LOCAL LINEAR LEARNED METHOD FOR IMAGE AND REFLECTANCE ESTIMATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Steven Paul Lansel
June 2011

Abstract

The recent increase in the number of megapixels in digital cameras has resulted in images that have a higher spatial resolution than is required by most imaging applications. This excess of pixels offers an opportunity to redesign imaging sensors to achieve improvements in other aspects of photography instead of spatial resolution. Improved low light sensitivity, expanded dynamic range, and improved color estimation are possible by altering the sensor's color filter array (CFA). The CFA is a mosaic of optical filters overlaying each photosensitive site on the sensor that controls the spectral sensitivity of the pixel. A few novel CFA designs are proposed to achieve the improvements stated above.

In addition to the above potential improvements to images for human perception, novel CFAs can enable multispectral imaging where the full spectrum of incoming light is estimated. There is a rich field of multispectral applications including computer vision, remote sensing, and medicine. Unfortunately expensive equipment is typically required to acquire multispectral measurements, which could be eliminated using new CFAs with a small number of color filters.

For cameras with new CFAs, there is a great challenge in designing the image processing pipeline, the series of calculations that transforms the raw output from the sensor into a desirable image. In general, one needs to perform demosaicking to estimate a full color image from the sensor's image where only a single band is measured at each pixel. Also denoising is required to suppress photon shot noise and noise from imperfections in the sensor electronics. Finally, a transformation is needed from the input color space defined by the sensor's spectral sensitivities to a desired output color space.

The Local Linear Learned (L^3) pipeline is presented that simultaneously performs demosaicking, denoising, and color conversion for an arbitrary CFA to an arbitrary output color space. Although there exist numerous image processing algorithms for camera pipelines, very few have the ability to operate on any CFA, estimate images in color spaces with any number or shape of bands, or perform any of the pipeline calculations simultaneously. The L^3 algorithm learns from a training set of images a method of clustering and adaptive linear filtering for each cluster. As a result, the algorithm can adapt to a specific dataset, CFA, and desired output color space that are appropriate for a particular imaging application. After training, the pipeline is applied to a pixel from a sensor image by identifying the appropriate cluster and applying that cluster's pre-computed filters. The algorithm enables rapid design and testing of future CFAs by automatically generating the image processing pipeline.

To address the spectral design of the CFA and potential multispectral applications, the problem of estimating a surface's reflectance is analyzed. The reflectance, which describes how much light a surface reflects at each wavelength, is estimated using a small number of colorband measurements along with knowledge of the spectrum of the light and camera sensitivities. The L^3 reflectance estimation method uses local filtering based on training reflectances that have similar color measurements to a given test point. The L^3 method and the optimal Bayesian estimator have similar performance. The localization of the estimate compared to a global linear estimator is most advantageous for constrained datasets that may appear in specific applications.

Acknowledgement

This thesis would not have been possible without the support of countless advisors, colleagues, friends, and family. I am most grateful for the constant support and patience of my advisor, Brian Wandell. He has taught me the fundamentals of vision and imaging systems and greatly contributed to this research. In addition, his suggestions have improved my speaking and writing skills, and he has provided great professional and personal advice. I am also grateful to the members of my committee, Tsachy Weissman, Marc Levoy, and David Donoho. Their questions and advice have allowed me to strengthen this work and explore possible future applications of these ideas.

I am indebted to fellow researchers Manu Parmar, Joyce Farrell, and Peter Catrysse who provided helpful feedback at all stages including significantly improving how to communicate the ideas and findings. Manu graciously explained key concepts to me when I was just starting in the field.

The many members of the VISTA Lab were friendly and helpful while providing a new perspective on my work. Pam Widrin and Sylvia Paris cheerfully assisted me with all administrative tasks, and I always enjoyed their positive attitudes and encouragement.

I am grateful for the generous funding provided by Olympus throughout my graduate study. Colleagues at Olympus were patient and enthusiastic during our collaboration. I appreciated the opportunity to meet with a number of people from Olympus who provided a valuable industrial perspective in both their visits to Stanford and my visit to their facilities in Japan. I especially want to thank Ken Ishii for his professional insight and personal friendship.

I owe my deepest gratitude to my family. From my parents, I have always received an abundance of love and support and are happy to make them proud of my accomplishments. Most importantly, I thank my wife, Arum, for her always loving support and encouragement no matter the circumstances.

Contents

List of Tables

List of Figures

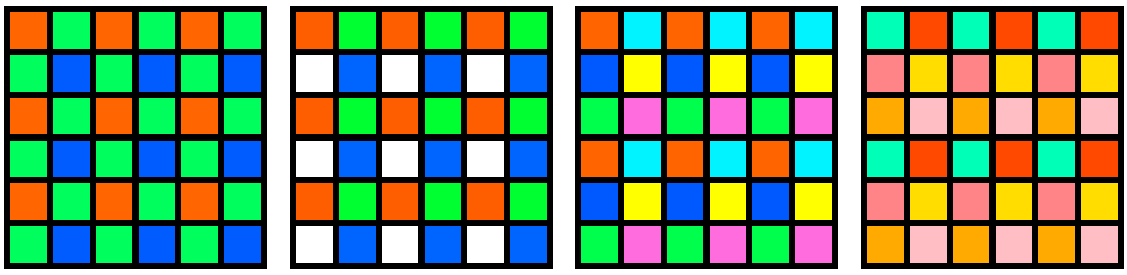
Chapter 1

Introduction

1.1 Motivation for novel sensors

The number of pixels on modern digital cameras has rapidly grown in the past few years. Cameras today typically have such high megapixel counts that they far exceed the resolution of existing displays. For example, more than 10 high-definition 720p displays are required in order to view an image from a 10 megapixel camera without downsampling. Also diffraction and aberrations from the camera's optics blur the image at the sensor, which limits the effectiveness of high spatial sampling [1]. Instead of simply increasing spatial resolution, the large number of pixels offer a possibility to improve other aspects of photographs. Possible improvements include increased sensitivity for low light photography, dynamic range expansion for scenes with bright and dark regions, and improved color accuracy.

Advances in these applications are possible by altering the color filter array (CFA) of sensors in modern cameras. The CFA is a set of optical filters in almost all digital cameras that overlay each photosensitive site on the sensor so each pixel measures only a particular color of light. The camera's sensitivity to light for each type of pixel in the CFA can be described by the camera's quantum efficiency for each channel. These functions give the probability that a photon with a given wavelength will excite an electron-hole pair in the sensor. Generated electrons in each pixel of the sensor are collected and counted to determine the amount of light at each pixel in that pixel's



(a) (b) (c) (d)

Figure 1.1: **Bayer and novel CFAs with expanded capabilities.** (a) Bayer pattern, (b) *RGBW* CFA with translucent pixel for low light, (c) six-band CFA for improved color or multispectral imaging, and (d) six-band CFA for potential medical use.

colorband.

The Bayer CFA shown in Figure 1.1(a) is in almost all cameras today [2]. Figure 1.1(b) shows a CFA where half of the green pixels from the Bayer CFA are replaced with white pixels [3]. The white pixels have transparent filters so their sensitivity is given by the sensor's silicon. This increased sensitivity enables very low light photography. Similar CFAs with pixels with increased sensitivity such as white pixels have been considered in other publications [4, 5] and a number of patents [6, 7, 8, 9, 10, 11].

The additional pixels could be used not just to improve photographs for human viewers, but to measure spectral features that cannot be directly observed by the human visual system. Human vision is enabled by the retina, a thin layer at the back of the eye that contains photoreceptors. These are millions of cells called cones that react to the incoming light. There are three types of cones each with a different sensitivity to light, which can be described as spectral curves over the visible range. Since there are only three types of cones in the human retina, human vision has a limited ability to differentiate between objects with similar spectra.

There are a number of optical imaging devices such as application specific cameras, microscopes, and endoscopes that have similar technologies to consumer cameras but are interested in light beyond its perception by people. Multispectral or hyperspectral imaging deals with trying to observe spectrums from a scene, not simply the three

color components that can be observed by the human visual system. The spectrums can also include wavelengths outside the visible range of 400-700 nm such as near infrared wavelengths of 700-1100 nm. The additional information in the spectrum may greatly simplify the task of classifying or detecting objects that have similar color appearances to human observers.

Multispectral applications include remote sensing [12], computer vision [13], displays and printers [14], art [15, 16], food safety and inspection [17, 18, 19], and medical applications including microscopy [20], fluorescence imaging [21], pathology [22], diagnosis [19], dermatology [23], and telemedicine [24].

There are a large number of specialized devices for capturing multispectral images [25]. Unfortunately all are very expensive and require a long acquisition time, which restricts measurements to stationary scenes without moving objects such as people. Typically these devices can simultaneously acquire two dimensions and sequentially scan the third dimension needed for multispectral imaging.

A camera with a CFA featuring a number of colorbands such as in Figure 1.1(c), offers the possibility of cheaply estimating multispectral scenes using a single acquisition that can occur very quickly. In addition to the typical red, green, and blue pixels, the CFA has cyan, magenta, and yellow pixels. The six-band sensor can be used in a multispectral camera to estimate the amount of light for any wavelength in each pixel. Development of such a multispectral camera should not cost much more than a regular consumer digital camera but could be used in many specialized applications.

Observing six colorbands allows improved color estimation over typical sensors with three bands. Since the cyan, magenta, and yellow filters transmit more light than the red, green, and blue filters, this CFA may also have increased low light performance. Similar CFA designs have been proposed [26], and the general design of CFAs for multispectral imaging has been studied previously [1, 27, 28, 29].

The CFA in Figure 1.1(d) shows a modified six-band design that may be used specifically in medical cameras that operate inside the body for superior performance in tissue classification. The sensor is designed to more heavily measure the red part of the spectrum, since internal tissue typically is more reflective in the red region.

These CFAs are only a few of the possible novel CFAs that may soon be developed. There are a number of publications that already propose new CFA designs or design methods [1, 4, 30, 31, 32, 33, 34, 35, 36, 37]. There is great potential to exploit spectral properties in specific applications of the future by designing CFAs for particular datasets. For imaging sensors that are designed to operate only in specific environments such as in the body, a factory, or a computer vision system, the sensor and processing can be optimized for the very narrow set of scenes that the sensor will be used to capture.

1.2 Image processing pipeline challenges for novel CFAs

A major challenge with novel CFAs is the difficulty of designing the image processing pipeline, which is a series of calculations that converts the output from the sensor into a desirable image for display or storage. The following are the calculations that comprise a typical image processing pipeline [38]. All pipelines generally have some form of each of these calculations although the order may be changed or additional algorithms may be introduced [39].

- **Defect correction** accounts for errors in the sensor such as dead or hot pixels.
- **Lens shading** adjusts the intensity at different locations in the image to account for the decreased exposure near the outside of the image.
- **Demosaicking** estimates the unobserved colorbands at each pixel to generate a full color image from a CFA image.
- **Denoising** attempts to remove any noise while still preserving the underlying content of the image.
- **Color transform** converts from the color space measured by the sensor into a desired standard color space. For images that are to replicate human perception

of the original scene, the output colorbands are typically a linear combinations of the XYZ color matching functions shown in Figure 3.1 [40].

- **Deblurring / Sharpening** attempts to create a more pleasing image with defined edges by restoring details that have been blurred by the camera.
- **Gamma transform** applies an exponential to the linear intensity values so the output image can be displayed properly and comply with standard color formats such as $sRGB$ [41].
- **Compression** reduces the file size while minimizing the perceptual change in the image.

The demosaicking, denoising, and color transform are the only parts of the pipeline that critically depend on the particular CFA. Although a large number of algorithms have been proposed for each of these three calculations, most do not apply to novel CFA designs. Also certain assumptions that are the basis of many image processing algorithms for standard RGB based images fail to generalize to arbitrarily defined color spaces.

Demosaicking novel CFAs is challenging because spatial and spectral correlations in the images must be exploited to generate high quality images. When estimating values at a particular pixel, nearby measurements are often from different colorbands while pixels of the same color are farther. The colorbands may overlap and have important correlations that should be exploited to properly demosaic the image.

Similarly, denoising is challenging because of the difficulty in separating the signal from the noise that exists throughout the different colorbands. The strength of the noise differs among the colorbands based on the amount of light in each band. Also the noise in each colorband differs across pixels because one band was measured while the others were estimated when demosaicking.

For Bayer cameras, the color transform is typically performed as a linear combination of the estimated RGB values to the output XYZ values at each pixel. Often the same transformation is used for all pixels regardless of content or noise level. This approach can seriously fail for certain CFAs or output color spaces especially if the

spectral filters overlap or there are significant differences in noise levels among the channels.

To illustrate the difficulty of processing novel CFAs, consider measuring with the $RGBW$ CFA shown in Figure 1.1(b) and estimating XYZ under the following conditions.

- **Very low light** - Red, green, and blue measurements are very noisy and unreliable, but the white measurements are much less noisy. Since it is impossible to accurately estimate rich colors from these measurements, the color transform should rely heavily on the white channel, and the output images should be close to grayscale.

- **Dim light** - Red, green, and blue measurements are noisy, and the white measurements contain little noise. The white channel can assist in demosaicking and denoising by identifying structures in the scene such as edges. All four channels should contribute to the output image and be in the color transform.

- **Bright light** - All measurements contain little noise although the white channel may saturate and become unreliable. The white measurements are not very helpful in the color transformation because the red, green, and blue sensitivities are designed to approximately lie in the subspace spanned by XYZ. The color transform depends primarily on the red, green, and blue channels.

Clearly the demosaicking, denoising, and color transformation must adapt to the peculiarities of this particular CFA and change significantly over different light levels. This illustrates the difficulty of creating processing pipelines for novel CFAs.

1.3 Local, linear, learned pipeline

The traditional method of using heuristics and assumptions about images to design the above stages of the image processing pipeline is inefficient and very difficult for many new CFA designs. A learning method is presented that automatically forms a pipeline by calculating filters and necessary parameters from a training set of data.

The proposed approach estimates the output image in one fast calculation instead of the separate calculations commonly required for demosaicking, denoising, and transforming color. The subtle correlations between the different channels of the CFA and the desired output color space are learned from the training set. Learning also enables the resultant algorithms to be optimized for particular applications and datasets, which in general was impossible with previous algorithms. Computational costs are kept to a minimum by requiring only linear calculations and scalar comparisons in the pipeline.

1.4 Reflectance estimation from few spectral measurements

With a method for processing any possible CFA, the next important question is how to design the CFA for a particular application. The bands should be designed to adequately measure the wavelengths of interest while minimizing sensor cost and measurement noise. In order to understand the design tradeoffs, one must study the problem of estimating spectra from a small number of measurements like the different channels of a camera. It is much easier to understand the tradeoffs of the spectral sensitivities by initially ignoring the spatial dimensions of the camera.

Chapter 4 investigates the problem of estimating an object's reflectance using measurements from a camera with known spectral sensitivity under a known illuminant. An object's reflectance is a function of wavelength that describes the percent of incident light on the object that is reflected. The reflectance determines the appearance and color of the object. The local, linear, learned reflectance estimation method is introduced, and its advantages and limitations are presented.

1.5 Thesis outline

Chapter 2 presents the L^3 (Local, Linear, Learned) approach used throughout the thesis. The L^3 image processing pipeline for arbitrary CFAs is developed in Chapter 3. The spectral problem of estimating a reflectance from camera measurements is considered in Chapter 4. Conclusions and possible future work are provided in Chapter 5.

Chapter 2

L^3 Overview

2.1 Local, linear, and learned concepts

The approach and algorithms introduced for image and reflectance estimation are referred to as L^3. This stands for **L**ocal, **L**inear, **L**earned, and highlights unique features of the method. In general, the L^3 approach is a way to leverage training data to estimate an unknown vector based on a vector of possibly noisy measurements. For image estimation, the CFA measurements surrounding a pixel are used to estimate a small number of spectral bands at the center pixel. For reflectance estimation, the measurements are a small number of spectral bands of light coming from an object that are used to estimate the object's reflectance. The following are unique features of the approach.

Local refers to the adaptive nature of the estimation. This is in contrast to global approaches where all estimates are formed in an identical manner. Such global algorithms fail when the relationship between the measurements and desired outputs changes significantly based on the observed measurement values. To perform the adaption, the measurement space is divided into clusters. The clusters are designed so similar objects or image features are grouped together. The estimates for all measurements that fall in the same cluster are calculated in an identical manner and are optimized for the members of that cluster.

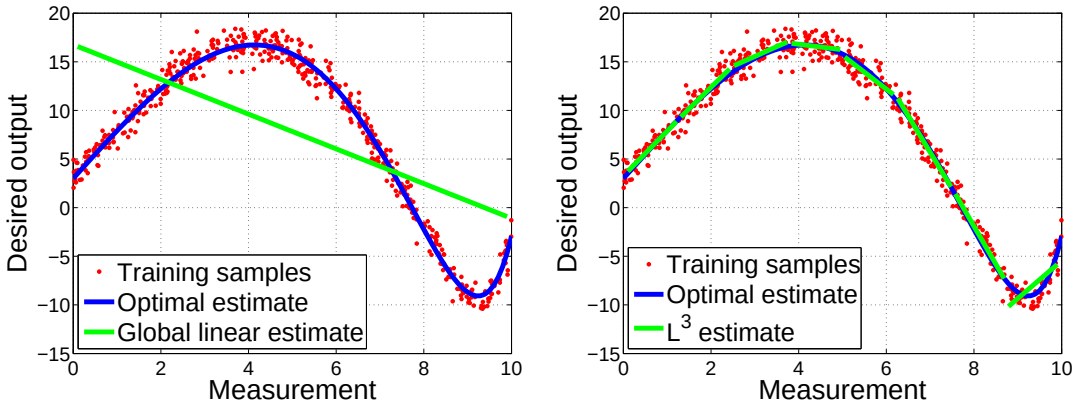
Linear emphasizes that almost all the algorithm's calculations are linear, hence

very fast to compute. Computational requirements are very important when operating on modern images and videos that contain millions of pixels. Many published image processing algorithms have very high computational requirements that may be too expensive for many applications. For a set of measurements, the proper cluster is identified using only linear calculations and comparisons. Then, a pre-computed linear filter for that cluster is applied to form the estimates. Under certain assumptions, the Wiener filter derived in Appendix A is the optimal linear filter for each cluster and is robust to the expected noise level of the measurements. In particular, assume the signal and noise are independent in a given cluster. Due to the signal-dependent nature of Poisson shot noise, this is not strictly true. But it is assumed since it simplifies the calculation and is reasonably valid when the training data in the cluster are similar.

Learned means that the estimation is driven by extracting statistics from a training set of data. Since estimation is generally an underdetermined problem, statistics from the datasets are needed to guide the estimates. Instead of relying on heuristics and general knowledge about images, which is common in image processing although challenging for non-RGB images, machine learning techniques are used to optimize the processing over a training set. The clustering method is learned from the training set. Then, the Wiener filter is calculated for each cluster, which achieves the least error over the training data in that cluster. As a result of the learning, the L^3 approach automatically generates estimates for a particular application. For applications where the reflectances or images are more constrained than typical scenes captured by consumers, the specialized algorithms may have significantly improved performance.

2.2 Illustrative example

Consider the fabricated estimation problem shown in Figure 2.2. The scenario is very simple where both the measurement and output are scalars. Measurements are randomly selected uniformly over the interval [0,10]. The corresponding desired output for each measurement differs from a fixed function of the measurement by a Gaussian



(a) (b)

Figure 2.1: L^3 **demonstration on one dimensional data.** (a) Global linear estimate derived from 500 training samples with $\sigma = 1$. (b) L^3 estimator derived over each of eight smaller intervals.

random variable with mean 0 and a constant standard deviation, σ. Therefore, the optimal estimate is this fixed function. The Gaussian random variable represents the deviation in the desired output that cannot be predicted from the measurement. A similar unavoidable error occurs in image and reflectance estimation because only limited measurements are available that do not perfectly predict the underlying signal. Note this unpredictability is different from measurement noise, which is absent from this example.

The optimal global linear (specifically affine) estimator derived from the training samples is shown in Figure 2.1(a). Notice that the global linear estimate is a poor fit for the data since the underlying function is very nonlinear. Figure 2.1(b) shows the result of dividing the interval [0,10] into eight intervals of equal width and deriving the optimal linear (affine) estimate based on the training samples that fall in each interval. Notice that the local linear estimate closely approximates the optimal estimate except in the [8.75,10] measurement interval. The optimal estimate varies too quickly in that interval to be well approximated. Now the estimator has enough freedom to closely track the optimal estimate. To make an estimate for a single measurement, first determine the interval that contains the measurement. The pre-calculated linear function or filter for that interval is then obtained from memory and applied.

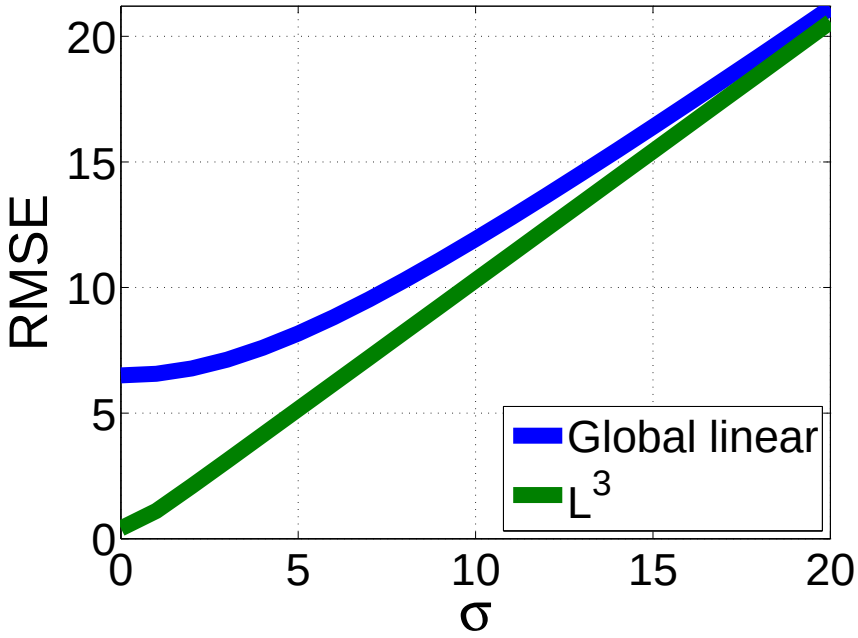


Figure 2.2: **L^3 is most valuable if relation between measurements and outputs is strong.** Root mean squared error (RMSE) of global linear and L^3 estimators using the underlying function and localization shown in Figure 2.2 with 500 similarly generated training points. Error was calculated over 10,000 randomly generated test points.

Kernel regression provides an alternate approach to estimate the underlying function and results in a continuous estimator. However, when performing an estimate one must access the original data and perform more computations. The local linear approach was chosen for its minimal computation and memory requirements during application, which is necessary for many modern imaging applications.

Figure 2.2 shows how unpredictability in the data affects the performance of the global and L^3 estimates. When $\sigma = 0$, there is a deterministic relationship between the measurement and the desired output. In this case, the RMSE directly reflects the estimator's bias or inability to approximate the underlying function. As σ increases, the relationship between the measurement and the desired output becomes weaker, which results in an unavoidable error for any estimator. Although the L^3 estimator is almost optimal for this example, when σ is large the error due to the unpredictability

of the data is large compared to the difference in bias between the global linear and L^3 estimators. As a result, the relative improvement of the L^3 estimator over the global linear estimator shrinks as σ increases. Measurement noise has a similar impact by degrading the performance of both estimators and shrinking the relative improvement of localization.

Therefore, the L^3 estimator is most valuable compared to the global linear estimator for datasets where there is a strong nonlinear relationship between the measurements and desired outputs and little measurement noise. Restricting the training dataset to only a subset of the possible objects or scenes in the world results in decreased uncertainty. Such constrained datasets exist for specific applications where imaged objects and scenes are more confined than in general photography.

2.3 Expansion to multiple dimensions

For estimation using multiple measurements, the division into local clusters and the linear estimation is performed in the higher dimensional space. If there are multiple outputs desired from the estimation, the clustering is performed once in the measurement space, and then separate linear estimators are calculated and applied for each desired output.

The method of dividing the measurement space into local clusters is an important design consideration. For computational reasons, clustering is performed hierarchically so that the number of clusters grows exponentially with the number of required branching operations. The division of a cluster into two smaller clusters involves a linear inner product and a scalar comparison, which are computationally efficient. The clusters are designed to have a strong linear relationship between the measurements and the desired outputs. If the linear relationship is not strong enough for a given cluster, further subdividing the cluster may be helpful. However, there is a risk of having too many clusters. In addition to increased computation and memory, the wrong cluster may be identified for a particular measurement due to the noise. There also is a need for each cluster to contain enough training data to avoid overfitting and generating regressions that may not perform well outside the training set.

Chapter 3

Image Estimation

The aim of this chapter is to create an image processing pipeline that offers accurate estimation and has the following important features:

- The pipeline must not demand too much computation or memory due to the large number of pixels in cameras today. Ideally the algorithm will require minimal computation and can expand in complexity as needed for increased performance.

- The pipeline must apply to any CFA. The CFA can contain any number of colorbands with any spectral shape and any spatial arrangement. Without a general approach, it is difficult to develop a pipeline that exploits the specific properties of a CFA. With a general approach, rapid evaluation through simulation of new CFA designs is possible, which otherwise requires reinventing the pipeline each time.

- The pipeline must allow output images to be in any color space. The color space can have any number of colorbands with any spectral shape. The output colorbands can be defined as an inner product between the incoming light and a sensitivity function that may have both positive and negative intervals.

- The pipeline must be robust to measurement noise. It should allow for the differing noise levels caused by the variation from dark to bright scenes. Also

it is important to recognize that there may be different noise levels across the color channels due to possible differences in overall sensitivity.

- The pipeline should be customizable for a given application. Algorithms tuned to the particulars of specific datasets may offer improved performance over pipelines for general imaging. This is especially important for cameras that will always be used for specific purposes where all imaged scenes are similar. In addition, the algorithm's task specialization may be valuable in consumer cameras if the type of scene can be determined. This may be possible through automatic scene detection or the user's input. For example, on many consumer digital cameras there exist pre-specified scene modes that the user can select, which could signal use of filters optimized for that scene type.

- The pipeline should be simple and understandable so that it can be adjusted to cause a desired result in the output images. Existing pipelines often require a large number of successive calculations, which makes understanding and modifying the whole system challenging. By combining several of the calculations, a designer can view the resultant images and adjust a few parameters to create a desirable change in certain images or image regions.

