Color Constancy and White Balancing

Earl Wong

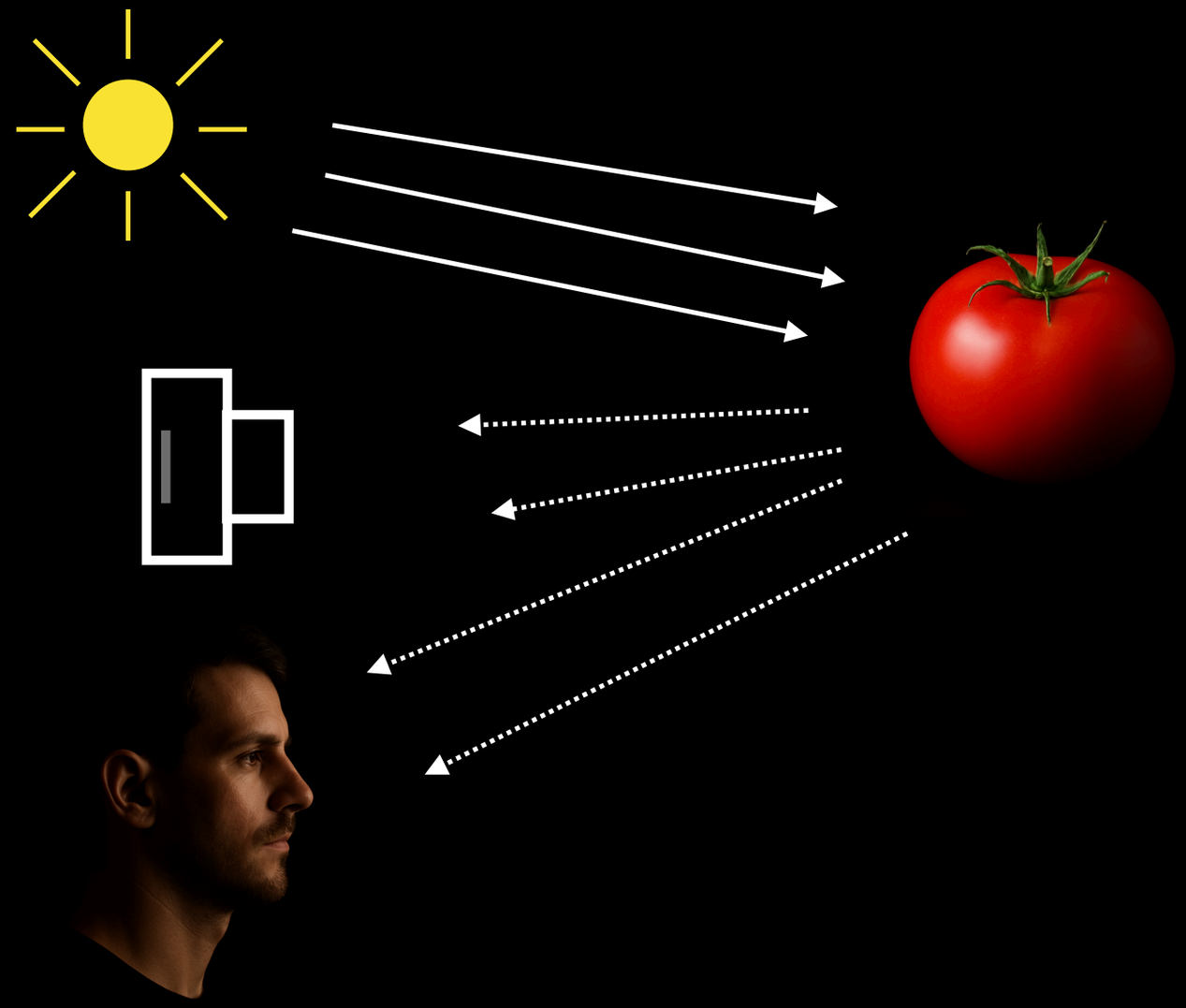
The Physical Problem

The colors recorded by the camera depend mainly on the surface reflectance of the object (i.e. its intrinsic color) and the color of the illumination.

The colors perceived by the human subject depend mainly on the surface reflectance of the object.

=> colors recorded by camera <> colors perceived by human subject

=> this difference can be large

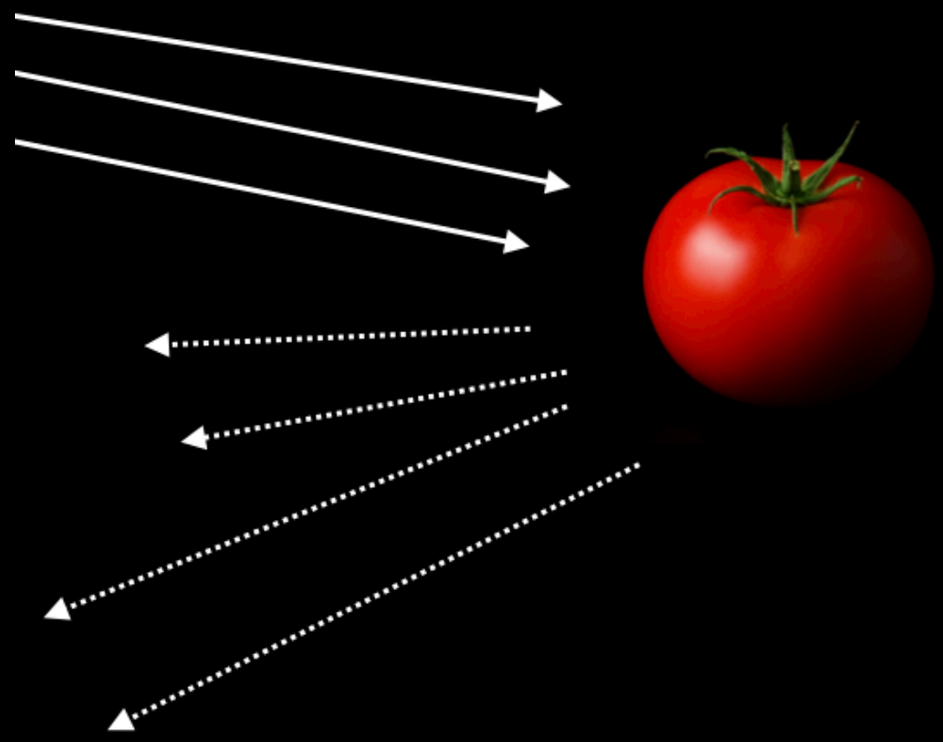


Identical Input to the Human and Camera

Different wavelengths of light are reflected by the red tomato.

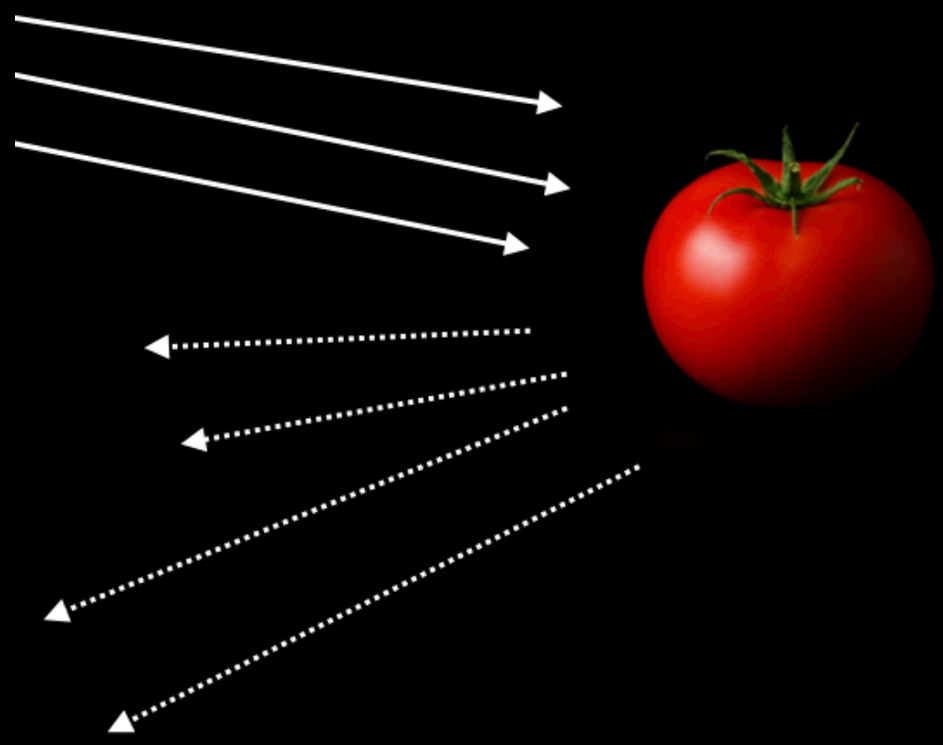
This is illustrated by the dotted white arrows.

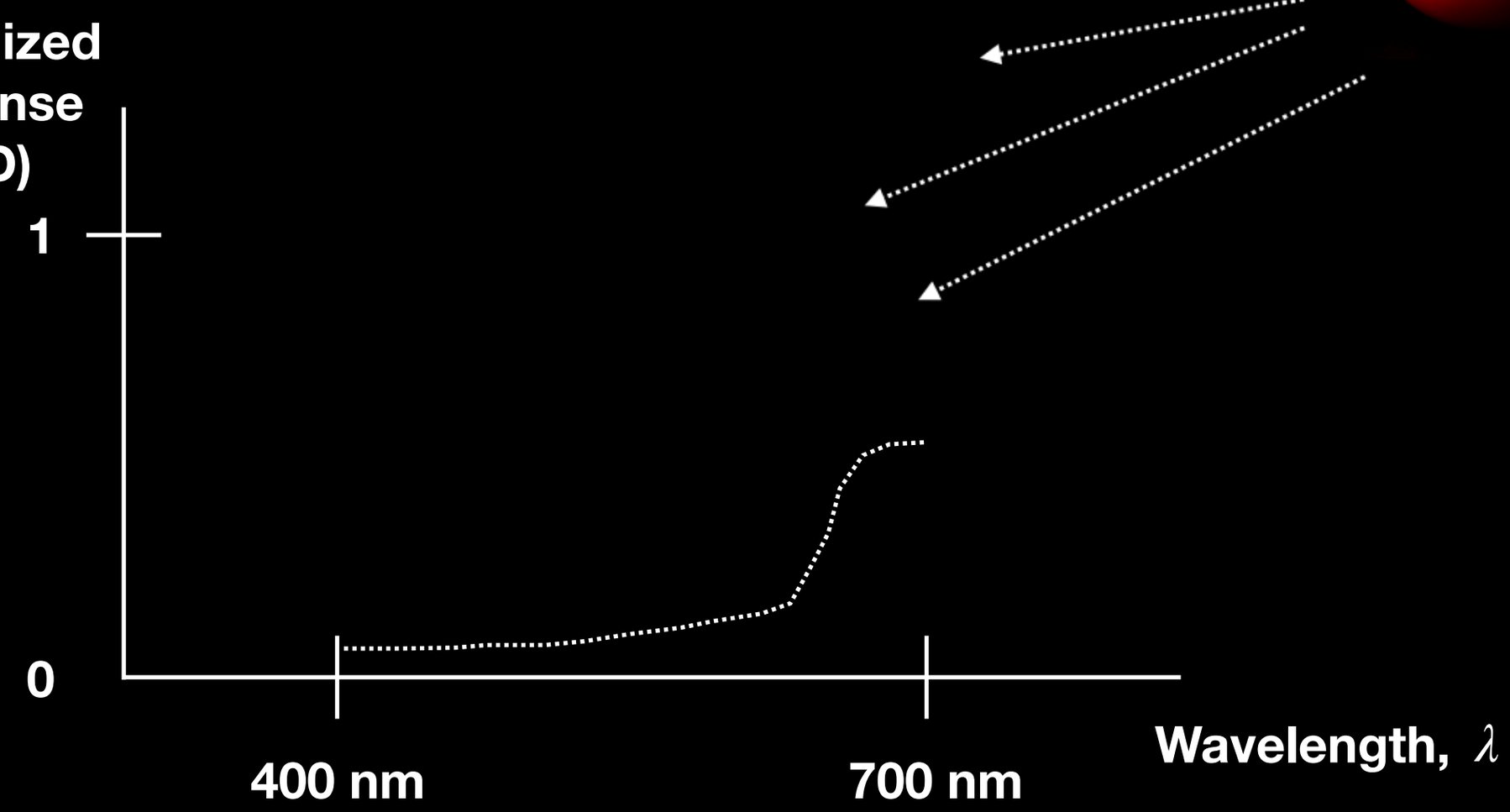
If we let the illumination be constant over all wavelengths, we obtain the spectral power distribution curve shown on the next slide.



Identical Input to the Human and Camera

The spectral power distribution (SPD) curve quantifies the amount of reflected light at different wavelengths.





Filtering

- Human subjects perceive color, by filtering the SPD using biological cones (S, M, L).

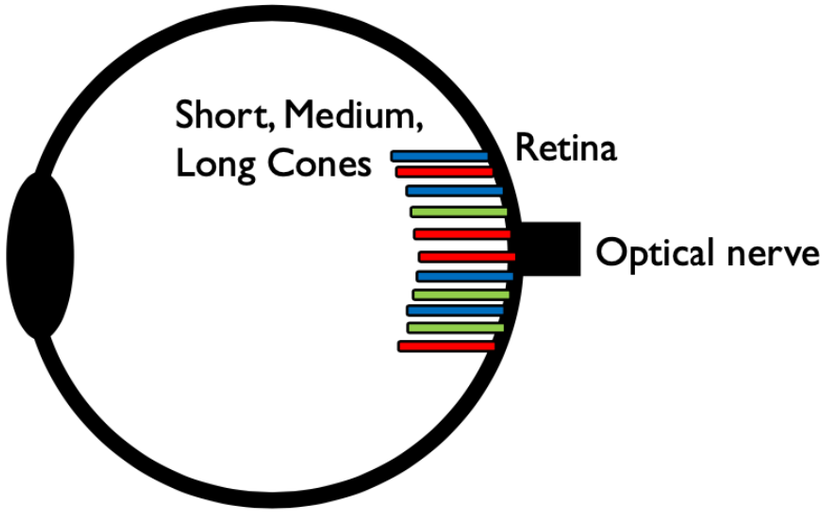
- Cameras capture color, by filtering the SPD using a color filter array (R, G, B).

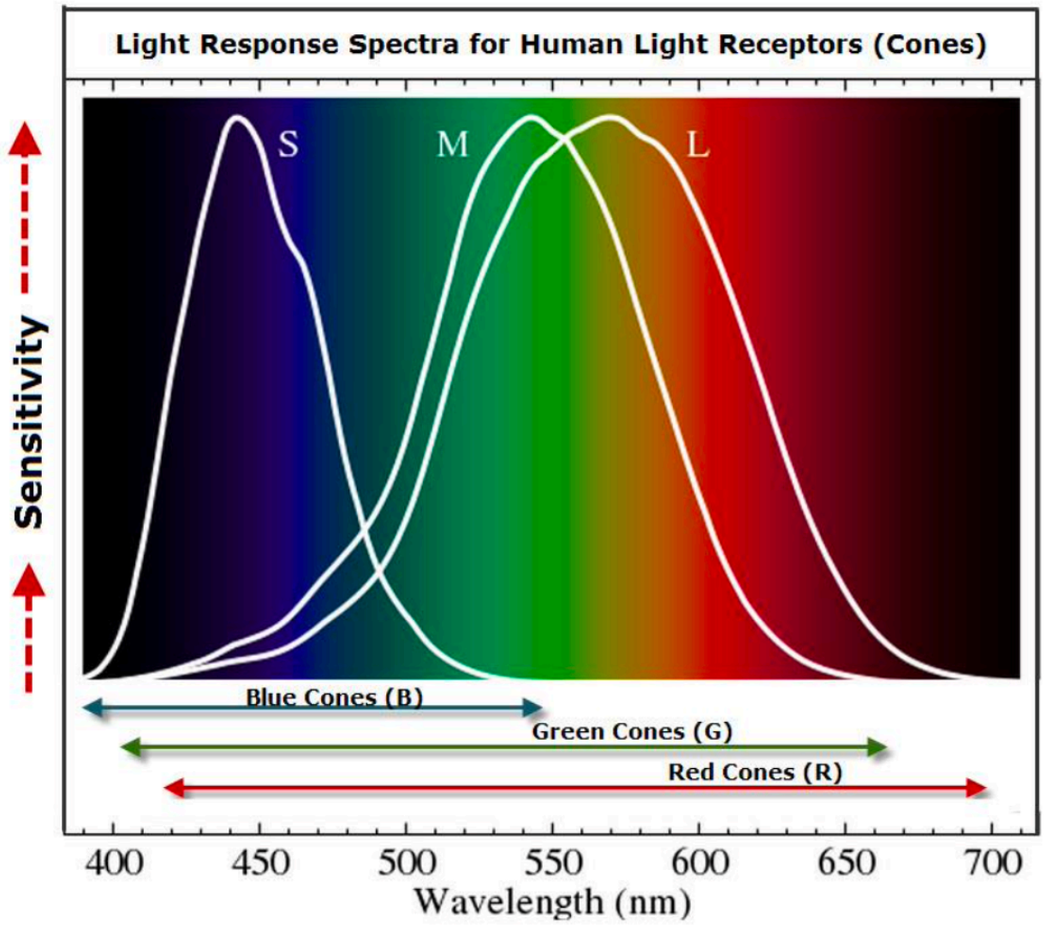
- The CFA is designed to mimic / approximate the functionality of the biological cones.

Biological Capture of Color

Biology of color sensations

- Our eye has three receptors (cone cells) that respond to visible light and give the sensation of color



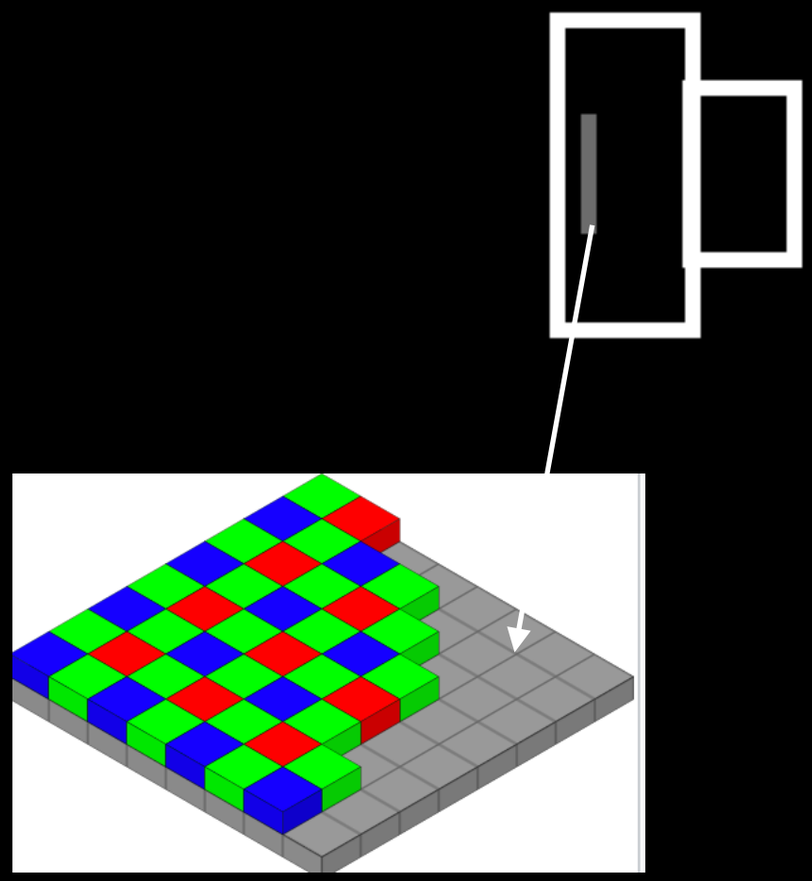


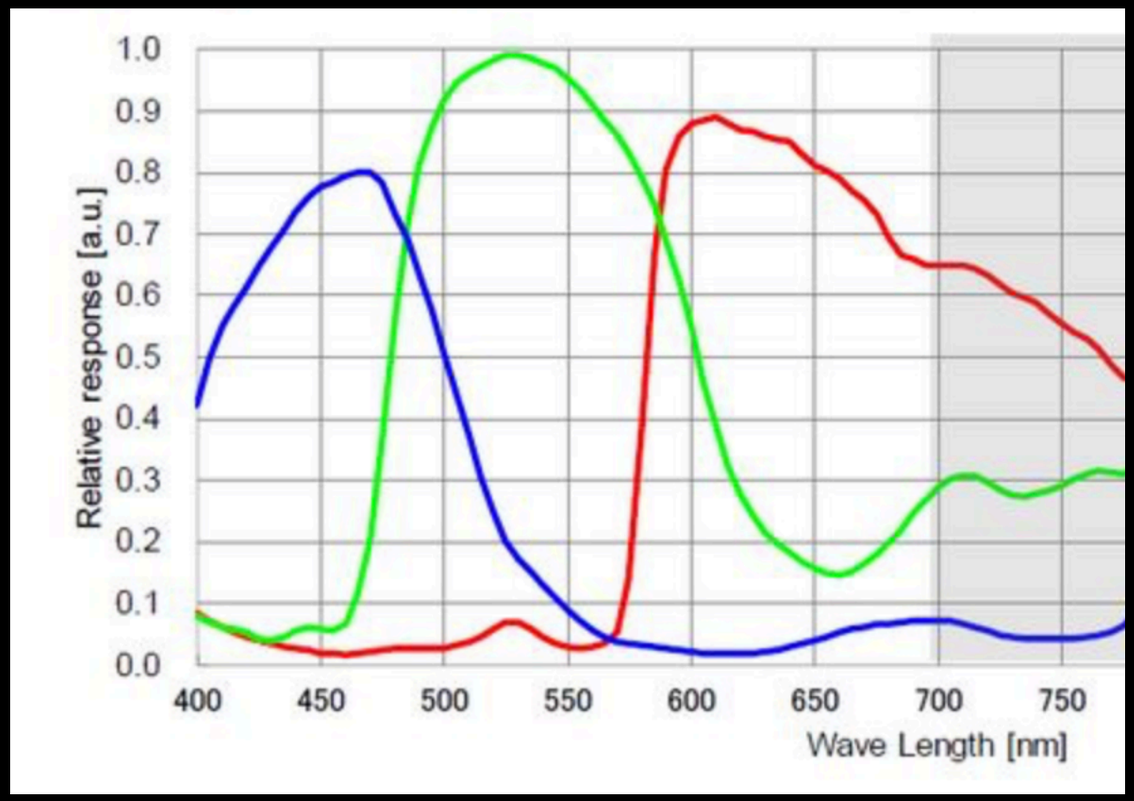
Slide credit: Michael Brown

Physical Device Capture of Color

Camera sensors have 3 color filters - R, G, B.

The R, G, B filter curves for a Sony IMX 533 sensor are shown below.





“Approximate” Relationship

- Mathematically, the “end result” is:

$$C_c = \int_\lambda S_c(\lambda) L(\lambda) R(\lambda) d\lambda \qquad\qquad c \in \{R, G, B\}$$

- Illumination: $L(\lambda)$
- Reflectance: $R(\lambda)$
- Biological (S, M, L) Cone / Sensor (R, G, B) Filter Response: $S_c(\lambda)$

- Previously: $SPD(\lambda) = L(\lambda)R(\lambda)$ - assume $L(\lambda) = 1$

But, Recall …

- ***Humans perceive a red tomato as being red, under a wide range of illuminants.

- Cameras will produce images with different color casts under the same wide range of illuminants.

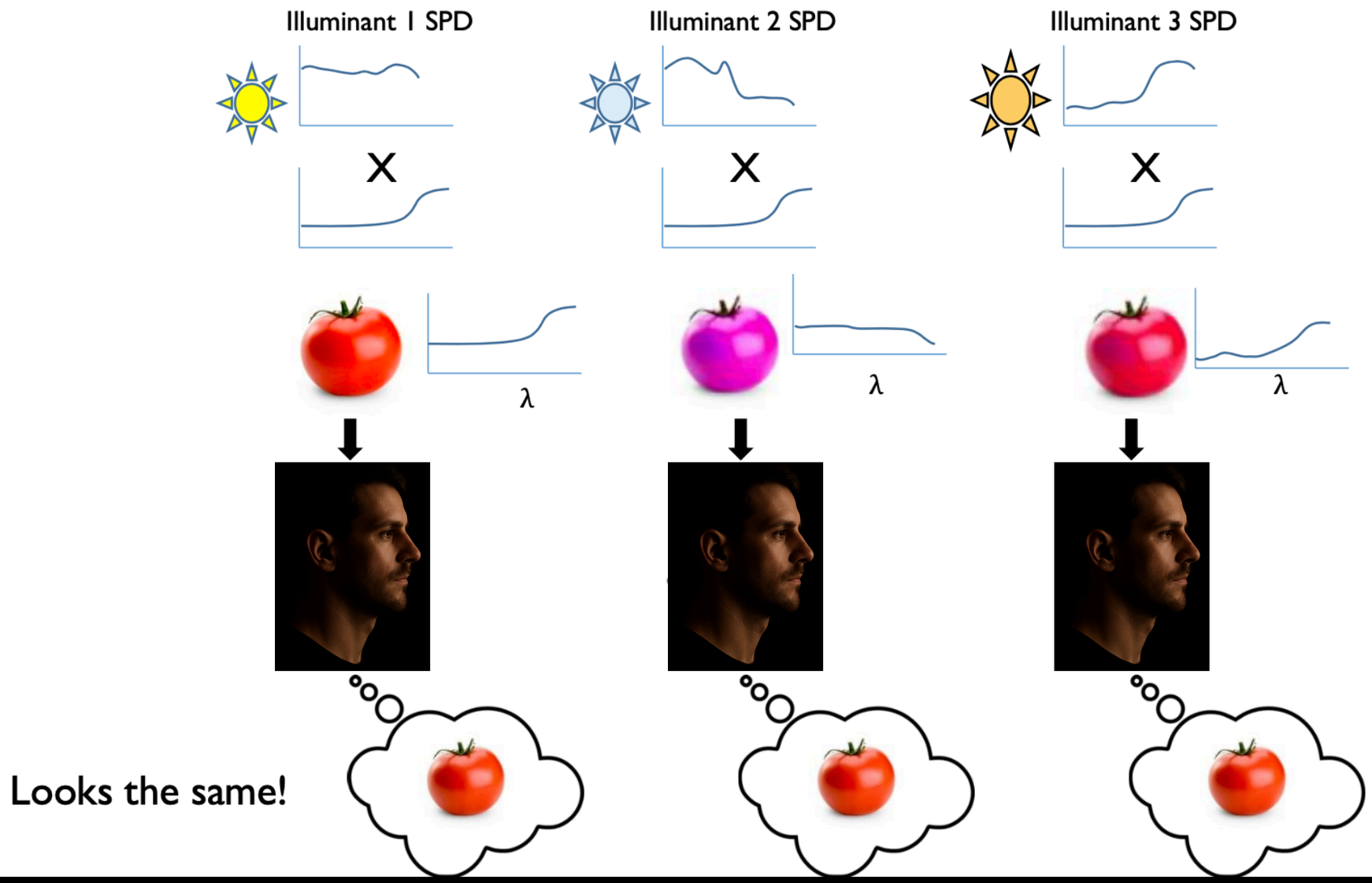
- The previous equation provides a good mathematical model for how a camera captures an image.

- However, it may be an oversimplification for humans.

- “There is reason to believe that the sensitivity of human receptors varies with brightness and duration of the stimulus.” $S_{c\_human} = S_c(\lambda, L, t, etc.)$ [Chromatic Adaptation, Bartelson 1977, High Level Trichromatic Color Matching and the Pigment Bleaching Hypothesis, Wyszecki and Stiles 1980]

Color constancy

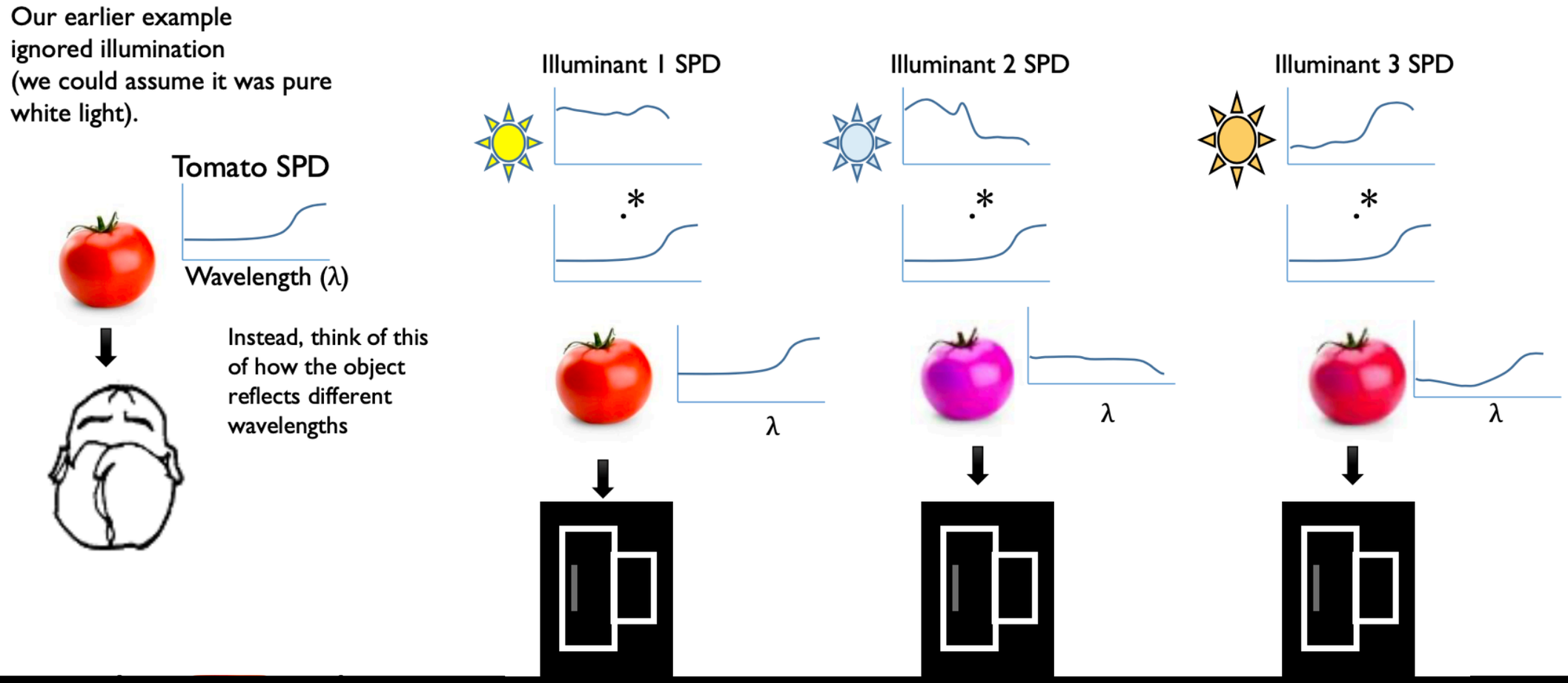
- Our visual system is able to compensate for the illumination



Slide credit: Michael Brown

An object's SPD

- In a real scene, an object's SPD is a combination of the its reflectance properties **and** scene illumination



Slide credit: Michael Brown

***Corner Cases

- In practice, if the illuminant spectrum is very narrow and the reflectance of the object is constant - $R(\lambda) = 1$ - human perception may not perceive the correct color.

- Example: Sodium vapor lamps (illumination spectrum ~570-610 nm) and a white car ($R(\lambda) = 1$).

- Problem: Your white car now looks “yellowish” at night.

Goal

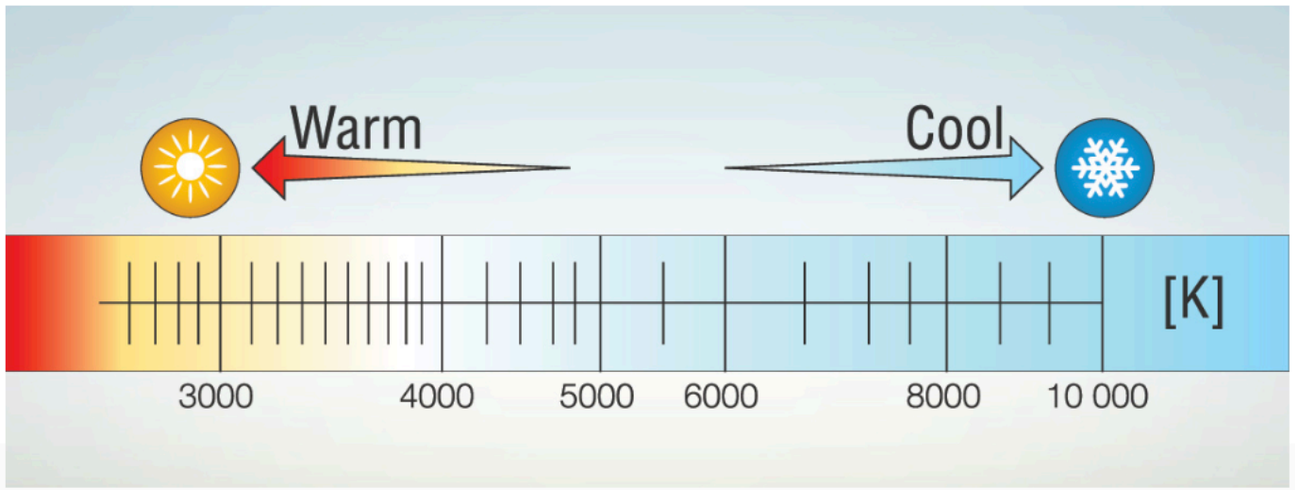
- For any scene captured by a camera under any type of lighting, remove the color cast in the resulting image.

Color Cast

Here's a brief rundown of some of the most common lighting situations you might encounter and what the corresponding Kelvin number is:

- Candlelight: 1900
- Incandescent light: 2700
- Sunrise/golden hour: 2800 to 3000
- Halogen lamps: 3000
- Moonlight: 4100
- White LEDs: 4500
- Mid-day: 5000 to 5500
- Flash: 5500
- Overcast/cloudy: 6500 to 7500
- Shade: 8000
- Heavy cloud cover: 9000 to 10000
- Blue sky: 10000

Note that the cooler the light, the higher the number. The warmer the light, the lower the number.



The Consumer Shortcut

The Approaches to The Problem

- Color constancy ⟷ **vision science**
- White balancing ⟷ **engineering**
- By construction, there is natural overlap between the two.

The Difference

- Vision science: Understand human perception and how it achieves color constancy / less focused on practical aspects.

- Engineering: Implement as algorithm to color correct an image, drawing on knowledge from vision science.

A Sampling of Algorithms

- GreyWorld
- White Patch / maxRGB
- GreyEdge
- Corrected Moments
- Gamut Matching
- Learning Based Estimation
- Convolutional Color Constancy / FFCC

[1] Greyworld Algorithm

- The average reflectance of a scene is constant / the scene has high color diversity.

- Compute the following quantities for an image: $R_{avg}, G_{avg}, B_{avg}$

- Apply the following scale corrections to the R and B pixels:

$$R_{new} = \frac{G_{avg}}{R_{avg}} R_{old} \qquad G_{new} = G_{old} \qquad B_{new} = \frac{G_{avg}}{B_{avg}} B_{old}$$

[2] White Patch / maxRGB Algorithm

- There exists a surface patch that is white / bright neutral in the scene. => It is the brightest (non clipped) surface patch.

- Compute the following quantities for an image: $R_{max}, G_{max}, B_{max}$

- Apply the following scale corrections to the R and B pixels:

$$R_{new} = \frac{G_{max}}{R_{max}} R_{old} \qquad G_{new} = G_{old} \qquad B_{new} = \frac{G_{max}}{B_{max}} B_{old}$$

[3] GreyEdge Algorithm

- The average edge color is grey = greyworld, but for edges.

- Compute the derivatives for the R, G and B color channels in an image: $\nabla R_{channel}, \nabla G_{channel}, \nabla B_{channel}$

- Then, compute the average of the derivatives, for the respective color channels: $(\nabla R_{channel})_{avg}, (\nabla G_{channel})_{avg}, (\nabla B_{channel})_{avg}$

- Apply the following scale corrections:

$$R_{new} = \frac{(\nabla R_{channel})_{avg}}{K_r} R_{old} \qquad G_{new} = \frac{(\nabla G_{channel})_{avg}}{K_g} G_{old} \qquad B_{new} = \frac{(\nabla B_{channel})_{avg}}{K_b} B_{old}$$

$$K_c = \frac{(\nabla C_{channel})_{avg}}{(\nabla R_{channel})_{avg} + (\nabla G_{channel})_{avg} + (\nabla B_{channel})_{avg}} \qquad C \in R, G, B$$

[4] Corrected Moment Algorithm

- Greyworld, WhitePatch / magRGB, GreyEdge are all moment based algorithms.

- The Corrected Moment algorithm seeks to improve these algorithms, by reducing their systematic bias.

- Systematic bias results from using a single illuminant with a single scene and a camera with overlapping spectral sensitivity curves.

- Corrected Moments accomplishes this by aggregating the information contained in a rich set of training example pairs (different illuminants, different scenes) into a correction matrix.

- The correction matrix then maps the appropriate correction (for an arbitrary illuminant and scene).

[4] Corrected Moment Algorithm

- Consider a set of M illuminants, applied to N different scenes.
- From this, you have M*N output images.
- Put the illuminants in a matrix L, and put the computed moments from the output images in a matrix P.
- Estimate the correction matrix C using pseudo least squares: $C = LP^T(PP^t)^{-1} \qquad dim(L) = 3xM, \quad dim(P) = 3x(M * N)$

[4] Corrected Moment Algorithm

- Compute the moment vector for your unknown illuminant, for the given scene - $m_{original}$

- Let $m_{corrected} = C * m_{original}, \quad dim(C) = 3x3, \quad dim(m_{original}) = 3x1$

- Now, compute the usual correction ratios using $m_{corrected}$ instead of $m_{original}$

[4] Corrected Moment Algorithm - Extension

- Now, instead of having 3 moment values (from the 3 color channels) associated with every illuminant and scene pair, 57 moment values are used.

- i.e. 19 edge moments.

- Now, a C matrix is learned, mapping the 57 moments associated with an unknown illuminant and scene pair, into a 3x1 vector.

[4] Corrected Moment Algorithm

- In addition, an iterative algorithm is used, to further improve the estimated matrix C.

1. initialise: $D^0 = \mathcal{I}$ (the identity matrix) and $i = 1$
2. $C^i = [D^{i-1}P]^+ L$ (where $^+$ denotes the 'pre multiplying' pseudo inverse $A^+ = [A^t A]^{-1} A^t$)
3. $D^i_{jj} = L_j [P_j C^i]^\dagger$ (here we find the best scalar using the 'post-multiplying' pseudo inverse $[\underline{v}^t]^\dagger = \frac{\underline{v}}{\underline{v}^t \underline{v}}$)
4. i = i+1, goto 2 until convergence

[5] Gamut Matching Algorithm

- Start with a reference illuminant (say, white light) and a large number of widely varying, surface reflectance patches (say, 160 mussels color chips).

- Compute and create the reference gamut C, in RGB space.

- Now, for the scene with unknown illumination, compute and create the observed gamut D.

- Determine the appropriate mapping from the observed gamut D, to the reference gamut C.

Notes

- The mapping will be more accurate, if the scene contains a diverse variety of surface reflectances (vs. several surface reflectances).

- The mapping from D to C is not unique. (i.e. Many mappings can exist.)

- The “best” mapping is selected from the list of plausible mappings.

[5] Gamut Matching Algorithm

- From the author's original paper:

1. Construct the canonical gamut, by observing as many surfaces as possible under a single given light. This light will be the canonical illuminant, and surface color is defined to mean the color that the surface has when observed under this light. The canonical gamut can be approximated by taking the convex hull of the union of the gamuts obtained by these observations. Call the canonical gamut C.
2. To construct the feasible set for any patch imaged under a constant illuminant:
   - Form the convex hull of the gamut observed. Call this gamut D.
   - Form the set of the L-by-L matrixes, M, of rank L such that $M(D) \subset C$. The particular minimal assumption under which the color constancy algorithm is operating determines a subset of this set, which is the feasible set.
3. Within this feasible set, use some estimator to choose that map most likely to correspond to the illuminant.

[5] Gamut Matching Algorithm

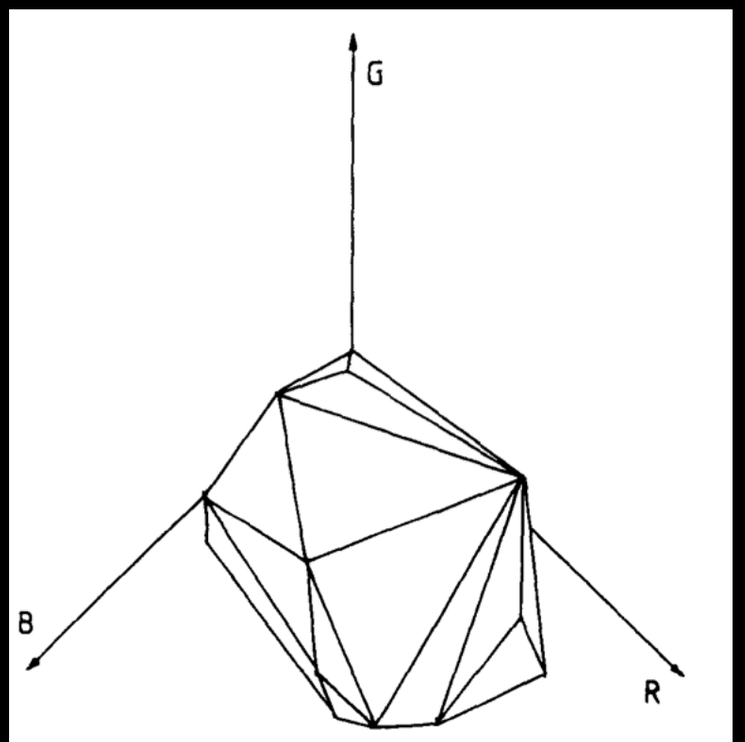


Fig. 1. The convex hull of the gamut obtained by observing 180 color chips under white light. This formed the observed canonical gamut for the experimental work. If $\underline{t}$ represents the green light for the gamut shown in figure 2, $\Psi(\cdot;\underline{t})$ would take this set to a superset of that gamut.

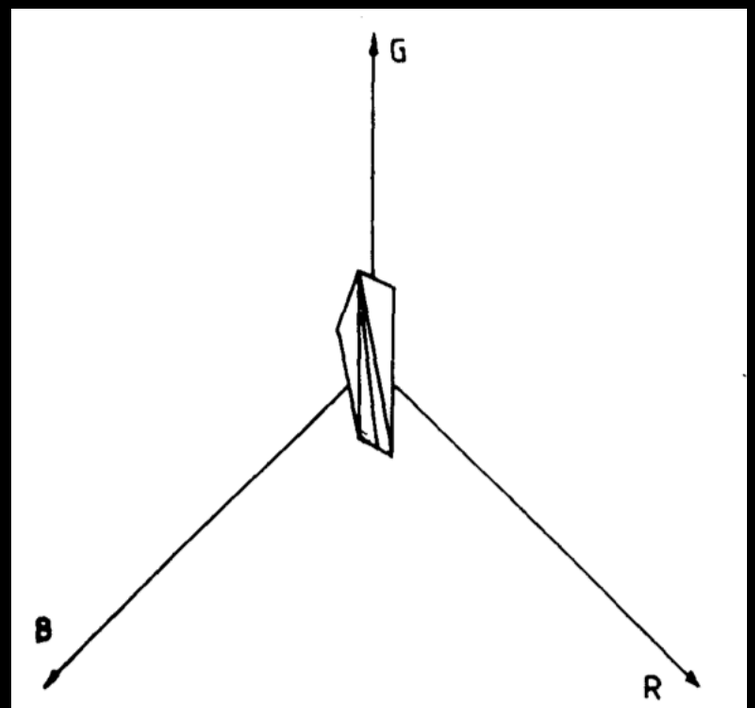


Fig. 2. The convex hull of the gamut observed for a Mondriaan image under green light. Notice that there are significant differences between this gamut, and that of figures 1 and 3. The feasible set consists of maps taking this gamut to a subset of the gamut of figure 1.

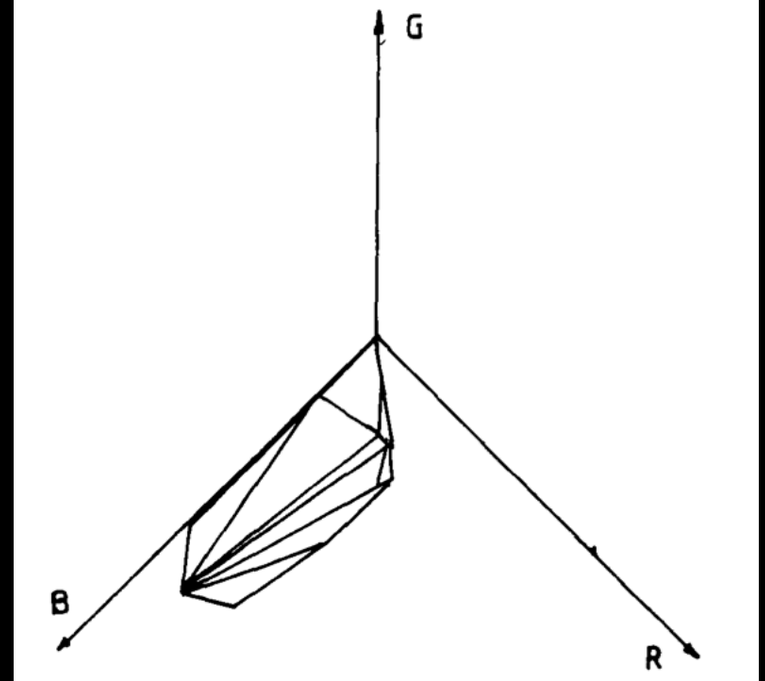


Fig. 3. The convex hull of the gamut observed for a Mondriaan image under blue light. Notice that there are significant differences between this gamut, and those of figures 1 and 2. **Crule** uses this skewing effect to infer possible illuminants.

Canonical Gamut

Resulting Gamut from Green Light

Resulting Gamut from Blue Light

[6] Learning Based Estimation

- A learning approach is employed, using regression trees.

4 hand crafted features are selected / designed for:

- 1) (f1) Average color - like GreyWorld
- 2) (f2) Max pixel R, G, B - $max(R(x,y) + G(x,y) + B(x,y))$
- 3) (f3) Dominant color - histogram R, G, B. Find the bin with the largest peak and average the pixels in that bin.
- 4) (f4) Color palette - largest mode in 2D xy space, based on (f3) histogram results for all bins greater than a tuned threshold.

[6] Learning Based Estimation

- The ground truth illuminants for the test set are mapped to rg space:

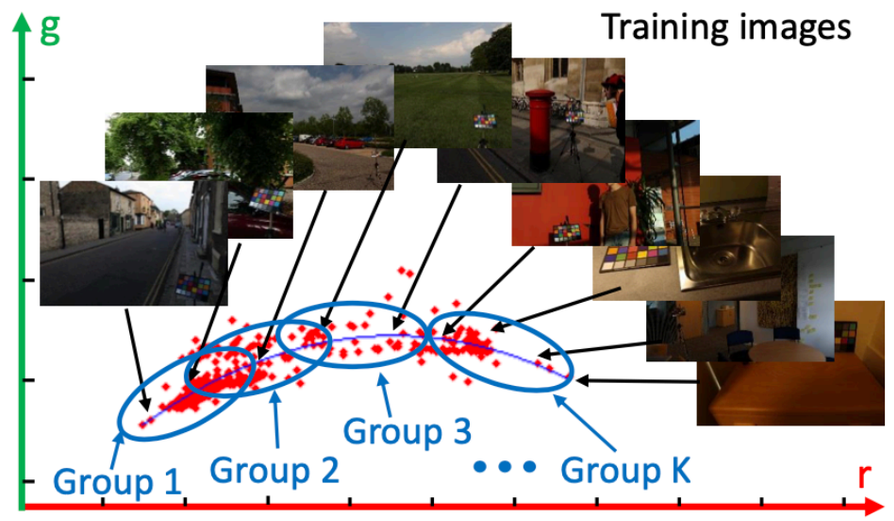


Figure 4: Illustration of sorting the training images and separate them into groups. The red dots indicate the ground truth illuminant rg-chromaticities from Gehler-Shi data set [24, 35] and the curve shows a quadratic fitting to these illuminant chromaticities. A number of training example images from the data set having different illuminant from blueish to reddish are also shown. The whole training data set is separated into K local overlapping groups.

$$r = \frac{R}{R+G+B}$$

$$g = \frac{G}{R+G+B}$$

- And are grouped into K = 30 overlapping groups.

[6] Learning Based Estimation

- 8K regression trees are then trained for the 4 features, as separate r and g regression trees are trained for each feature.

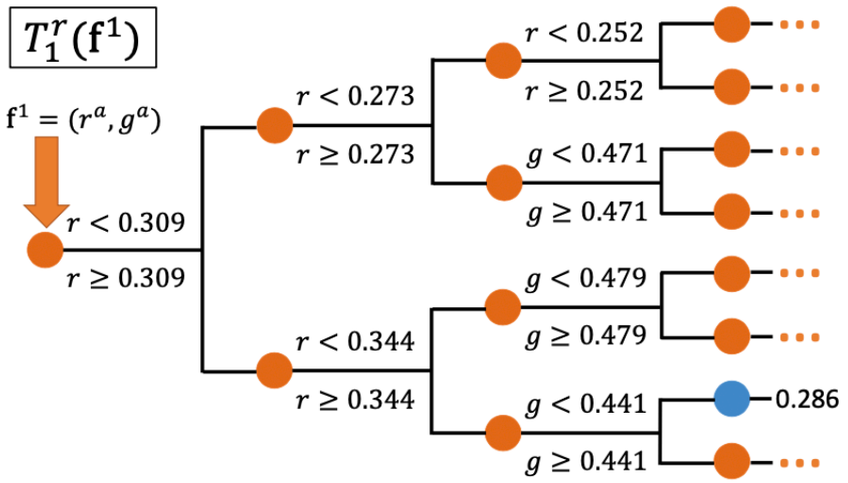


Figure 3: An example of the regression tree $T_1^r(\mathbf{f}^1)$ (the r-chromaticity illuminant prediction on feature 1 with the training subset 1). The orange dots denote non-leaf nodes where a decision is made according to the split rule. The blue dot denotes a leaf node where the final regression value is determined. This figure shows only the first four layers (out of 17) which already contain a leaf (end) node.

Error in author's figure.

All decision branches should be labeled r.

[6] Learning Based Estimation

- The author does not use random forest bagging to estimate the final illuminant though.

- Instead, using the same input image, the r and g outputs are computed using the 4 features.

- The 4 results are then plotted on the rg plot.

- If more than 3 of the 4 results are less than .025 euclidean distance from each other, the outputs from all of the results are appended to a $\tilde{R}$ set and a $\tilde{G}$ set.

- The rg chromaticity estimate is the computed from $median(\tilde{R}, \tilde{G})$

CART Trees - Breiman

1️⃣ At each node in the tree:

Suppose your input has **two features** x_1, x_2 and output y.

2️⃣ Step A: Candidate splits per feature

For each feature x_j:

1. Sort the training samples by x_j.
2. Try all possible thresholds t along x_j (usually midpoints between consecutive feature values).
3. For each threshold:
   - Left child = samples with $x_j \leq t$
   - Right child = samples with $x_j > t$
   - Compute **total SSE** of y in both children:

$$\text{SSE}_{\text{total}} = \sum_{i \in L} (y_i - \bar{y}_L)^2 + \sum_{i \in R} (y_i - \bar{y}_R)^2$$

4. Keep the threshold t_j^* for this feature that **minimizes total SSE**.

3️⃣ Step B: Select the best feature to split

- After doing this for all features $x_1, x_2, \ldots, x_d$, you have the **best threshold for each feature**.
- Compare the **minimum SSE across all features**.
- Choose the feature x_{j^*} whose split gives the **overall smallest SSE**.
- Split the node on $x_{j^*} \leq t_{j^*}^*$.

4️⃣ Step C: Recurse

[7] Convolutional Color Constancy / FFCC

- Like the previous algorithm, Convolutional Color Constancy operates in the chroma space.

Advantage:

- Intensity is removed from the picture (=good, since hue / color - the quantity of interest - is not a function of intensity)

- Instead of dealing with 3 parameters - R, G and B, you are now only dealing with 2 chroma parameters.

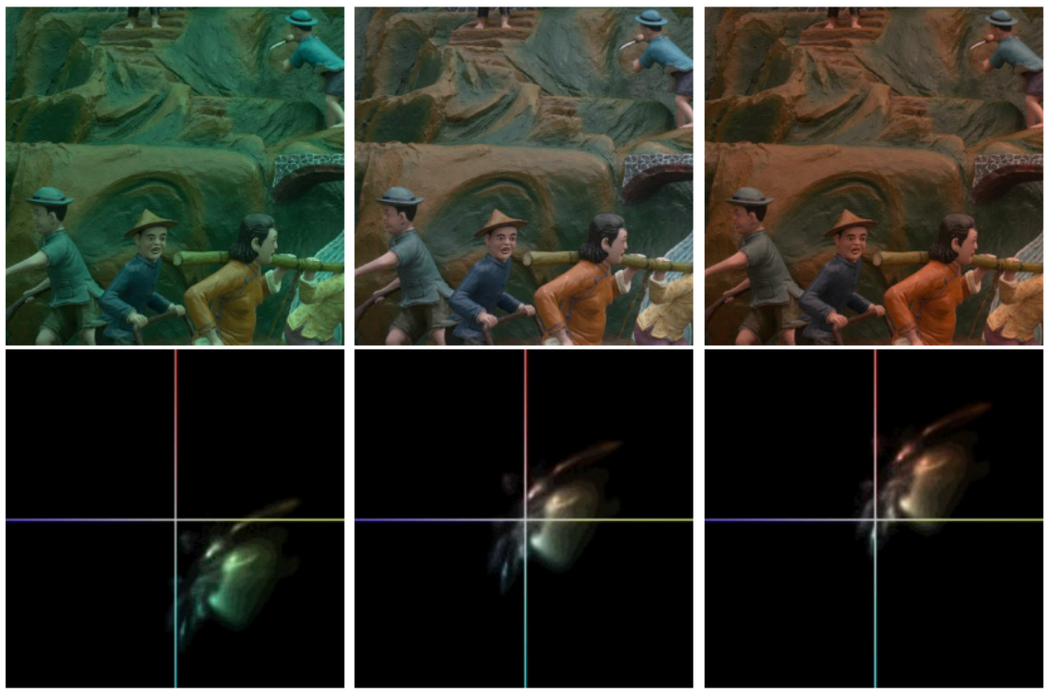
[7] Convolutional Color Constancy / FFCC

- In addition, Convolutional Color Constancy operates in the log( ) chroma space (log(u), log(v)).

Advantage:

- log(ab) = log(a) + log(b)
- As a result, changes due to a color cast will now result in shifts versus of multiplicative scaling.

[7] Convolutional Color Constancy / FFCC



(a) Input Image (b) True Image (c) Tinted Image

Figure 2: Some images and their log-chrominance histograms (with an axis overlayed for easier visualization, horizontal = u, vertical = v). The images are the same except for "tints" — scaling of red and blue. Tinting an image affects the image's histogram only by a translation in log-chrominance space. This observation enables our convolutional approach to color correction, in which our algorithm learns to localize a histogram in this 2D space.

We form a 2D histogram in chroma space, from the RGB image.

The non color cast image has a centroid located at approximately the origin in chroma space.

By determining the shift for the color cast image, we have the needed correction.

Ideally, some form of template matching would be best.

However, the True Image is what we are trying to recover = unknown.

[7] Convolutional Color Constancy / FFCC

- The author proposes to learn a convolutional filter F, to improve the color cast estimate.

- The author proposes to learn this convolutional filter R, in an optimization framework:

framework. Formally, our optimization problem is:

$$\min_F \lambda \sum_{u,v} F(u,v)^2 + \sum_{i,u,v} P(u,v)\, C\Big(u, v, L_u^{(i)}, L_v^{(i)}\Big)$$

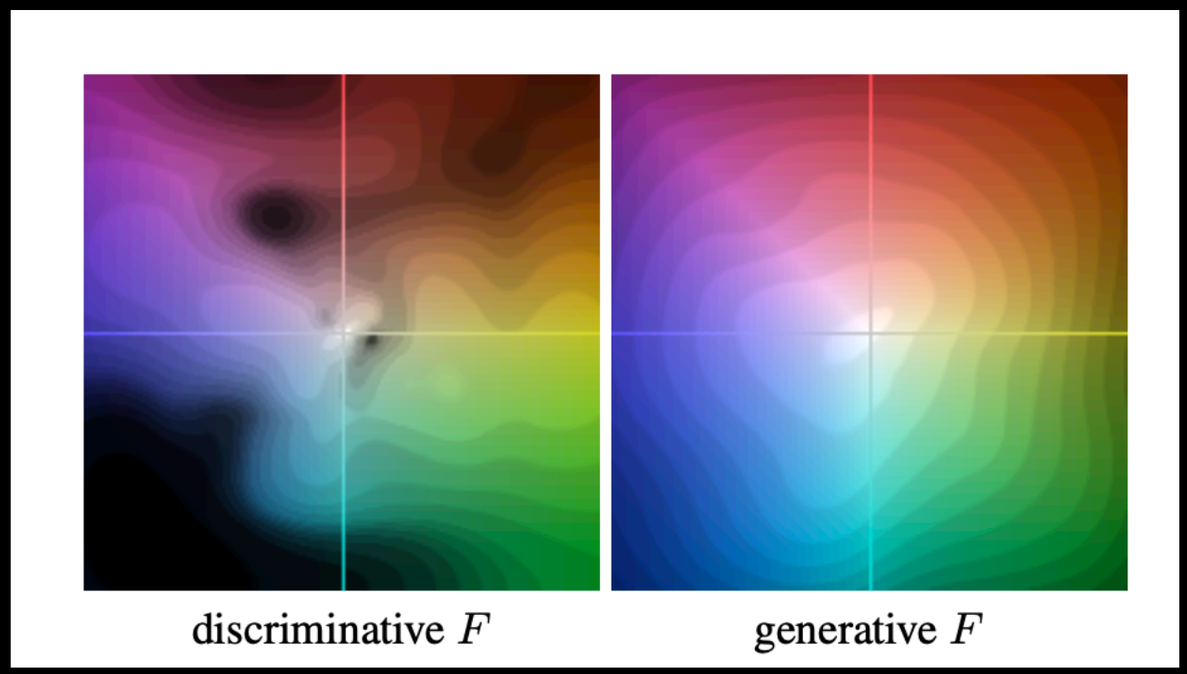
$$P(u,v) = \frac{\exp\big((N^{(i)} * F)(u,v)\big)}{\sum_{u',v'} \exp\big((N^{(i)} * F)(u',v')\big)} \tag{10}$$

Where F is the filter whose weights we learn, $\{N^{(i)}\}$ and $\{L^{(i)}\}$ are our training-set chrominance histograms and ground-truth illuminations, respectively, and $(N^{(i)} * F)(u,v)$ is the convolution of $N^{(i)}$ and F indexed at location (u,v). For convenience we define $P(u,v)$ which is a softmax probability for each (u,v) bin in our histogram as a function of $N^{(i)} * F$. We regularize our filter weights by

- Two types of learned filters are possible: generative and discriminative.

[7] Convolutional Color Constancy / FFCC

- The generative framework results in “smoothing” filters.
- The discriminative framework results in filters that emphasize and de-emphasize various regions of transformed space.
- As a result, the discriminative filter is more powerful.



[7] Convolutional Color Constancy / FFCC

- The key difference, is the introduction of the following term into the optimization problem:

$$C\left(u, v, u^*, v^*\right) = \arccos\left(\frac{\langle \ell, \ell^* \rangle}{\|\ell\| \, \|\ell^*\|}\right)$$

$$\ell = [\exp(-u),\ 1,\ \exp(-v)]^{\mathrm{T}}$$

$$\ell^* = [\exp(-u^*), 1, \exp(-v^*)]^{\mathrm{T}}$$

- Essentially, this term provides valuable information, regarding the illumination estimation error, associated with the existing filter.

- i.e. The filter now learns with discrimination / with weighting, since it now possesses knowledge of how good or bad the illuminant estimate is for the current input image and the evolving learned filter.

[7] Convolutional Color Constancy / FFCC

- The author proposes that the image is also preprocessed with several different filters:

$$
\begin{aligned}
I'_1 &= I \\
I'_2 &= \max\left(0, I * \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}\right) \\
I'_3 &= \text{blur}(I^4, 11)^{1/4} \\
I'_4 &= \sqrt{\text{blur}(I^2, 3) - \text{blur}(I, 3)^2}
\end{aligned}
$$

- Hence, 3 new convolutional filters are learned using these images.

- The results from the convolutional filters are combined, yielding the third picture on the next slide, leading to the subsequent correction.

[7] Convolutional Color Constancy / FFCC

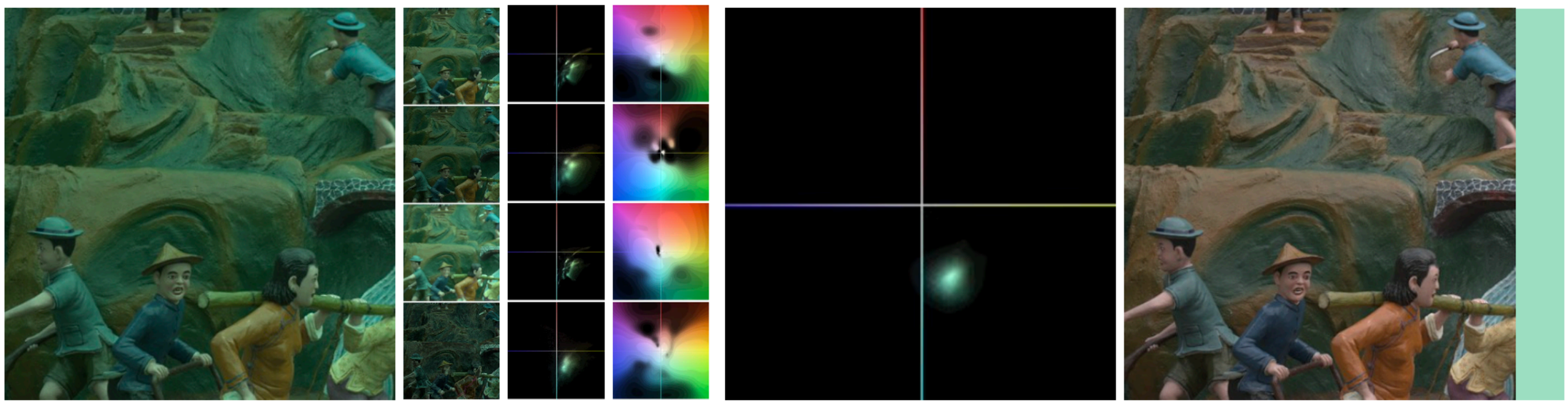


Figure 7: An overview of inference in our model for a single image. An input image I is transformed into a set of scale-preserving augmented images $\{I'_j\}$ which highlight different aspects of the image (edges, patches, etc). The set of augmented images is turned into a set of chroma histograms $\{N_j\}$, for which we have learned a set of weights in the form of pyramid filters $\{F_j\}$. The histograms are convolved with the filters and then summed, giving us a score for all bins in our chroma histogram. The highest-scoring bin is assumed to be the color of the illuminant $\hat{L}$, and the output image $\hat{W}$ is produced by dividing the input image by that illuminant.

[7] Convolutional Color Constancy / FFCC

- As the procedure just described can be “wall clock” heavy, Fast Fourier Color Constancy (FFCC) introduces a mapping to speed up the performance.

- As I do not fully understand this mapping (yet), I will not try to explain it.

Von Kreis Model

- The Von Kreis Model finds widespread use in color constancy.
- It assumes that when the illumination changes from L_1 *to* L_2 the resulting sensor signals are related by simple per channel scaling.

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} k_R & 0 & 0 \\ 0 & k_G & 0 \\ 0 & 0 & k_B \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

- For example, the diagonal quantities represent the correction factors for GreyWorld, White Patch / magRGB, GreyEdge, etc.

Von Kreis Model

In practice this model begins to deteriorate:

- If the spectral sensitivity curves are wide and overlap significantly (= channel crosstalk)
- If there are sharp spikes in the illuminant
- If the surfaces are highly chromatic / reflectance surface is narrowband
- To address channel crosstalk / non-channel independent gain, a 3x3 correction matrix is also used.

Color Correction Matrix (CCM)

- Going through the linear pipeline, the raw RGB input is first rescaled via the Von Kries model to the remove color cast.

- Then, a 3x3 CCM is applied, to correct color errors introduced by the overlapping spectral sensitivity curves (i.e. crosstalk).

- In addition, the 3x3 CCM simultaneously transforms the signal into the appropriate color space, like sRGB.

- To perform the color error correction, calibration points are captured for two different illuminants - A and D65, resulting in two CCM matrices.

- The CCM matrices for all other illuminants are then estimated from these calibration points.

Main Test Sets Used for Color Evaluation

- Color Checker Dataset - 568 images, reprocessed to be linear, raw format

- NUS 8 - 1736 images from 8 commercial cameras, linear, raw format

- SFU Laboratory - 321 images, (31 objects, 11 light sources), linear

Algorithm References

- 1) A Spatial Processor Model for Object Color Perception, G. Buchsbaum, Journal of the Franklin Institute, 1980
- 2) Analysis of the Retinex Theory of Color Vision, Brained and Wandell, JOSA A, 1986
- 3) Edge Based Color Constancy, Van Der Weijer, et. al., Transactions on Image Processing, 2007
- 4) Corrected Moment Illuminant Estimation, G.D. Finlayson, ICCV 2013
- 5) A Novel Algorithm for Color Constancy, D. Forsyth, Int. Journal of Computer Vision, 1990
- 6) Effective Learning Based Illuminant Estimation Using Simple Features, Cheng, et. al., CVPR 2015
- 7) Convolutional Color Constancy, J. Barron, ICCV, 2015
- 8) Fast Fourier Color Constancy, J. Barron and Y.T. Tsai, ICCV 2017