Colour Constancy from Both Sides of the Shadow Edge

Stuart E. Lynch
University of East Anglia
Norwich, UK
stuart.lynch@uea.ac.uk

Mark S. Drew
Simon Fraser University
Vancouver, Canada
mark@sfu.ca

Graham D. Finlayson
University of East Anglia
Norwich, UK
graham@cmp.uea.ac.uk

Abstract

That image colours depend equally on the colour of surfaces and the colour of the prevailing light means that raw RGB pixel values are not correlates of object features and so, cannot be used directly in applications such as recognition and tracking. By estimating the colour of the light and then removing the colour bias due to illumination, image colours can be made to correlate with object reflectance. Yet, even the best algorithms can fail badly. The problem is made even harder in the many typical scenes where there are multiple lights, e.g. sun (yellow) and shadow (blue).

We demonstrate that the second, harder case actually provides the basis of a novel solution strategy: an illumination edge in an image provides a powerful cue to determining the underlying material surface. This paper makes three important technical contributions. First, we take an existing analytic one-surface two-lights estimation algorithm (which assumes that lights lie on a line in chromaticity space) and re-engineer it as a voting algorithm which relaxes the prior assumption about where lights might lie. Second, we model illumination change using a 3×3 linear matrix as opposed to the more approximate diagonal model that is used in the prior art for this problem. Being able to more accurately account for illumination change provides a step change in the performance of our method. Lastly, we re-engineer a prior art shadow-edge generator to suit this problem.

1. Introduction

The aim of a visual system is to understand a scene given only the sensory responses to light. Given that the spectrum of light can vary (i.e. daylight – blue, sunlight – yellow), the observed colours of surfaces will have an intimate relationship with the lighting conditions of the scene. Humans go about their lives unaffected by this relationship and perceive colours in an unchanging way. A white T-shirt appears white to us whether we are indoors under fluorescent lighting, or outdoors under sunlight. This remarkable ability is known as Colour Constancy.

The automatic mode in all modern digital cameras uses an algorithm (white balance) to balance the colours in a photograph to normalise against the colour cast caused by the illuminant. A typical camera uses tristimulus sensors. Each pixel is represented by three intensities recorded in the Red, Green and Blue parts of the spectrum. The effect of illumination at a pixel is often modelled as a diagonal matrix (a scaling of the channels)

$$\begin{bmatrix} p_r \\ p_g \\ p_b \end{bmatrix} = \begin{bmatrix} E_r & 0 & 0 \\ 0 & E_g & 0 \\ 0 & 0 & E_b \end{bmatrix} \begin{bmatrix} S_r \\ S_g \\ S_b \end{bmatrix} \quad (1)$$

where S_k and E_k are the surface reflectance and illuminant RGBs respectively ($k = \{r, g, b\}$). Usually, the process of white balancing in a camera assumes that the illuminant is uniform across the whole image. The implication is that a single E_k can be associated with every pixel in a scene, and is divided out in the balanced image. This assumption is used as the foundation for most colour constancy algorithms. However, it is not hard to imagine a scene that contains more than one illuminant. Any outdoors scene containing shadows will contain more than one illuminant. Direct sunlight is yellowish. Cast shadows (with direct sunlight blocked) will appear blue, causing a change of illumination at the shadow edge. In this paper, we make no assumptions about known or partially known geometry (as in [25]) and in fact need not necessarily focus on images with shadow boundaries, but instead work with any illumination change.

Gijsenij *et al*. [20] proposed dividing an image into segments, with the assumption that each segment has uniform illumination. They then used traditional colour constancy algorithms on the segments. These acted as illuminant votes. Instead we propose a method that more robustly finds areas of illumination change in an image, and propose a new algorithm which actually uses the change in illumination to get a solid estimate.

Forsyth, in his gamut mapping algorithm [16], examined the case where many surfaces are lit by a single illuminant. He was able to derive a set of mappings which could all feasibly render the image under a canonical illuminant. He observed that the process of illuminating a scene limits the colours that can be viewed. If an image contains strongly chromatic reds, then the scene couldn't have been illumi-

nated by a strongly chromatic blue. There have since been many different variants of this algorithm [8, 19, 2], all assuming that a scene is lit by a single illuminant. Images with many shadows are generally considered hard cases. In this paper, we not only aim to address this problem, but actually use the varying illumination directly to derive a solution.

Our method is tested on a set of images due to Gehler [17], taken on the Canon 5D camera, which is commonly used to benchmark estimation algorithms. As proof of concept, on a manually annotated image set and for the 52 images where there is indeed an illumination change, remarkably, given as little as only a single surface viewed under two lights (two pixel RGBs), we estimate the illuminant to a greater accuracy than all existing algorithms benchmarked on this data set. Then by adjusting to the problem at hand a prior art photometric-invariant method (that provides pixels only weakly dependent on light colour), we engineer an algorithm that can automatically find illumination edges to suit the problem. We then obtain almost the same excellent performance when our automatic illumination edge finder is used on the original non-annotated images.