The L^3 pipeline fulfills these design requirements by leveraging machine learning on training images to simultaneously perform in a single calculation at least the demosaicking, denoising, and color transformation calculations from a traditional image processing pipeline. The lens shading stage of the pipeline presented in Chapter 1 is not initially included in the L^3 pipeline because it is not spatially shift-invariant. The gamma transform and compression calculations are also not included in the L^3 pipeline because they are highly nonlinear and easily performed in post-processing.

3.1 Background

There exist a large number of publications related to the algorithms in the image processing pipeline. Almost all existing research is directed to a single part of the

pipeline, and each step is considered in isolation. Only considering one of these tasks can create problems in camera pipelines due to the complex interactions of the calculations and rigid assumptions of some of the algorithms [39]. The following publications are exceptions that consider jointly performing demosaicking and denoising [42, 43, 44, 45, 46, 47], demosaicking and deblurring [48, 49], or all three at once [50].

The problems of demosaicking and denoising as commonly defined in the literature are well addressed. The differences between output images among leading demosaicking or denoising algorithms are often imperceptible [51]. Unfortunately the generally accepted problem setups for these algorithms does not match the needs of actual digital cameras. When performing demosaicking and denoising separately, the algorithms can be placed in either order. However, both approaches introduce challenges for the algorithms.

Demosaicking

Demosaicking is the process of estimating all measured colors at each pixel from an image with a CFA structure where only a single color is measured at each pixel. For a sensor with a three-channel CFA, two-thirds of the output image is never measured but estimated by demosaicking.

The simplest demosaicking algorithms independently interpolate each of the measurement channels. One such approach uses bilinear interpolation. Algorithms that operate on the channels independently yield poor results because they ignore the correlations between the colors. It is challenging to accurately estimate the red and blue channels of the Bayer CFA in this manner because each is measured at only a quarter of the total pixels.

Demosaicking algorithms for the Bayer CFA use cross-channel correlations by leveraging the high frequency information present in the green channel to help infer the high frequency information in the red and blue channels where it was not measured. It is common to first interpolate the green channel. Many algorithms assume that the high frequency coefficients in the red, green, and blue channels are

highly correlated [52]. The demosaicked image is often estimated by trying to reduce high frequency coefficients in color difference channels while matching observed measurements.

Surveys of existing demosaicking algorithms are available [53, 54]. Most algorithms are based on common assumptions or heuristics about images. Only a few algorithms clearly optimize for performance of a specific error metric over a set of images. A few algorithms include linear minimum mean squared error filtering or Wiener filtering as some part of demosaicking [30, 42, 47, 55, 56, 57, 58, 59, 60]. Generally these algorithms are globally linear and non-adaptive. Another demosaicking algorithm that incorporates training is provided in [61].

A number of demosaicking algorithms attempt to classify each pixel into a few different classes based on a small number of calculations often involving gradients in a few directions. The classes may reflect the presence or absence of large horizontal or vertical gradients. Once the class is determined, filters or calculations specific to that class are applied to estimate the missing values at the pixel. The L^3 pipeline uses a related clustering and filtering procedure.

Unfortunately almost all demosaicking algorithms are designed to work exclusively on the Bayer CFA. Demosaicking algorithms for the Bayer CFA generally offer little insight into the demosaicking of arbitrary CFAs. For example, the high frequency information that exists in the Bayer green channel that is essential to most algorithms may not exist for novel CFAs. Papers that consider alternate CFAs include [4, 30, 62, 63], however they require the spectral bands in the CFAs to be a linear combination of particular red, green, and blue sensitivities.

Demosaicking is becoming less important as pixel counts increase and errors on the order of a couple pixels that are common with demosaicking algorithms are imperceptibly small. In a pipeline, performing demosaicking before denoising poses the challenge of demosaicking noisy measurements. Noise severely degrades the performance of all demosaicking algorithms.

Denoising

Image denoising is the task of trying to remove unwanted random measurement noise from an image. Since only the noisy image is observed, denoising algorithms attempt to detect the underlying signal to filter out the noise without distorting the desired image. The simplest approach to denoising is to convolve the noisy image with a centrally weighted function such as a Gaussian. This reduces noise in uniform regions of the image but has the unwanted effect of blurring the entire image. All denoising algorithms try to detect the underlying structure such as edges in the image and preserve these features while averaging out noise.

The bilateral filter attempts to selectively blur the image while preserving edges [64, 65]. At each pixel, the filter is the product of a two dimensional Gaussian in the spatial domain and a one dimensional Gaussian in the radiometric domain. The hope is that any noise will be removed by averaging over space while avoiding blurring the edges. These filters are just a subset of the filters that can be obtained by learning the optimal filters using the L^3 algorithm. The bilateral filter is adaptive, spatially localized, nonlinear, and based on heuristics. It has also been applied to demosaicking [50].

A number of denoising algorithms rely on identifying a transform that compacts clean images into a small number of coefficients with large magnitudes [66]. Wavelets are an example of such a transform for most natural images. Since the noise is random, it will appear with equal power in all coefficients in the transform domain if the transform is orthonormal. In the transform domain, coefficients with large magnitudes are assumed to represent the signal and are unchanged. Coefficients with small magnitudes are assumed to be dominated by the noise and are either zeroed or reduced. After applying this nonlinear thresholding, the inverse transform yields the denoised image.

The L^3 image processing pipeline is perhaps most similar to the K-LLD denoising algorithm [67]. K-LLD clusters regions of the noisy image based on local geometric structure. A least squares problem is later solved for each cluster to determine how to form the denoised values. In contrast, the L^3 approach is based on training data separate from the image and does not perform any training on the noisy image.

A review of existing denoising algorithms is provided in [68]. Although many denoising algorithms exist, they generally are restricted to fully sampled grayscale images and assume additive white Gaussian noise. It is common for denoising algorithms to require a large amount of computation, which may be challenging to perform on a high resolution camera. Denoising is growing in importance as pixels shrink, capture less light, and are increasingly noisy despite improvements in sensor design.

In a pipeline, denoising could either be applied before or after demosaicking. Before demosaicking, images are not fully sampled because of the CFA pattern and have a complex noise distribution that is generally dominated by Poisson distributed photon shot noise. Denoising these images is very challenging. After demosaicking, images have an unknown noise distribution. The noise is not spatially white anymore and is harder to distinguish from the underlying image.

Color transform

The color transform converts measurements in the sensor's color space to a desired output color space such as for display. The most common color space standard is $sRGB$, which is derived by applying a nonlinear transform to the XYZ tristimulus values under a $D65$ illuminant [41]. The standard $D65$ illuminant shown in Figure 3.16 resembles daylight. The desired color transform uses sensor measurements defined by the camera's sensitivities under a known illuminant to estimate the appearance of the scene using a predefined set of colorbands as if the scene were under $D65$. For cameras to be viewed by humans, the output colorbands will be XYZ although this will be adjusted for multispectral imaging. The color transform generally has to account for two separate processes: sensor correction and illuminant correction.

Sensor correction is the process of using the measurements defined by the spectral sensitivity of the camera to estimate the desired output color space of the incoming light even though it was not directly measured. Since the camera's spectral sensitivities are determined by the actual device, which has cost and manufacturing

constraints, the sensitivities are not a linear combination of the XYZ functions. Consequently there is no linear transformation that perfectly estimates the XYZ values from the observed measurements. Also the camera's sensitivity functions determine how much light enters each channel, which affects the signal to noise ratio (SNR), so there is a desire to broaden the filters. The shape of the sensitivities also plays a role in the amplification of noise by the color transform as described in Chapter 4.

Illuminant correction is the process of estimating the scene's illuminant and calculating the scene as if it were illuminated by the standard $D65$ illuminant. Most displays render a white point with the same appearance as the $D65$ illuminant so the estimated scene should match its appearance under this illuminant. This process is often referred to by a number of names including white balancing, color correction, or color conversion. The human visual system attempts to determine the scene's illuminant and factor it out so object's in the scene can have a relatively constant appearance under a variety of illuminations [40]. This process helps a person identify an object despite the fact that the light reflected off the object may be spectrally very different depending on the illuminant. The most challenging part of illuminant correction is estimation of the spectral shape or color temperature of the illuminant. The difficulty lies in separating the effects of the illuminant and reflectance using only the observation of their product, the reflected light entering the camera. To avoid this problem, the illuminant will be assumed.

Generally the color transform is performed by a single linear combination of the measured channels and depends on the estimated illuminant of the scene. Local linear regressions for color transforms have been used previously [69, 70, 71]. A more thorough background of the literature related to the color transform is provided in Chapter 4.

Chapter contents

In Section 3.2, the camera simulation to generate training and testing data is detailed. A pipeline using learned global linear filtering is presented in Section 3.3. This method is very fast and performs well in most parts of images but poorly in regions with

texture or edges. Section 3.4 introduces the L^3 pipeline that improves upon the global algorithm by making it localized and adaptive to different image regions. Results of the pipelines are presented in Section 3.5 with extensions of the base L^3 pipeline presented in Section 3.6.

3.2 Camera simulation for generating training and testing data

Description of training set and generation methods

To generate a learning algorithm for an image processing pipeline, one needs a high quality set of input images and the corresponding desired output images. The learned algorithm will try to estimate the output images as closely as possible. Perfect estimation is impossible due to noise and a non-deterministic relationship between the input and output images.

Specifically, the training set consists of a collection of CFA measurements that contain little or no noise and the corresponding desired output, which may be calculated, designed, or measured. There is no need for the output images to be a scientifically accurate measurement of the scene if an altered image is preferable. For instance, camera manufacturers may adjust the appearance of certain colors such as the sky to make it more saturated, which might appeal to consumers. A large number of images are not required since each image contains many pixels, but the training images should be representative of the variety of scenes for the intended application. The training data must be generated only once for a camera design and an intended application, so it is reasonable to require sophisticated measurement or simulation equipment for acquisition.

There are a couple methods to generate the training data. One is computer simulations of known multispectral scenes based on basic properties of the camera. This has the advantage that there is no requirement for a physical camera prototype, which can be expensive to build especially for novel CFAs. It is very simple to experiment with new camera designs in simulation, which enables fast optimization

and testing without prototype cameras and a physical lab with calibrated test scenes. An existing camera can be characterized by photographing standard test charts to obtain the simulation parameters. As long as the simulations statistically resemble measurements from the camera's sensor, output images from the learned pipeline will match the simulated outputs.

For an existing camera, one could alternatively photograph a scene with a known multispectral distribution and use the actual sensor measurements for training. This removes any error caused by a mismatch between the simulation and the existing device. The multispectral data from the scene could be measured using one of a number of sophisticated and often expensive pieces of scientific equipment [25].

Simulation of sensor measurements

To generate training data and evaluate algorithms, the Image Systems Evaluation Toolkit (ISET) was used to simulate digital cameras [72, 73]. The simulations accurately account for the scene, optics, and sensor while offering an environment to test different camera designs and algorithms [74].

The simulation starts with a description of the scene to be measured. Multispectral images provide the reflectance at each spatial location for a uniform sampling of the visible wavelengths. The scene is modeled as appearing on a plane that is perpendicular to the optical axis of the camera at a set distance from the camera. The screen is assumed to be Lambertian, meaning it reflects incident light equally in all directions. This simulated reflectance target is illuminated using a spatially uniform light source with a known power spectral distribution. Unless otherwise stated, $D65$ is the illuminant. The intensity of the illuminant is varied to achieve different scene luminance levels. In actual applications, scenes often have depth, non-Lambertian surfaces, and illuminants that may vary in spectral shape and intensity across the scene. Although these simplifying assumptions have implications for the simulated photographs, removing the assumptions would require additional data that is difficult to capture and typically not included in published multispectral datasets.

The light from the simulated scene passes through the optics of the virtual camera.

The lens focuses the incident light on the sensor. The image on the sensor is blurred by the optics. The blur is described by the optical transfer function of the lens, which is a wavelength dependent function. For this simulation, lenses have no defects and are diffraction-limited for a given f-number. The simulated lens has f-number 4 and a focal length of 3 mm. For diffraction-limited lenses a point light source appears at the sensor as a dispersed pattern called an Airy disk. Real lenses may exhibit a number of defects called aberrations, although they are ignored here. A brightness falloff called vignetting occurs near the edges of the lens, but is not simulated due to the presumption that this will be corrected before applying the algorithms in this chapter. Care is taken to ensure the underlying multispectral scene is not oversampled or undersampled. The number of pixels in the sensor that observe the simulated screen is kept between 25% and 100% of the number of pixels in the original dataset. This is achieved by adjusting the field of view of the camera since the lens focal length, sensor pixel size, and distance to the screen are fixed. The aspect ratio of the sensor is adjusted to match the aspect ratio of the multispectral scene.

After the light from the scene passes through the camera's optics, it arrives at the sensor. For most cameras, the light first passes through a filter called a hot mirror that blocks any infrared light. It is then focused toward the photosensitive silicon substrate by a microlens that is placed on each individual pixel. The light for each pixel passes through the color filter that makes up the CFA. Part of the incident light that passes through the CFA is absorbed by metal connects on top of the photodiode that are used for controlling and reading the sensor. This loss of light is described by the sensor's fill factor.

Finally, the light is incident on the silicon photodiode where a photon with a given wavelength may generate an electron-hole pair with probability given by the quantum efficiency of the photodiode. The spectral effect of the infrared-blocking filter, color filter of the CFA, fill factor, and efficiency of the photodiode can be combined to give the spectral sensitivity of the corresponding color channel of the camera. The spectral sensitivity describes the probability a photon entering the camera towards a pixel will generate an electron-hole pair. The electrons are collected and converted to a voltage using an amplifier with a particular conversion gain. The voltage is quantized

with an analog to digital converter. Quantization is ignored in the simulations since quantization errors are typically small compared to other noise sources.

For a single picture, each pixel has a limited ability to hold generated electrons, which is given by its full well capacity. When the number of electrons generated by the light exceeds this amount, the pixel saturates. The measurement is at the maximum value and is not very useful because it is impossible to determine the correct measurement. In general for photography, the exposure duration, lens aperture, or gain (ISO) are adjusted to try to prevent saturation. However when using the $RGBW$ CFA in Figure 1.1(b), if the illumination is bright enough to achieve noise-free measurements in the RGB channels, the white channel may saturate. Fortunately the white channel is not needed for such bright settings and only results in a small loss of spatial resolution. Saturation is disabled from presented simulations to simplify the pipelines needed for processing.

Ignoring any noise, the voltage from each pixel in the sensor, y, is:

$$y = eg \int r(\lambda) l(\lambda) s(\lambda) d\lambda \tag{3.1}$$

where e is the exposure time, g is the conversion gain, λ is the wavelength of light, $r(\lambda)$ is the reflectance, $l(\lambda)$ is the illuminant in number of photons per second arriving at the pixel if the surface was perfectly reflective, and $s(\lambda)$ is the spectral sensitivity of a single channel in the camera. The exposure time for all simulations is 1/60th of a second. The spectral sensitivities for the R, G, B, and W channels used for the simulations are provided in Figure 3.1.

The ISET noise model gives z, the voltage measured by the sensor at a pixel, using the following equation.

$$z = \left(y + c + k_1 n_1 \sqrt{y + c} + k_2 n_2\right)(k_3 n_3 + 1) + k_4 n_4$$

Here n_1, n_2, n_3, n_4 are Gaussian random variables with mean 0 and standard deviation 1. These variables and the signal y are all independent. The dark voltage is given by c, which is caused by unwanted electrons in the circuit and scales linearly with exposure time. The term $k_1 n_1 \sqrt{y + c}$ represents the Poisson shot noise of electrons in

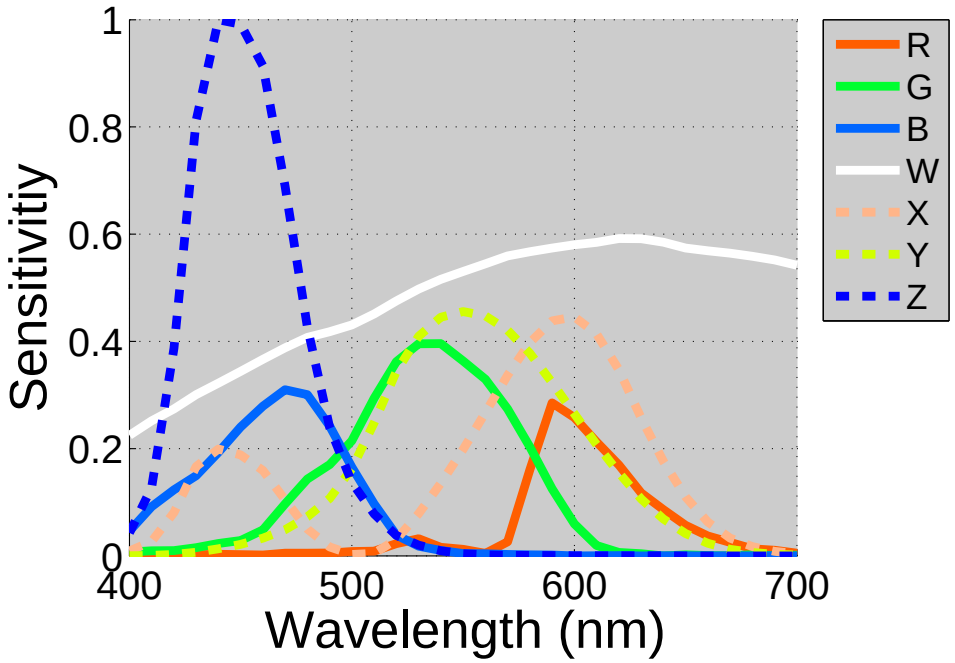


Figure 3.1: **Camera spectral sensitivities and XYZ color matching functions.** Solid lines are the simulated spectral sensitivities of the $RGBW$ camera channels. Dashed lines are XYZ sensitivity functions for the output color space. The RGB channels were measured from a Nikon D100 camera while the W band is similar to the spectral sensitivity of the silicon photodiode. The maximum values of the RGB, W, and XYZ curves were set to 0.4, 0.6, and 1, respectively.

the circuit where k_1 is the square root of the conversion gain. The Poisson distribution of the arrival of electrons in the circuit is well approximated for large values by the Gaussian distribution with mean 0 and variance equal to the variance of the Poisson random variable. Noise caused by reading and resetting the circuit are represented by $k_2 n_2$ where k_2 is the combined read and reset noise. Fixed pattern noise caused by undesired differences across the sensor consists of photo response non-uniformity (PRNU) and dark signal non-uniformity (DSNU). PRNU describes the differences in gain across pixels and is given by the multiplicative term $k_3 n_3 + 1$ where k_3 is the PRNU parameter expressed as a ratio. DSNU, given by $k_4 n_4$ where k_4 is the DSNU parameter, describes the voltage offset that can be expected among pixels when they are under the same illumination. The parameter values used for the simulations are

Sensor parameter	Value
Pixel width	2.2 μm
Fill factor	45%
Dark voltage	0.01 mV/s
Read and reset noise	1.34 mV
Dark signal non-uniformity	1.41 mV
Photo response non-uniformity	0.22%
Conversion gain	200 μV/e
Voltage swing	1.8 V
Exposure	1/60 s

Table 3.1: **Sensor simulation parameters.**

provided in Table 3.1.

The additive noise defined as $n = z - y$ is given by

$$n = \left(c + k_1 n_1 \sqrt{y + c} + k_2 n_2\right)(k_3 n_3 + 1) + k_3 n_3 y + k_4 n_4 \tag{3.2}$$

The noise power is needed later so it is derived below.

$$\begin{aligned} E[n^2] =& E\left[\left((c + k_1 n_1 \sqrt{y + c} + k_2 n_2)(k_3 n_3 + 1)\right)^2\right] + E\left[(k_3 n_3 y)^2\right] + \\ & E\left[(k_4 n_4)^2\right] + 2E\left[(c + k_1 n_1 \sqrt{y + c} + k_2 n_2)(k_3 n_3 + 1) k_3 n_3 y\right] \\ =& E\left[(c + k_1 n_1 \sqrt{y + c} + k_2 n_2)^2\right] E\left[(k_3 n_3 + 1)^2\right] + k_3^2 y^2 + \\ & k_4^2 + 2E\left[c + k_1 n_1 \sqrt{y + c} + k_2 n_2\right] E\left[(k_3 n_3 + 1) k_3 n_3\right] y \\ =& \left(c^2 + k_1^2 (y + c) + k_2^2\right)(k_3^2 + 1) + k_3^2 y^2 + k_4^2 + 2c k_3^2 y \end{aligned} \tag{3.3}$$

For most cameras today, the photon shot noise dominates all of the noise sources caused by imperfections in the electronic circuits except when the number of photons captured by the camera is very low. The shot noise is an inescapable property of imaging that cannot be corrected with improved electronics and is increasingly problematic as pixels continue to shrink in size.

Calculation of desired output images

In this chapter, all output images are calculated directly from the multispectral scene. The output images are inner products of the light that falls on the sensor with any number of defined sensitivity functions. This is accomplished by appropriately modifying $s(\lambda)$ in (3.1). However, practical limitations placed on the sensor's sensitivity can be ignored when calculating the desired output image. For example, the output sensitivities can have any shape and may be negative, which is not possible in actual sensors. In addition, the ideal output images are calculated without any shot noise or noise due to sensor imperfections. The resolution of the output images matches the sensor's resolution.

With the described simulation, the light passes through the lens before generating the output images so they have the effects of the camera's optics built-in. If it is desirable to modify or correct the impact of the camera's lens, the output image can be calculated using another more desirable lens. Then, the processing pipeline will attempt to reconstruct the image that would have been formed with the more desired lens instead of the actual one on the camera as described in Section 3.6.

When the goal of the output image is to recreate the perception of viewing the original scene, the output color space is the XYZ tristimulus values or a linear combination of them. The XYZ curves shown in Figure 3.1 are used as the sensitivities in (3.1). Since the equation requires the illuminant be expressed as the number of photons, the shown curves differ slightly from the standard XYZ curves defined by CIE (International Commission on Illumination) that assume the illuminant is expressed in energy units [40]. The depicted curves are the product of the standard curves, Planck's constant, the speed of light, and the reciprocal of the light's wavelength. The curves are also scaled by a constant so that the calculated voltage from the camera simulation is converted to give accurate XYZ measurements of the light that falls on the sensor.

Conversion from XYZ to $sRGB$

The XYZ values generally describe the appearance of a light to a human observer but need to be converted to a standard color space such as $sRGB$ for display [41]. For displays, images typically need to be described using $sRGB$ for the rendered image to closely match the original scene. The white point for $sRGB$ displays has the same appearance as $D65$ so the XYZ values need to be calculated using $D65$ as the illuminant by performing illuminant correction as described in Section 3.1. First, the overall intensity of the scene needs to be adjusted so it can be rendered on a device with a limited output range. This can be achieved by scaling the XYZ values by a constant. If the scaled values are $X^*Y^*Z^*$, the scaling should give $Y^* = 1$ for the desired white point. While the XYZ values provide a physical description of the scene, the $X^*Y^*Z^*$ values describe how the image should be rendered on a display.

The conversion from $X^*Y^*Z^*$ to $sRGB$ requires a linear conversion, clipping, and a nonlinear transform [75]. The linear color conversion is defined below.

$$\begin{bmatrix} R_{lRGB} \\ G_{lRGB} \\ B_{lRGB} \end{bmatrix} = \begin{bmatrix} 3.2406 & -1.5372 & -0.4986 \\ -0.9689 & 1.8758 & 0.0415 \\ 0.0557 & -0.2040 & 1.0570 \end{bmatrix} \begin{bmatrix} X^* \\ Y^* \\ Z^* \end{bmatrix}$$

To ensure output values can be displayed, these values are now clipped between 0 and 1 using $\tau(x)$.

$$\tau(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x \leq 1 \\ 1 & \text{if } 1 < x \end{cases}$$

A nonlinear transform, often referred to as gamma encoding, converts the linear luminance scale to an approximately linear brightness scale. For the $sRGB$ standard, the following transform is used.

$$\gamma(x) = \begin{cases} 12.92x & \text{if } x \leq 0.0031308 \\ 1.055x^{\frac{1}{2.4}} - .055 & \text{if } x > 0.0031308 \end{cases}$$

The final $sRGB$ values are calculated below.

$$R_{sRGB} = \gamma(\tau(R_{lRGB})), \quad G_{sRGB} = \gamma(\tau(G_{lRGB})), \quad B_{sRGB} = \gamma(\tau(B_{lRGB}))$$

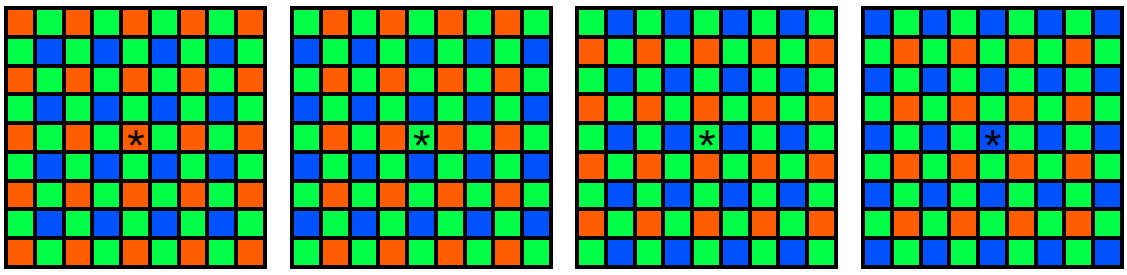
3.3 Global linear pipeline

A global linear learned pipeline is one of the simplest pipelines to implement and requires minimal computation. The output estimates at each pixel for a particular mean luminance are a fixed linear combination of the sensor measurements at and nearby the pixel. In this section, the pipeline introduced for a particular light level will be globally linear. However, the pipelines for different light levels will differ due to the differing SNRs. The approach is similar to some published algorithms mentioned in Section 3.1.

The algorithms in this chapter are spatially localized meaning values at each pixel in the final image are a function only of the sensor measurements at and nearby the pixel. This enables parallel processing. The existing implementation uses a square set of sensor measurements called a patch, which measures $\sqrt{m} \times \sqrt{m}$ pixels where $\sqrt{m}$ is odd. Since only the output values at the center pixel are estimated, the number of patches that must be processed equals the number of pixels in the image. There do not exist enough neighboring pixels to form a patch for pixels within $(\sqrt{m} - 1)/2$ pixels of the outside of the image, so no outputs are calculated in this border region.

It is common to have a similar spatial localization in demosaicking algorithms. Such a property is uncommon in denoising because non-local regions of the image may contain similar structure that can be used to determine the underlying signal and remove noise. Although locally restricting the estimation ignores possible patterns or statistics in the sensor image that may be helpful in estimation, the restriction significantly reduces computation and memory requirements.

There are several different patch types depending on how the CFA pattern lines up with the patch. Figure 3.2 shows the four patch types for the Bayer CFA, which will initially be used to illustrate the algorithms in this chapter. In general there are as many patch types as pixels in the repeating block of the CFA pattern, 2×2 for



(a) (b) (c) (d)

Figure 3.2: **Patch types for Bayer CFA.** (a) R-patch, (b) G_1-patch, (c) G_2-patch, and (d) B-patch. The * symbol denotes the center pixel. Patch size can differ from 9×9 pixels as shown here.

the Bayer CFA. Each patch type is denoted by the color of its center pixel where the output color channels will be estimated.

In general, separate filters are learned for each patch type. For the Bayer CFA, the G_1 and G_2 patches differ only by a rotation. Under the assumption that there is no fundamental differences in the vertical and horizontal directions in images, the optimal filters for these patches will also be rotations of each other. This assumption is approximately true for cameras since photographing the same scene in landscape or portrait orientations performs this rotation. Exploiting this rotation reduces the number of filters trained and stored.

Let $y \in \mathbb{R}^m$ be a vectorized sensor patch that is assumed to be noise-free. When testing, the measurements will be corrupted with noise $n \in \mathbb{R}^m$ so $z = y + n$ is observed. Note these definitions for y, n, and z that represent patches will be used throughout the remainder of the chapter and are different than the related terms for individual pixels previously used in Section 3.2. Let $x \in \mathbb{R}^o$ be the desired values of the o output colorbands at the center pixel of the patch.

To form the training data, k patches are extracted at random from a set of training images. Let the columns of $Y \in \mathbb{R}^{m \times k}$ and $X \in \mathbb{R}^{o \times k}$ be a collection of the y and x vectors from each patch. Although Y is assumed to be noise-free for training, the filtering needs to be robust to measurement noise. Let $N \in \mathbb{R}^{m \times k}$ be a random matrix representing measurement noise where each column is generated independently according to (3.2).

The linear estimator $W \in \mathbb{R}^{o \times m}$ is desired that makes the estimate $\widehat{X} = W(Y+N)$ most similar to X. The affine estimator presented in Appendix A.3 was tested but not chosen because it offered little improvement over the linear estimator. Generally $m < k$, so perfect estimation is not possible. Instead the sum of the squares of the errors of the estimates is minimized. Specifically, minimize $|\widehat{X} - X|_F^2$ where $|A|_F = \sqrt{\sum_{i,j} a_{i,j}^2}$ is the Frobenius norm. This corresponds to an optimal estimate as measured by mean squared error (MSE) or peak signal to noise ratio (PSNR). Unfortunately these metrics do not correspond well with perceptual errors, but are convenient for optimization [76].

For the Wiener filter derivation, a few simplifying assumptions are needed. To remove the signal dependence of the noise, N is assumed to be independent of X and Y, which may not be true in practice but is required for the Wiener filter. Instead the columns of N are assumed to be independent and identically distributed with mean 0 and autocorrelation R_n. The diagonal elements of R_n are calculated using (3.3) except the specific observed patch, y, is replaced by the average training patch. As a result, the Wiener filter solves for the average noise expected over the dataset even though it will vary depending on the measurement at each pixel. This assumption is reasonable for low dynamic range scenes but may be poor for high dynamic range scenes where there is a large variation in noise across the image.

Under these assumptions, the optimal linear filter is the Wiener filter, W, derived in Appendix A. Using (A.1), W is found by solving $W(YY^T + kR_n) = XY^T$. When applying the filter to a patch, the estimate of the output colorbands at the center pixel, $\widehat{x} \in \mathbb{R}^o$, is given by $\widehat{x} = Wz$. If there is no noise, denoising is automatically disabled to give a demosaicking algorithm using the optimal demosaicking filter, $W = XY^+$, where Y^+ is the Moore-Penrose pseudoinverse.

Figure 3.3 shows the Wiener filters trained on some natural images. Each row of the W matrix is the filter required to estimate a certain color channel and is converted from a vector to its corresponding patch for display. To obtain the global linear estimate of the XYZ channels at the center of a patch, simply calculate the inner product between the patch and each of the three filters.

As a reference, 2 cd/m^2 is similar to a dim indoor scene or an outdoor scene 15

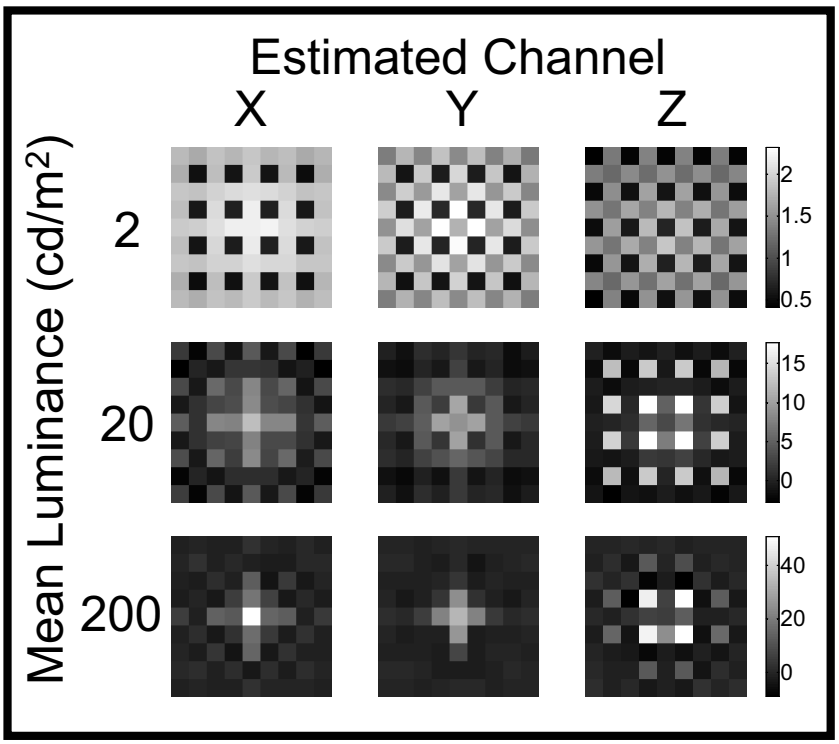


Figure 3.3: **Optimal filters are more dispersed for darker scenes.** Derived Wiener filters to estimate the X, Y, and Z channels at the center pixel of an R-patch. Each row of filters are scaled as shown on the right.

minutes after sunset with a clear sky. A brightly lit indoor scene or an outdoor scene at sunset under a cloudy sky is around 20 cd/m^2. Finally, 200 cd/m^2 is similar to the output from an LCD display or an outdoor scene at noon under a gray sky [77].

The filters in Figure 3.3 have a number of interesting features. Compared to bright scenes, the filters for dim scenes are more dispersed and have a smaller range of coefficients, which improves the filter's noise robustness. For bright illumination, the few pixels near the center of the patch are a good predictor of the desired channels because they have little noise. Also the signal is more likely to be the same at the pixels in the center instead of the outside of the patch. But as noise increases, it is risky to rely so much on the pixels at the center of the patch, so the other pixels are used more heavily. By spreading out the filters, the measurement noise can be smoothed away, however the signal is also blurred.

For the two brighter illumination levels, the patch's R, G, and B pixels have the largest coefficients for the X, Y, and Z channels, respectively. This occurs because these are the pairs with the most spectral overlap. For the darkest illumination, the total of the coefficients of the green channel is larger than the total of the coefficients for the red or blue channels for each of the X, Y, and Z filters. The result is an image with dull colors, which causes a systematic color bias although reduces the risk of incorrectly estimating more saturated colors in the output image.

Notice that only for the two brighter illumination levels are there negative coefficients. Part of the estimates are from a weighted difference between the pixels at the center and along the edges. This helps cancel out the overall effect of colors that may not correlate well with the desired output channel while finding any differences in intensity across the patch that assist with estimating the desired output channel. This difference is particularly sensitive to noise because subtracting two random variables results in a random variable with variance equal to the sum of the original random variables.

Figure 3.4 illustrates the performance of the global linear estimation method for a bright scene with little noise. The training was performed on six similar portrait scenes that were created at Stanford. The method performs very well under such little noise and the difference between the ideal output and reconstruction is imperceptible with a Spatial CIELAB value of 0.49. The details for the perceptual metric are provided in Section 3.5. The algorithm makes errors near sharp edges such as around the necklace or specular highlights in the skin and textured regions such as in the flowers. Slight color bias can exist for saturated colors like in the top-left of the image. This is due to the difficulty of estimating saturated colors using only three measured channels.

3.4 L^3 pipeline

Since only a single set of filters are allowed in the global linear pipeline for each light level, the filters must balance the competing interests of averaging to reduce noise and attempting not to blur edges in the image. To overcome this limitation, the

(a) (b) (c)

Figure 3.4: **Global linear estimation performs well except near texture.** (a) Ideal output image converted from XYZ to $sRGB$. (b) Result of global linear estimation using Bayer CFA with mean luminance of 500 cd/m^2 and 9×9 patch. Image was converted from XYZ to $sRGB$. (c) Difference between images (a) and (b). Gray value corresponds to mean over XYZ channels of absolute difference between the images at each pixel. The whitest point corresponds to errors of 10% or greater of the average value in the ideal XYZ image.

L^3 pipeline is introduced where a small set of Wiener filters are pre-computed and applied when appropriate to adapt to the local content of the image. The patches from the training data are organized into clusters of similar patches. For each cluster, the Wiener filter is found as described in Section 3.3 using just the patches in that cluster. If the clusters are designed well, the resultant filters are able to adapt to the particular properties of each cluster and differ between the clusters, which results in improved estimation. The clustering procedure and filters are derived from the training data once and stored. To apply the pipeline to a patch, the appropriate cluster is identified and the corresponding filters are accessed in memory and applied to give the estimate of the output image at the center pixel. Figure 3.5 provides an overview of the L^3 pipeline.

Classification into flat and texture patches

Since the largest errors caused by global linear filtering are near edges and textures, patches will be divided into two groups, flat and texture. Flat patches are relatively

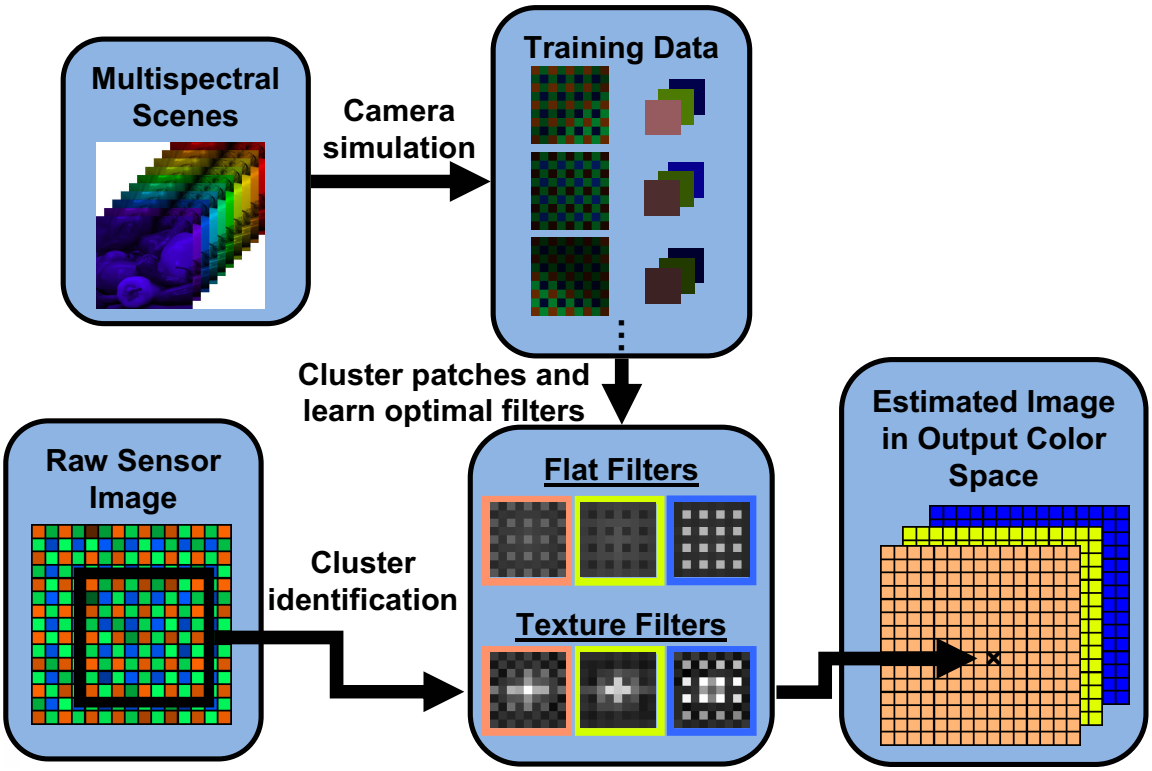


Figure 3.5: **Overview of training and applying L^3 pipeline.**

uniform areas of the image that contain only low spatial frequencies. Texture patches contain higher frequencies and appear as edges or texture. A diagram depicting the calculations of applying the L^3 pipeline is provided in Figure 3.6.

To distinguish between flat and texture patches, global Wiener filters from Section 3.3 are calculated except the output color space is the same as the CFA measurement bands (RGB for the Bayer pattern). These Wiener filters are referred to as the CFA filters and are given by $O \in \mathbb{R}^{h \times m}$ where h is the number of colorbands in the CFA. The CFA filters estimate the values in the different CFA measurement bands at the patch's center pixel. This is called the overall color of the patch, denoted by $\widehat{z} \in \mathbb{R}^h$, and calculated as $\widehat{z} = Oz$.

For each pixel in the patch, subtract the estimate that corresponds to that pixel's color. This results in a patch called the residual patch denoted by $z_0 \in \mathbb{R}^m$ and given by $z_0 = z - D\widehat{z}$. Here $D \in \{0,1\}^{m \times h}$ is a matrix that describes the CFA pattern in

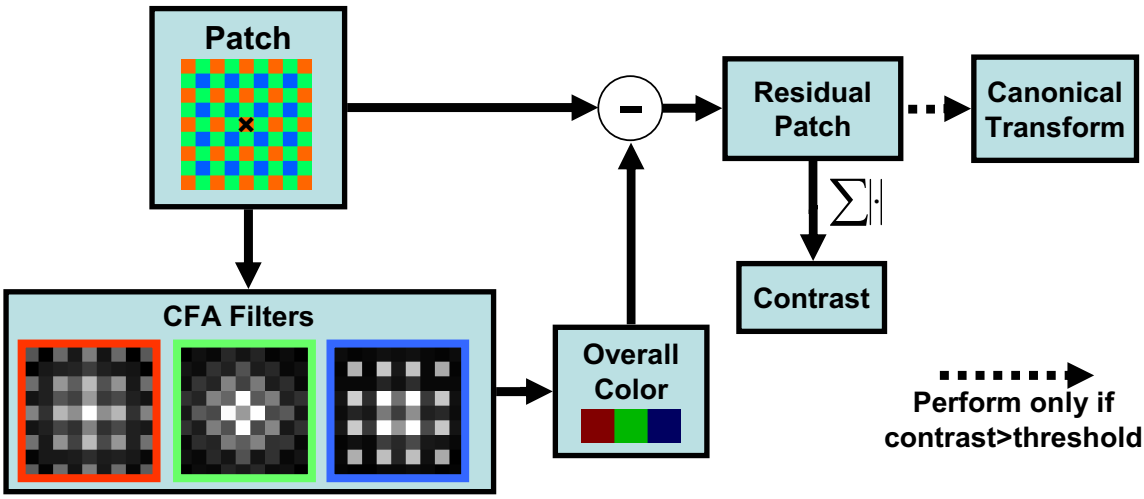


Figure 3.6: **Calculations for applying the L^3 pipeline.** A patch is classified as flat or texture by calculating the contrast and comparing it to a threshold. If the patch is texture, it is oriented before applying the texture filters. The CFA filters shown are optimized for 20 cd/m^2.

the patch where $D_{i,j} = 1$ if and only if the pixel in entry i of z is the same color as entry j of $\widehat{z}$. If the patch is perfectly flat, meaning all measurements are equal in the different colorbands, the residual patch is identically 0. The amount that each value in the transformed patch deviates from 0 is a measure of the amount of texture in the patch. Therefore, the contrast of the patch is defined as $c = \sum(\|z_0\|)$. A patch is flat if $c \leq c^*$ and is texture if $c > c^*$ where c^* is a predetermined threshold.

The threshold is chosen by simulating the noisy measurement process for the training patches and setting the threshold so a certain percentage of patches are classified as flat. This percentage is a design parameter of the algorithm. Since modern digital cameras tend to have a high pixel density, a large percentage of pixels can be considered flat, but this may depend on the content of the training scenes. It is important to realize that neglecting noise by calculating the threshold using noise-free training patches results in a threshold that is too low to achieve the desired percentage of flat patches when testing. This is because noise increases the contrast for all patches so a higher threshold is required to achieve the same percentage.

Figure 3.7(a) shows an example of the classification of flat and texture pixels.

(a) (b)

Figure 3.7: **Texture pixels have large global linear estimation errors.** (a) Classification of pixels into flat (black) and texture (white) patches. The threshold between flat and texture patches was set to classify 80% of the training patches as flat. (b) Error of global linear estimation from Figure 3.4(c).

Notice that the global linear pipeline results in small errors for flat pixels while the errors for texture patches tend to be higher as shown in Figure 3.7(b). Demosaicking and denoising errors are generally largest near edges and textures due to the difficulty of estimating the higher frequencies. The flat regions have relatively small errors. This is because there is little risk of blurring the signal, and noise can be filtered out aggressively. For these reasons, the flat patches will simply be filtered by a Wiener filter specifically trained on the flat patches. But the texture patches will be further processed in an effort to reduce their error. Fortunately a large majority of the pixels in an image are flat so calculation is still fast.

Canonical orientation of texture patches

A common goal in demosaicking and denoising algorithms is to avoid averaging or filtering across edges because this will blur the edge. The texture patches often have an edge or gradient, however these currently occur in any direction. No single filter for the texture patches can adapt to the orientation of the edge or gradient. The solution is to orient the patches so that the gradient is always in the same direction so a single optimal filter can be applied that takes advantage of the orientation.

Fortunately CFA structures often have one or more axes of symmetry that can be exploited. For CFA patterns with a 2×2 repeating pattern such as the Bayer and *RGBW* CFAs, each patch type has vertical and horizontal axes of symmetry. For the Bayer pattern, the red and green patches are also symmetric along either of the diagonals.

If the CFA pattern of a patch type is symmetric over a particular axis, mirroring the patch over the axis does not change the corresponding color of any of the pixels while keeping the center pixel in the same location. The ability to mirror the patches allows one to orient them so the gradient is in a particular direction. The patches are mirrored so that one side of each axis of symmetry will have a higher sum than the other side. Arbitrarily the left and top regions of each patch are chosen to have the higher value assuming vertical and horizontal axes of symmetry. Therefore, the oriented patches are always brighter on the left and top sides than the right and bottom sides. A Wiener filter is learned for and applied to the oriented texture patches, which can now take advantage of the orientation of the gradient across the patch.

Since which sides of the axes of symmetry that have a higher value is ignored, it is only advised when the information has no particular significance. Typically this is safe because the orientation of an image contains no interesting properties. There may be slight macroscale features that occur in images such as blue from sky appearing in the top of landscape images, but these features are relatively weak on the scale of the patches.

Wiener filters for the global, flat, and texture patches are provided in Figure 3.8. The global filters are a compromise between the flat and texture filters. This is because the global filters optimize the overall squared error, which is largest for the texture patches although 80% of patches are flat. The filters for flat patches are more dispersed in order to reduce noise by averaging with little risk of blurring the underlying signal. But filters for texture patches must be more centrally weighted to avoid using measurements where the signal has changed such as across an edge. Also notice the global and flat filters are approximately symmetric, which is not true of the texture filters because they have been oriented.

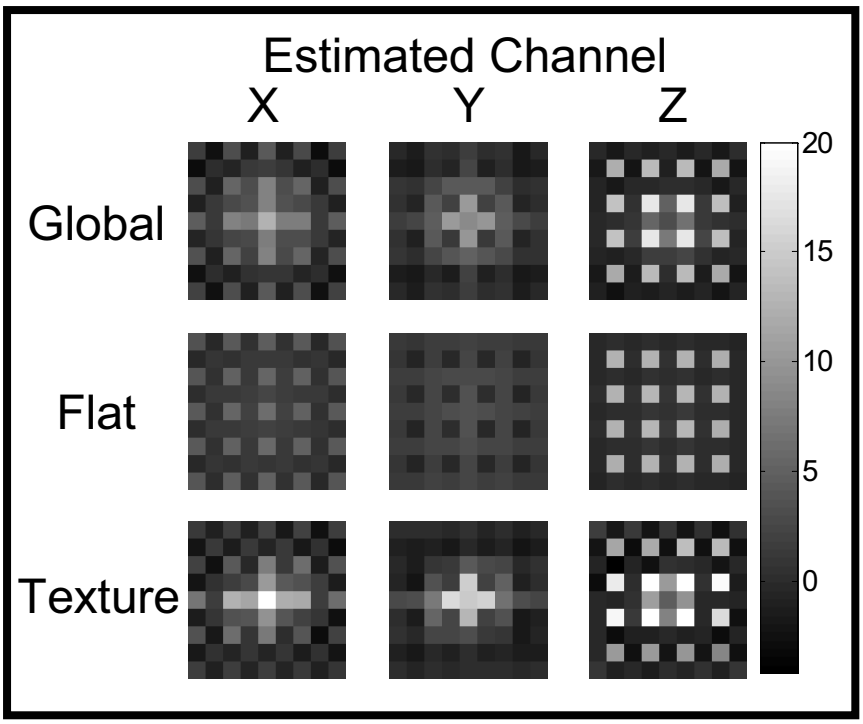


Figure 3.8: **Learned global, flat, and texture filters differ.** Derived Wiener filters for estimating the X, Y, and Z channels at the center pixel in an R-patch for a scene with mean luminance of 20 cd/m^2. The threshold between flat and texture patches was set to classify 80% of the training patches as flat.

Further clustering of texture patches

Since texture pixels contain a majority of the processing errors, it is tempting to further process them by clustering and finding an optimal linear filter for each cluster. The hope is that the texture clusters can more accurately capture differences among the texture patches. If the learned filters for each of these clusters are significantly different, the overall estimation may be improved. However, if the filters are similar, little improvement can be expected from subdividing the texture clusters. Details for how the texture patches are further clustered are provided in Appendix B. Only slight improvements of tested images are realizable by further clustering the texture patches, which means simply having a single texture cluster may be sufficient.

Implementation of flat and texture filters

Since the overall color is removed from a patch to give the residual patch during the flat/texture classification, it is more efficient to implement the flat and texture filters on the residual patch and overall color. The alternative is to either replicate the patch in memory before subtracting the overall color and apply the canonical transformation to both patches or convert the residual patch back to the original patch. To avoid this extra step, the flat and texture filters are instead converted to apply directly to the residual patch and overall color.

Let W be the flat or texture filter that generates the desired estimate by $\widehat{x} = Wz$. The removal of the overall color to obtain the residual patch can be written as

$$\begin{bmatrix} z_0 \\ \widehat{z} \end{bmatrix} = \begin{bmatrix} I - DO \\ O \end{bmatrix} z$$

where $I \in \mathbb{R}^{m \times m}$ is the identity. Let the matrix on the right side of the above equation be $A \in \mathbb{R}^{(m+h) \times m}$. Since the columns of A are linearly independent, $A^+A = I$ [78]. Therefore, $\widehat{x} = Wz = WA^+Az$. Let $\Psi \in \mathbb{R}^{o \times (m+h)}$ where $\Psi = WA^+$ and

$$\widehat{x} = \Psi \begin{bmatrix} z_0 \\ \widehat{z} \end{bmatrix}$$

The Ψ filters produce the same estimates as the W filters but can be applied directly to the residual patch and overall color as desired. These are the filters that are stored and used for computation in the pipeline.

A pseudocode description of the L^3 pipeline is provided in Algorithm 1. The `left`, `right`, `top`, and `bottom` functions are the sum of the entries in the corresponding side of the patch excluding the center row or column. The functions `mirrorv` and `mirrorh` flip the patch over the vertical or horizontal axes of symmetry, respectively, which are assumed to exist in the pseudocode.

Algorithm 1 L^3 Processing Pipeline

Input: Vectorized patch from sensor, z; CFA filter, O; binary matrix describing CFA structure, D; contrast threshold, c^*; tree height for texture clustering, t; texture branching filters, B_n; texture branching thresholds, b_n; filter for flat cluster, Ψ_f; filters for each texture cluster leaf, $\Psi_{t,n}$

Output: Vector of estimated values at center pixel for each band in output color space, $\widehat{x}$

$\widehat{z} \leftarrow Oz$
$z_0 \leftarrow z - D\widehat{z}$
$c \leftarrow \sum(\|z_0\|)$
if $c < c^*$ **then**
  return $\widehat{x} = \Psi_f \left[z_0^T \quad \widehat{z}^T\right]^T$
else
  if left(z_0)<right(z_0) **then**
    z_0 =mirrorv(z_0)
  end if
  if top(z_0)<bottom(z_0) **then**
    z_0 =mirrorh(z_0)
  end if
  $n \leftarrow 1$
  for $\tau = 1$ **to** t **do**
    $b \leftarrow B_n z_0$
    if $b < b_n$ **then**
      $n \leftarrow 2n$
    else
      $n \leftarrow 2n + 1$
    end if
  end for
  return $\widehat{x} = \Psi_{t,n} \left[z_0^T \quad \widehat{z}^T\right]^T$
end if

Table 3.2: **Computation required per pixel by L^3 pipeline.**

Stage	Multiply	Add/Subtract	Compare
Calculate overall color	hm	$h(m-1)$	
Subtract overall color		m	
Classify as flat/texture		$m-1$	1
Canonical orientation		$2p(m-\sqrt{m}-2)$	$2p$
Hierarchical clustering	ptm	$pt(m-1)$	pt
Filter with Ψ	$o(m+h)$	$o(m+h-1)$	
Total	$(h+o+pt)m$	$(h+o+(t+2)p+2)m$	$(t+2)p+1$

Computational complexity

The computational complexity of applying the L^3 pipeline is presented in Table 3.2. Recall that patches are $\sqrt{m} \times \sqrt{m}$ pixels, and the number of colorbands in the CFA and output color spaces are h and o, respectively. The fraction of patches that are classified as texture is p. The tree height for texture clustering, explained in Appendix B, is t. Most of the computations involve an inner product of two vectors that are of length m, which requires m multiplications and $m-1$ additions. The number of multiplications can be reduced if there is some symmetry in the filters such as can be enforced on the CFA and flat filters, however this is not assumed. Note m absolute values are also required to calculate the contrast when determining if a patch is flat or texture. For the canonical orientation, two axes of symmetry are assumed for the table although this may vary depending on the CFA. Terms that did not scale with m were omitted from the total count of multiplications and additions/subtractions in the table.

The global linear and L^3 pipelines can be modified to estimate the outputs for more than one pixel in each patch. For example, with a 10×10 patch, the output color space could be estimated at the center 2×2 pixels. This would result in the reduction by a factor of 4 in the number of patches that need to be calculated and an elimination of the need for multiple patch types. The resultant images should have a similar quality to filtering with the presented method using patches of 9×9 pixels because each of the four center pixels in the 10×10 patch are at least eight pixels away from the edge of the patch like in the 9×9 patch.

3.5 Results of L^3 pipeline

Noisy sensor images with CFA patterns are processed with the L^3 pipeline to make the estimated images in this section. These images are compared to the ideal desired output XYZ images calculated from the multispectral scene at each pixel in the sensor. Unless otherwise stated, the L^3 algorithm has the following parameter values: 9×9 pixels, 80% of training patches are flat, and tree height is zero so there is only a single texture cluster.

For display, ideal and estimated XYZ images are converted to $sRGB$ as described in Section 3.2. The largest Y value in the ideal XYZ image is used for the white point for the conversion. This is calculated separately for each set of parameters, which explains why scenes with different mean luminance values have the same ideal rendered images.

Low light enhancement from $RGBW$ CFA

Figure 3.9 shows estimated images under a number of different settings. The capture of a scene was simulated using both the Bayer and $RGBW$ CFAs. The images were processed using the L^3 pipeline and a basic pipeline consisting of demosaicking with bilinear interpolation and a linear color transformation optimized for a small set of reflectances under the assumption of no noise. Bilinear demosaicking was chosen because it applies to any CFA. The L^3 pipeline was trained on six similar portrait scenes, one of which is shown in Figure 3.4(a). All portrait images are 517×481 pixels. The estimated XYZ images from the pipelines were converted to $sRGB$ for display in Figure 3.9.

The two columns on the left are from the basic pipeline. The two columns on the right are from the L^3 pipeline, which has reduced the noise for all light levels compared to the basic pipeline. The images from the basic pipeline appear noisy because no denoising is performed. After scaling the XYZ values so the ideal image has $Y^* = 1$ as described in Section 3.2, any values less than zero or greater than one are clipped. For the basic pipeline, the noise can cause estimated values to be outside this range. For the 2 cd/m^2, 20 cd/m^2, and 200 cd/m^2 scenes, 80%, 24%, and 2% of

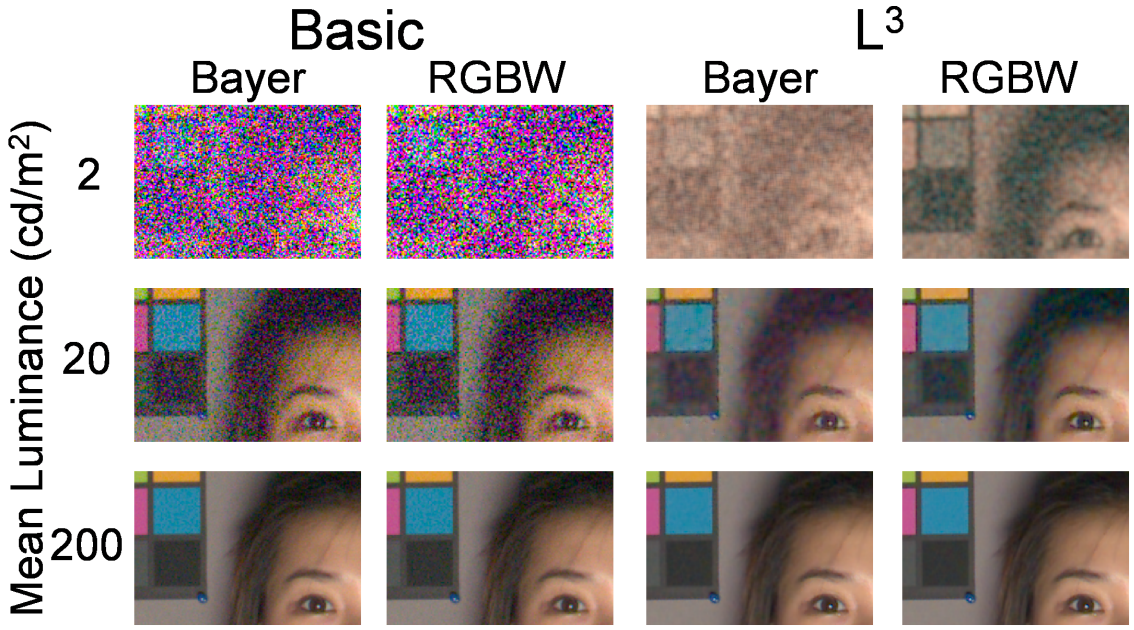
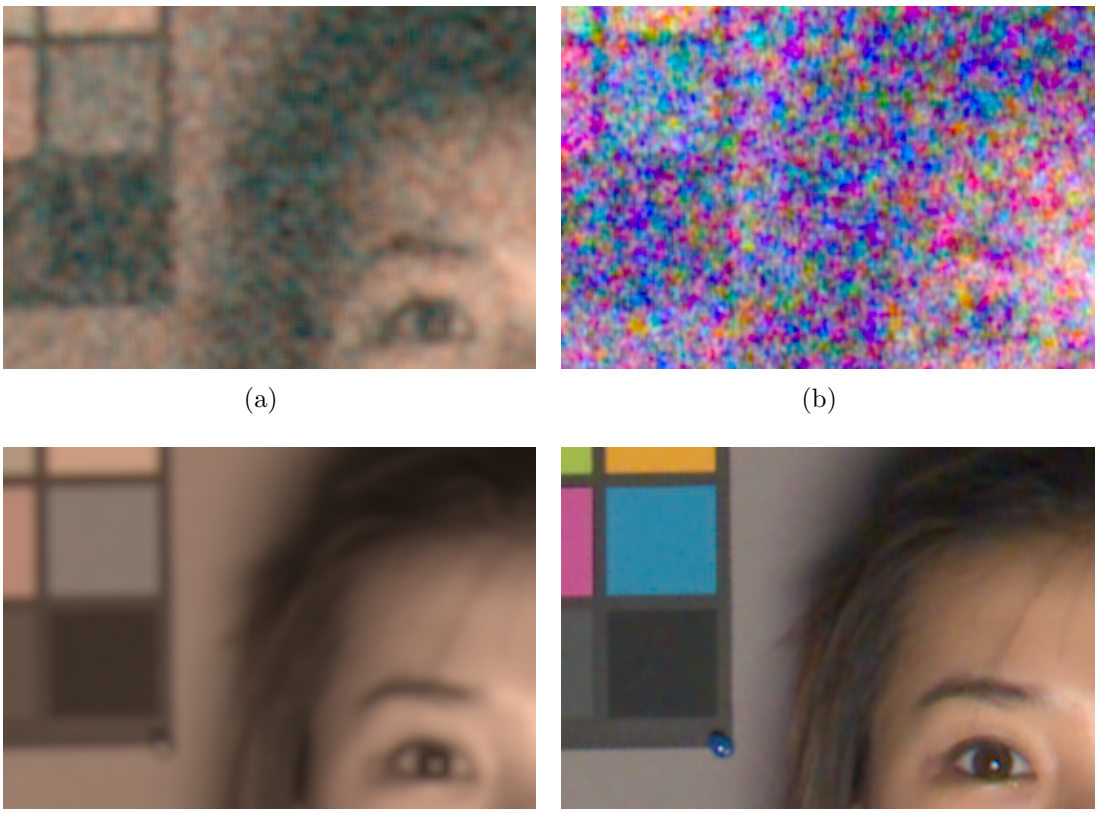


Figure 3.9: L^3 **pipeline and** *RGBW* **CFA improve low light imaging.** Images are the result of reconstructing a scene from simulated camera measurements and are cropped to 190×280 pixels. Ideal output image is omitted because it is very similar to the bottom-right image. Images from the darkest illumination for the L^3 pipeline use a single cluster, which is equivalent to the global linear pipeline, since clustering is not reliable or helpful for such a dark scene.

the pixels in the estimated images have at least one channel that needs clipped. These clipped pixels appear as saturated colors in the rendered images. On the other hand, the L^3 pipeline avoids this problem by compressing the range of estimated values to suppress the noise and results in almost no clipped values for any light level.

For the darkest scene, the L^3 pipeline generates images with desaturated colors. The *RGBW* CFA with the L^3 pipeline yields a recognizable image for the darkest scene by relying on the white channel while the other methods generate images that are generally unusable. The *RGBW* CFA with the L^3 pipeline enables imaging of dark scenes and assists in low light but has little impact, positive or negative, on relatively bright scenes compared to the Bayer CFA.

Figure 3.10 shows the impact of the L^3 pipeline's adaptation to the mean scene luminance. Images in Figures 3.10(a) and 3.10(d) are from pipelines trained on the



(a) (b)

(c) (d)

Figure 3.10: L^3 **adapts to luminance in a meaningful way.** Images are result of applying L^3 pipeline to simulated $RGBW$ images used for Figure 3.9. Pipeline used in (a, c) was trained for 2 cd/m^2. Pipeline used in (b, d) was trained for 200 cd/m^2. Scene for (a, b) had 2 cd/m^2 and for (c, d) had 200 cd/m^2.

mean luminance that is being used for testing. These images are identical to those in Figure 3.9. Figure 3.10(b) is the result of applying the L^3 pipeline trained on bright scenes to an image from a dark scene. Since the measurement noise is much stronger than anticipated by the pipeline, it is not aggressively removed, and the resultant image is very noisy like those from the basic pipeline. Although the saturated colors of the large uniform areas in the test chart in the top left of the images may appear more visible with the pipeline trained on the bright scene than the one trained on the dark scene, smaller features such as the eye and eyebrow are much more difficult to see.

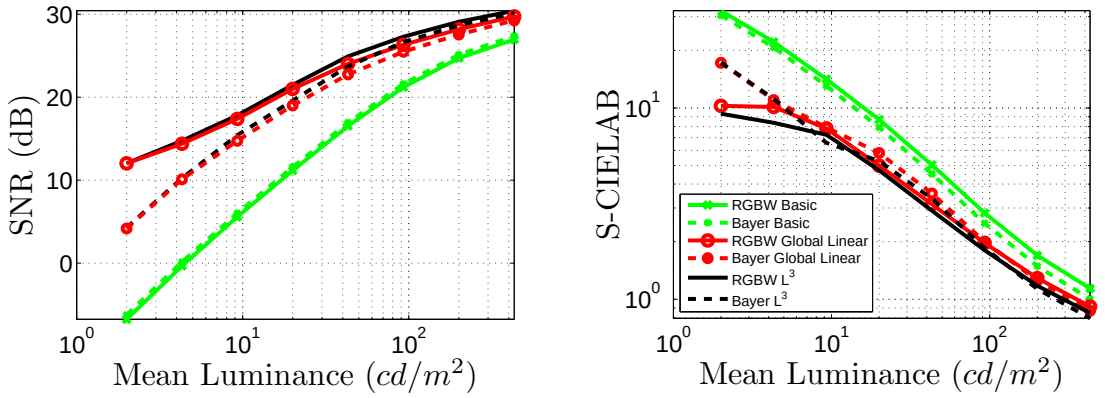
Figure 3.10(c) is the result of applying the L^3 pipeline trained on dark scenes to an image from a bright scene. Although there is little measurement noise, the pipeline significantly blurs the details of the image because it is trying to remove the noise it anticipates. The colors in the image are desaturated and similar to grayscale because the pipeline mainly relies on the white channel instead of the noisy RGB channels. The color bias of the L^3 pipeline for dark scenes can be viewed by comparing Figures 3.10(c) and 3.10(d).

The filters in the L^3 pipeline optimized for different luminance levels are different in a significant way as viewed by the large impact on the quality of the estimated images. The L^3 pipeline is learning to adapt to the simulated measurement noise in the sensor images. In an actual camera that may capture scenes with a large range of luminance values, a number of L^3 filters should be stored for different intensity intervals. More filters should probably be chosen for low light than bright light because of the more rapid change in SNR caused by photon shot noise for low light.

Image quality metrics

To evaluate the quality of estimated images, they are compared to the ideal XYZ images. The most common metric in image processing is the signal to noise ratio or related quantities. The SNR of the estimated image is given by $10 \log_{10} \frac{\chi^2}{(\chi - \widehat{\chi})^2}$ decibels (dB) where χ and $\widehat{\chi}$ are the ideal and estimated XYZ images. Figure 3.11(a) illustrates the SNR of the estimated images for different light levels. Although the filters are optimized for SNR, a perceptual quality metric is needed since SNR does not well predict error visibility [76].

The Spatial CIELAB (S-CIELAB) full reference perceptual quality metric is used to evaluate the visibility of the distortions introduced by the measurement and image processing pipelines [79]. The S-CIELAB metric spatially filters the XYZ images to simulate the blurring and sensitivity of the human visual system. The amount of blurring depends on the person's viewing distance and the size of the image. This is described by the visual angle that the image subtends horizontally. Unless otherwise stated, the visual angle of the full image is set to 20 degrees.

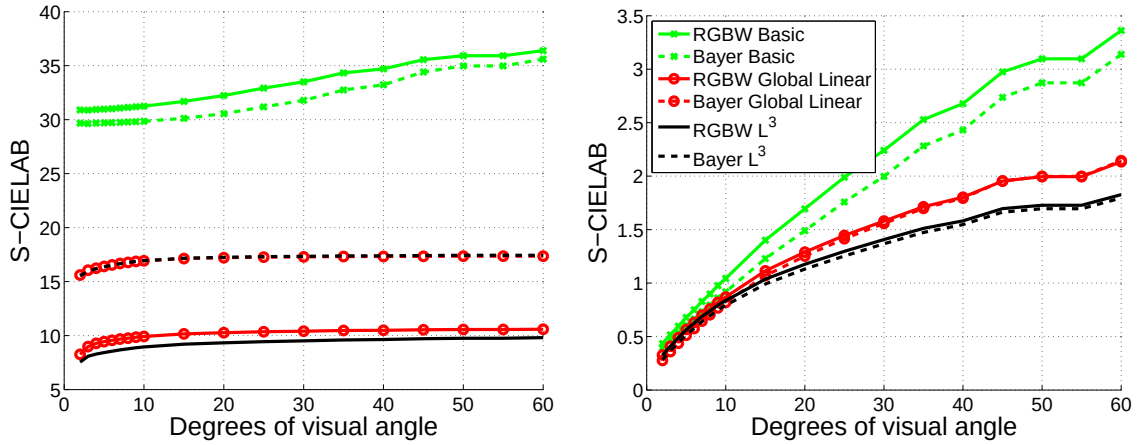


(a) (b)

Figure 3.11: *RGBW* **CFA with the L^3 pipeline improves image quality.** (a) SNR and (b) S-CIELAB error metrics of estimated XYZ images. Training and testing images are same as in Figure 3.9. Legend applies to both plots.

Once the image is spatially filtered, the standard CIELAB color metric is calculated at each pixel using the color appearance of the white point. The white point's XYZ value is calculated from the intensity of the simulated illuminant at the sensor. The units for the metric are ΔE where small values indicate a less visible distortion, and a value of one is generally a just noticeable difference. The metric gives a value for each pixel in the image. The mean across the pixels is reported as the S-CIELAB value for the image. Figure 3.11(b) illustrates the S-CIELAB metric of the estimated images. The border of the image of width $(\sqrt{m}-1)/2$ pixels is not included when calculating the S-CIELAB and SNR metrics because there are not enough measurements to form a patch as described in Section 3.3.

The basic pipeline has worse performance than the global linear or L^3 pipelines for all luminance values. The difference shrinks and is less visible as luminance increases since the noise is weaker. The global linear and L^3 pipelines have similar performance for both CFAs when luminance is small because the division into flat and texture clusters is problematic when SNR is low. There is a large difference between the *RGBW* and Bayer CFAs for dark scenes due to the improved SNR of the white channel. As luminance increases, the difference between the *RGBW* and



(a) (b)

Figure 3.12: **Low light improvement for *RGBW* persists for any image size.** S-CIELAB as a function of visual angle from same images as Figure 3.11. Mean luminance is (a) 2 cd/m^2 and (b) 200 cd/m^2. Legend applies to both plots.

Bayer CFAs decreases because the white channel is not as valuable when the *RGB* channels have little noise. With the improved SNR of brighter scenes, the L^3 pipeline offers a performance improvement over the global linear pipeline.

Since the resolution of images is often varied due to limited display resolution, Figure 3.12 illustrates the visibility of errors in the estimated images as the visual angle varies. This directly corresponds to changing the viewing distance from a full resolution display while the image maintains a constant size on the display. The setup is similar to adjusting the size of the image while maintaining a constant viewing distance from a display with a fixed resolution. Smaller visual angles indicate that high frequencies such as exist in noise are less visible.

Figure 3.12(a) shows that quality improvements caused by the *RGBW* CFA and L^3 algorithm are persistent for any visual angle. All images have S-CIELAB improvements for decreased visual angle due to the decreased visibility of fine details and noise. The images output by the global linear and L^3 pipelines for the *RGBW* CFA are significantly different from the other methods in both low and high frequencies, which enables the large improvement for any visual angle. On the other hand, the differences in estimated images for bright illumination exist primarily in the fine

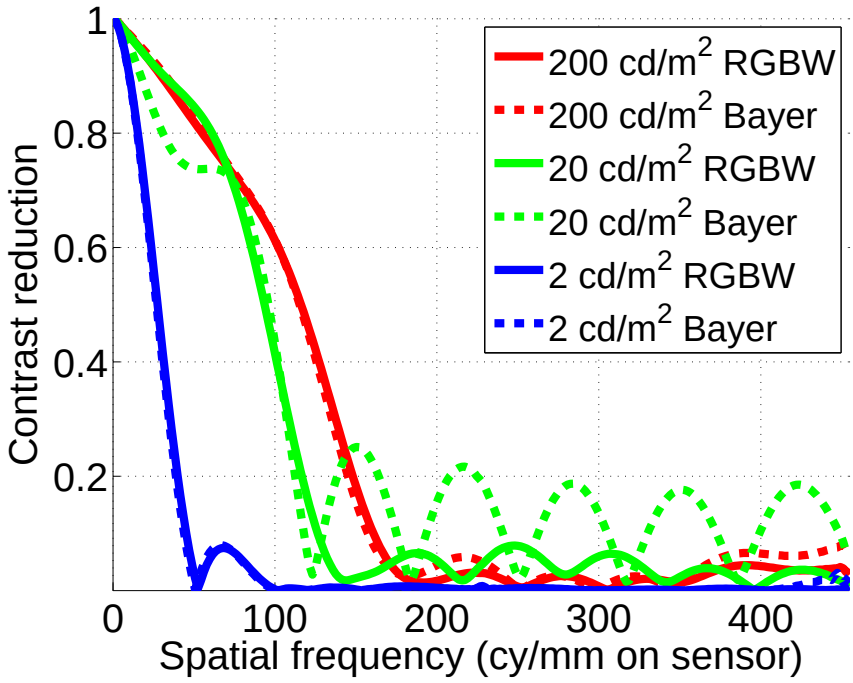


Figure 3.13: L^3 **pipeline automatically reduces high frequencies to counter noise.** MTF of luminance channel, Y, from images output by camera using L^3 pipeline with same setup as Figure 3.9.

detail and noise. As a result, the visibility of the estimated images from bright scenes varies significantly with visual angle as shown in Figure 3.12(b). For images from bright scenes that will be viewed at a lower resolution, only modest improvements are possible beyond the basic pipeline.

The spatial resolution of optical systems is often described by the modulation transfer function (MTF), which gives the magnitude of the system response to different spatial frequencies. Figure 3.13 provides the MTF for the measurement and L^3 pipeline for different light levels. This was generated with a noise-free simulation of the slanted edge defined in ISO 12233 [80]. The faster falloff of the MTF for darker scenes reflects the smoothing of the L^3 filter to try to reduce noise, which results in the loss of spatial resolution. The Bayer and $RGBW$ CFAs have similar MTFs, which indicates spatial resolution is similar for the L^3 pipeline with either type of sensor. If saturation were simulated, the MTF for the $RGBW$ CFA may be reduced.

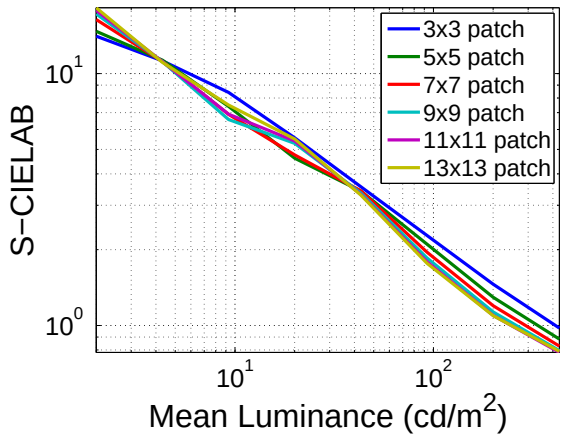


Figure 3.14: **Larger patches improve L^3 performance for bright scenes.** Scenes of various mean luminance were simulated with the Bayer pattern and then processed by the L^3 pipeline. Calculated from same scene as appears in Figure 3.9.

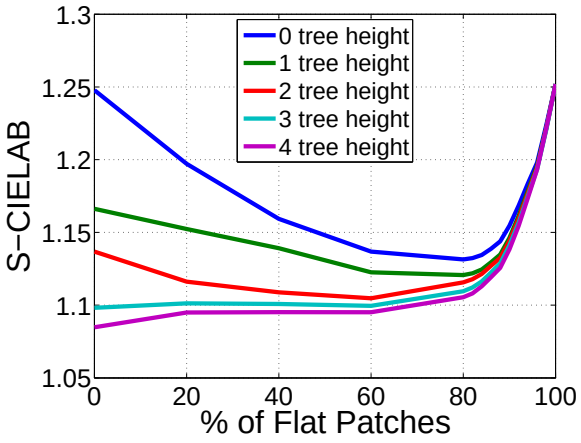


Figure 3.15: **Correct percent of flat patches slightly improves performance.** Result of setting threshold between flat and texture patches to achieve a desired percentage of flat training patches. Scenes were simulated with a mean luminance of 200 cd/m^2, sampled with the Bayer pattern, and estimated with the L^3 pipeline. Calculated from same scene as appears in Figure 3.9.

Impact of parameters

Figure 3.14 illustrates the visual impact of adjusting the size of the patch for various light levels. The largest trend in the data is that performance increases with increasing luminance, which is to be expected due to the increased SNR of the measurements. Since very little improvement is achieved for patches larger than 9×9 pixels and the computation generally scales with the number of pixels in the patch, 9×9 patches were chosen.

Figure 3.15 illustrates the visual impact of the threshold between flat and texture patches under bright illumination. Note choosing 100% flat patches corresponds to global linear filtering. Tree height describes the amount of clustering of texture patches as presented in Appendix B. A tree height of zero indicates use of a single texture cluster. For zero tree height, there is an ideal percent of flat patches. For

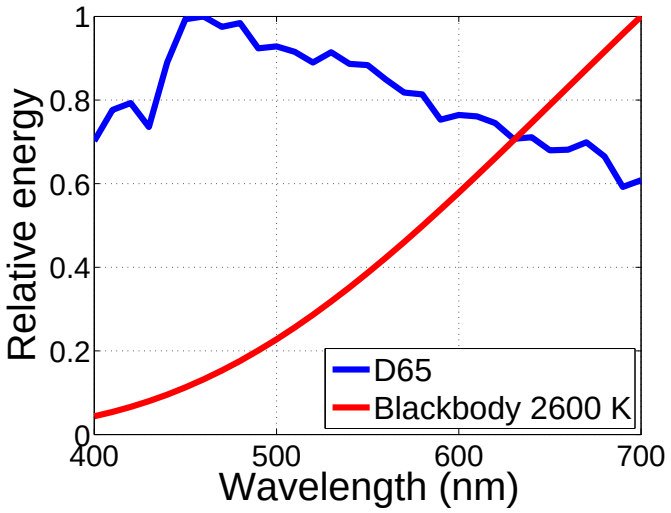


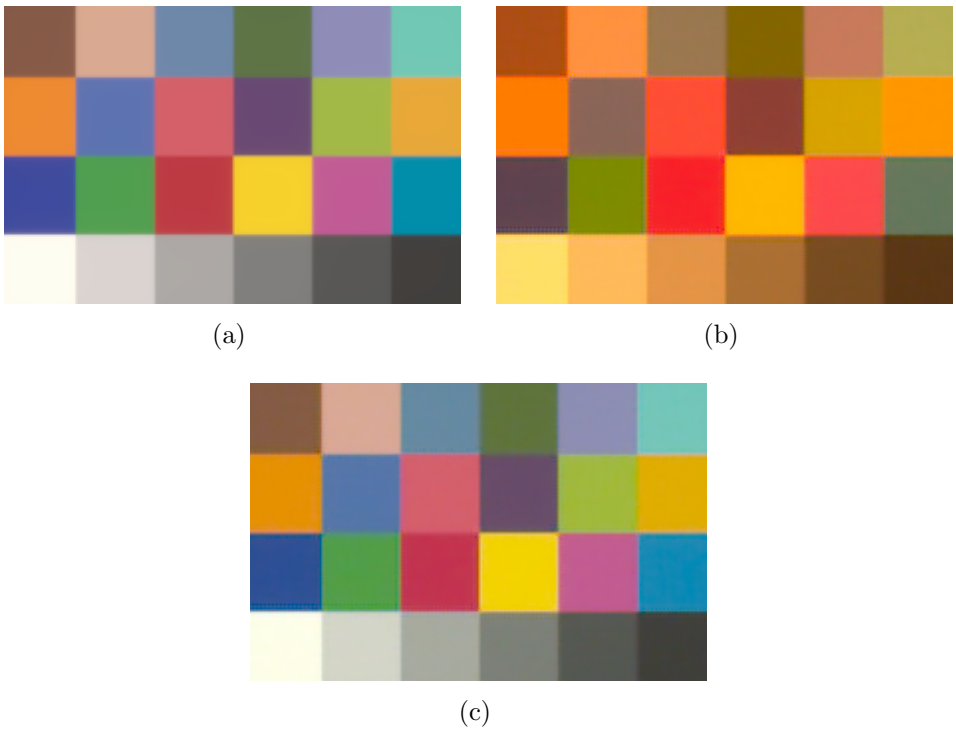
Figure 3.16: **Illuminants:** $D65$ **and Blackbody at 2600 K.** Relative spectral power distribution of standard illuminants. Maximum values are set to one although illuminants are scaled in simulation to achieve a desired mean scene luminance. Blackbody curve is given by Planck's law.

small tree heights, there is not enough flexibility in the filters to fit the data properly if too many patches are considered texture, which is why the curves increase to the left. For zero tree height, there is only one cluster for 0% and 100%, which is why the curve has approximately identical peaks at the endpoints. For larger tree heights, considering too many patches as texture is overcome by the added flexibility in the clustering procedure.

Color transform with illuminant correction

To test illuminant correction, a scene was simulated using an illumination from a blackbody at 2600 K shown in Figure 3.16. Using the measurements of the scene under this blackbody illuminant the scene's appearance under $D65$ needs to be estimated to generate a perceptual reproduction of the original scene.

Figure 3.17 shows estimated images of the standard GretagMacbeth ColorChecker reference chart under the 2600 K blackbody illuminant. Figure 3.17(b) is the result of applying the L^3 pipeline trained on scenes with both inputs and outputs rendered



(a) (b)

(c)

Figure 3.17: **Illuminant correction as part of L^3 color transform.** (a) Desired $sRGB$ rendering under $D65$. (b) Estimated image of scene with mean luminance 2000 cd/m^2 under 2600 K blackbody illuminant, sampled with Bayer CFA, and reconstructed with L^3 pipeline trained on $D65$ images. (c) Same as (b) except L^3 pipeline was trained with measurements from scenes under 2600 K blackbody illuminant and output scenes under $D65$. Training for all pipelines was performed on six similar charts with reflectances drawn randomly from the SOCS database [81]. All images are 150×230 pixels.

under $D65$. The estimated image has a reddish appearance compared to the correct rendering of the scene shown in Figure 3.17(a), which is a result of the blackbody illuminant containing more energy in the red part of the spectrum than $D65$.

The L^3 algorithm can be trained to automatically perform the illuminant correction in the color transform. The illuminant for the sensor measurements in the simulated scenes for training should match the illuminant from the scene later captured. If also the desired output image for training is under $D65$, the L^3 pipeline learns the illuminant correction into the filters.

The color correction of the L^3 pipeline is illustrated in Figures 3.17(c). The S-CIELAB values for Figures 3.17(b, c) are 21.3 and 3.7, respectively. For consumer cameras that may capture scenes under a wide variety of illuminants, a number of filters may need to be stored for the L^3 pipeline to perform illuminant correction. Generally a limited number of illuminants are considered in illuminant estimation algorithms, so a different set of filters could be stored for each possible illuminant. Alternatively, there may be a simple way to adjust the filters designed for one illuminant to perform the illuminant correction for another illuminant. Simply scaling the coefficients of each of the colorbands in the L^3 filters to adjust for the color of the illuminant could approximate the ideal filters and eliminate the need to store so many filters.

Learning on constrained sets

The L^3 pipeline adapts to the statistics of the training images. The amount of improvement of a pipeline optimized for a particular application, as opposed to a general pipeline designed for a huge variety of scenes, can be very large. However, the improvement depends on the dataset. The value of optimizing the pipeline increases with more constrained datasets because of their increased predictability.

For consumer cameras, the type of scene can be automatically detected by the camera or the user can selecting a number of pre-defined scene modes. Scene modes currently implemented on many digital cameras include portrait, sunset, landscape, document, fireworks, beach, and snow. Filters can be optimized for each of these types of modes. There also are a number of scientific, industrial, or medical imaging devices that are designed to only image a single type of scene. The processing pipeline can be designed and optimized for the expected scenes instead of employing a general pipeline that is designed for all types of photography.

To illustrate the potential of learning on constrained datasets for specific applications, consider the imaging of a document containing text. For general processing pipelines, images with text are challenging because edges are very small and sharp.

(a) (b) (c)

Figure 3.18: L^3 **learning of restricted images such as text.** (a) Ideal image. (b, c) Reconstructions after sampling with Bayer pattern and processing with (b) basic pipeline and (c) L^3 pipeline. Images are 250 × 135 pixels. The scene had a mean luminance of 200 cd/m^2.

When the edges are on the order of a couple pixels in width, all demosaicking algorithms for general images introduce color artifacts to the estimated image depending on how the edges line up with the CFA.

Fundamentally, images of black text on white paper have only a single color channel like grayscale images. If the processing pipeline is aware that the estimated images should resemble grayscale, the demosaicking problem becomes trivial. The mosaic can be removed by scaling each of the measurement channels by a constant to adjust for each channel's differing amount of the illuminant captured. No spatial interpolation is required except for denoising.

Figure 3.18 compares an original text image and output from the basic and L^3 pipelines. The scene was generated by creating an image of text in software. The reflectance at each point in the scene was then set for all wavelengths to a constant

given by the intensity in the image. The camera simulation blurred the scene and created the sensor image. The L^3 pipeline was trained on four similar scenes that used different fonts.

There are a large number of color artifacts in the image from the basic pipeline, which makes the text appear multicolored and difficult to read. The image from the L^3 pipeline is entirely grayscale because there were no colors in the training set's output images. It is impossible for any color artifacts to appear in the L^3 images because the derived XYZ filters only differ by a scalar multiple. For these images, the flat filters are spread out as much as possible to reduce the noise in the white page. The texture filters are very centralized so as not to blur the edges. Normally the texture filters need to be a little more dispersed because they require measurements from other colorbands in order to estimate the color.

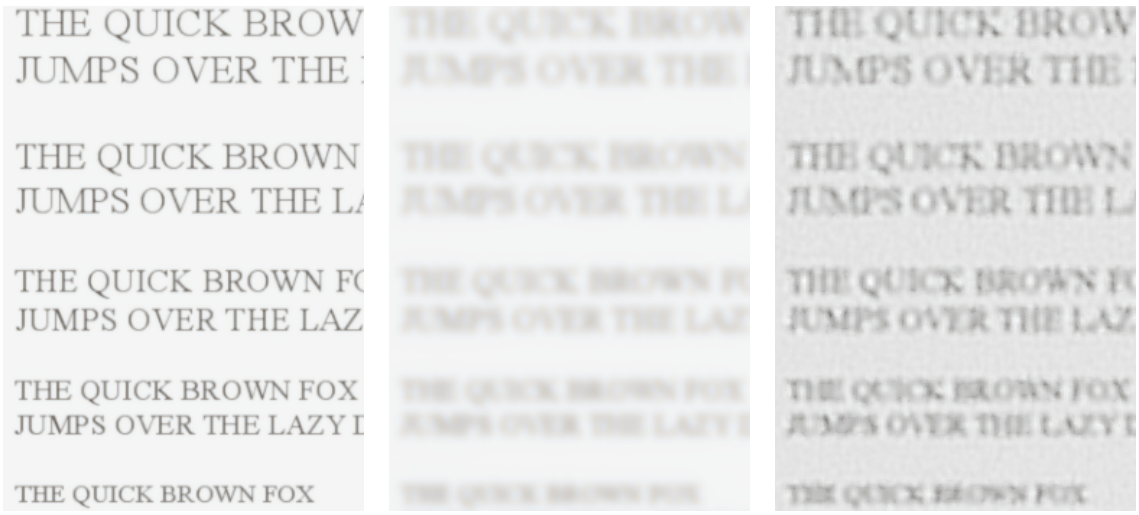
3.6 Extensions of L^3 pipeline

The following examples extend the L^3 pipeline's ability to perform demosaicking, denoising, and color transformation.

Deblurring

A common goal of image processing pipelines is to estimate images with well-defined edges since they may be more pleasing to consumers. Another way of describing this is to remove any blur caused by the optics or CFA through a process of deblurring. The L^3 pipeline can be adapted to automatically deblur an image by using sharp output images for training. The algorithm learns filters that can sharpen the blurred sensor images while simultaneously performing the demosaicking, denoising, and color transformation.

Figure 3.19 illustrates deblurring performed by the L^3 pipeline. For a fixed focal distance, lenses with larger f-numbers have smaller apertures, which causes increased diffraction. Without sharpening, the image from a lens with f-number of 32 is very



(a) (b) (c)

Figure 3.19: L^3 **pipeline can perform automatic deblurring.** (a) Ideal image formed with diffraction-limited lens with f-number of 4. (b, c) Reconstructions after imaging with diffraction-limited lens with f-number of 32, sampling with Bayer pattern, and applying L^3 pipeline trained on input images from lens with f-number of 32 and output images from lens with f-number of (b) 32 and (c) 4. Images are 350×250 pixels.

blurry as shown in Figure 3.19(b). The L^3 pipeline was trained with sensor measurements taken from these blurry images and output images taken from the sharp images formed by a lens with f-number of 4. Four similar scenes that used different fonts were chosen for training. The result of the pipeline is the much sharper image shown in Figure 3.19(c). Some letters that are not discernible in the blurred image can be made out in the processed image. The smallest letters are not as sharp since most of the information is lost in the blurring of the lens. Unfortunately deblurring generally is susceptible to noise, which is visible in the processed image. A large SNR of 40 dB was used for the measurements in these simulations.

To understand how deblurring is performed, MTFs of the system are presented in Figure 3.20. The MTFs are of the L^3 pipelines used in Figure 3.19 combined with the sharp lens with an f-number of 4. The standard L^3 pipeline optimized for the blurry image allows only low frequencies to pass through the system. This eliminates high

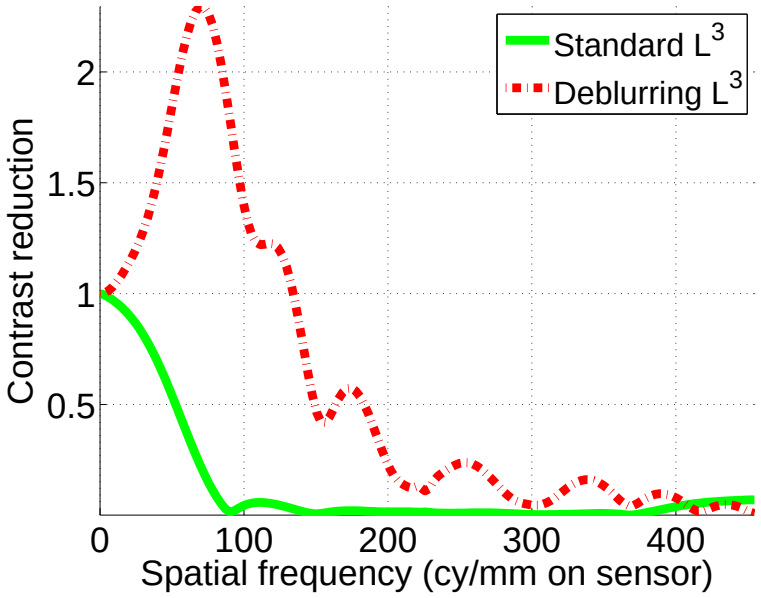


Figure 3.20: L^3 **pipeline boosts spatial frequencies to deblur.** MTF of luminance channel, Y, of estimated images after sampling with Bayer pattern using lens with f-number of 4 and L^3 pipelines used in Figure 3.19.

frequency noise while preserving the low frequency terms that make up the blurry image. The deblurring L^3 pipeline preserves the DC term while boosting low and medium frequencies. This counteracts the attenuation of these frequencies by the blurry lens. The high frequency terms below the Nyquist rate of 227 cycles/mm are passed through the system but not boosted due to the risk of enhancing the noise.

With the L^3 pipeline's ability to deblur, it may be possible to image with a cheaper lens that has more aberrations and correct the errors in processing to achieve images of the same quality. Jointly optimizing the optics and processing could reduce the cost and weight of the lens or increase the quality of an existing camera.

Multispectral estimation with six-channel CFA

Using a six-channel CFA such as in Figure 1.1(c), it is possible to estimate multispectral images from a single acquisition. The simulated camera adds cyan, magenta, and yellow filters to the previously described red, green, and blue filters. Generally one

wants to know an object's reflectance instead of the spectral power distribution of the light from the scene. The spectral shape of the illuminant is continued to be assumed since this must be known or estimated before performing reflectance estimation.

The problem of estimating a reflectance from a few measurement bands is explored in Chapter 4. Although the L^3 method can directly estimate each of the desired wavelength samples, this demands a large amount of computation and memory. Due to the smooth nature of reflectances, they can be well approximated using a few principal components as described in Appendix C. For the simulations here, the desired output space consists of six principal components, which offers a compact representation of reflectances. Since measurements of the reflectance are desired, the assumed illuminant is factored out before calculating the coefficients of the principal components when generating the desired output images for the training data. Once the coefficients of the principal components are estimated, the reflectance estimate is formed as a linear combination of the principal components using the coefficients. Note this is a linear estimator as opposed to the affine estimator detailed in Appendix C.

The reflectance estimation method was tested using the CAVE multispectral image database [1]. The L^3 algorithm was trained on the multispectral scene containing jelly beans. Since the scene contained many edges, the parameter of the amount of training patches classified as flat was set to 50%. Figure 3.21 shows the ideal and estimated reflectances from the L^3 pipeline on a different scene for a bright illumination. The reflectance estimates are very accurate except for wavelengths around 400 and 700 nm because the camera has limited sensitivity at the ends of the visible spectrum.

Multispectral object detection

A common task in multispectral imaging is to detect or classify different objects based on their spectral components. Many objects that appear identical to a human observer under common illuminations may actually have different reflectances, which could be used to distinguish the objects in a computer vision application. There are naturally occurring examples of similar materials such as if a disease slightly alters

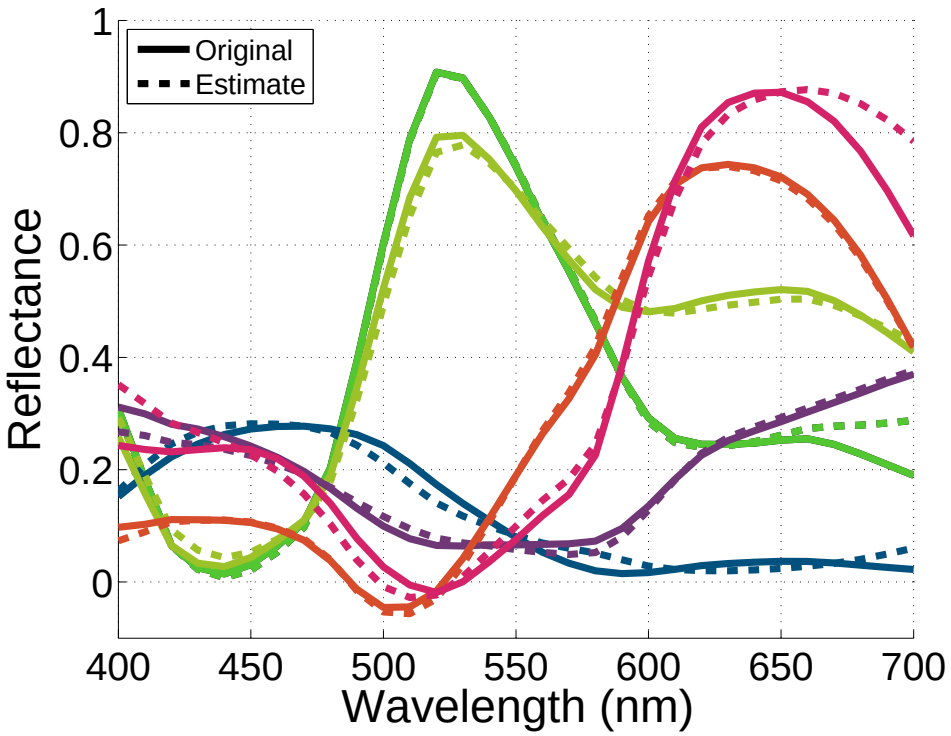
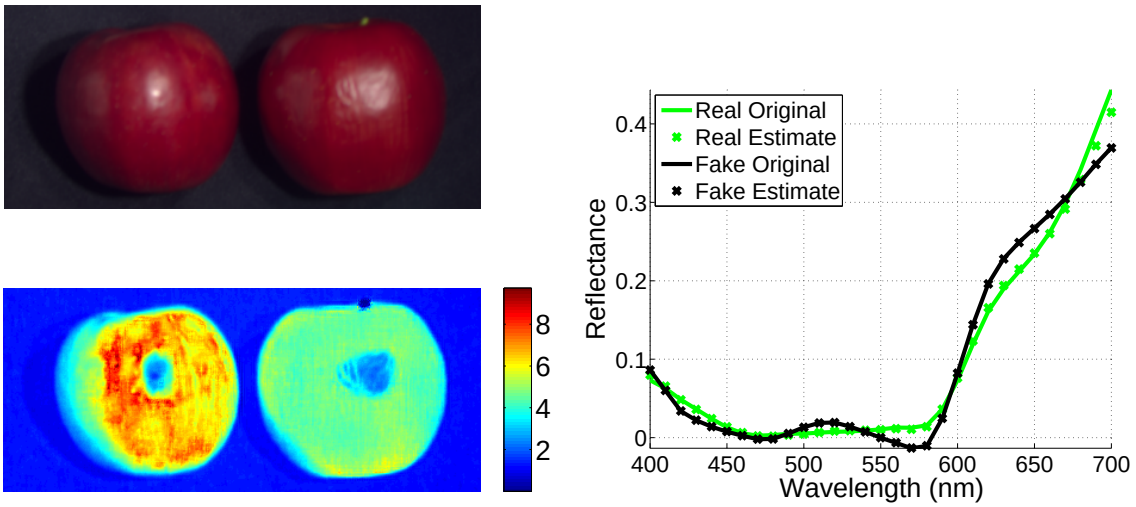


Figure 3.21: L^3 **pipeline can perform reflectance estimation at each pixel.** Estimation using principal components from simulated acquisition with a six-channel CFA of a scene containing multicolored beads with mean luminance of 5000 cd/m^2. Original and estimated reflectances represent averages over a number of pixels contained within a single bead. Curves with the same color represent the same object.

the reflectance of some cells or tissue of an organism. Camouflage is a man-made example of objects that appear similar but may differ spectrally.

Figure 3.22 shows a scene containing real and man-made apples [1]. The two apples appear nearly identical in color due to their deliberately similar reflectances in the visible spectrum. If data were available in the infrared portion of the spectrum, 700 nm to 1100 nm, the apples may be very easily detected using a camera that can measure in these wavelengths because the reflectances need not be similar over that interval. By sampling with the six-channel CFA in the visible spectrum and using the L^3 pipeline, the reflectances can be estimated with great accuracy assuming the L^3 algorithm is trained specifically for this task.

To differentiate the real and fake apples, a distance measure is introduced to



(a) (b)

Figure 3.22: L^3 **pipeline applied to multispectral object detection.** (a, top) Scene of real and fake apples shown in $sRGB$. (b) Original and estimated reflectances after sampling with six-channel CFA and averaging over a small number of pixels in each apple. Training data was taken from the same scene due to a lack of similar scenes. Simulated mean scene luminance was 5,000 cd/m^2. (a, bottom) Similarity of the estimated reflectance to the target reflectance. The colors indicate the value of $-log(1-d)$. Large values indicate similarity to the target reflectance of the real apple.

compare the estimated reflectances. Let $x_t \in \mathbb{R}^o$ be a representative reflectance of the target object to be detected where o is the number of wavelength samples. Let $\widehat{x} \in \mathbb{R}^o$ be a reflectance estimate that is to be evaluated if it belongs to the desired object class. Using the PCA estimation, $\widehat{x} = P\widehat{\theta}$ where $\widehat{\theta} \in \mathbb{R}^6$ are the estimated PCA coefficients and the columns of $P \in \mathbb{R}^{o\times 6}$ are the PCA reflectance vectors as described in Appendix C.

Due to the difficulty of estimating the absolute intensity of the illuminant at each pixel in the scene, which is described in Chapter 4, the distance measure is designed to be invariant to the overall height of the reflectances. The distance measure chosen is given by $d = \frac{x_t^T \widehat{x}}{|x_t|_2 |\widehat{x}|_2}$ where Euclidean norms are used in the denominator. Therefore, d is the cosine of the angle between the reflectance vectors in $\mathbb{R}^o$. The maximum value of $d = 1$ is achieved by reflectance estimates that have the same shape as the

target reflectance. This value decreases as the shapes differ.

It is more efficient to calculate the distance using the six PCA coefficients instead of the reflectances. Since over 30 wavelength samples are generally used for multispectral imaging, the memory and computational requirements are reduced by working on the PCA coefficients. Let $\theta_t \in \mathbb{R}^6$ be the first six PCA coefficients of the reflectance given by $\theta_t = P^T x_t$. Since most energy in the reflectances is contained in the PCA subspace, $x_t \approx P\theta_t$. The distance measure is approximated by $d = \frac{\theta_t^T P^T P\widehat{\theta}}{|P\theta_t|_2 |P\widehat{\theta}|_2}$. This can be simplified as $d = \frac{\theta_t^T \widehat{\theta}}{|\theta_t|_2 |\widehat{\theta}|_2}$, because the columns of P are orthonormal meaning $P^T P$ is the identity.

The bottom of Figure 3.22(a) illustrates the similarity between the estimated reflectance and the target reflectance from the real apple. The real apple on the left is clearly distinguished from the fake apple on the right and can be selected by thresholding the image. The specular reflections from the surface of the apples are not detected because they have the spectral shape of the illuminant with no modulation by the apple's reflectance. Notice the distance measure yields relatively uniform values across the curved surface of the apple despite large changes in the brightness of the reflected light.

Chapter 4

Reflectance Estimation

4.1 Background

Light incident on a surface can be absorbed, transmitted, or reflected. For opaque objects, the reflectance is the amount of incident light reflected for each wavelength, which determines the objects's color. The reflectance conveys a great deal of information about the object and is central to many aspects of multispectral imaging as discussed in Chapter 1. For example, the reflectance can be used to detect the nature of the material such as in computer vision and remote sensing tasks. For medical applications, the reflectance could be used to diagnose healthy or diseased tissue. For transparent objects, transmissivity describes the amount of light transmitted through the object for each wavelength. This chapter applies equally to transmissivity although reflectance is mentioned for convenience.

If one can measure the spectral power distribution, the amount of energy in each wavelength of light, of both the illuminant and the light reflected from a surface, the reflectance is calculated by dividing the functions at each wavelength. Properly characterizing the camera and illuminant is a common task in the field of color science and is critical for the effect of the reflectance to be isolated. Unfortunately measuring spectra at each wavelength requires sophisticated equipment that often is very expensive, may require long acquisition time, or lack spatial resolution.

There are also advantages of knowing an object's reflectance for general photography for human observers. For example, the color conversion from a camera's sensor to a standard output color space such as XYZ or $sRGB$ may be improved by having advanced knowledge of reflectances. These spectral estimates may be used for estimation of an illuminant's color temperature for improved white balancing. With knowledge of the reflectances in a scene, one could render an image as if it were taken under a different illuminant.

Reflectance estimation is also useful if one is not interested in the entire spectrum of light but instead only certain wavelengths or colorbands as needed by the application. For cameras that try to reproduce the scene as perceived by a human, one needs to estimate the XYZ values of the scene as if it were under a standard illuminant such as D65, which is modeled after daylight and shown in Figure 3.16. Reflectance estimation can explain how to do the necessary transform from the sensor's color space under the scene's illuminant to the XYZ channel under a different illuminant. The reflectance can first be estimated, and then the spectral components of interest can be calculated. Alternatively, reflectance estimation algorithms can be modified to directly estimate the desired output.

For these reasons, there is excitement about the possibility of leveraging technology in common digital cameras to measure a few colorbands of light from a surface and then estimate the reflectance. As previously mentioned, modifying the sensor's CFA to have around six different filters is a possibility for multispectral imaging. With existing cameras that use the Bayer CFA, multispectral imaging is possible by taking multiple images of a surface either under different colored illuminants and/or using different optical filters over the main lens [82]. Generally an increase in the number of color measurements results in increased accuracy in estimated reflectances.

If both the spectral power distribution of the illuminant and the spectral sensitivities of the camera are known, it is possible to use the small number of measurements of the light reflected from an object captured with one of the above procedures to estimate the reflectance function. However, since only a few colorbands are observed, the reflectance function is critically underdetermined. Therefore, it is impossible to guarantee accurate estimation.

Although reflectance is a continuous function of wavelength, it varies slowly so it can be sampled without much loss of information. For example, in color imaging analysis, light is often discretized by sampling every 10 nm in the visible wavelength range of 400 nm to 700 nm. The resulting length 31 vector has been found to well represent common illuminants and surface reflectances [83]. For illuminants other than fluorescent lights, it has been shown that light reflected from the representative Munsell set of spectra [84] is adequately sampled in the visible range at every 20 nm [85].

Since it is common to have fewer colorband measurements than the number of samples required above, one must rely on the statistics of reflectances to form a reasonable estimate. Reflectance functions tend to be smooth functions of wavelength, are always bound between 0 and 1, and fall within a small subset of all such possible functions. Reflectance estimation methods offer different ways to leverage the common statistics of reflectances to overcome the underdetermined nature of the problem.

A number of reflectance estimation methods have been developed [86]. The most complete way to incorporate knowledge about reflectances is to use Bayesian methods [87]. For many applications, this may be impractical due to the necessity of large amounts of data. A simpler approach is to summarize the knowledge of the likely wavelength functions by creating a low dimensional linear model of reflectance functions and restrict the estimate to this linear subspace [88, 89]. An intermediate strategy between Bayesian and linear models has also been proposed [90]. A two-step estimation was proposed in which a linear estimate is altered by values in a look-up table. The table is designed by encoding the typical estimation errors of the linear model for different parts of the sensor space.

Often measurement noise is not accounted for in these reflectance estimation algorithms. The estimates formed under the assumption of no noise may be very poor when tested with actual noise. This is especially problematic for spectrally overlapping measurements.

Scale Invariance

It may not always be feasible to know the intensity of the illuminant at an object but only its shape [90]. For non-planar scenes, it is very difficult to estimate the intensity of the light incident on each point in the scene due to the varying distance from the light source or sources. For example, this typically occurs when using a flash with the camera. Since the illuminant intensity at a particular point is not known, it is not possible to distinguish between a dark object that is close to the light source and a bright object that is farther from the light source. In this setting where the illuminant intensity is not known at each point, the scale of the reflectance cannot be reliably estimated. However, the shape or relative values of the reflectance can be estimated if the shape of the illuminant is known.

Scale invariance can be incorporated into reflectance estimation algorithms by factoring out the average of the measurements and scaling each of the training reflectances by the same factor. As a result, the shape of the reflectance is estimated but the overall height is lost.

Chapter contents

In this chapter, the Bayesian, global linear, and L^3 estimators are introduced. The PCA subspace estimation method is presented in Appendix C. The accuracy of the estimation methods are compared for different numbers of filters and measurement SNRs. The L^3 estimator has nearly optimal performance in general and performs significantly better than the global linear estimator only for constrained sets of reflectances.

4.2 Problem formulation and notation

In order for the results of this chapter to apply to any camera, an ideal camera is simulated. Even for an ideal sensor that generates perfect measurements, photon shot noise is still present. Photon shot noise dominates all noise sources in modern cameras except for very dark scenes. The simulation of an ideal camera at a single

pixel is far simpler than a realistic camera simulation such as in Chapter 3.

For this section, assume there is a single illuminant and m different colorbands acquired by some combination of optical filters on the lens or in the CFA. Let $l \in \mathbb{R}^o$ be the spectral distribution of the illuminant sampled at o different wavelengths. Specifically, l is the number of photons per second that would arrive at the sensor's pixel if the imaged object in the scene were perfectly reflective. The illuminant is $D65$ for all simulations in this chapter. Let $x \in \mathbb{R}^o$ be a sampled reflectance. Let the columns of $s \in \mathbb{R}^{o \times m}$ be the spectral sensitivity of the m different colorbands of the camera. Unless otherwise stated, the spectral sensitivities are from a Nikon D100 as shown in Figure 3.1. All entries in s are between 0 and 1 and indicate the likelihood that a photon with a given wavelength that enters the camera heading toward a pixel of a particular color will generate an electron that is captured by the sensor.

Ignoring any noise sources, let the number of electrons captured by a pixel be $y \in \mathbb{R}^m$. If a photo is taken with an exposure duration of e, $y = es^T diag(l)x$ where $diag(l)$ is a diagonal matrix with diagonal entries given by l. This is similar to (3.1). Let $\Phi \in \mathbb{R}^{m \times o}$ be given by $\Phi = es^T diag(l)$ so that $y = \Phi x$. The illuminant and camera have been combined into Φ to express the measurements as a linear projection of the reflectance.

The vector y will be treated as the noise-free measurements even though actual cameras do not output the number of electrons captured but must quantize the resultant voltage. In alternate measurement setups with multiple illuminants, Φ should have a column for each combination of illuminant and optical filter that is measured. The training data consists of a collection of k known reflectances in the columns of $X \in \mathbb{R}^{o \times k}$ and their resultant noise-free measurements $Y \in \mathbb{R}^{m \times k}$, where $Y = \Phi X$.

Since Poisson shot noise is considered, the noisy measurements, $z \in \mathbb{R}^m$, are given by $z = \eta$ where $\eta \in \mathbb{R}^m$ is a Poisson random vector with mean y. For large values of y, this is well approximated by $z = y + n$ where n is a Gaussian random vector with components that are independent given y and have mean 0 and standard deviation $\sqrt{y}$. Therefore, the SNR for each measurement is given by $10 \log_{10}(y)$ decibels (dB). For simplicity, the Gaussian approximation is used to generate the noise.

Since the SNR varies for different measurement bands and reflectances, a single

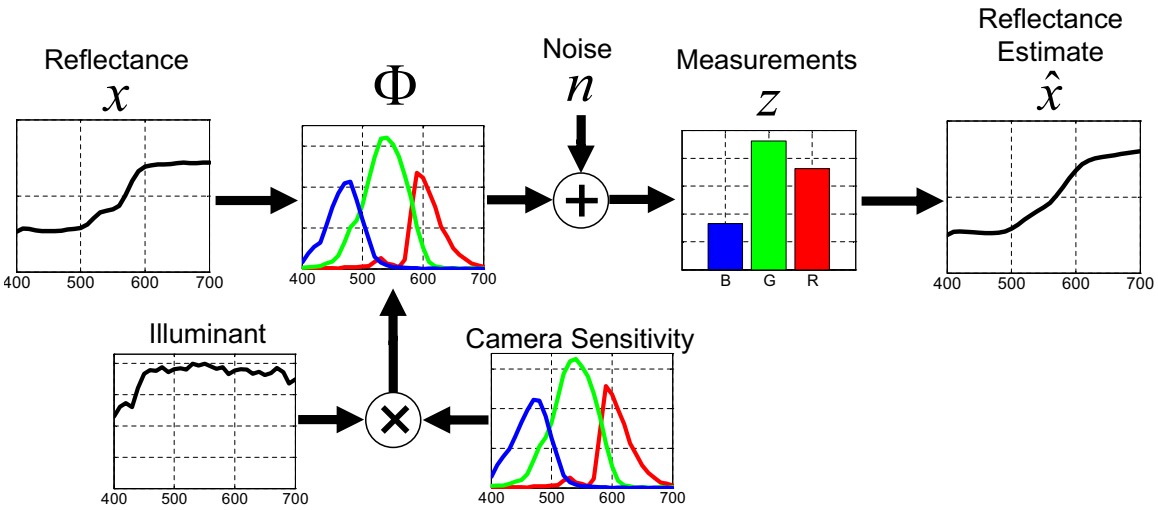


Figure 4.1: **Reflectance estimation overview.** The illuminant and spectral sensitivities of the camera are combined to make Φ, which measures the reflectance. Noise is added to produce the measurements, which are used to estimate the original reflectance.

number is desired to describe the noise level for a sensor and a set of reflectances. The overall SNR of the measurements is chosen. This value is $10log_{10}\frac{mean(Y^2)}{mean(Y)}$ expressed in decibels where mean is over all entries in the matrix.

The original reflectance function is estimated from the noisy measurements. Let $f : \mathbb{R}^m \rightarrow \mathbb{R}^o$ be the estimation function where the reflectance estimate is $\widehat{x} = f(z)$. An overview of the reflectance estimation problem is provided in Figure 4.1. In lieu of a more application specific error measure, the root mean squared error (RMSE) of the estimate, $\frac{1}{\sqrt{o}}|\widehat{x} - x|_2$, is minimized. This measure weights all chosen wavelength samples equally, which may not be appropriate for certain applications.

4.3 Visible and invisible components of reflectances

Ignoring noise, the measurement process is linear. Consider the reflectances that result in measurements of zero. Assuming the rows of Φ are linearly independent, as should occur for a well-designed measurement system, these reflectances form an $o - m$ dimensional subspace of $\mathbb{R}^o$ called the kernel of Φ. Any vector in $\mathbb{R}^o$ can be

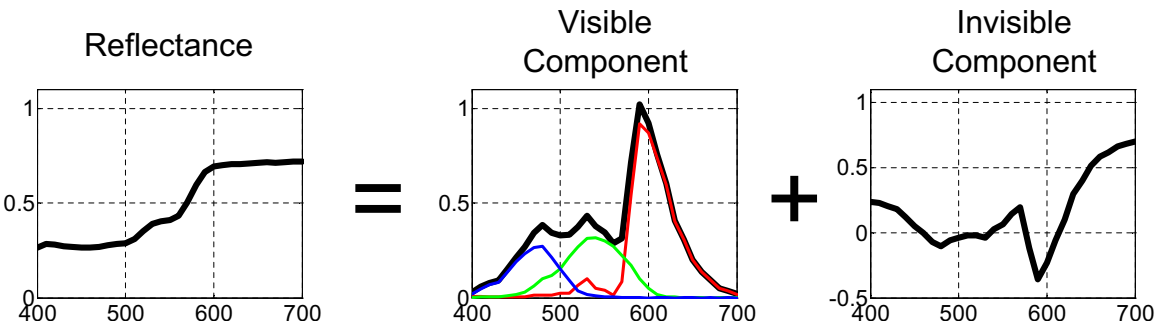


Figure 4.2: **Reflectances can be decomposed into visible and invisible components.** Each reflectance can be decomposed uniquely into the sum of visible and invisible components. The visible component lies in the subspace spanned by the rows of Φ. The red, green, and blue curves are scaled copies of the rows of Φ. The sum of the curves gives the visible component. The invisible component generates 0 measurements from the camera.

expressed uniquely as the sum of a vector that lies in this subspace and its orthogonal complement. For any $x \in \mathbb{R}^o$, define $x = v + i$ where $v^T i = 0$ and $\Phi i = 0$. The set of all possible vectors v forms an o dimensional subspace. One can find v by projecting x onto the rows of Φ,

$$v = \Phi^T(\Phi\Phi^T)^{-1}\Phi x = \Phi^+ y \tag{4.1}$$

where Φ^+ is the Moore-Penrose pseudoinverse. The v vector will be called the visible component of the reflectance because it captures all of the information the camera can obtain. The i vector is the invisible component of the reflectance because it results in no measurement and is invisible to the camera. Any reflectances that have the same visible components but different invisible components will result in the same measurements. Such objects are metamers to the camera. These terms were similarly described previously [15]. Figure 4.2 provides an example of the visible and invisible components of a reflectance.

For a noise-free measurement, y, the visible component of the reflectance can be found using (4.1). However, it is very challenging to estimate the invisible component using only the measurements. The challenge in reflectance estimation is using the measurements to estimate the invisible component based on the properties and

statistics of reflectances. Fortunately there are correlations between the measurements and the invisible component that can be exploited.

4.4 Bayesian reflectance estimation

Consider the scenario where the desired reflectance, x, is drawn with equal probability from the surfaces in the training set. In this manufactured setting, the measured reflectance must already be in the training set, which is unrealistic in practice. However, the setting offers an optimal solution that illustrates a lower bound that is not normally discernible.

With Bayes' rule, the probability each reflectance in the training set, x, was chosen given the noisy measurement, z, is:

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)}$$

Note $p(x) = 1/k$ for all x in the training set since all reflectances are equally likely in this scenario. Using the law of total probability

$$p(z) = \sum_x p(x)p(z|x) = \frac{1}{k}\sum_x p(z|x)$$

Therefore, $p(x|z) = \frac{p(z|x)}{\sum_x p(z|x)}$.

The conditional distribution $p(z|x)$ comes from the noise distribution:

$$p(z|x) = p(z = y + n) = p(n = z - y) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi y_i}} \exp\left(-\frac{(z_i - y_i)^2}{2y_i}\right)$$

Since a squared error loss metric is used, the Bayesian estimator given by the conditional expected value, $\widehat{x} = E(x|z)$, is optimal. The estimate is a weighted average of the reflectances in the training set where the weights are given by the above conditional probability. The estimate is most heavily influenced by those reflectances in the training set that have noise-free measurements close to the observed measurements. The weights are Gaussian with standard deviation in each band given by the square

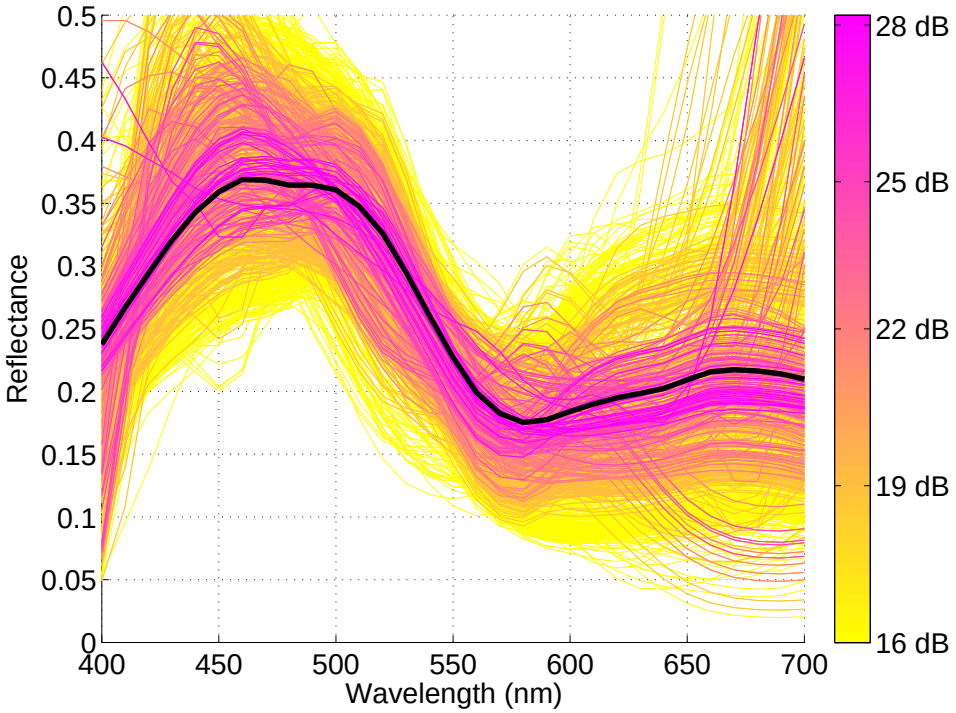


Figure 4.3: **Bayes' rule finds possible reflectances.** A particular reflectance (black) was measured with the illuminant and camera sensitivity shown in Figure 4.1. For each measurement SNR, the probability of each reflectance in the training set having generated the measurements is calculated using Bayes' Rule. Reflectances in the training set are drawn with the color corresponding to the maximum SNR where their probability is at least one-tenth the probability of the true reflectance.

root of the observed measurement. For this scenario where the measured reflectance is in the training set, there exists no reflectance estimate with a smaller expected squared error.

For low noise cases, the Bayesian estimator is able to identify the correct reflectance from the test set with high probability. The estimator's error can be made arbitrarily small by shrinking the noise. As the noise is increased, it is more difficult to select the correct reflectance. A larger set of reflectances are plausible candidates because the increased noise can explain larger deviations between the observed and noise-free measurements. Figure 4.3 provides an illustration of this phenomenon.

When noise is small enough so the true reflectance can be selected from the training

set with high probability, the Bayesian estimator is benefitting from the assumption that the reflectance is in the training set, which is unreasonable in practice. For this reason, consider the Bayesian estimator as described above but where the particular test reflectance was removed from the training data. This setup ensures there is no overlap between the training and testing sets and is more realistic. The left-out Bayesian estimator does not have any optimal performance guarantee anymore. When noise is high, the Bayesian and left-out Bayesian estimators will have similar performance but diverge as noise decreases. For noise levels where the estimators yield similar results, the Bayesian estimator gives a lower bound on estimation performance that may be achievable in practice. Unfortunately Bayesian solutions may not be practically reasonable due to the need for a large training set of reflectances and high computational requirements for each estimation.

4.5 Global linear reflectance estimation

The global linear estimator is one of the simplest and fastest. The estimate is $\widehat{x} = Wz$ where $W \in \mathbb{R}^{o \times m}$. For the squared error loss, the optimal W is the Wiener filter from (A.1):

$$W(YY^T + kR_n) = XY^T$$

The autocorrelation of the noise, R_n, is a diagonal matrix with diagonal entries y. Since y varies for each reflectance and a simple linear estimator is desired, the signal dependence is ignored. Instead the average of the measurement vectors from the training set is used for the diagonal elements of R_n. The global affine estimate is another option but is not presented because the results are generally similar. Also the linear estimator can more easily be made scale invariant as previously described.

Due to its simplicity, the global linear estimator has minimal computational requirements. However, performance of the estimator is poor if the relationship between the measurements and reflectances is highly nonlinear. As previously shown, there is a linear relationship between the noise-free measurements and the visible component

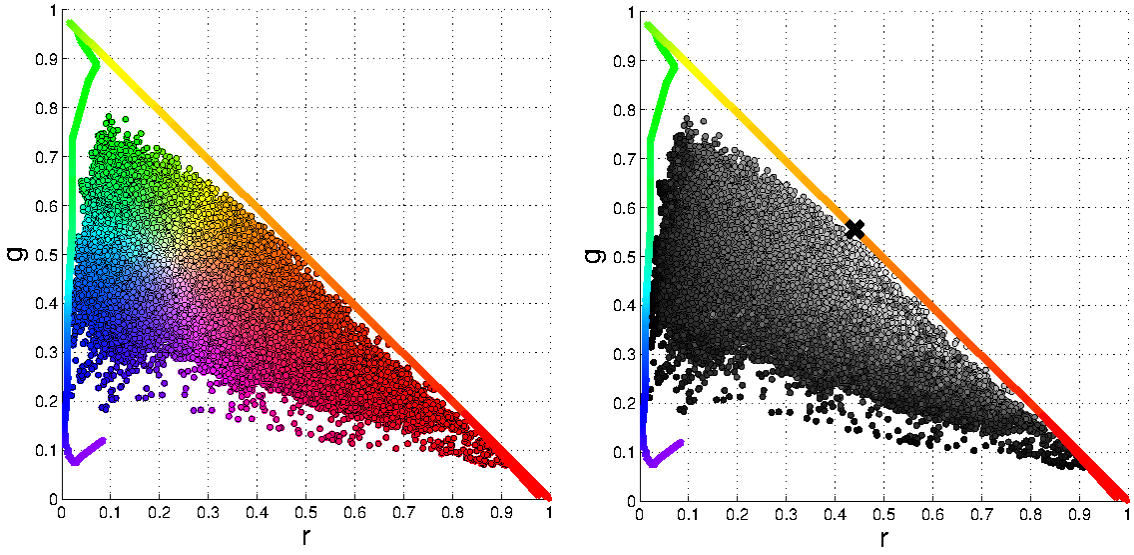
of the reflectance. However, there is no guarantee of a linear relationship between the measurements and the invisible component.

To illustrate the degree of linearity, consider a typical three color camera with measurements, R, G, and B. Since it is difficult to depict points in $\mathbb{R}^3$, consider the measurement chromaticity values given by $r = R/(R+G+B)$, $g = G/(R+G+B)$, and $b = B/(R+G+B)$. The chromaticity values factor out the overall intensity of the reflectance as measured by $R+G+B$. Since $r+g+b = 1$, only two coordinates are required to describe the color (hue and saturation) of light reflected from a surface. These measurement chromaticity coordinates, r and g are similar to the CIE x and y chromaticity coordinates for human perception [40]. These values can be plotted on a chromaticity diagram as shown in Figure 4.4.

In Figure 4.4(a), a circle has been placed at the chromaticity coordinates for each reflectance in the training set. The color of the circle is the $sRGB$ rendering with maximum brightness for that surface under the illuminant. Notice that reflectances with nearby chromaticities have similar perceptual appearances after ignoring the overall level of reflectance. This is a result of the camera sensitivities being designed to be very close to the subspace spanned by the XYZ functions for human perception.

Due to the linearity of the measurement process, a reflectance formed by the linear combination of two reflectances will have a point on the chromaticity diagram along the line connecting the chromaticity points from the two reflectances. Since reflectances are always between zero and one, all feasible reflectances must lie in the convex hull of the spectral locus. The shape of the spectral locus is heavily influenced by the camera's spectral sensitivities. For example, the spectral locus for wavelengths greater than 550 nm is along the line connecting $(r, g) = (1, 0)$ and $(0, 1)$ because the camera's blue channel has negligible sensitivity above 550 nm. Reflectances near this line have very little reflectance below 550 nm.

The linear reflectance estimate is given by $\widehat{x} = W \begin{bmatrix} R & G & B \end{bmatrix}^T$. By removing the



(a) (b)

Figure 4.4: **Global linear reflectance estimation fails for certain wavelengths.** For each reflectance in the training set, a circle is placed at the corresponding measurement chromaticity coordinates, (r, g). The curve around the data is the spectral locus of monochromatic reflectances and is colored according to the $sRGB$ representation of the light. (a) The color of each circle is the $sRGB$ rendering with maximum brightness and chromaticity equal to $D65$ reflected from the surface. (b) The gray intensity of each circle is equal to the reflectance at 580 nm divided by $R + G + B$. The black $\times$ symbol represents the location of a reflectance that is zero everywhere except at 580 nm.

brightness, $R + G + B$, the equation becomes $\frac{\widehat{x}}{R+G+B} = K \begin{bmatrix} 1 & r & g \end{bmatrix}^T$ where

$$K = W \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix}.$$

Therefore, a linear relationship between the measurements and the reflectance implies an affine relationship between the measurement chromaticity coordinates and the reflectance scaled by the sum of the measurements.

In Figure 4.4(b), the gray value for each reflectance is equal to the reflectance at

580 nm divided by $R + G + B$. Factoring out the overall brightness illustrates the suggestion from Section 4.1 of how to implement scale invariance when the illuminant intensity is unknown. Although the gray values are similar for nearby points, there is some unpredictable local variation that illustrates that perfect estimation of the reflectance is not possible using only three measurements.

Notice that points close to the $\times$ symbol have brighter gray values than more distant points. A reflectance that only reflects light at 580 nm has chromaticity coordinates at the $\times$ symbol and maximum gray value in Figure 4.4(b). A high reflectance around 580 nm increases both the R and G coordinates. Therefore, reflectances with chromaticity coordinates, (r, g), near $(0, 0)$, $(0, 1)$, or $(1, 0)$ must have very little reflectance at 580 nm. Meanwhile, reflectances near the $\times$ symbol must have almost no reflectance in the B region of the spectrum and some reflectance in both the G and R parts of the spectrum. Due to the smoothness of reflectances, these reflectances must have relatively large values around 580 nm. No affine function of r and g can fit this data well because it cannot simultaneously have large values around $(.45, .55)$ and small values around $(0, 1)$ and $(1, 0)$ because the points are collinear. The conclusion is that the global linear estimator cannot perform well at least for certain wavelengths.

Linear estimation of new measurements

Perhaps instead of estimating the reflectance, one prefers to estimate how the reflectance would appear under a certain light using a particular sensor. Specifically $\psi = \Theta x$ is desired where $\Theta \in \mathbb{R}^{m \times o}$ is the combined measurement matrix for the new illuminant and sensor.

The problem can be approached by first forming the linear reflectance estimate, $\widehat{x} = Wz$, and then simulating the noise-free measurement for the new system to give

the desired estimate as $\widehat{\psi} = \Theta\widehat{x}$. As a result,

$$\begin{aligned}\widehat{\psi} &= \Theta W z \\ &= \Theta X Y^T (YY^T + kR_n)^{-1} z \\ &= \Psi Y^T (YY^T + kR_n)^{-1} z\end{aligned}$$

where $\Psi = \Theta X$ can be viewed as a matrix of training data for the desired measurements ψ. Notice this is the form of the optimal linear estimator of $\widehat{\psi}$ given observation of z. Therefore, forming the optimal linear reflectance estimate and then simulating its noise-free measurement is equivalent to the optimal linear estimate of the desired measurement.

4.6 L^3 reflectance estimation

To improve the performance of the global linear estimator, a local linear estimator is developed. It consists of linear regressions that are fit only on reflectances from the training set that have similar measurements to the observed test measurement. The local dependence means only materials with similar reflectances to the test material are considered in the regression, which is similar to Bayesian estimation. For example, there is no need to include green materials when estimating the reflectance of a red material. This locality allows the estimator to more closely adapt to the statistics of reflectances in different regions of the measurement space. The resultant algorithm is similar to that proposed previously [90].

The design of the selection of nearby reflectances for an observed measurement should depend on the application. To limit computational and memory requirements, it may be best to divide the measurement space, $\mathbb{R}^m$, into regions where the linear filter is pre-computed and saved. Then, the appropriate region can be identified for each measurement and the corresponding filter can be applied. Alternatively, a mixture of filters from nearby regions can be combined based on the location of the measurement, which will help smooth the transitions between regions.

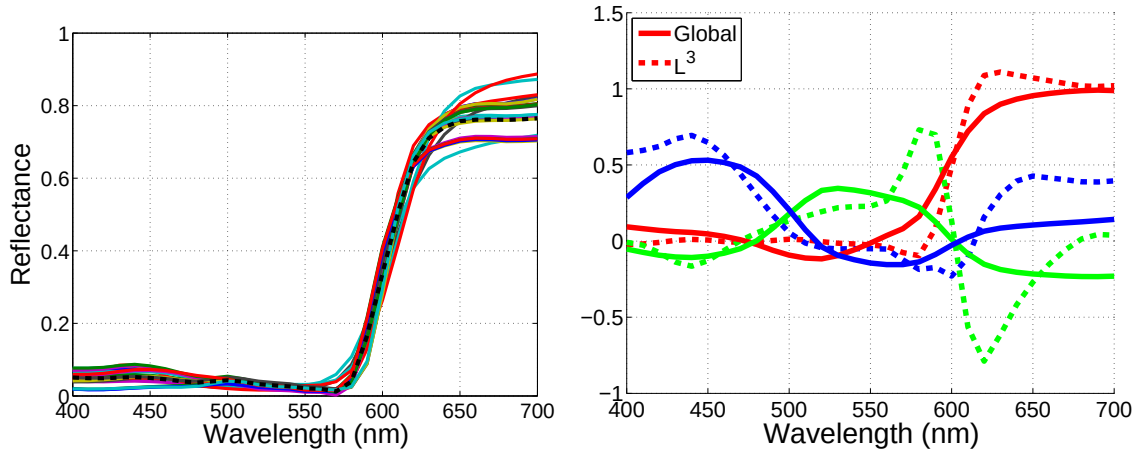
In order to avoid the complex tradeoffs in designing a specific implementation of

the algorithm, the ideal case is analyzed where the local neighborhood is calculated for each observed measurement. This requires access to the entire training set for each reflectance estimate and will have slightly improved performance over other implementations that require less memory and computation.

To select local reflectances from the training set, the squared Euclidean distance in measurement space is proposed. For an observed measurement, z, calculate the distance to each of the noise-free measurements from the training set, y, using $|z-y|_2^2$. This could be altered to weight directions in the measurement space differently to account for differences in the color channels due to the data or measurement process.

There are two possibilities for choosing the local reflectances based on this metric. One is to select all training reflectances that have measurements within a certain distance of the observed measurements, which will result in an inconsistent number of reflectances for different observed measurements. Or the number of reflectances can be fixed with the reflectances with the smallest distance chosen, which results in the distance threshold varying based on the observed measurements. The first option allows the possibility to relate the distance threshold to the noise level. However, the second option is chosen because it ensures a minimum number of reflectances are chosen for all observed measurements.

Figure 4.5 provides an illustration of the L^3 estimation for a particular reflectance. The 32 reflectances with closest simulated measurements to the observed measurement are selected as shown in Figure 4.5(a). Notice that there are slight differences among the selected reflectances but they all have a similar shape. The Wiener filter is calculated on these reflectances using the expected noise power in each channel for the observed measurements. The resultant global and local vectors are shown in Figure 4.5(b). The global and local filter vectors differ in shape due to the differences between the overall dataset and the selected set of 32 similar reflectances. The global vectors can reasonably estimate reflectances from the entire dataset. The local vectors provide poor estimates of dissimilar reflectances but can more accurately capture the differences among reflectances with similar measurements to the observed test point.



(a) (b)

Figure 4.5: L^3 **reflectance estimation is fit on similar reflectances.** (a) Local neighborhood of 32 reflectances with measurements closest to the test reflectance (black dashed line) that has chromaticity coordinates $(r, g) = (0.70, 0.19)$. (b) Global and local regression vectors (columns of W) assuming no noise. The vectors are multiplied by $R + G + B$ from the test measurement to remove the scale of the measurements. The color of the curves corresponds to the measurement band.

4.7 Results

For all results, leave-one-out cross-validation was performed. This means the estimation was repeated using each reflectance in the dataset for testing by simulating noisy measurements and trying to estimate it using a training set consisting of all remaining reflectances from the dataset. The quality of the estimate for a single reflectance is measured by the RMSE. The overall estimate quality for a dataset is the average of the RMSE values over all reflectances in the dataset. The exposure duration was varied to give a variety of noise levels for the simulations.

Although there are a number of spectral databases, there are very few with over 5,000 reflectances [91]. Large datasets are required for the Bayesian estimator to not gain an unfair advantage for reasonable noise levels by identifying the test reflectance with high probability from the dataset. The Standard Object Color Spectra (SOCS) database contains nearly 53,000 reflectances and transmittances, which is the largest publicly available set known to the author [81]. The normalized and interpolated

data in the dataset were used.

A majority of the SOCS database, 27,840 reflectances, were obtained by printing standard test charts with offset printing. The charts were printed with all combinations of ten sets of inks and three types of paper. For a single combination of inks and paper, the set of possible printed reflectances is a four dimensional surface since each reflectance is generated by mixing the cyan, magenta, yellow, and black inks. The surface is nonlinear due to the mixing of the inks. Since this set contains multiple sets of ink and paper, there is no clear restriction of the dimensionality of the reflectances.

For robustness, two smaller datasets were also tested. The first consists of 1,995 reflectances including standard test surfaces and natural objects from Simon Fraser University, which will be denoted as SF [92]. Included in this set are 1,269 Munsell chips, which were combined with three different Munsell sets from another dataset including two matte and one glossy [93]. For the calculations, all reflectances were sampled every 10 nm from 400 to 700 nm. Resampling or linear interpolation of the original data was performed as needed.

Effect of number of reflectances in L^3 estimate

Figure 4.6 shows the effect of the size of the local neighborhood on the L^3 estimate. The curves have a U shape with a clear minimum. The number of reflectances in the neighborhood with minimal error is approximately 10 and is consistent across all four datasets. Left of the minimum, too few reflectances are included, and the regressions overfit the data. For example, if only a single reflectance is included in the regression, the resultant estimate is a scaled copy of the nearest reflectance from the dataset. This estimator is not smooth enough to have good performance and relies too heavily on the single reflectance. Alternatively, including too many reflectances in the regression does not allow the estimator to adjust adequately to match the complexity of the data.

The right endpoint on each curve in Figure 4.6 corresponds to using all reflectances in the dataset for training, which yields the global linear estimator. The value of the L^3 estimator compared to the global linear estimator is given by comparing the

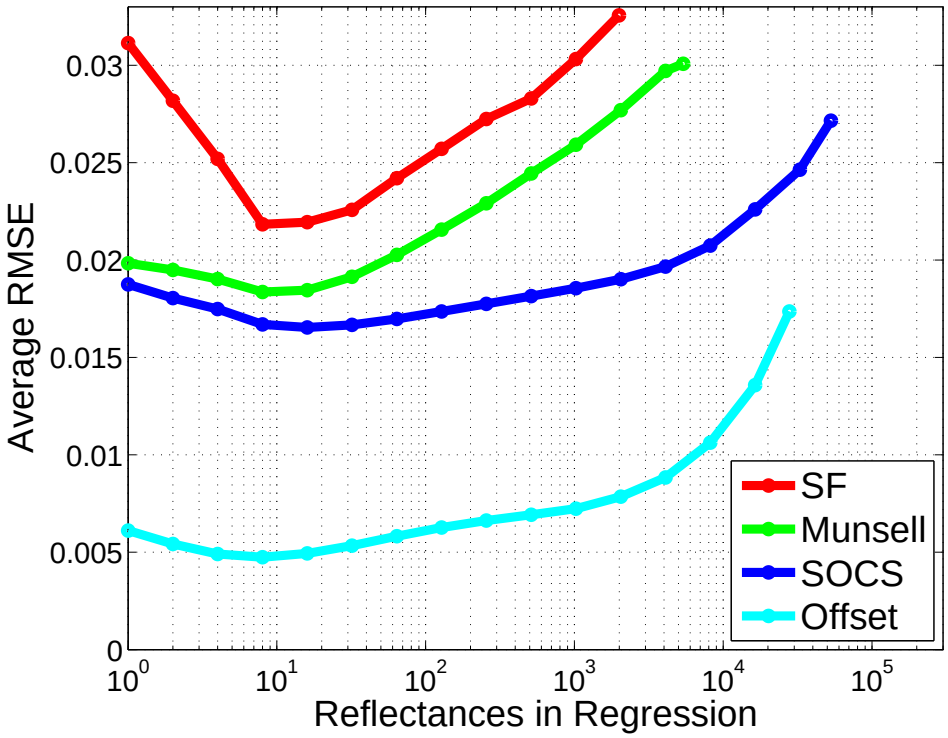


Figure 4.6: **Ideal number of reflectances in L^3 regression is consistent across datasets.** Performance of the L^3 estimate based on the number of reflectances from the training set included in the regression for each test point. The corresponding dataset for each curve is provided in the legend. Very little noise was added to give an overall SNR of approximately 45 dB.

minimum of the curves to the location of the right endpoint. The complexity and memory requirements of the L^3 estimator when implemented by pre-computing the Wiener filters increase with fewer reflectances in the local neighborhood because more regressions are required to cover the measurement space with such local precision.

When noise is increased, the curves move up and become flatter due to the increased difficulty of the estimation problem. The minimums move slightly to the right because the additional uncertainty in the test reflectance's noise-free measurements requires a larger neighborhood. The value of 32 reflectances in the regression was chosen for all further calculations because it is close to the minimum for all tested noise levels and datasets.

L^3 is most valuable for low noise and constrained datasets

The performance of the global Wiener and L^3 estimators are compared in Figure 4.7. Since the ratios are all greater than one, the L^3 algorithm is outperforming the global Wiener filter. The ratios increase as SNR increases showing the L^3 has a larger impact when noise is low. This is because noise makes estimation challenging for all algorithms, which dilutes any improvement due to better reflectance modeling.

The complexity of the set of reflectances impacts the amount of improvement of the L^3 estimator as described in Section 2.2. For example, the SOCS database contains reflectances from a large variety of natural and man-made objects. The set of offset reflectances are all designed for human vision and come from a single source, a printer. Given a measurement, there is a larger variation among the reflectances that are possible metamers to the camera if the reflectance is from the SOCS database compared to the offset subset.

The L^3 algorithm is most valuable for applications where possible reflectances are drawn from a more constrained set than from all naturally occurring reflectances. Fortunately many multispectral applications naturally consider confined datasets based on the task at hand. A scanner is a device that almost exclusively captures printed material. Consumer cameras might also be used in a mode for capturing printed material. The L^3 algorithm may reduce the error of the reflectance estimate in such devices by a factor of three. However, the improvement from the L^3 algorithm is limited for general photography applications where the set of reflectances may be very broad.

L^3 estimator is nearly optimal

Figure 4.8 compares the performance of the different estimation algorithms for various noise levels. Note that the vertical axis is a log scale so small distances for high values on the graph actually represent large performance differences. The pseudoinverse refers to the global Wiener filter with no noise. For high SNR, the noise is small so fitting for it is not very important, and there is almost no difference between the pseudoinverse and the global Wiener filter. For low SNR, the difference grows but is

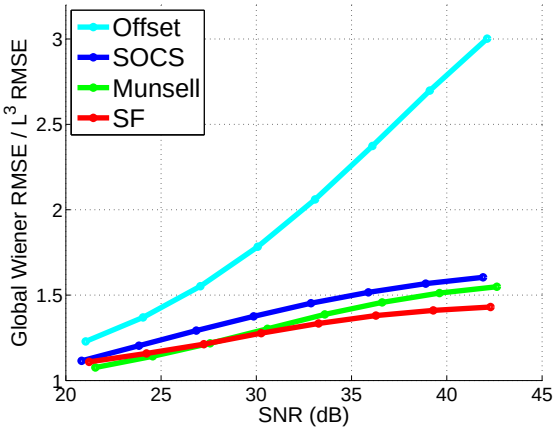


Figure 4.7: L^3 **is superior to global Wiener filter especially for restricted datasets.** The ratio of the average RMSE from the global Wiener and L^3 estimators is plotted for various SNRs. The legend indicates the dataset for each curve.

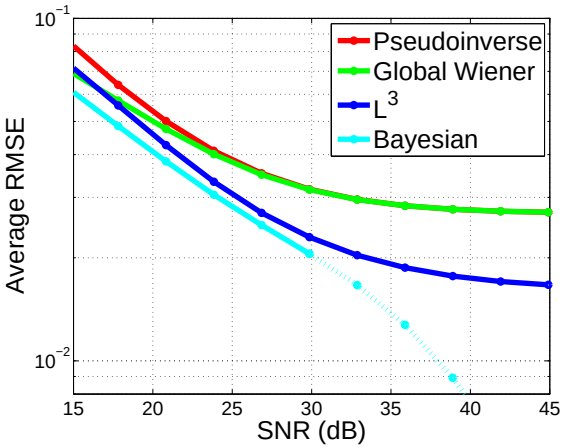


Figure 4.8: L^3 **is nearly optimal and superior to global linear estimators for high SNR.** Estimation of SOCS database. The Bayesian curve is dashed starting at the first point where the ratio of the average RMSE from the left-out Bayesian estimator and the Bayesian estimator is more than 1.2.

still modest for this RGB sensor.

As described in Section 4.4, the Bayesian estimator is optimal for this setup and reflects the unavoidable difficulty of estimating the reflectance among near metamers in the presence of noise. However, the lower bound is not realistic for high SNRs where the dataset is not dense enough to prevent the Bayesian estimator from identifying the true reflectance with high probability. The dashed part of the Bayesian curve in Figure 4.8 represents the interval where the lower bound may not be realistically achieved.

Notice that the L^3 algorithm has similar performance to the optimal Bayesian estimator over the achievable interval. There is little room for possible improvement of the L^3 algorithm at least for medium and high noise levels. It is difficult to compare the performance for low noise levels because of the necessity of even larger reflectance databases for the Bayesian estimator. It is possible that other local or nonlinear reflectance estimation algorithms may have similar performance but may differ in

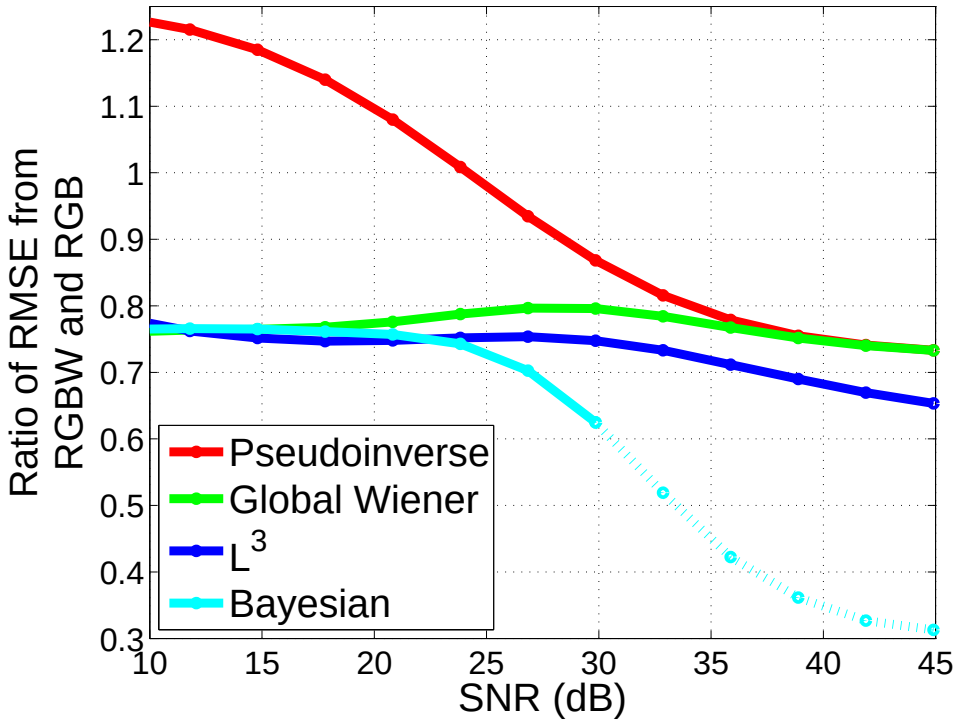


Figure 4.9: **White measurement improves estimators that account for noise.** Ratio of average RMSE of estimation from $RGBW$ and RGB sensors using the SOCS database. The plotted SNR is calculated as described in Section 4.2 but only over the RGB channels by ignoring the increased SNR of the white channel. The Bayesian curve is dashed over the same interval as in Figure 4.8.

memory and computational requirements.

Improvement due to additional white measurement

The addition of more measurement bands generally improves reflectance estimation. For the remainder of this section, the white filter shown in Figure 3.1 will be added to the RGB filters. Figure 4.9 shows the influence of the additional channel on the performance of the various reflectance estimation methods. The global Wiener, L^3, and for low SNRs the Bayesian estimators have reductions in the estimate's error by approximately 25%. For high SNRs where the Bayesian estimator may have unrealistic performance, the white channel adds a new dimension that significantly

helps identify the correct reflectance from the training set.

The pseudoinverse method, which ignores noise, significantly differs from the global Wiener filter except for high SNRs. In fact, the addition of the white measurement harms the pseudoinverse estimator for SNRs below approximately 25 dB. For these low SNRs, the pseudoinverse method can be improved by just ignoring the extra white channel. These observations show the importance of properly accounting for the noise when the measurement filters overlap such as occurs with $RGBW$. For the RGB measurements alone, the filters have little overlap, which explains why there is only a small performance difference between the pseudoinverse and global Wiener filters shown in Figure 4.8.

Figure 4.10 shows how the global Wiener vectors change for different noise levels. When SNR is high, the vectors have large positive and negative intervals. To form an estimate, the sum of scaled versions of these vectors often destructively interfere to achieve the best estimate possible. Due to the overlap of the filters, this cancellation is required to optimize the fit. However as SNR drops, the subtraction becomes problematic due to the noise power adding while the signal subtracts. This amplification of noise explains the decreased performance of the pseudoinverse estimator. Notice how the vectors become flatter as SNR decreases in order to suppress the noise. The global Wiener filter balances the noise amplification and the bias that is caused by using non-negative filters.

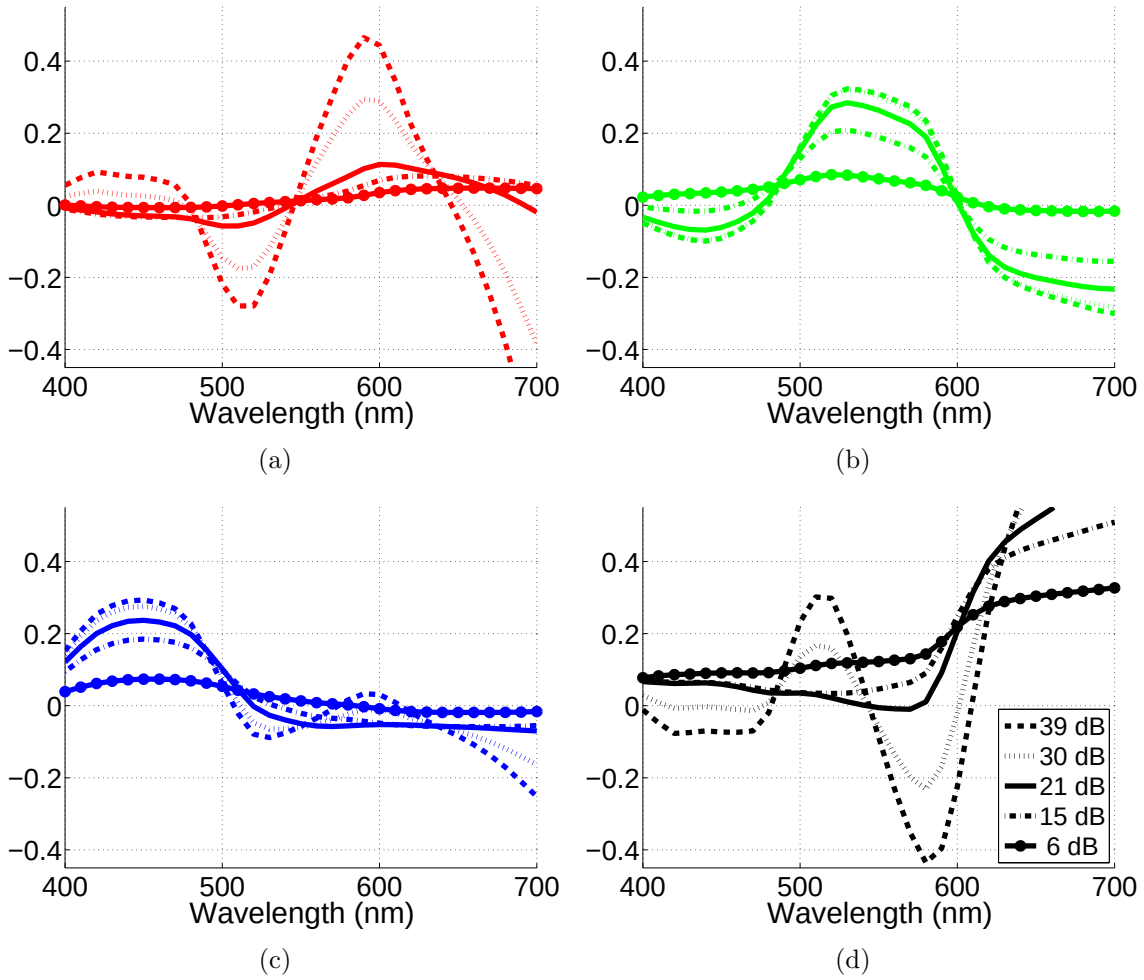


Figure 4.10: **Global Wiener vectors for $RGBW$ vary with SNR.** The regression vectors for the R, G, B, and W measurements are provided in (a), (b), (c), and (d), respectively. The vectors were found for the SOCS database and are multiplied by the average measurement in the corresponding channel so their amplitudes can be compared. For each curve, the legend in (d) provides the SNR as described in Figure 4.9.

Chapter 5

Conclusions

The local linear learned (L^3) algorithm learns how to cluster training data and finds the linear filter that minimizes the error of the estimate over the training data in each cluster. The approach is fast, robust to noise, adapts to the particulars of a dataset, and generalizes to a large variety of problems.

The L^3 pipeline has the ability to process a noisy image from a camera sensor with any CFA design and simultaneously perform demosaicking, denoising, and color transformation to estimate a high quality image in any output color space. Patches from the sensor image are classified as belonging to flat or texture regions. Texture patches are oriented into a canonical form and may be further subdivided. Then, the output estimate at the center of the patch is found by applying pre-computed Wiener filters that are based on the patch's cluster and the luminance of the scene. The Wiener filters are optimized for reducing noise while preserving the signal. The L^3 pipeline's ability to automatically perform illuminant correction and deblurring was also presented.

The L^3 pipeline enables the use of novel CFA designs, which are a promising technology due to the huge number of pixels on modern sensors. Illustrated examples include CFAs with white pixels for low light imaging and six-channel CFAs for multispectral imaging from a single capture. With software to perform camera simulations, the L^3 pipeline enables designers to rapidly test and improve novel CFA designs. There is an exciting possibility of designing application specific sensors and

processing pipelines that can leverage the particular properties of the scenes encountered by application specific devices.

The L^3 approach can also be applied to estimating a reflectance given a few measurements of the reflected light assuming a known illuminant. The L^3 estimator has nearly optimal performance, which is limited by the amount of noise and variation of the reflectances in the dataset. For low noise and constrained datasets that are common in specific applications, the L^3 algorithm offers significant improvement over the global linear estimator and has fast computation. The global Wiener filter and L^3 algorithm both take the noise into account when forming the estimate, which is critical for measurements with spectral sensitivities that overlap.

5.1 Possible improvements and extensions of L^3 pipeline

Although images from the L^3 pipeline are generally of high quality, their visual appeal may be improved for noisy scenes through post-processing. Currently the output value for each pixel is independently estimated with the goal of maximizing MSE or PSNR. This approach ensures the pipeline has minimal computational requirements and can be processed in parallel. A post-processing step that further smoothes estimated values in flat regions may decrease the PSNR but make the image more pleasant for the human visual system by eliminating any noise that still exists in regions that should contain little texture.

The following are possible applications of the L^3 pipeline that might be achieved by redesigning patch clustering and/or the desired output image.

- **Dead pixel correction** - Cluster to detect dead pixels by checking if the measurement at a patch's center pixel differs greatly from the overall color estimate. The cluster's filters will be specifically optimized to estimate without using the dead pixel's measurement.

- **Adapt to pixel saturation** - If a pixel saturates due to being overexposed, the measurement is useless. Nearby pixels of the same color will probably also

saturate. A cluster could be made that detects the saturation of a channel and performs the estimation without the measurements from the saturated pixels.

- **Improved spectral estimation** - Currently the overall color in a patch is ignored when clustering because it is not helpful for general imaging. For datasets from more specific applications, the color may be very meaningful as shown in Chapter 4. Clustering could be based on the overall color so clusters contain only patches with similar colors. If the patch's overall color has a strong correlation with spatial features for a dataset, estimation may significantly improve.

- **High dynamic range imaging** - Currently images are processed based on the mean scene luminance. This allows high and low light images to be processed well, but performs poorly for images with both high and low light regions. Instead the pipeline could be based on local light levels, which would enable improved processing of high dynamic range scenes.

- **Optical correction** - By considering the optical system when processing, it may be possible to correct aberrations by training to an image captured using superior optics. Clusters could be designed for different spatial regions of the image where there may exist different geometric or chromatic errors due to lens aberrations.

- **Object classification or detection** - Cluster labeled training data in CFA space to try to separate different objects of interest such as with support vector machines. The likelihood of a test patch belonging to a certain object of interest is predicted by the percent of the training data in the same cluster that are from that object. This allows classification and identification based on the raw sensor data instead of estimated output images that are larger in file size but contain no new information.

- **Illuminant estimation using skin** - The shape of the reflectance of skin is relatively invariant across humans. With automatic detection of faces, light reflected from the skin can be measured. The chromaticity of the skin region

can then be used to estimate the scene's most likely illuminant by leveraging skin training data. [94, 95]

- **Estimation from multiple images** - Multiple images are automatically taken by the camera in fast succession and then processed into a single image. Examples include images taken with identical settings for improved SNR, multiple shutter speeds for high dynamic range imaging, and multiple focus points for all focus imaging. Patches would need to be extended to three dimensions by including measurements from each of the images.

- **User-defined custom processing** - By observing how a camera user edits photos, the processing pipeline could be adjusted to automatically make similar adjustments. For example, if the color of the sky is always made more saturated or red eye is always removed, a cluster could be learned for the relevant patches. The filters for the cluster could be optimized using the adjustments the user has made.

Appendix A

Wiener filter derivation

A.1 Setup

The goal is to use a vector of m measurements to estimate a vector of o variables. The training data consists of noise-free measurements, $Y \in \mathbb{R}^{m \times k}$, and the data to be estimated, $X \in \mathbb{R}^{o \times k}$. The columns of these matrices represent k different observations. Here $m < o$ so perfect estimation is generally impossible.

Although Y is assumed to be noise-free for training, the measurements are expected to be corrupted by noise when testing. The estimator should be optimized for this expected measurement noise. Let the noise be given by $N \in \mathbb{R}^{m \times k}$, where each column is independent and identically distributed with mean 0 and autocorrelation R_n. The rows of N may have different distributions since they correspond to different types of measurements such as different colorbands. Assume N is independent with X and Y. The task is to estimate X as a linear function of the measurements $Y + N$ using knowledge of X, Y, and R_n.

A.2 Linear estimator

The Wiener filter $W \in \mathbb{R}^{o \times m}$ minimizes the expected value of the sum of the squares of the errors between the desired output and the estimate, $\widehat{X} = W(Y+N)$. Specifically, it minimizes $E\left|\widehat{X} - X\right|_F^2$ where $|A|_F = \sqrt{tr(A^T A)} = \sqrt{\sum_{i,j} a_{i,j}^2}$ is the Frobenius

norm and tr is the trace operator. The objective function can be simplified as follows

$$
\begin{aligned}
&E\left|W(Y+N)-X\right|_F^2 \\
&\quad = E|(WY-X)+WN|_F^2 \\
&\quad = E\left[tr\left(((WY-X)+WN)^T((WY-X)+WN)\right)\right] \\
&\quad = E\left[tr\left((WY-X)^T(WY-X)\right)+tr\left((WN)^T(WN)\right)+2tr\left((WY-X)^TWN\right)\right] \\
&\quad = tr\left((WY-X)^T(WY-X)\right)+E\left[tr((WN)^T(WN))\right]
\end{aligned}
$$

where the last term was dropped because N is 0 mean and independent of X and Y. To find the minimizing W, differentiate the above expression using matrix calculus and the identity $\frac{\partial}{\partial W}tr(A^TA) = 2A\frac{\partial A}{\partial W}$ [96]. The first term simplifies as follows.

$$
\begin{aligned}
\frac{\partial}{\partial W}tr\left((WY-X)^T(WY-X)\right) &= 2(WY-X)\frac{\partial}{\partial W}(WY-X) \\
&= 2(WY-X)Y^T
\end{aligned}
$$

Now for the second term.

$$
\begin{aligned}
\frac{\partial}{\partial W}E\left[tr((WN)^T(WN))\right] &= E\left[2WN\frac{\partial}{\partial W}(WN)\right] \\
&= E\left[2WNN^T\right] \\
&= 2WkR_n
\end{aligned}
$$

Now put the terms together and equate to 0.

$$
\begin{aligned}
2(WY-X)Y^T + 2WkR_n &= 0 \\
WYY^T - XY^T + WkR_n &= 0 \\
W(YY^T + kR_n) &= XY^T
\end{aligned} \tag{A.1}
$$

Solving under the assumption that the inverse exists gives $W = XY^T(YY^T+kR_n)^{-1}$, which is the optimal linear filter. For calculations, it is more efficient to solve the

linear system in (A.1) using standard linear algebra methods to avoid the calculating the matrix inverse. Note that if there is no noise, $R_n = 0$, the optimal filter becomes $W = XY^T(YY^T)^{-1} = XY^+$ where Y^+ is the Moore-Penrose pseudoinverse.

SVD analysis

To offer some insight into the Wiener filter, consider the singular value decomposition of Y. Let $Y = U\Sigma V^T$ where U and V are orthogonal matrices and Σ is a rectangular diagonal matrix with diagonal elements s_i for $1 \leq i \leq m$. Equation (A.1) can be rewritten as

$$\begin{aligned} W(U\Sigma V^T V\Sigma^T U^T + kR_n) &= XV\Sigma^T U^T \\ W(U\Sigma\Sigma^T U^T + kR_n) &= XV\Sigma^T U^T \\ W(U\Sigma\Sigma^T + kR_n U) &= XV\Sigma^T \\ WU(\Sigma\Sigma^T + U^T kR_n U) &= XV\Sigma^T \\ W &= XV\Sigma^T(\Sigma\Sigma^T + U^T kR_n U)^{-1}U^T \end{aligned}$$

Consider what would happen if the rows of N are independent and identically distributed. This would occur if the noise is equal for all measurement entries. Then, R_n is a diagonal matrix. Let the diagonal elements of R_n be σ^2. As a result, $D = \Sigma^T(\Sigma\Sigma^T + U^T kR_n U)^{-1}$ is a diagonal matrix with diagonal elements $s_i/(s_i^2 + k\sigma^2)$. The optimal filter is given by $W = XVDU^T$.

When s_i is large relative to $k\sigma^2$, the diagonal elements are approximately $1/s_i$. If instead, s_i is small relative to $k\sigma^2$, the diagonal elements are approximately 0. This results in not using measurements corresponding to singular vectors where the noise is much stronger than the signal. If the linear estimator was found without anticipating measurement noise, the solution would have the same form but the diagonal elements of D would be $1/s_i$. For small singular values, these reciprocals are poorly conditioned and unstable in the presence of noise. This is why the optimal procedure for noisy measurements is to use the derived diagonal elements, which are more robust to noise.

Random vector framework

The previous derivation applies to a linear regression over a large number of objects in the training data. Now consider the problem where $x \in \mathbb{R}^o$ and $y \in \mathbb{R}^m$ are random vectors that are drawn uniformly from the training data by randomly selecting corresponding columns from X and Y. Similarly, let $n \in \mathbb{R}^m$ have the same distribution as the columns of N. If the goal is for the estimate $\widehat{x} = W(y + n)$ to have minimum squared error, $E\left[(x - \widehat{x})^2\right]$, the solution is given by (A.1). This vector notation is now adopted since it more closely reflects the intended use of the Wiener filter.

A.3 Affine estimator

For signals with non-zero mean, the linear estimator may be improved by forming an affine estimate $\widehat{x} = K(y + n) + b$ where $K \in \mathbb{R}^{o \times m}$ and $b \in \mathbb{R}^o$. Again the objective is to minimize the sum of the squared errors of the estimate. The optimal vector b is derived by forcing the mean of the estimate to agree with the signal mean

$$\begin{aligned} \overline{x} &= E[\widehat{x}] \\ &= E\left[K(y + n) + b\right] \\ &= K\overline{y} + b \end{aligned}$$

where $\overline{x} \in \mathbb{R}^o$ and $\overline{y} \in \mathbb{R}^m$ is the expected value of the vectors x and y, respectively, which are calculated from the training data. The optimal value is given by $b = \overline{x} - K\overline{y}$. Therefore, the estimate is

$$\begin{aligned} \widehat{x} &= K(y + n) + \overline{x} - K\overline{y} \\ &= K((y - \overline{y}) + n) + \overline{x} \end{aligned}$$

Let $\widehat{x}_0 = \widehat{x} - \overline{x}$ and $y_0 = y - \overline{y}$, so the above equation can be rewritten as the following.

$$\widehat{x}_0 = K(y_0 + n)$$

Using (A.1), gives

$$K = X_0 Y_0^T (Y_0 Y_0^T + kR_n)^{-1} \tag{A.2}$$

where X_0 and Y_0 are the training matrices, X and Y, after removing their column mean. This can be thought of as removing the mean from the measurements, forming a linear estimate, and then adding the mean of the desired output. Of course the optimal W for a linear estimate generally differs from the optimal K for an affine estimate. Depending on the particular dataset, the affine offset may or may not increase the accuracy of the estimation.

Appendix B

Clustering of texture patches

B.1 Motivation

Using a single texture cluster, if a patch contains an edge it is not possible to determine whether the edge straight/curved or sharp/soft. By subdividing texture patches and learning optimal filters for each cluster, the L^3 pipeline can become more adaptive to particular features in the sensor images. For example, with a cluster only for corners, the filters can be optimized to detect corners and generate them in the estimated image. Although clustering for specific tasks can enable interesting applications as mentioned in Chapter 5, in this appendix clustering will be to achieve improved estimation accuracy for the current simulation setup. In general, the improvements are modest.

There are a number of important requirements for the clustering process. First, all patches in a cluster should be similar enough so that a single linear filter may exist that can offer reliable estimation. Of course this is not a sufficient requirement because it is satisfied by having a huge number of clusters, which enables very adaptive filtering. But the required amount of training data scales with the number of clusters in order to avoid overfitting. Trying to limit the number of clusters also keeps memory and computation requirements at a reasonable level. There also is less risk of misidentifying a patch's cluster due to noise when there are a smaller number of clusters. Finally, the procedure for identifying a patch's cluster needs to require

minimal computation since it is performed on all texture patches.

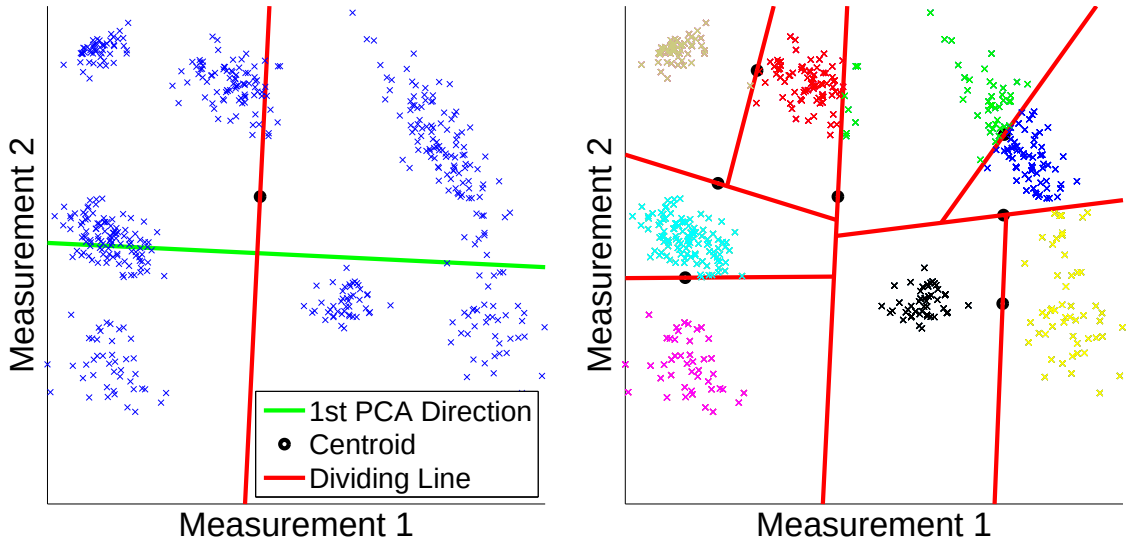
B.2 Clustering procedure

For minimal computation, a top-down hierarchical clustering method is employed. Initially all oriented residual texture patches are in a single cluster. Then, the cluster is split into two clusters of approximately equal size. The splitting continues until the desired number of clusters is achieved. For simplicity, all branches of the tree terminate at the same level, however it may be advantageous to only subdivide the clusters with high errors.

The advantage of the tree approach is that a patch's cluster can be identified out of 2^t possible leaf clusters by requiring only t branching decisions. The number of branching decisions required to reach one of the leafs will be referred to as the tree height. For speed and simplicity, the branching operation is restricted to the calculation of an inner product between the patch and a pre-determined filter, called the texture branching filter. The patch is placed into one of the two smaller clusters by comparing the inner product to a pre-determined threshold, called the texture branching threshold.

In order to reduce the variation among the patches in each of the smaller clusters, the texture branching filter is the first PCA (principal component analysis) vector of the patches in the cluster. This vector is in the direction of maximum variation among the patches. The two resultant clusters each have a smaller variation in this direction than the parent cluster. The use of PCA is also motivated by the fact that it is the continuous solution of the cluster assignment vector for the K-means cluster objective for two clusters [97]. This continuous variant of K-means clustering may be more appropriate here because points near the boundary between the two clusters risk being misclassified due to noise. The chosen texture branching threshold is the mean value of the first PCA component of all patches in the original cluster. This results in an approximately equal number of patches in each of the smaller clusters.

A manufactured example illustrating the hierarchical clustering procedure is provided in Figure B.1. Using just three branching operations, the data has been divided



(a) (b)

Figure B.1: **Hierarchical clustering quickly identifies clusters.** (a) Each of the blue points represent an observation that is in the cluster to be divided. The dividing line between the two new clusters is normal to the first PCA direction and passes through the cluster's centroid. The two new clusters contain all points on each side of the dividing line. (b) The original cluster has been divided into eight clusters by repeated division. All points in each cluster are drawn in the same color.

into eight relatively compact clusters. For graphical simplicity, each observation consists of two scalar measurements. As a result, one dimensional dividing lines form the boundary between the two new clusters. For the pipeline, the dimension of the data is the number of pixels in each patch, m, and the boundary between the two new clusters is a hyperplane of dimension $m-1$ that is normal to the first PCA direction.

A number of modifications of the clustering were tested. For example, the described texture clustering is based on Euclidean distance between patches. This might be improved by centrally weighting the patches to account for the fact that pixels near the center of the patch are more important for estimation as evidenced by the centrally weighted Wiener filters. Alternatively, including the overall color values in the clustering to separate patches of different colors could reduce the variation in the noise power across the patches in the cluster. Or perhaps normalizing the contrast across texture patches before clustering could develop filters that specifically reflect

the geometry of the patch. These alternate clustering methods offered only minor improvements for the tested datasets but might be helpful in other settings.

B.3 Results

Figure B.2 illustrates the quality improvement of an estimated image due to clustering of texture patches. The estimates for a test image generally improve in quality with more clusters for moderate tree heights, but the gains are typically small. Of course, there is a limit to the performance increases with larger tree heights because exponentially more data is required to have sufficient training patches in each cluster to avoid overfitting. This explains the decrease in the curves around tree height 10 for this dataset.

When noise or tree height are large, there is a risk that in practice the wrong cluster will be identified for a texture patch due to noise. Since the filters are optimized assuming patches are placed in the proper cluster, the result can be a loss in performance. This is why the curve for 9 cd/m^2 decreases for large tree height. For darker scenes, the texture clustering can result in worse images for any positive tree height due to the difficulty of clustering under such low signal to noise ratio.

Since the number of clusters and filters grows exponentially with the tree height, there is a point where the slight improvement in quality does not justify the increase in computation and memory requirements. It is possible that for other datasets that might have sharper edges or different image features, improvements may be larger.

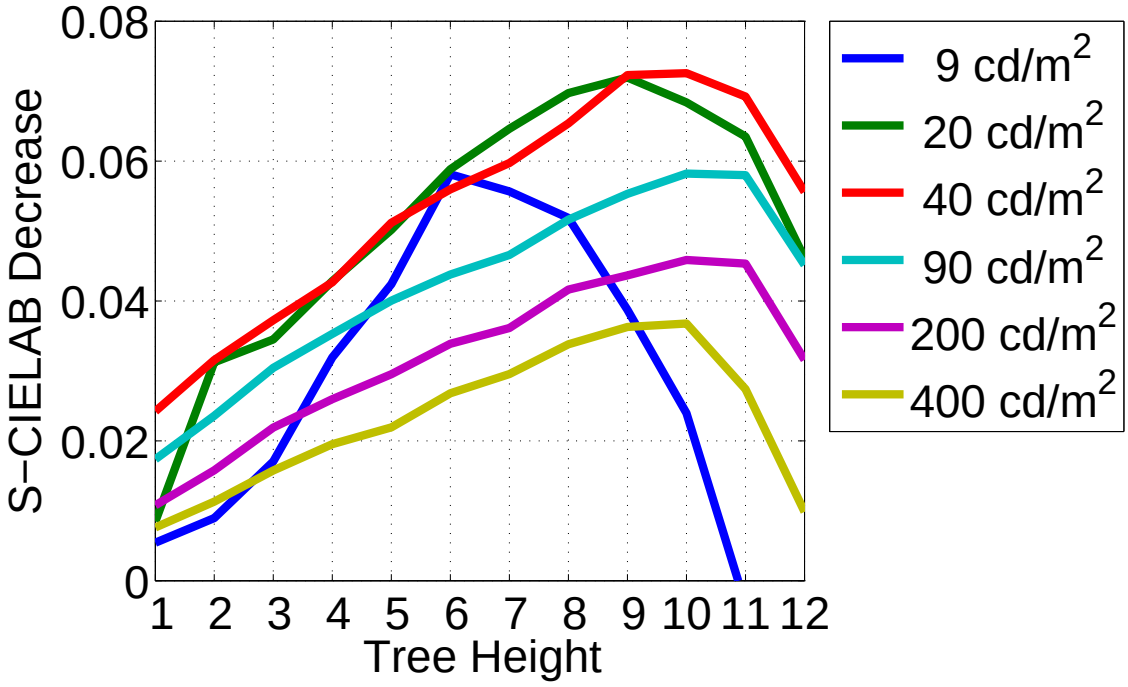


Figure B.2: **Texture clustering offers slight performance improvements.** Decrease in S-CIELAB metric caused by clustering texture patches with a certain tree height compared to a single cluster for texture patches. Scenes were simulated with mean scene luminance provided in the legend, sampled with the Bayer pattern, and estimated with the L^3 pipeline. The same test and training images were used as in Figure 3.9. Patches were 9×9 pixels with 80% of training patches classified as flat.

Appendix C

PCA subspace reflectance estimation

A common approach to compactly representing reflectances is to store only a few coefficients of a reflectance in a particular basis while ignoring any other coefficients that are deemed to have little information. The most common basis comes from principal component analysis (PCA), which is a common tool in color science [98]. The first PCA vector is in the direction of most variance in the data. Each subsequent vector is in the direction of maximum variance that is orthogonal to the previously selected vectors. Figure C.1 shows the first six PCA vectors for the SOCS dataset [81] where 99.9% of the energy in the reflectances is contained in the subspace spanned by these vectors. Note that the number of zero crossings increases by one for each subsequent PCA vector, which illustrates that reflectances vary slowly over wavelength.

A reflectance estimate can be formed by finding the reflectance that lies in the subspace spanned by the first few PCA components and most closely matches the observed measurements. In other words, the estimate is the point in the subspace that projects onto the observed measurements. Noise is not typically considered in these estimation methods except possibly when selecting the number of PCA vectors in the subspace.

There are two common PCA subspace estimation procedures [99]. One is a linear estimation in the subspace spanned by the chosen PCA vectors. The other approach

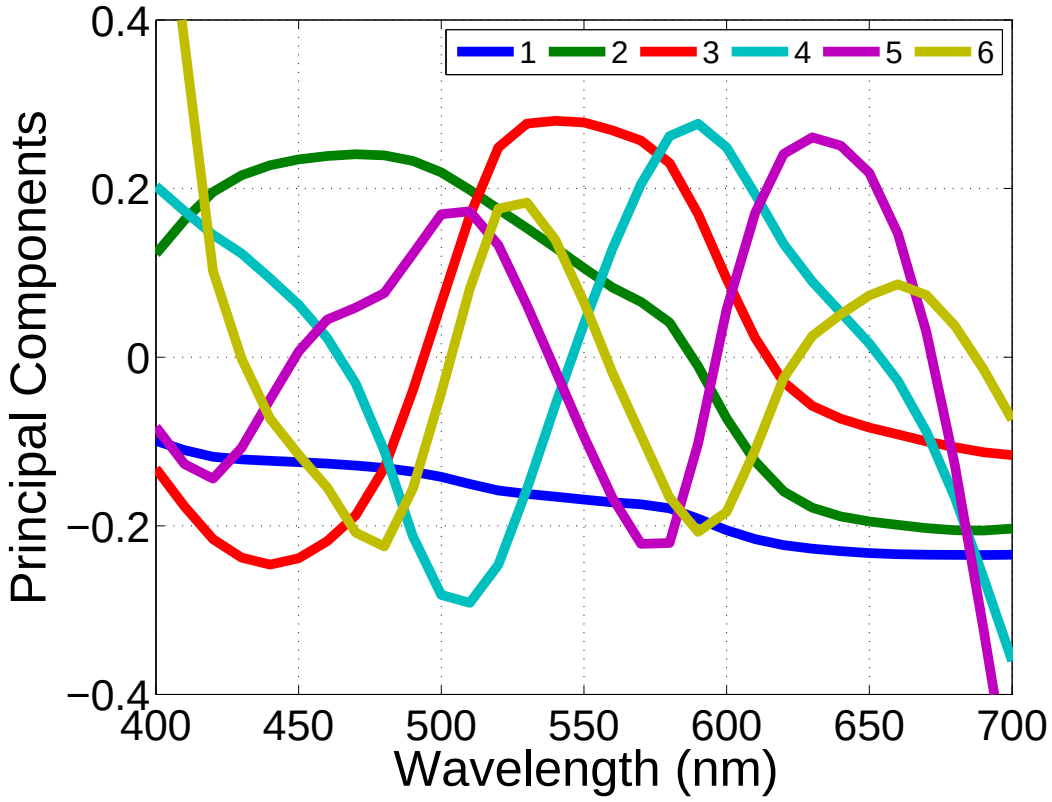


Figure C.1: **Principal components of reflectances.** PCA vectors are calculated from SOCS dataset [81]. The legend gives the order of the vectors arranged in decreasing order of the associated singular value.

is an affine estimation where the deviation from the mean reflectance is in the PCA subspace. The affine approach is analyzed under the assumption of no noise since this estimate is motivated by the properties of the PCA basis.

To find the PCA basis from a set of k reflectances, $X \in \mathbb{R}^{o\times k}$, average the columns to find the mean reflectance, $\overline{x} \in \mathbb{R}^o$. Subtract the mean reflectance from each of the training reflectances to form $X_0 \in \mathbb{R}^{o\times k}$. The PCA basis is given by the columns of $U \in \mathbb{R}^{o\times o}$ in the singular value decomposition $X_0 = U\Sigma V^T$ where U and V are orthogonal matrices and Σ is a rectangular diagonal matrix. Note these matrices are different than those introduced in Section A.2. Let $P = U\Delta$ where $\Delta \in \mathbb{R}^{o\times q}$ contains ones along the main diagonal and zeros elsewhere. Here q is the number of PCA vectors used. Now the columns of P are the first few principal components that

form a basis for the linear subspace. The estimate has the form $\widehat{x} = P\widehat{\theta} + \overline{x}$ where $\widehat{\theta} \in \mathbb{R}^q$ are the estimated PCA coefficients.

Using the notation of Chapter 4, the observed measurements for a reflectance are $z \in \mathbb{R}^m$. Assuming no noise, the measurements of a reflectance are given by Φx where $\Phi \in \mathbb{R}^{m \times o}$. For the estimate, find θ to minimize the squared error between the observed measurements and the noise-free measurements that would be observed by measuring the estimate. This can be expressed as

$$\begin{aligned} \|\Phi\widehat{x} - z\|_2^2 &= \|\Phi P\widehat{\theta} - (z - \Phi\overline{x})\|_2^2 \\ &= \|\Phi P\widehat{\theta} - z_0\|_2^2 \end{aligned}$$

where $z_0 = z - \Phi\overline{x}$. The solution is found using the Moore-Penrose pseudoinverse

$$\begin{aligned} \widehat{\theta} &= (\Phi P)^+ z_0 \\ &= (\Phi P)^T (\Phi P (\Phi P)^T)^{-1} z_0 \\ &= P^T \Phi^T (\Phi P P^T \Phi^T)^{-1} z_0 \\ &= \Delta^T U^T \Phi^T (\Phi U \Delta \Delta^T U^T \Phi^T)^{-1} z_0 \end{aligned}$$

where the existence of the inverse was assumed. Now the estimate is

$$\widehat{x} = U\Gamma U^T \Phi^T (\Phi U \Gamma U^T \Phi^T)^{-1} z_0 + \overline{x}$$

where $\Gamma = \Delta\Delta^T$ is a square matrix with zero entries except for ones on the first q entries of the main diagonal. Using the notation of Section A.3, this can be expressed as

$$\widehat{x}_0 = U\Gamma U^T \Phi^T (\Phi U \Gamma U^T \Phi^T)^{-1} z_0 \tag{C.1}$$

From (A.2) the optimal affine estimate for no noise is given by the pseudoinverse

$$\begin{aligned}
\widehat{x}_0 &= X_0 Y_0^T (Y_0 Y_0^T)^{-1} z_0 \\
&= X_0 X_0^T \Phi^T (\Phi X_0 X_0^T \Phi^T)^{-1} z_0 \\
&= U\Sigma V^T V \Sigma^T U^T \Phi^T (\Phi U \Sigma V^T V \Sigma^T U^T \Phi^T)^{-1} z_0 \\
&= U\Sigma\Sigma^T U^T \Phi^T (\Phi U \Sigma\Sigma^T U^T \Phi^T)^{-1} z_0 \\
&= USU^T \Phi^T (\Phi USU^T \Phi^T)^{-1} z_0 \qquad \text{(C.2)}
\end{aligned}$$

where $S = \Sigma\Sigma^T$ is a square diagonal matrix with the diagonal entries given by the squares of the singular values of X_0.

Comparing (C.1) and (C.2) shows the similarity of the PCA subspace and affine least squares estimators assuming no noise. The PCA subspace estimation forces the estimate to lie strictly in the first few PCA directions, while the least squares estimate allows a small amount of energy outside this subspace. The least squares estimates have as much energy in each PCA direction as exists in the training set of reflectances. The PCA subspace method allows the estimate to reside anywhere in the subspace without any preference for placing more energy in the first PCA components than the later allowable PCA components, even though the training data has this feature.

The PCA subspace estimate results in a higher squared error over the training set compared to the least squares estimator. Another important distinction between the PCA and least squares estimators is that the PCA method requires knowledge of the measurement system, Φ. Characterizing the system in practice requires extra measurements and calculations. On the other hand, the least squares estimator does not require any such characterization. PCA subspace methods allow the estimate to be expressed more compactly, which can reduce computational and memory requirements.

Bibliography

[1] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar, "Generalized assorted pixel camera: Postcapture control of resolution, dynamic range, and spectrum," *Image Processing, IEEE Transactions on*, vol. 19, no. 9, pp. 2241–2253, 2010.

[2] B. E. Bayer, "Color imaging array," July 20, 1976, United States patent number 3971065.

[3] M. Parmar and B. A. Wandell, "Interleaved imaging: an imaging system design inspired by rod-cone vision," in *Digital Photography V*, B. G. Rodricks and S. E. Susstrunk, Eds., vol. 7250. SPIE, January 18, 2009, p. 725008.

[4] K. Hirakawa and P. J. Wolfe, "Spatio-spectral color filter array design for optimal image recovery," *Image Processing, IEEE Transactions on*, vol. 17, no. 10, pp. 1876–1890, 2008.

[5] M. Kumar, E. O. Morales, J. E. Adams, and W. Hao, "New digital camera sensor architecture for low light imaging," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, 2009, pp. 2681–2684.

[6] E. B. Gindele and A. C. Gallagher, "Sparsely sampled image sensing device with color and luminance photosites," Nov 5, 2002, United States patent 6476865.

[7] J. F. Hamilton and J. T. Compton, "Processing color and panchromatic pixels," February 1, 2007, United States patent 2007/0024879 A1.

[8] T. Kijima, H. Nakamura, J. T. Compton, and J. F. Hamilton, "Image sensor with improved light sensitivity," March 30, 2010, United States patent 7688368.

[9] I. Sato, K. Ooi, K. Saito, Y. Takemura, and T. Shinohara, "Color image pick-up apparatus," June 28, 1983, United States patent 4390895.

[10] G. Susanu, S. Petrescu, F. Nanu, A. Capata, and P. Corcoran, "RGBW sensor array," July 2, 2009, United States patent 2009/0167893.

[11] T. Yamagami, T. Sasaki, and A. Suga, "Image signal processing apparatus having a color filter with offset luminance filter elements," June 21, 1994, United States patent 5323233.

[12] J. Campbell, *Introduction to Remote Sensing.* The Guilford Press, November 1, 2006.

[13] B. Bhanu and I. Pavlidis, *Computer Vision Beyond the Visible Spectrum.* Springer, November 12, 2004.

[14] M. Yamaguchi, H. Haneishi, and N. Ohyama, "Beyond red–green–blue (RGB): Spectrum-based color imaging technology," *Journal of Imaging Science and Technology*, vol. 52, no. 1, p. 010201, January-February 2008.

[15] J. E. Farrell, J. Cupitt, D. Saunders, and B. A. Wandell, "Estimating spectral reflectances of digital artwork," in *Chiba Conference on Multispectral Imaging*, 1999.

[16] S. Tominaga and N. Tanaka, "Spectral image acquisition, analysis, and rendering for art paintings," *Journal of Electronic Imaging*, vol. 17, no. 4, p. 043022, October - December 2008.

[17] J. Blasco, N. Aleixos, J. Gómez, and E. Moltó, "Citrus sorting by identification of the most common defects using multispectral computer vision," *Journal of Food Engineering*, vol. 83, no. 3, pp. 384–393, December 2007.

[18] T. Brosnan and D.-W. Sun, "Improving quality inspection of food products by computer vision a review," *Journal of Food Engineering*, vol. 61, no. 1, pp. 3–16, January 2004.

[19] O. Carrasco, R. B. Gomez, A. