can formally be expressed as [1]:

$$\mathbf{w} = S_W^{-1}(\mathbf{m_2} - \mathbf{m_1}),$$

where

$$S_W = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^T$$

$$\mathbf{m_1} = \frac{1}{N} \sum_{n \in \mathcal{C}_1} x_n, \ \mathbf{m_2} = \frac{1}{N} \sum_{n \in \mathcal{C}_2} x_n.$$

With respect to application of this method for determining color cues, its performance was not overall satisfactory. Clusters of projected data in the training and validation sets overlapped, and classification quality for points within the overlapping area was very poor.

*2) Binary decision trees:* Decision trees is a common method in data mining. The main idea is to create a structure that predicts the values of a target attribute based on several input variables. In this approach data have the following form:

$$(\mathbf{x}, c) = (x_1, x_2, \dots, x_n, c),$$

where $x_n$ are attributes, and $c$ is the target variable. In our problem $c$ is the class of an area to which this point belongs ("grassy" or "non-grassy"). Each node in such a tree contains some attribute name and its threshold value. Leaves of the tree contain values of target variables. The tree can be constructed from training data by splitting initial cluster into subsets determined by values of attributes. The process continues recursively by splitting clusters into smaller subsets until stopping conditions are reached. Each area is then classified as follows:

$$c(\text{area}) = \begin{cases} \text{grassy}, & N_g/(N_g + N_{ng}) > c_0 \\ \text{non-grassy}, & \text{otherwise}, \end{cases}$$

where $N_g$ and $N_{ng}$ are the number of grass and non-grass points in that area, and $c_0$ is a threshold. Several different algorithms exist that enable automated construction of decision trees [5]. The simplest ones among these are the so-called $ID3$ [3] and $C4.5$ [4]. In our work we have chosen $ID3$ due to simplicity of its implementation.

The process of constructing a decision tree with $ID3$ can be briefly described as follows. Let $x = (x_1, \dots, x_n)$ be the vector of attributes taking values in $\mathbb{R}$, $(x_{1,i}, \dots, x_{n,i})$, $i = 1, \dots, N$ be the training set, and $S$ be the initial (bounded) set from which the training set is drawn. In addition, for each attribute $x_j$ we introduce a set of thresholds $\{t_{j,1}, \dots, t_{j,M}\}$ that are equally spaced in the interval $[\min x_j, \max x_j]$. With each threshold $t_{j,k}$ we will associate two subsets $S^+_{j,k} = \{x \in S | \ x_j \geq t_{j,k}\}$ and $S^-_{j,k} = \{x \in S | \ x_j < t_{j,k}\}$. It is clear that $S = S^+_{j,k} \cup S^-_{j,k}$, and in this sense thresholds $t_{j,k}$ split the original set $S$ into two disjoint subsets with respect to the values of attribute $x_j$. All points in the training set are supposed to be already correctly classified into classes $c$ from the set of admissible classes $C$. Furthermore, the following statistical characterizations of the training set are supposed to be readily available: 1) $|S|$, $|S^+_{j,k}|$, $|S^-_{j,k}|$ – the total numbers of elements in the sets $S$, $S^+_{j,k}$, and $S^-_{j,k}$; 2) $p(c, S)$, $p(c, S^+_{j,k})$, and $p(c, S^-_{j,k})$ – the ratios of the number of elements from $S$, $S^+_{j,k}$, and $S^-_{j,k}$ classified as from class $c$ to the total number of elements in $S$, $S^+_{j,k}$, and $S^-_{j,k}$ respectively.

For the sets $S$, $S^+_{j,k}$, and $S^-_{j,k}$ defined above we introduce quantities specifying variability of classes within these sets. In this case standard Shannon entropy [6] is used:

$$H(S) = -\sum_{c \in C} p(c, S) \log_2 p(c, S),$$

$$H(S^+_{j,k}) = -\sum_{c \in C} p(c, S^+_{j,k}) \log_2 p(c, S^+_{j,k})$$

$$H(S^-_{j,k}) = -\sum_{c \in C} p(c, S^-_{j,k}) \log_2 p(c, S^-_{j,k}).$$

Obviously, if e.g. $H(S^+_{j,k}) = 0$ (or $H(S^-_{j,k}) = 0$) then the set $S^+_{j,k}$ (or $S^-_{j,k}$) contains objects of only one class. Finally, we specify conditional entropy $H(S|t_{j,k})$

$$H(S|t_{j,k}) = \frac{|S^+_{j,k}|}{|S|} H(S^+_{j,k}) + \frac{|S^-_{j,k}|}{|S|} H(S^-_{j,k}),$$

and relative information gain $RIG(S|t_{j,k})$

$$RIG(S|t_{j,k}) = (H(S) - H(S|t_{j,k}))/H(S).$$

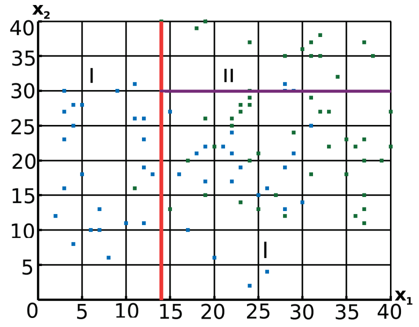


Fig. 2: An illustration for ID3 algorithm. See the article text for comments.

The algorithm for constructing binary decision trees can now be described as follows:

1) Consider initial set $S$
2) Create a set of thresholds $\{t_{j,k}\}$
3) For every $t_{j,k}$ calculate $RIG(S|t_{j,k})$
4) Determine

$$t_{l,m} = \arg \max_{j=1,\dots,n;k=1,\dots,M} RIG(S|t_{j,k})$$

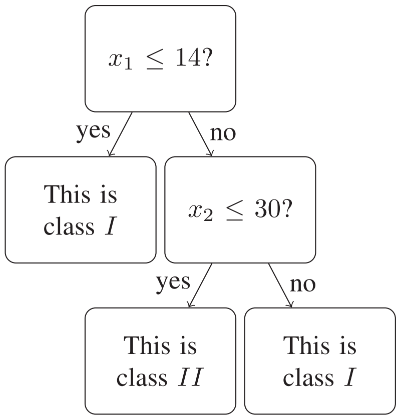
5) Create a node with attribute $x_l$ being a decision variable, and $x_l < t_{l,m}$, $x_l \geq t_{l,m}$ being its corresponding branching conditions; split the initial set $S$ into two sets $S^+_{l,m}$ and $S^-_{l,m}$
6) Remove $t_{l,m}$ from the list of thresholds and repeat this procedure recursively for each subsequent subsets $S^+_{l,m}$ and $S^-_{l,m}$ until a stopping condition is met

In order to illustrate how the algorithm works for the relevant case of data points comprising of two attributes $x_1$ and $x_2$ we consider the following simple example. Let training data be as in the diagram in Figure 2. The data points are already classified into two classes (marked by blue and green rectangles on figure respectively), and the initial set $S$ is the entire white rectangle. We continue with step 2 and chose $t_{1,k} = k$, $t_{2,k} = k$, $k = 1, \dots, 40$. This completes step 2. For the chosen set of thresholds we are to calculate $RIG(S|t_{j,k})$, $j = 1, 2; k = 1, \dots, 40$ (step 3). Suppose that the maximal value of $RIG$ is reached at $t_{1,14}$ (red line on Figure 2) (step 4). Proceeding to step 5, we split the initial set $S$ into $S^+_{1,14}$ (to the right of the red line) and $S^-_{1,14}$ (to the left of the red line). These steps are repeated until stopping conditions in each branch are met. Figure 3 shows an example of possible final decision tree for this case.

## III. EXPERIMENTAL RESULTS

### A. Testing

The approach has been applied for detection of sky and grass fragments in images. The training set consisted of about 500 photos of different types including ordinary indoors and outdoors images. Examples of typical grass patches detection are shown in Fig. 4. White and dark cells it the right panel label patches in which the probability of grass is high or low, respectively.



$$\begin{tikzcd}
 & x_1 \leq 14? \arrow[dl, "yes"'] \arrow[dr, "no"] & & \\
\text{This is class } I & & x_2 \leq 30? \arrow[dl, "yes"'] \arrow[dr, "no"] & \\
 & \text{This is class } II & & \text{This is class } I
\end{tikzcd}$$

Fig. 3: An decision tree example

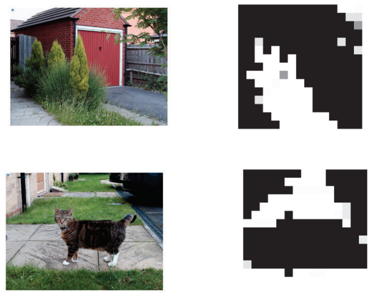


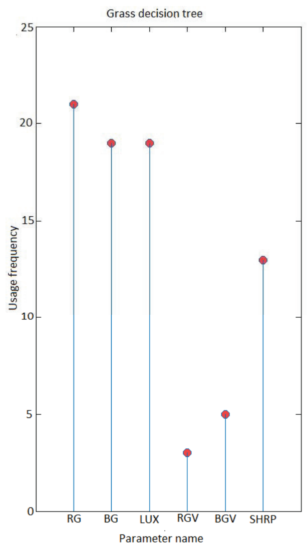
Fig. 4: Examples of grass detection. Left column: source images. Right column: probability/likelyhood maps.

### B. How informative parameters were?

Not all parameters are equally informative. We found that, in some problems better results can be achieved if some parameters are removed from consideration. For example, in the sky detection problem excluding variances of $R/G$,$B/G$ helped to increase quality of detection. In addition, we investigated the influence of the number of parameters used on the size of the final decision tree. In some cases exclusion of few specific parameters led to reduction of the tree's depth. Finally, frequencies of appearance of parameters/attributes in constructed trees varied too suggesting that some parameters are less informative than others. Variability of parameter/attribute frequencies in the final trees is shown in Figure (5).

### C. Manual calibrations routines

Further improvements can be achieved by manual calibration. The crucial point concerning manual calibration is first to decide if having low rate of false-positives is more preferable over an occasional mis-detection, or it is the other way around. Depending on this choice, in manual calibration we looked for points which have not been correctly classified and forced the algorithm to associate this point and a small hypercube of pints around it with either false or positive case. Note that complete elimination of mistakes may not always be possible due to potential inseparability of certain domains in the feature space.



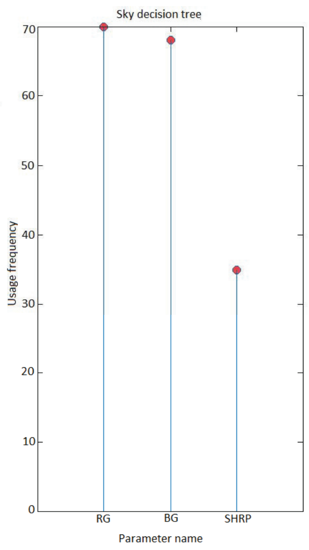


Fig. 5: Frequency of parameters usage in each decision tree

### D. Problems and issues

The following issues emerged during experimental validation/tuning.

1) The problem of how to form training sets ensuring best possible quality of detection
2) In some lighting conditions, the classification of sky patches can be unreliable as it may have statistical characterizations very similar to non-sky objects. This issue can be resolved by increasing the number of attributes employed for classification.
3) Color characteristics of sky may vary substantially which imposes additional constraints and demands on the choice of appropriate training sets.

## IV. Application of scene elements probabilities in AWB algorithm

In order to illustrate application of our method we first take classic grey world algorithm and then add our proposed scene-analysis based AWB setting correction to improve color reproduction. Recall that the grey world algorithm works as follows. Let $I(x,y)$ be an image of the size $M \times N$, where $x, y$ are indices of pixel positions. Variables $I_R(x,y)$,$I_G(x,y)$ and $I_B(x,y)$ denote red,green and blue channels of the image, respectively. Average values within channels can be calculated [2] as

$$R_{avg} = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N} I_R(x,y) \quad (1)$$

$$G_{avg} = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N} I_G(x,y) \quad (2)$$

$$B_{avg} = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N} I_B(x,y) \quad (3)$$

Grey world algorithm keeps the green channel unchanged and defines correction ratio for the red and blue channels as

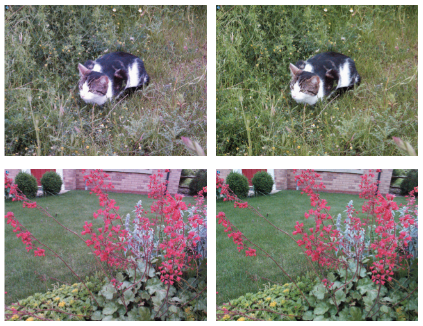


Fig. 6: Examples of how information about grass probabilities can help in improving AWB. Left column: AWB setting with conventional GW algorithm. Right column: AWB setting with grass probability information.

$\alpha_{GWA} = \frac{G_{avg}}{R_{avg}}$ and $\beta_{GWA} = \frac{G_{avg}}{B_{avg}}$. Red and blue channels are adjusted as follows: $I_{GWA,R}(x,y) = \alpha_{GWA} I_R(x,y)$, $I_{GWA,B}(x,y) = \beta_{GWA} I_B(x,y)$. Now we split our image onto $15 \times 15$ patches and calculate proportions of grassy points in every patch according to our method. Let $P$ be a matrix $15 \times 15$ containing these values, and derive $W = 1 - P + 0.001$, where the pedestal 0.001 is introduced to avoid divisions by zero. Then instead of calculating means in (1)– (3) we calculate their weighted means (with the weight matrix $W$). The rest of the GW algorithm remains unchanged. Fig. 6 shows examples of how the proposed addition improves the outcomes of the GW algorithm.

## V. Conclusion

In this work we proposed a technique for automatic AWB settings improvement based on grass and sky detection in pictures taken by digital cameras. The technique employs construction of binary decision trees with manual post calibration. On the given database of images the resulting algorithm showed very low level of false positives giving additional clues on the light sources analyzed by the AWB algorithm. This enables us to hope that it can be successfully implemented in existing AWB algorithms without compromising their operational quality.

## References

[1] Christopher M. BISHOP. *Pattern Recognition and Machine Learning*. Pearson, 2008.

[2] Edmund Y. Lam. *Single-Sensor Imaging: Methods and Applications for Digital Cameras*. Taylor, Francis Group, 2009.

[3] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1,1:81–106, 1986.

[4] Quinlan J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc, 1993.

[5] L. Rokach and O. Maimon. Decision trees. In Lior Rokach and Oded Maimon, editors, *Data Mining and Knowledge Discovery Handbook*, pages 154–192. Morgan Kaufmann Publishers Inc, 2010.

[6] C. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, 1948.