2. Background

Image formation at a pixel can be modelled as:

$$p_k^x = \int_\omega S^x(\lambda)E(\lambda)R_k(\lambda)d\lambda \tag{2}$$

with $k = 1..3$ for colour samples p_k^x at pixel location x. The function $S^x(\lambda)$ represents the material reflectance properties of the surface and $E(\lambda)$ is the illumination spectrum of the scene illuminant. This is often assumed to be constant across the entire scene. $R_k(\lambda)$ is the sensitivity function of the kth sensor, with the product integrated across the visible spectrum ω.

Suppose we were to model our camera sensors as Dirac Delta functions (yielding the von Kries model [28]). For such narrowband sensors eq.(2) then simplifies to:

$$\underline{p}^x = \mathbf{E}\underline{S}^x \tag{3}$$

where $\mathbf{E}$ is a diagonal matrix with elements $\mathbf{E}_{kk} = \int_\omega E(\lambda)R_k(\lambda)d\lambda$, and column vector $\underline{S}^x = \underline{S}_k^x = \int_\omega S^x(\lambda)R_k(\lambda)d\lambda$. This model roughly holds for real cameras as well, and the sensors can be 'sharpened' – the RGBs are mapped by a 3×3 matrix – to maximise the validity of this model [9]. While spectral sharpening makes the factor model of image formation (3) work better, modelling light change using a full 3×3 matrix (rather than the diagonal matrix in the factor model) works still better (a point we return to later in the paper).

Most colour constancy algorithms do not try to recover the brightness of the illuminant as this is deemed to be unrecoverable since light and surface are multiplied. Therefore authors have often tried to recover the 2-dimensional chromaticity of the illuminant instead [8], normalising for brightness. A projective band-ratio chromaticity space which projects RGBs onto the plane $b = 1$ is:

$$\mathrm{chrom}\left(\begin{bmatrix} p_r & p_g & p_b \end{bmatrix}\right) = \begin{bmatrix} \frac{p_r}{p_k} & \frac{p_g}{p_k} & \frac{p_b}{p_k} \end{bmatrix} \tag{4}$$

where $k \in \{r, g, b\}$. Since one of these 3 values will always equal 1, it is common to only use the remaining 2 dimensions to represent the chromaticity (i.e. $[p_r/p_b, p_g/p_b]$). It is trivial to show that $\mathrm{chrom}(x) = \mathrm{chrom}(\alpha x)$ where α is an arbitrary scalar. A 3×3 diagonal matrix, however, does have a 2×2 diagonal matrix equivalent in band-ratio chromaticity space. This means eq.(3) can be expressed in exactly the same form in this space.

2.1. Colour Constancy under Varying Illumination

The range of illumination colours is often linked to the temperature of blackbody radiators. These are modelled using Planck's formula: $E = E(\lambda, T)$ where T is the temperature in Kelvin of the blackbody. Calculating tristimulus values for the Planckian spectra yields a curve in chromaticity space known as the Planckian locus. Assuming all real-world illuminants lie near this curve is a powerful constraint. In fact modern day 'Colour Rendering Indices' which effectively score commercial lamps only give a high score for lights which behave as if they were Planckian. It is remarkable that the vast majority of artificial lights mimic nature and have colours on or close to the Planckian locus. By assuming lights were Planckian, Finlayson and Hordley [12] demonstrated that colour constancy was – to some extent – theoretically possible at a per-pixel level. Specifically, that they could find one number (a non-linear function of the band ratios) that was intrinsic, i.e., it only depended on the reflectance and was independent of illumination.

The Planckian locus has a well known characteristic curved shape when plotted in chromaticity space. Importantly, Finlayson *et al*. [11] observed that when the factor model of image formation is adopted Planckian illuminants lie very close to a line in *inverse chromaticity space*: when we plot b/r against b/g (assuming we use $k = b$).

Further, suppose that a white light has an RGB of [1,1,1]; then clearly its spectral band ratio maps to [1,1]. The mapping taking the reference white light (assuming the factor model in spectral band ratio chromaticity space) to another light with coordinates (a,b) is a diagonal matrix with these terms along its diagonal. When we invert this map then it becomes the mapping required to take some light back to the reference white light conditions. That is, we can think of the inverse chromaticity space as the set of mappings that take spectral band ratios from one light back to the reference. This approximate linearity is shown in Fig. 1.

$$\mathrm{inv\_chrom}\left(\begin{bmatrix} p_r & p_g & p_b \end{bmatrix}\right) = \begin{bmatrix} \frac{p_b}{p_r} & \frac{p_b}{p_g} \end{bmatrix} \tag{5}$$

We introduce the caret $\hat{}$ to symbolise spectral band-ratio chromaticities, such that $\hat{\underline{p}} = \mathrm{chrom}(\underline{p})$. We also introduce

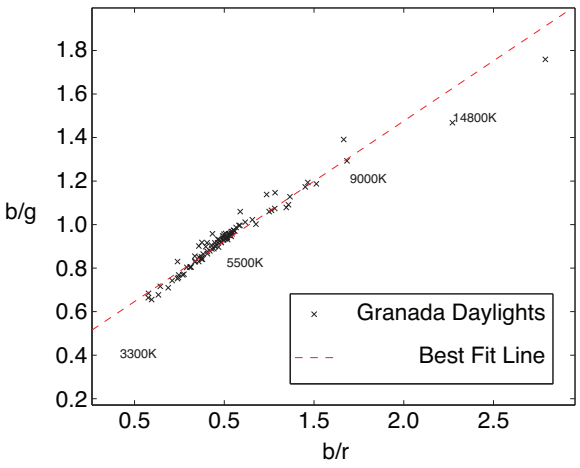

Figure 1. Illuminant chromaticities from the Granada, Spain dataset [21] and the Planckian Locus plotted in inverse chromaticity space. The chromaticities were rendered using the measured sensitivities of the Sony DXC-930 camera [1].

the additional superscript ϵ into eq.(3) to denote different illuminants. So $\underline{p}^x$ shall hence be written as $\underline{p}^{\{x,e\}}$.

Let us denote the set of mappings back to the reference light (the line shown in the Fig.1) as Φ. Given a chromaticity under a first unknown light $\hat{\underline{p}}^{\{x,1\}}$ we denote the set of possible corresponding chromaticities under the reference light as $\hat{\underline{p}}^{\{x,1\}}\Phi$: i.e., we apply all possible mappings. Also, $\hat{\underline{p}}^{\{x,1\}}\Phi$ must define a line a chromaticity space. It follows that, given two chromaticities of the same surface under two unknown lights, each induces a line of possible corresponding matches under the reference and their intersection can be used to estimate the actual true chromaticity:

$$\hat{\underline{S}}^x = \hat{\underline{p}}^{\{x,1\}}\Phi \bigcap \hat{\underline{p}}^{\{x,2\}}\Phi \tag{6}$$

This method works well when all the model assumptions hold. However, for real world data the method is not very stable. Indeed, the lines of chromaticities (under the reference lights) generated for two similar unknown lights are very close to one another. Extremely small perturbations from perfect model conditions results in the intersection of these lines changing markedly and this leads to an incorrect estimation of surface colour. Indeed, for the method to work well the two unknown lights need to be quite different in colour.

Kawakami and Ikeuchi [22] argued that real world illuminants are constrained in chromaticity space so they are not well represented by an infinite line, and proposed representing the set as a finite line segment. The line segment was chosen to span the colour temperatures of daylight. However, this formulation posed the problem of the line segments not intersecting in many cases. To enforce an intersection they translate one of the line segments and this generally forces an intersection. However, by the nature of how they remedy the non-intersection problem the intersection point always contains the endpoints of at least one of the segments.

Finlayson and Lynch [7] extended the Kawakami and Ikeuchi work and proposed a solution which solved for the point simultaneously closest to both lines. The least squares optimal solution was shown to produce better performance.

While progress has been made the degree of constancy the 'line intersection' approach provides is limited and does not seem easy to extend to multiple surfaces. Further, at the heart of the method is the assumption that we can find illumination change in an image. Kawakami *et al*. [23] also examine this problem using shadows, as well as imaging the same scene over time. They do not, however, include an explicit shadow edge detection algorithm. The problem of finding illumination change in a single image is interesting and yet has not been integrated with illuminant estimation algorithms.

2.2. Detecting Illumination Change in Images

Finlayson *et al*. [13] proposed a method for removing shadows in images which crucially involves finding those edges in the image due to an illumination change. Key to their method was the adoption of Wien's approximation of Planck's formula as a theoretical guide to model the possible illuminants in a scene (with the algorithm developed working for real lights notwithstanding). Using Wien's approximation, pixel response for a light of colour temperature T, $p^{\{x,T\}}$ is written as:

$$E(\lambda,T) \approx c_1\lambda^{-5}e^{\frac{c_2}{T\lambda}} \rightarrow p_k^{\{x,T\}} = c_1\lambda^{-5}e^{\frac{c_2}{T\lambda_k}}S^x(\lambda_k) \tag{7}$$

where we assume the sensors are Dirac Delta functions – or via Spectral-Sharpening [9] behave like Delta functions – centred at wavelengths λ_k. The logarithm of the spectral band ratio chromaticities can be written as:

$$\log \hat{p}_k^{\{x,T\}} = \log(s_k/s_b) + (e_k - e_p)/T \tag{8}$$

where $s_k \equiv c_1\lambda_k^{-5}S(\lambda_k)$ and $e_k \equiv -c_2/\lambda_k$. As temperature changes, $\log \hat{p}^{\{x,T\}}$ traverses a straight line. Those authors showed that projecting all chromaticities onto the perpendicular to this line normalises against illumination change: the resulting chromaticity image has no shadows. In [14] the edges found in the 'illumination independent' image is compared with the edges found in the colour original. The edges in the latter but not the former are illumination edges. We extend this approach later in the paper.

Finlayson *et al*. [10] also showed that taking two images of the same scene with the second exposure taken through a coloured filter made illumination change an easier problem to solve. However, here we attempt to develop a solution for conventional cameras. Indeed, our method is tested on a ground truth data set which is not of our creation.

3. Estimating the illuminant at illumination edges

Broadly, this paper makes two important contributions: First we re-engineer the Kawakami and Ikeuchi and Finlayson and Lynch algorithms to work in a voting framework. We shall see two benefits of this approach. First, our method is more robust to imperfect input data. And, second it extends naturally to the situation where we have more than one surface colour observed under the different lights illuminating a scene. Second, we develop a method for automatically finding illumination edges in an image.

3.1. The Voting Algorithm

The 3×3 matrix $\mathbf{M}^i$ takes all RGBs measured under the ith light to a reference light (typically white light). That a linear mapping suffices for this task is well understood [6] and furthermore is in practice easy to measure (we relate the RGBs for known surface colours recorded under a reference and another light). For a large range of typical lights we pre-calculate a set of mappings.

In analogy to the line intersecting approach we wish to map two RGBs (for the same surface under two unknown lights) to a single reference colour. At the same time we would also like to introduce more accuracy into the mapping model. We currently use the diagonal model. Ideally we would like to use 3×3 maps having 9 degrees of freedom. The problem with this previously is that a full 3×3 matrix does not have a 2×2 equivalent in chromaticity space. Therefore the line approach would not translate directly. However, suppose we have a 3×3 matrix $\mathbf{M^1}$ which maps RGBs under illuminant 1 to a canonical illuminant. Similarly suppose we also have a matrix $\mathbf{M^2}$ which maps RGBs under illuminant 2 to a canonical illuminant. We would expect $\text{chrom}(\mathbf{M}^1\underline{p}^{\{x,1\}}) = \text{chrom}(\mathbf{M}^2\underline{p}^{\{x,2\}})$. Let's assume that we have some representative set of real-world illuminant RGBs. We then calculate a mapping matrix $\mathbf{M}$ for each illuminant. Given pixels of a set of m surfaces x under two unknown lights $\underline{p}^{\{x,1\}}$ and $\underline{p}^{\{x,2\}}$ then we can calculate the following distance measure

$$\theta^x_{i,j} = \text{ang}\left\{\mathbf{M}^i\underline{p}^{\{x,1\}}, \mathbf{M}^j\underline{p}^{\{x,2\}}\right\} \tag{9}$$

where i and j index the angle being solved for a pair of lights. The angle between the two vectors is used because it is a uniform measure of difference that is intensity invariant. If we test all the pairs of n illuminants for a single surface then we would calculate n^2 angles. We could now just pick the illuminant pair with the smallest angle. However, this does not make use of this extra surface information so we propose to frame this in a voting scheme. We want to convert the n^2 angles into meaningful votes for each light. First, we map our angular measure to the interval [0,1]:

$$v(\theta, \sigma) = \exp(-\theta^2/(2\sigma^2)) \tag{10}$$

The smaller the σ the more bias the voting schema will have towards smaller angles (small angle = big vote). As we combinatorially evaluate pairs of candidate lights, we maintain two voting polls for each of the n illuminants. A vote is cast by adding $v(\theta^x_{i,j}, \sigma)$ to each illuminant's total in a pair. The outcome of this process is graphically represented in Fig.2. Introducing additional surfaces means we have more votes for each illuminant. Since all surfaces should vote for the same illuminant pair, then the process becomes more robust.

We then have to choose two illuminants from the voting polls. We could simply take the illuminants that had the highest number of votes. However, suppose several of our test illuminants circle the correct answer, which isn't in our set: then they will yield an equally high vote. Taking the average of these will produce a better result than just picking one. We threshold illuminants that are highly voted for and use their vote as a weight to output a weighted average. This has shown to perform better than choosing the pair of illuminants with the smallest angle. We have found that weighting the lights that account for the top 20% of votes yielded good performance.

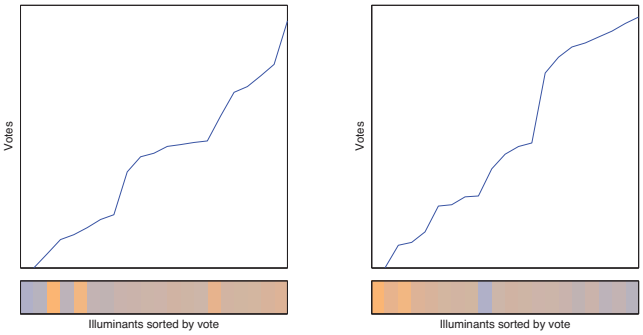

Figure 2. The votes for each illuminant on both sides of the shadow edge. The illuminants are sorted in ascending order by the number of votes.

Notice that this algorithm easily incorporates multiple surfaces into its formulation. In contrast, it is not clear how the other two-illuminant algorithms could introduce additional surface information to stabilise the answer.

3.2. Estimating the Illuminants on Both Sides of an Illumination Edge

Here we wish to find pixel pairs where illumination changes but material properties are the same. We make the general assumption that there are two illuminations in the image, in-shadow and out-of-shadow. The algorithm in the previous section works because there is an underlying predictability regarding illumination colours in the world. This property has been previously exploited to create illumination invariant images. In section 2.2, illumination changes are absent, or at least greatly attenuated. Then comparing the gradient fields in the original image with those in the invariant image, we should be able to find edges that appear

in the original but are not evident in the invariant.

The objective, however, is not to diminish these edges in the original, zeroing them out or inpainting across them as in [14], with a view to re-integrating the modified gradient field into a shadowless RGB image.

Instead, we simply wish to identify illumination-change boundaries with the aim of establishing chromaticities on either side of the edge, for our illumination-voting scheme. We make use of a Mean-Shift (MS) segmentation [3] of the image for establishing relevant edges. This has the advantage that shadow-edges can be quite blurry, in that what we are after is colour on either side: and the MS segments can very simply supply an estimate of RGB triples on either side without need for accuracy since each segment draws upon many pixels to colour its region. This represents an advance over the necessity for sharp illumination edges in [14].

3.3. Detecting Shadow Edges

Suppose we have used the invariant-image formulation in [14] to generate a greyscale, illumination-free image. In fact, we need not remove colour completely, and can generate a more well-behaved invariant L_1 chromaticity image as in [5]. Denote this image by $\boldsymbol{\chi}$, a 3-band image where each pixel sums to unity. The shadow removal technique in [14] requires a calibration step. This step ties well with that of the voting algorithm, since they both require collecting a set of surfaces under many different illuminants. From the collected data we can calculate the direction of illumination change in chromaticity log space, and also pull out a representative set of lights and mappings.

Now form the MS segmentation of the input image; let us call this RGB segmentation image $\boldsymbol{R}_M$. Here, colour helps us segment the input image. It is straightforward to generate a binary edge map for $\boldsymbol{R}_M$ using the Canny edge-finder; however here we use the Canny mechanism but instead generate a gradient pair R_x, R_y instead of a binary output, making use of a grayscale version ρ of $\boldsymbol{R}_M$ formed by adding the colour-channels of $\boldsymbol{R}_M$.

Now we form the MS segmentation of the invariant $\boldsymbol{\chi}$. Call this 3-band image $\boldsymbol{\chi}_M$. To make a simple gradient field out of this chromaticity image, we would once again like to form a grey image X as the sum of colour bands. However, chromaticities add to unity; therefore instead we use a grey formed in the L_2 norm, as $X = (\sum_{k=1}^{3} \boldsymbol{\chi}_k^{\,2})^{1/2}$. Denote the gradient pair thus formed out of the invariant-chromaticity image by χ_x, χ_y.

We then compare the image MS segmentation with the invariant MS image using a similar edge-comparison procedure as in [14]: perform non-maximum suppression as in Canny edge detection, using 4 quadrants of directions, and as well remove non-shadow edges by removing edges occurring in the original image with the same orientations as edges occurring in the same location in the invariant image. Finally, noting that isolated dots and lines in the original image segmentation appear weakened in the invariant-chromaticity image, these pixels are removed from the generated shadow-edge map. The result is a set of 1-pixel-wide shadow boundaries, bordering on input-image MS edges.

3.4. Culling Edge Pixels

For this application it is not required to be completely successful in identifying all shadow-edge pixels – see, e.g., Fig. 3, showing candidate edges identified. As well, we do not need every instance of the same across-edge RGB pairs from the MS image of the input RGB (even a single pixel pair would do, in theory).

Figure 3. Candidate illumination-edge pixels automatically identified.

We next find across-edge pairs by finding, for each pixel on a MS image edge, the closest pixel in a different segment than that pixel's own segment. This always works, to find across-edge colour pairs (and is fast). We then have one RGB triple and a second RGB triple for each illumination-edge pixel. We put these in luminance order by asking which triple is darker (has smaller R+G+B), and then remove all but the unique 6-vector (3 plus 3) RGB pairs, RGB_1 and RGB_2.

