Illuminant Estimation using Ensembles of Multivariate Regression Trees

Peter van Beek, Cheriton School of Computer Science, University of Waterloo
R. Wayne Oldford, Department of Statistics & Actuarial Science, University of Waterloo

Abstract

White balancing is a fundamental step in the image processing pipeline. The process involves estimating the chromaticity of the illuminant source and using the estimate to correct the image to remove any color cast. Given the importance of the problem, there has been much previous work on illuminant estimation. Previous work is either more accurate but slow and complex, or fast and simple but less accurate. In this paper, we propose a method for illuminant estimation that uses (i) fast features known to be predictive in illuminant estimation and (ii) single feature decision boundaries in ensembles of multivariate regression trees, (iii) each of which has been constructed to minimize a multivariate distance measure appropriate for illuminant estimation. The result is an illuminant estimation method that is simultaneously fast, simpler, and more accurate.

Introduction

White balancing is either performed onboard the camera—if the image delivered by the camera is in JPEG format, for example—or in a post-processing phase—if the image delivered by the camera is in the camera's native RAW format. When done onboard the camera, real-time and space considerations place additional restrictions on white balancing algorithms. Most commercial cameras use simple algorithms based on the gray-world assumption [13], such as using the mean color of the image as an estimate of the illumination, as these algorithms have the advantage of being fast and simple, albeit at the expense of accuracy [11].

Given the importance of the problem, there has been much previous work on illuminant estimation (e.g., [3, 8, 15, 16, 18, 19, 21, 33, 34]; see [22] for a recent survey). Presently, convolutional neural networks (CNN) can provide state-of-the-art accuracy [4, 32]. However, in terms of simplicity and speed, the CNN approach is at a definite disadvantage. To get credible speed performance requires a GPU to process each image. Even then, a CNN can take approximately three seconds on a GPU to process a single image with present state-of-the-art methods [32]. While it is unclear whether this waiting time is tolerable for post-processing, it is certainly the case that the slow speed and the need to add specialized hardware is unacceptable for onboard a camera.

The most successful methods to date in illuminant estimation are based on regression and minimize the squared error loss when *constructing* their predictive models [11, 32]. However, when *evaluating* the performance of the predictive models, the squared error loss is not used and distance measures more appropriate for illuminant estimation, such as angular error, are used instead.

In this paper, we propose a method for illuminant estimation that uses (i) fast features known to be predictive in illuminant estimation and (ii) single feature decision boundaries in ensembles of multivariate regression trees, (iii) each of which has been *constructed* to minimize a multivariate distance measure appropriate for illuminant estimation. The result is an illuminant estimation method that is fast, simple, and accurate. Figure 1 summarizes the improvement in accuracy over Cheng et al. [11].

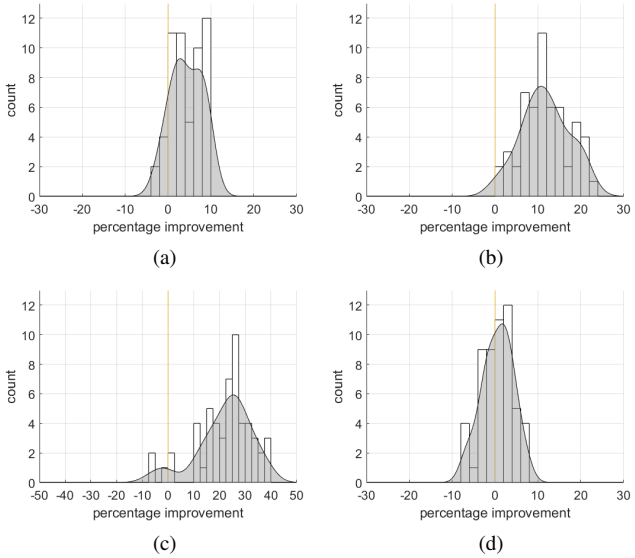

Figure 1. *Percentage improvement over Cheng et al. [11] for the (a) mean, (b) median, (c) mean of the best 25%, and (d) mean of the worst 25% of the distances measures for an image set, where each image set is captured by a single model of a camera, and there are 11 image sets and 5 distance measures for each image set.*

The rest of the paper proceeds as follows. We first review distance or performance measures that have been proposed for evaluating the effectiveness of white balancing algorithms. We then present our proposed method of constructing predictive regression models based on minimizing an appropriate distance measure. We next present the results of experimentally evaluating our proposed method, comparing against the most successful methods to date [11, 32]. Finally, we end the paper with some conclusions.

Distance Measures

Finlayson and Zakizadeh [17] have shown that the ranking of white balancing algorithms changes depending on the chosen distance or performance measure. The distance measures assume

normalized RGB; i.e.,

$$r = \frac{R}{R+G+B}, \quad g = \frac{G}{R+G+B}, \quad b = \frac{B}{R+G+B},$$

where R, G, and B are the red, green, and blue channel measurements and $r+g+b=1$. In what follows, let all vectors be row vectors, $\hat{\mathbf{e}} = (\hat{r}, \hat{g}, \hat{b})$ be the estimated illuminant, and $\mathbf{e} = (r, g, b)$ be the ground truth illuminant.

The most widely used distance measure [20, 25], *recovery angular error*, measures the angular distance between the estimated illuminant and the ground truth illuminant,

$$dist_{\theta}(\hat{\mathbf{e}}, \mathbf{e}) = \cos^{-1}\left(\frac{\hat{\mathbf{e}} \cdot \mathbf{e}^T}{\|\hat{\mathbf{e}}\|\|\mathbf{e}\|}\right),$$

where $\hat{\mathbf{e}} \cdot \mathbf{e}^T$ is the dot product of the vectors, $\|\cdot\|$ is the Euclidean norm, and we assume the angular distance is measured in degrees. Gijsenij et al. [20] note that human observers sometimes judge the recovery angular error underestimates the perceived differences between two images. One reason is that the recovery angular error ignores the direction of the deviation from the ground truth, which can be important from a perceptual point of view.

Finlayson and Zakizadeh [17] propose *reproduction angular error*,

$$dist_{rep}(\hat{\mathbf{e}}, \mathbf{e}) = \cos^{-1}\left(\frac{r/\hat{r} + g/\hat{g} + b/\hat{b}}{\sqrt{(r/\hat{r})^2 + (g/\hat{g})^2 + (b/\hat{b})^2}\ \sqrt{3}}\right),$$

as a distance measure. In words, once the image has been corrected with the estimated illuminant, the angular error between the white balanced ground truth and the target of uniform gray $\mathbf{1} = (1,1,1)$ is determined.

Gijsenij et al. [20] discuss the use of the Minkowski distance for measuring the distance between $\hat{\mathbf{e}} = (\hat{r}, \hat{g}, \hat{b})$ and $\mathbf{e} = (r, g, b)$. Two special cases are the *Taxicab distance*,

$$dist_1(\hat{\mathbf{e}}, \mathbf{e}) = |\hat{r} - r| + |\hat{g} - g| + |\hat{b} - b|,$$

and the *Euclidean distance*,

$$dist_2(\hat{\mathbf{e}}, \mathbf{e}) = \sqrt{(\hat{r} - r)^2 + (\hat{g} - g)^2 + (\hat{b} - b)^2}.$$

Gijsenij et al. [20] note that the Euclidean distance treats each of the RGB channels uniformly whereas it is known that the sensitivity of the human eye to perceived differences varies across color channels. To this end, they define the *perceptual Euclidean error*,

$$dist_{ped}(\hat{\mathbf{e}}, \mathbf{e}) = \sqrt{w_r(\hat{r} - r)^2 + w_g(\hat{g} - g)^2 + w_b(\hat{b} - b)^2},$$

where weights w_r, w_g, and w_b capture this sensitivity and $w_r + w_g + w_b = 1$. In experiments, Gijsenij et al. [20] found that the weight vector $(w_r, w_g, w_b) = (0.21, 0.71, 0.08)$ gave a higher correlation with the judgment of human observers than that of $dist_{\theta}$, $dist_1$, and $dist_2$.

Our Proposal

In this section, we present our approach for white balancing based on ensembles of multivariate regression trees. We begin with a brief review of univariate (ordinary) regression trees and how they were applied by Cheng et al. [11] for white balancing.

Univariate regression trees

Univariate regression trees are constructed in a greedy, top-down manner from a set of labeled training examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i = (x_{i1}, \ldots, x_{im})$ is a vector of feature values and y_i is a scalar response variable (see, e.g., [7, 29, 23]). As it is sufficient for our purposes, we assume that the feature values and the response variable are real-valued. The root node of the tree is associated with all the training examples. At each step in the construction of the tree, the training examples at a node are partitioned by choosing the feature $j \in \{1, \ldots, m\}$ and partition p that minimizes the total of the squared-error loss functions,

$$\underset{c_1}{\operatorname{argmin}} \sum_{i \in L(j,p)} (y_i - c_1)^2 \ + \ \underset{c_2}{\operatorname{argmin}} \sum_{i \in R(j,p)} (y_i - c_2)^2, \qquad (1)$$

where the partition is into two subsets, $L(j,p)$ and $R(j,p)$, formed by branching on feature j according to $x_{ij} \leq p$ and $x_{ij} > p$, respectively. For any choice of feature j and partition p the minimization is solved by taking the scalar c_1 to be the mean of the y_i values in the left branch and the scalar c_2 to be the mean of the y_i values in the right branch. Once the best pair (j,p) is found, a left child node and right child node are added to the tree and are associated with the subsets $L(j,p)$ and $R(j,p)$. The partitioning continues until some stopping criterion is met, in which case the node is a leaf and is labeled with the scalar c associated with the subset of examples at the node.

To estimate an illuminant $\mathbf{e} = (r, g, b)$, Cheng et al. [11] predict the r chromaticity and the g chromaticity independently using separate univariate trees fit with the squared-error loss function and from these two values the estimate of the b chromaticity can be determined using $b = 1 - r - g$. Cheng et al. [11] use four pairs of simple features in their univariate trees: (f_r^1, f_g^1), the mean color chromaticity as provided by the gray-world algorithm; (f_r^2, f_g^2), the brightest color chromaticity, an adaptation of the white-patch algorithm; (f_r^3, f_g^3), the bin average of the mode of the RGB histogram, and (f_r^4, f_g^4), the mode of the kernel density estimate from the normalized chromaticity plane.

Two important points are that Cheng et al.'s [11] method (i) predicts the r and g chromaticities *independently*, and (ii) minimizes the distance measure of interest—in their work, the recovery angular error—*indirectly* by minimizing the squared-error loss function. Our starting point is (i) the observation that our response variable is not a scalar but a vector $\mathbf{e} = (r, g, b)$ of chromaticities, and (ii) the related observation that independently fitting using the squared-error loss function is not necessarily a good surrogate for minimizing distance measures used in white balancing; Example 1 illustrates the point.

Example 1. *Let $\mathbf{e} = (r, g, b)$ be the ground truth illuminant. Consider the two estimates of the illuminant $\hat{\mathbf{e}}_1 = (\hat{r}_1, \hat{g}_1, \hat{b}_1)$ and $\hat{\mathbf{e}}_2 = (\hat{r}_2, \hat{g}_2, \hat{b}_2)$, where*

$$\begin{aligned} \hat{r}_1 &= r + \alpha, & \hat{r}_2 &= r + \alpha, \\ \hat{g}_1 &= g + \alpha, & \hat{g}_2 &= g - \alpha, \\ \hat{b}_1 &= 1 - \hat{r}_1 - \hat{g}_1 = b - 2\alpha, & \hat{b}_2 &= 1 - \hat{r}_2 - \hat{g}_2 = b, \end{aligned}$$

and $\alpha > 0$ is some residual error. Here, the pair of estimates $\hat{r}_1$ and $\hat{r}_2$ and the pair of estimates $\hat{g}_1$ and $\hat{g}_2$ both have equal squared error (α^2). Thus, from the point of view of independently

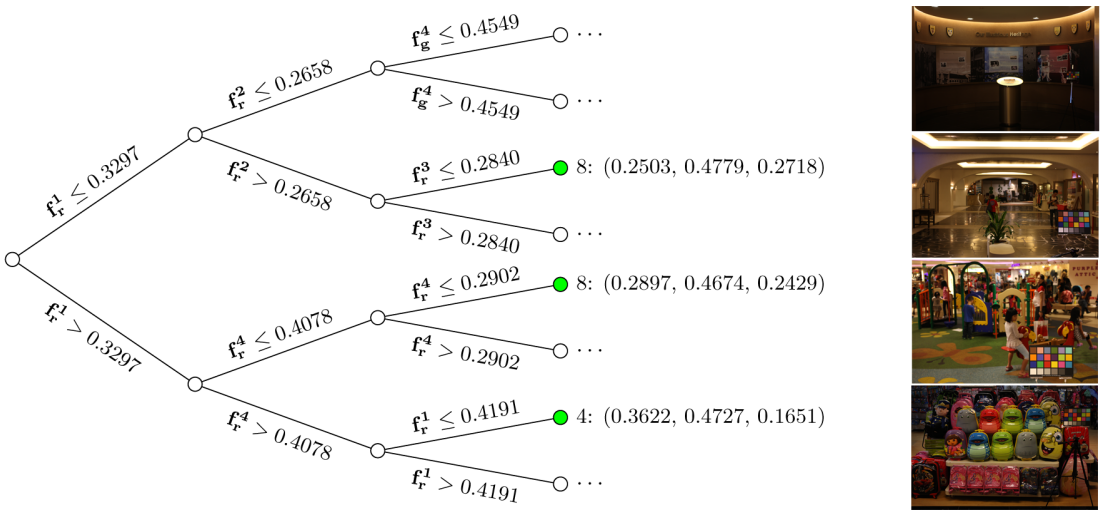

Figure 2. *An example multivariate regression tree when the distance measure is recovery angular error, showing only the first four layers (out of 18 layers). The three (green) leaf nodes are labeled with the number of training examples at that node and the illuminant estimation if that node is reached. Also shown are the four images associated with the green bottom-most leaf node; the images have been gamma corrected for presentation.*

fitting using the squared-error loss function, the two estimates of the illuminant have equal error. However, for distance measures that have been proposed for white balancing, the error of the estimate $\hat{\mathbf{e}}_1$ *is larger (and can be much larger) than that of* $\hat{\mathbf{e}}_2$*. For example, let* $\mathbf{e} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ *and* $\alpha = 0.05$*. For the recovery angular error,* $dist_\theta(\hat{\mathbf{e}}_1, \mathbf{e}) = 11.977$ *and* $dist_\theta(\hat{\mathbf{e}}_2, \mathbf{e}) = 6.983$*. Thus, minimizing the squared error is not necessarily a good surrogate for minimizing a distance measure of interest in white balancing.*

Multivariate regression trees

We show how multivariate regression trees [30, 12, 27], where each tree predicts multiple responses, can be used to effectively estimate an illuminant. In the case of multiple responses, multivariate trees are more compact than univariate trees and can be more accurate when the response variables are correlated [28]. Multivariate regression trees are constructed in a greedy, top-down manner from a set of labeled training examples $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, where as before $\mathbf{x}_i$ is a vector of feature values but now $\mathbf{y}_i = (y_{i1}, \ldots, y_{ik})$ is a *vector* of response variables [30, 12, 27].

Our proposed method for illuminant estimation is for a single tree to simultaneously predict all three chromaticity components of an illuminant $\mathbf{e} = (r, g, b)$, rather than multiple trees predicting them independently, thus taking into account that the chromaticity components of the illuminant are correlated and constrained (see Figure 2). We fit a multivariate regression tree by directly minimizing a distance measure (loss function) of interest. Previous work on multivariate regression trees minimize only loss functions that are variants of the squared-error loss function [12, 30]. In our proposed method, at each step in the construction of the multivariate regression tree, the training examples at a node are partitioned by choosing a feature $j \in \{1, \ldots, m\}$ and partition p that minimizes the total of the loss functions,

$$\underset{\hat{\mathbf{e}}_1}{\operatorname{argmin}} \sum_{i \in L(j,p)} dist(\hat{\mathbf{e}}_1, \mathbf{e}_i) + \underset{\hat{\mathbf{e}}_2}{\operatorname{argmin}} \sum_{i \in R(j,p)} dist(\hat{\mathbf{e}}_2, \mathbf{e}_i), \quad (2)$$

where the partition is into two subsets, $L(j,p)$ and $R(j,p)$, formed by branching on feature j according to $x_{ij} \leq p$ and $x_{ij} > p$, respectively; $dist(\cdot,\cdot)$ is any distance measure that has been proposed for white balancing (see Section "Distance Measures"); and $\hat{\mathbf{e}}_1$, $\hat{\mathbf{e}}_2$, and $\mathbf{e}_i$ are normalized RGB vectors in $\mathbb{R}^3$, i.e., the chromaticities sum to one. The partitioning continues until some stopping criteria is met, in which case the node is a leaf and labeled with the estimated illuminant $\hat{\mathbf{e}}$ associated with the subset of examples at the node. When such a tree is used to estimate the illuminant of an image that has not been seen before, one starts at the root and repeatedly tests the feature at a node and follows the appropriate branch until a leaf is reached. The label of the leaf is the estimated illuminant of the image.

Ensembles of multivariate regression trees. To improve predictive accuracy, an ensemble of trees is learned where each tree is a multivariate regression tree. Various methods have been proposed for constructing ensembles including manipulating the training examples, manipulating the input features, injecting randomness into the learning algorithm, and combinations of these methods (see [14]). We adopt the method for constructing ensembles that injects randomness into the construction of an individual tree. When constructing the tree, each feature j partition p pair considered in Equation 2 is accumulated along with the total of the loss functions associated with the pair. Rather than taking the (j,p) pair that minimizes the total of the loss functions, a (j,p) pair is chosen at random from all pairs within a given percentage of the minimum. This ensures that diverse yet accurate trees are constructed. We also experimented with bagging [5] and Cheng et al.'s method [11], two methods that manipulate the training examples to construct diverse trees, and with random forests [24, 6], a method that manipulates both the training examples and the input features. However, in each case the alternative led to a decrease in accuracy.

Given a new image, the individual predictions of the trees in the ensemble are combined into a single estimated illuminant $\hat{\mathbf{e}}$ for the image as follows (see Figure 3). Let $S = \{\hat{\mathbf{e}}_1, \ldots, \hat{\mathbf{e}}_k\}$ be the set of predictions from the individual trees. The final estimated

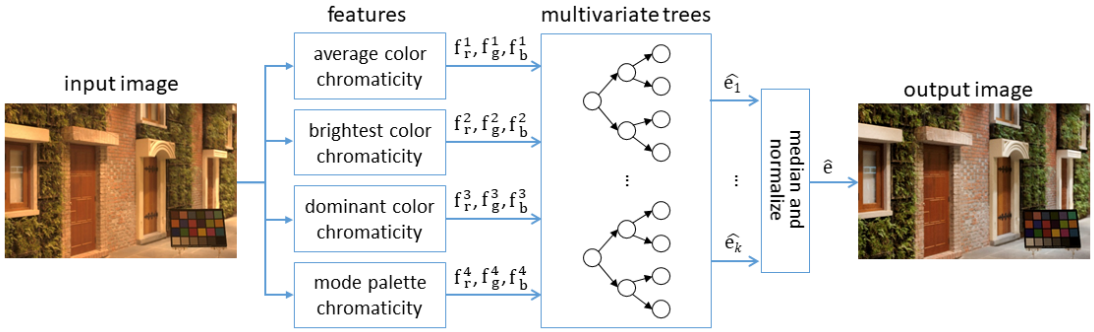

Figure 3. *Our adaptation of Cheng et al.'s [11] method. Given an input image, four triples of feature values are calculated, the feature values are used by each tree in the ensemble to predict the illumination, and the individual predictions $\{\hat{\mathbf{e}}_1,\ldots,\hat{\mathbf{e}}_k\}$ are combined into a single estimate $\hat{\mathbf{e}}$ of the illuminant that is used to white balance the image.*

illuminant is the normalized RGB vector $\hat{\mathbf{e}} = (\hat{r},\hat{g},\hat{b})$ that minimizes $\sum_{i\in S} dist(\hat{\mathbf{e}},\hat{\mathbf{e}}_i)$; i.e., the same minimization problem as the one that is solved in partitioning the training examples at a node and in labeling a leaf when constructing the trees (Equation 2).

Fitting multivariate regression trees. In fitting a multivariate regression tree to a set of labeled training examples, the idea is to repeatedly, in a greedy and top-down manner, find the best feature j and partition p that results in the largest drop in the total error, as measured by a given distance measure (see Equation 2). The minimization in Equation 2 is a constrained, non-linear optimization problem. The problem must be solved many times when learning the ensemble of trees (approximately 40,000 calls to the minimization routine is typical for constructing a single tree) and also once every time the ensemble is applied to a new image and the individual predictions of the trees are combined to obtain a final estimated illuminant. Note that learning the trees is an offline process that occurs once while applying the ensemble to a new image is an online process that will occur many times, as it occurs each time an image is captured by the camera.

The minimization problem can be solved using *exact* but sophisticated and computationally expensive numerical optimization routines such as those based on interior-point or sequential quadratic programming methods. Exact methods are suitable for offline learning of a single ensemble of multivariate trees, but unfortunately are too slow for experimental evaluation where cross-validation and repeated trials are necessary to obtain accurate estimates of performance. This impediment also arises in univariate trees when using loss functions other than squared error (see, e.g., [23, p. 342]). As well, sophisticated exact methods are unlikely to be suitable for online application of the ensemble onboard the camera, given the real-time and space considerations.

Instead, we propose to solve the minimization problem *approximately* by taking the median of the ground truth illuminants of the training examples at a node and normalizing so that the RGB values sum to one (see Example 2). The method is simple and fast. As well, the method often finds the exact solution and otherwise finds good quality approximate solutions for distance measures that arise in white balancing. Further, solving the minimization problem approximately does not appear to have a significant impact on the accuracy of the multivariate regression trees, perhaps because the tree construction itself is a greedy, and hence approximate, process.

Example 2. *Consider the follow three training examples, where the ground truth illuminants are shown but feature values are not shown, and let $dist_2$ be the distance measure.*

example	*r*	*g*	*b*
$\mathbf{e}_1$	*0.4285*	*0.4468*	*0.1247*
$\mathbf{e}_2$	*0.4221*	*0.4473*	*0.1306*
$\mathbf{e}_3$	*0.4098*	*0.4682*	*0.1220*

The median of the training examples gives (0.4221, 0.4473, 0.1247) and normalizing so that the RGB values sum to one results in the illuminant estimate (0.4246, 0.4500, 0.1254) with an associated cost of 3.5134. The mean is (0.42012, 0.45411, 0.12577) with an associated cost of 3.7619. The illuminant estimate that exactly minimizes the sum of the distance measure over the training examples is (0.4228, 0.4487, 0.1285) with an associated cost of 3.4049.

Experimental Evaluation

We compare a MATLAB implementation of our multivariate regression tree method[1] to Cheng et al.'s [11] univariate regression tree method and Shi et al.'s [32] CNN method.

We used the following image sets in our experiments. The *SFU Laboratory image set* consists of images of objects in a laboratory setting captured by a video camera under 11 different illuminants [1]. Following previous work, we use a subset of 321 images (the minimal specularities and non-negligible dielectric specularities subset). The *Gehler-Shi image set* consists of 568 indoor and outdoor images captured by two different cameras [18, 31]. As done in previous work, we used the reprocessed version of the image set which starts from a camera RAW image file and creates a linearly processed, lossless 12-bit PNG file using the well-known dcraw program. The *NUS 8-camera image set* consists of 1736 images captured by eight different cameras [9], where in most cases each camera has photographed the same scene. Each image is captured as a minimally processed camera RAW image file and is linearly processed to create a lossless 16-bit PNG file. The *NUS-Laboratory 8-camera image set* is a complementary laboratory set of 840 images captured with the same cameras. These additional images correct for a bias towards images captured outdoors in daylight [10].

[1] Our software is available at:
`https://cs.uwaterloo.ca/~vanbeek/Research/research_cp`

Experimental methodology

To assess our approach, we followed Cheng et al.'s [11] original experimental setup closely. In particular, we used the following methodology in our experimental evaluation.

Training data generation. Cheng et al. [11] use four pairs of simple features in their univariate trees (see the description in Section "Univariate regression trees") and in our main set of experiments, we used the same features. In addition, to each of these pairs of features we add a blue component f_b^i, $i = 1,\ldots 4$. Although redundant, these additional features come for free and significantly improve the accuracy of our multivariate trees. In both approaches, training and testing is done on each camera separately; i.e., a predictive model is built for a particular model of camera rather than a generic model that can be used by any camera. To construct the machine learning data for a camera, each image taken by the camera in an image set is processed by (i) normalizing the image to a [0, 1] image, using the saturation level and darkness level of the camera, (ii) masking out the saturated pixels and the color checker, if present, (iii) determining the feature values for the image, and (iv) labeling the example using the ground truth illuminant. The result is a set of labeled training examples $\{(\mathbf{x}_1,\mathbf{y}_1),\ldots,(\mathbf{x}_n,\mathbf{y}_n)\}$, where $\mathbf{x}_i = (f_r^1, f_g^1, f_b^1, \ldots, f_r^4, f_g^4, f_b^4)$ is a vector of feature values and $\mathbf{y}_i = (r_i, g_i, b_i)$ is the ground truth illuminant.

Parameter selection. In both approaches, parameters were set on 1/3 of the Gehler-Shi image set using $dist_\theta$ as the distance measure. The parameters were then fixed for all other cameras, image sets, and distance measures. Cheng et al. [11] set the following parameters: (i) the number of trees in an ensemble ($30 \times 4 \times 2 = 240$ trees, where each feature pair is used to build a tree for predicting r and a tree for predicting g, and this is repeated 30 times), (ii) the amount of overlap and the number of slices of the training data used in constructing the ensemble, and (iii) a threshold value for determining the consensus of the ensemble. In our method, we set the following parameters: (i) the number of trees in an ensemble (30 trees), (ii) the amount of randomization used in constructing the ensemble (10%), and (iii) a threshold value, where a node is partitioned only if the average error at the node is greater than the threshold (0.5).

Cheng et al. [11] use the MATLAB routine `fitrtree` for fitting a tree using the squared-error loss function. The routine has two parameters that were used at their default values: each branch node in the tree has at least "MinParentSize = 10" examples and each leaf has at least "MinLeafSize = 1" examples per tree leaf. These values are used as a stopping criteria when building the tree. We used the same stopping criteria in our implementation for fitting a tree using a chosen white balancing distance measure.

Performance evaluation. We used standard k-fold cross validation, to evaluate and compare the accuracy of our method and Cheng et al.'s [11] method. In k-fold cross validation, the image set is randomly partitioned into k approximately equal folds and each of the folds is, in turn, used as a testing set and the remaining $k-1$ folds are used as a training set. In our experiments we used 10-fold cross validation, as 10-fold is the most widely recommended, especially for our setting where the amount of data per camera is limited in some regions of the output space [26, 2, 23]. To reduce variance, the statistics we report are the result of performing 30 runs of 10-fold cross validation with different random seeds. Both methods share the same random seeds so that the partitions into training and test are the same for each algorithm for each experiment. Although for space considerations we only directly compare against Cheng et al.'s [11] method, we note that results comparing [11] to many other algorithms can be found in their original paper (the results reported here differ somewhat to those reported in [11] on common image sets as that paper uses 3-fold cross validation and reports results based on only a single run of a method).

Experimental results

We compared the approaches on accuracy, simplicity, and speed. All experiments were performed on a PC with an Intel i7-6700K, 4GHz running MATLAB R2016a.

Accuracy. Figure 1 shows a comparison of the accuracy of our method of ensembles of multivariate regression trees against Cheng et al.'s [11] method on the five distance measures discussed in Section "Distance Measures". On these image sets, our method gives comparable accuracy as measured by the mean of the worst 25% errors and a significant improvement in accuracy as measured by the mean, median, and mean of the best 25% errors. The current best-performing CNN approach [32] reports percentage improvements over Cheng et al. [11] of 5.1% for the mean, 8.2% for the median, 2.0% for the mean of the best 25%, and 4.7% for the mean of the worst 25%, where the improvement is measured over the geometric means of the eight cameras on the NUS 8-camera image set. As a point of comparison, on the combined NUS and NUS-Laboratory 8-camera image set, we achieve an improvement of 3.5% for the mean, 11.1% for the median, 24.7% for the mean of the best 25%, and -0.6% for the mean of the worst 25%, again as measured over the geometric means.

Simplicity. The only source of complexity of trees is their size and number. For the Canon 1Ds Mark III camera, we recorded the size of each tree and the size of each ensemble of trees, as measured by the number of nodes. Our method leads to ensembles that are an order of magnitude smaller. For example, averaging over 30 runs, Cheng et al.'s method has approximately 207 nodes per tree and 49,575 nodes per ensemble compared to 143 nodes and 4,317 nodes for our method, when using the $dist_\theta$ distance measure.

Speed. For the processing of an image, our method and Cheng et al.'s method are for practical purposes identical, ours being negligibly faster due to having fewer and shallower trees. As noted by Cheng et al. [11], processing an image takes 0.25 seconds or less and such speeds are comparable to simple gray-world algorithms, as used in many commercial cameras [13]. In contrast, Shi et al. [32] state that the CNN processing of an image takes approximately 3 seconds *on a GPU*.

Conclusion

Multivariate regression trees, where each tree predicts multiple responses, can be used to effectively estimate an illuminant for white balancing an image. In our method a multivariate regression tree is fit by directly minimizing a distance measure of interest. We show empirically that overall our method leads to improved performance on diverse image sets. Our ensembles of multivariate regression trees are more accurate and simpler, while inheriting the fast run-time of previous work.

Acknowledgments

The authors would like to thank Tengyu Cai, Felix Chen, Akshaya Senthil, and Weijie Wang for their assistance on the project. We also would like to thank D. Cheng, B. Price, S. Cohen, and M. S. Brown for making their code and benchmark image sets available.

References

[1] K. Barnard, L. Martin, A. Coath, and B. Funt. A comparison of computational color constancy algorithms II: Experiments with image data. *IEEE Trans. on Image Processing*, 11:985–996, 2002.

[2] Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *JMLR*, 5:1089–1105, 2004.

[3] S. Bianco, G. Ciocca, C. Cusano, and R. Schettini. Automatic color constancy algorithm selection and combination. *Pattern Recognition*, 43:695–705, 2010.

[4] S. Bianco, C. Cusano, and R. Schettini. Color constancy using CNNs. *CoRR*, abs/1504.04548, 2015.

[5] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[6] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[7] L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. CRC press, 1984.

[8] V. C. Cardei, B. Funt, and K. Barnard. Estimating the scene illumination chromaticity by using a neural network. *JOSA A*, 19:2374–2386, 2002.

[9] D. Cheng, D. K. Prasad, and M. S. Brown. Illuminant estimation for color constancy: Why spatial-domain methods work and the role of the color distribution. *JOSA A*, 31:1049–1058, 2014.

[10] D. Cheng, B. Price, S. Cohen, and M. S. Brown. Beyond white: Ground truth colors for color constancy correction. In *Proc. of the IEEE International Conference on Computer Vision*, pages 298–306, 2015.

[11] D. Cheng, B. Price, S. Cohen, and M. S. Brown. Effective learning-based illuminant estimation using simple features. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1000–1008, 2015.

[12] G. De'Ath. Multivariate regression trees: A new technique for modeling species-environment relationships. *Ecology*, 83(4):1105–1117, 2002.

[13] Z. Deng, A. Gijsenij, and J. Zhang. Source camera identification using auto-white balance approximation. In *Proc. of the IEEE International Conference on Computer Vision*, pages 57–64, 2011.

[14] T. G. Dietterich. Ensemble methods in machine learning. In *Proc. of the International Workshop on Multiple Classifier Systems*, pages 1–15, 2000. Available as: Springer Lecture Notes in Computer Science 1857.

[15] G. D. Finlayson. Corrected-moment illuminant estimation. In *Proc. of the IEEE International Conference on Computer Vision*, pages 1904–1911, 2013.

[16] G. D. Finlayson and E. Trezzi. Shades of gray and colour constancy. In *Proc. of the Color and Imaging Conference*, 2004.

[17] G. D. Finlayson and R. Zakizadeh. Reproduction angular error: An improved performance metric for illuminant estimation. In *Proc. of the British Machine Vision Conference*, 2014.

[18] P. V. Gehler, C. Rother, A. Blake, T. Minka, and T. Sharp. Bayesian color constancy revisited. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[19] A. Gijsenij and T. Gevers. Color constancy using natural image statistics and scene semantics. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33:687–698, 2011.

[20] A. Gijsenij, T. Gevers, and M. P. Lucassen. Perceptual analysis of distance measures for color constancy algorithms. *JOSA A*, 26:2243–2256, 2009.

[21] A. Gijsenij, T. Gevers, and J. van de Weijer. Generalized gamut mapping using image derivative structures for color constancy. *International Journal of Computer Vision*, 86:127–139, 2010.

[22] A. Gijsenij, T. Gevers, and J. van de Weijer. Computational color constancy: Survey and experiments. *IEEE Trans. on Image Processing*, 20:2475–2489, 2011.

[23] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data mining, Inference, and Prediction*. Springer, 2nd edition, 2009.

[24] T. K. Ho. Random decision forests. In *Proc. of the 3rd Int'l Conference on Document Analysis and Recognition*, pages 278–282, 1995.

[25] S. D. Hordley and G. D. Finlayson. Reevaluation of color constancy algorithm performance. *JOSA A*, pages 1008–1020, 2006.

[26] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc. of the International Joint Conference on Artificial Intelligence*, pages 1137–1145, 1995.

[27] D. R. Larsen and P. L. Speckman. Multivariate regression trees for analysis of abundance data. *Biometrics*, 60(2):543–549, 2004.

[28] W.-Y. Loh and W. Zheng. Regression trees for longtitudinal and multiresponse data. *Annals of Appl. Statistics*, 7(1):495–522, 2013.

[29] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[30] M. R. Segal. Tree structured methods for longitudinal data. *J. American Statistical Association*, 87:407–418, 1992.

[31] L. Shi and B. Funt. Re-processed version of the Gehler color constancy dataset of 568 images. `http://www.cs.sfu.ca/~colour/data`.

[32] W. Shi, C. C. Loy, and X. Tang. Deep specialized network for illuminant estimation. In *Proc. of the European Conference on Computer Vision*, pages 371–387, 2016.

[33] H. Vaezi Joze and M. Drew. Exemplar-based colour constancy and multiple illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36:860–873, 2014.

[34] W. Xiong and B. Funt. Estimating illumination chromaticity via support vector regression. *J. of Imaging Science and Technology*, 50(4):341–348, 2006.

Author Biography

Peter van Beek received a BSc from the University of British Columbia (1984) and an MMath and PhD in Computer Science from the University of Waterloo (1986, 1990). He has been a faculty member in Computer Science at the University of Waterloo since 2000. In 2008, he was named a Fellow of the Association for Artificial Intelligence. His research interests span the field of AI with a focus on constraint programming and applied machine learning.

Wayne Oldford received a BMath (in Statistics and in Combinatorics & Optimization) from the University of Waterloo (1977), and an MSc and PhD in Statistics from the University of Toronto (1979, 1982). He worked at Statistics Canada (1977–79) and at MIT's Center for Computational Research in Economics and Management Science (1982–86). He has been a faculty member in Statistics at the University of Waterloo since 1986. His research includes computational statistics, data analysis, and visualization.