At this point we can remove further edge pixels by asking, amongst all the pixel-pairs, which ones are well described by a *single* representative 3×3 transform $\boldsymbol{M}$ from RGB_1 to RGB_2. To do so, we make use of the Least Median of Squares (LMS) regression [26], which automatically identifies outliers. The LMS procedure also delivers a robust version of the Coefficient of Determination, R^2, which must be sufficiently high in order to endorse having indeed found illumination boundaries, in the current input image.

At this point, we have a much reduced set of RGB-triple pairs, for which we can apply our illuminant-voting algorithm.

4. Experiments

Gehler *et al.*'s "ColorChecker" dataset is one of the most widely tested image sets to evaluate illuminant estimation; therefore, for the subset of images therein where there are clearly two lights, we will evaluate both our algorithms for

finding illumination change and our voting algorithm. The ColorChecker dataset also the includes the same 24 patches in every image. Thus, we have many pairs of surfaces under many lights, which is ideal to evaluate how well our voting approach works (independent of the illumination edge finding algorithm). We also evaluate our algorithm on a smaller set of 30 images we captured ourselves where there was a clear change in illumination (the kind of scenes where conventional illuminant estimation algorithms fail).

4.1. A linear ColorChecker Dataset

Because light change is linear (a 3×3 matrix maps RGBs across illumination) only for linear data, we decided to use the linearised ColorChecker images helpfully supplied by Funt and Shi [27]. However, we found that the linearised versions available did not in fact correct for the dark-level values. We note that they have recently provided an accompanying Matlab script which does precisely this.[1]

However a great deal of work has been processed using the uncorrected version (e.g. in [18]). One manifestation of this error is that these images (because they have a black point that is higher than they should) can look flat when rendered for display.

Using DCRAW we rerendered our own images[2], accounting for the black point. DCRAW automatically colour corrected the images to a D65 white point. The colour correction is simply a diagonal matrix with diagonal elements [2.7, 1.15, 1.41]. The correct whitepoint is chosen to be the mean of the non-clipped grey-patches on the Macbeth ColorChecker. Note that if the illuminant is redder or bluer than D65, achromatic surfaces in these images will be respectively redder or bluer than they should be: i.e. by applying this correction there is still a need for illuminant estimation. In Fig.4 we show two images linearised by Shi and Funt [27], the same images with the black level corrected, and our D65 set (gamma is applied for display). The original 568-image Gehler data set comprises images for two different cameras. Some of our experiments use the same surface under different illuminants across different images; for this approach to be valid it is important that the camera space is constant over the images. We therefore experiment only on the 482 images taken on the Canon 5D camera.

4.2. Macbeth ColorChecker Experiments

Since every image in the dataset contains a Macbeth ColorChecker chart, we have 24 surfaces under many illuminants. We can use these as input for our algorithm since edges per se are not required. To run the algorithm we choose two thirds of the images at random. The illuminant set is chosen such that they best represent the span of all of the illuminants captured. We do this by evenly sampling the gamut of illuminants and choosing ones that lie closest to our samples. We can calculate the associated 3×3 mapping matrix for each illuminant by solving a least squares optimisation between each illuminant and the chosen canonical illuminant in the set (we choose one that is closest to white).

Figure 4. **Top:** The Shi and Funt reprocessing of the Gehler Dataset. **Middle:** Corrected Black Level. **Bottom:** Our Processing with Black Level Correction and D65 Colour Correction.

Our algorithm is then run on the remaining third of the images. We randomly select a single surface from two images then calculate the illuminants and repeat this experiment 1000 times. This protocol is repeated assuming we have 9 surfaces under two lights (we can do this because the checker has many surface colours). The whole experiment is cross validated 3 times.

In common with almost all illuminant estimation algorithms we compare our recovered illuminants to the actual lights and calculate the angular error. The mean and median angular error performance of 1- and 9-surface two light voting illuminant estimation are shown in the last two rows of Table 1.

Of course, it is important to benchmark our algorithm against conventional algorithms. This not as immediately easy as it first sounds. Indeed, while there is a plethora of statistics for the ColorChecker dataset the algorithms are tested (naturally enough) on the whole images. Further, the community is focussing on the Shi and Funt data set which we found to be lacking black-point correction.

However, based on a large number of algorithm evaluations, simple pixel gamut mapping [16] is a leading algorithm. Gamut mapping works by trying to find the illuminant correction that takes RGBs observed under a single unknown light mapped back inside a known gamut of colours for the reference light. We build the gamut for the Macbeth

[1]File http://www.cs.sfu.ca/∼colour/data/process_568.m, create date: 9/16/2013 - modify date: 9/16/2013.

[2]Data at http://colour.cmp.uea.ac.uk/datasets/reprocessed-gehler.html

patches under the reference light. We adopt the same randomised protocol as for evaluating our voting algorithm but now for each light in the pair we randomly select 5 patches and run gamut mapping. We remark that 5 patches constitutes 20% of the colours in the checker which is a large range of colour diversity. Indeed, it is large enough that gamut mapping should work well.

The results of gamut mapping and three other algorithms are presented in Table 1. For the one-surface voting algorithm we observe about the same performance as gamut mapping. This is extremely encouraging. It appears a single surface viewed under two lights provides a powerful cue to determining what those lights are. Performance gets better when we can have more surfaces available.

	Our Rendering	
Algorithm	Mean	Median
Grey World	7.3	6.3
Max RGB	6.5	5.4
Shades of Grey [15]	6.6	5.3
Gamut Mapping [16]	4.3	2.3
Voting Algorithm 1 Surface	4.2	2.1
Voting Algorithm 9 Surfaces	3.0	1.6

Table 1. Using known ColorChecker patches

4.3. Shadow Edge Detection Experiments

Here we incorporate the automatic finding of illumination edges with our voting algorithm. We have chosen a subset of 52 images that actually contain two illuminants (shadow edges). For each image the illumination-edge detection system outputs pairs of pixels which represent the same surface under two lights. We can directly feed this into the voting algorithm to get an estimate for the illuminant colour on either side of the shadow edge. However, since the images in fact contain only a single ColorChecker, we can only measure our performance for one of the white point outputs (the one in the same illumination as the checker).

To build our set of illuminant maps we used the 430 $(482 - 52)$ images that were not in the 52 test images. In Table 2 we again compare this against Gamut Mapping[18].

	Our Rendering	
Algorithm	Mean	Median
Grey World	5.5	4.3
Max RGB	5.3	4.4
Shades of Grey [15]	4.6	3.4
Grey Edge [4]	4.5	2.6
Gamut Mapping [16]	5.6	3.2
Voting Algorithm with Shadows	4.0	1.3

Table 2. Using automatically-detected shadow edges, in 52 images that have shadows; compared to prior art run on these images.

Results are reported in Table 2. The voting algorithm's median error is really very good though the higher mean (still the best compared with the other algorithms) indicates some poorer performance on some images. Of course it is not entirely surprising that gamut mapping doesn't perform as well here as does our algorithm: gamut mapping is expecting surfaces rendered under the same illuminant as input.

4.4. Custom Dataset Experiments

To further evaluate our method we captured a new set of 30 raw images using a Nikon D60 SLR camera. Images were taken at different locations in the UK and the USA. Immediately after each photograph was taken, two separate images were taken of a Macbeth ColorChecker, inside and outside of shadow. We can use the white patches of the ColorChecker as an estimate for the two illuminant colours. Importantly, these images always have two illuminants. As such they are not suitable for estimation using a conventional algorithms.

Now, from the initial images, with no colour charts in them, we again attempt to automatically detect shadow edges and hence form an estimate of the two illuminant colours. We show an example image from our set in Fig.5. Results are shown in Table 3. For comparison we also implemented and ran a version of the Finlayson and Lynch [7] algorithm, which assumes all illuminant mappings lie on a line segment. This was conceived as a single-surface two-light algorithm.

Figure 5. **Top Left:** Original Image with shadow edge candidates. **Top Right:** MS Image **Bottom Left:** Shadow Free Greyscale Image **Bottom Right:** Shadow Free Chromaticity Image

The first 2 rows of Table 3 show performance of the line segment intersection approach. The first entry is using paired surfaces from the ColorCheckers. The second row shows performance calculated for an actual illumination change in the image. Row 3 reports our 1 surface voting algorithm again for the checker patches. There is a significant gain in performance delivered by voting. We again randomly select 9 surfaces and see a modest performance improvement. This result (and those in Table 1) indicate most of the power in two-light illuminant estimation is for

the first pair of surfaces. Lastly, in row 5 we report the results for our automated approach where we automatically find and exploit – what we think are – illuminant edges. Again performance is significantly better than the prior art.

	Our Rendering	
Algorithm	Mean	Median
Line Segments [7] on ColorChecker	5.2	3.2
Line Segments [7] on Shadows	5.1	4.4
Voting Algorithm 1 Surface	2.2	1.6
Voting Algorithm 9 Surfaces	2.0	1.5
Voting Algorithm with Shadows	3.2	2.2

Table 3. New Nikon dataset, automatic shadow-edge plus voting

5. Conclusions

We have proposed an algorithm which detects pixels on both sides of shadow edges in an image, and then uses this information to estimate the two illuminant colours. Further, we developed a voting strategy for illuminant estimation which we show provides a significant improvement over the prior art.

The results we present here are important for three reasons. First, images with two lights are notoriously hard (impossible) to solve for the majority of algorithms which assume there is a single illuminant present. Second, we show that we can find illumination change in images using quite a simple algorithm. Third, even when there is just a single surface under two different lights, we get a very strong cue to what the illuminant colour actually is.

This work captures the zeitgeist of current illuminant estimation research in that we are not proposing to solve the whole problem but, rather, have identified a particular case which we can solve (and for which traditional algorithms fail). This leads us to believe that general algorithms for illuminant estimation should be engineered from more specialised components that are designed to solve particular classes of images (e.g. those with faces or those with shadow edges)

References

[1] K. Barnard, L. Martin, B. Funt, and A. Coath. A data set for color research. *Color Research & Application*, 27(3):147–151, 2002.

[2] S. Bianco and R. Schettini. Color constancy using faces. In *CVPR 2012*, pages 65–72. IEEE, 2012.

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002.

[4] J. V. de Weijer and T. Gevers. Edge-based colour constancy. *Image Processing*, 16:9:2207–2214, Jan 2007.

[5] M. S. Drew, G. D. Finlayson, and S. D. Hordley. Recovery of chromaticity image free from shadows via illumination invariance. In *Workshop on Color and Photometric Methods in Computer Vision, ICCV*, pages 32–39, 2003.

[6] M. S. Drew and B. V. Funt. Natural metamers. *CVGIP*, 56(2):139–151, 1992.

[7] G. D. Finalyson and S. E. Lynch. Revisiting surface colour estimation under varying illumination. *CIC18*, pages 301–306, Nov 2010.

[8] G. D. Finlayson. Color in perspective. *PAMI*, 18(10):1034–1038, 1996.

[9] G. D. Finlayson, M. S. Drew, and B. V. Funt. Spectral sharpening: sensor transformations for improved color constancy. *JOSA*, 11(5):1553–1563, 1994.

[10] G. D. Finlayson, C. Fredembach, and M. S. Drew. Detecting illumination in images. In *ICCV*, pages 1–8. IEEE, 2007.

[11] G. D. Finlayson, B. V. Funt, and K. Barnard. Color constancy under varying illumination. In *ICCV*, pages 720–725. IEEE, 1995.

[12] G. D. Finlayson and S. D. Hordley. Color constancy at a pixel. *JOSA*, 18(2):253–264, 2001.

[13] G. D. Finlayson, S. D. Hordley, and M. S. Drew. Removing shadows from images. *ECCV*, pages 129–132, 2002.

[14] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew. On the removal of shadows from images. *PAMI*, 28(1):59–68, 2006.

[15] G. D. Finlayson and E. Trezzi. Shades of gray and colour contancy. In *CIC12*, pages 37–41, 2004.

[16] D. A. Forsyth. A novel approach to colour constancy. In *ICCV*, pages 9–18. IEEE, 1990.

[17] P. V. Gehler, C. Rother, A. Blake, T. Minka, and T. Sharp. Bayesian color constancy revisited. In *CVPR*, pages 1–8, 2008.

[18] A. Gijsenij, T. Gevers, and J. V. de Weijer. Computational color constancy: Survey and experiments. *Image Processing*, (99):1–1, 2011.

[19] A. Gijsenij, T. Gevers, and J. van de Weijer. Generalized gamut mapping using image derivative structures for color constancy. *IJCV*, 86(2):127–139, 2010.

[20] A. Gijsenij, R. Lu, and T. Gevers. Color constancy for multiple light sources. *Image Processing, IEEE Transactions on*, 21(2):697–707, 2012.

[21] J. Hernández-Andrés, J. Romero, R. L. Lee Jr, et al. Colorimetric and spectroradiometric characteristics of narrow-field-of-view clear skylight in granada, spain. *JOSA*, 18(2):412–420, 2001.

[22] R. Kawakami and K. Ikeuchi. Color estimation from a single surface color. In *CVPR*, pages 635 – 642, 2009.

[23] R. Kawakami, K. Ikeuchi, and R. T. Tan. Consistent surface color for texturing large objects in outdoor scenes. In *ICCV*, volume 2, pages 1200–1207. IEEE, 2005.

[24] S. E. Lynch, G. D. Finalyson, and M. S. Drew. Reprocessed linear version of the gehler dataset, with black level corrected. http://colour.cmp.uea.ac.uk/datasets/reprocessed-gehler.html, 2013.

[25] A. Panagopoulos, T. F. Y. Vicente, and D. Samaras. Illumination estimation from shadow borders. In *ICCV*, pages 798–805. IEEE, 2011.

[26] P. J. Rousseeuw. Least median of squares regression. *JOSA*, 79(388):871–880, 1984.

[27] L. Shi and B. Funt. Reprocessed version of the gehler color constancy dataset of 568 images. http://www.cs.sfu.ca/ colour/data/.

[28] J. Von Kries. Influence of adaptation on the effects produced by luminous stimuli, 1970.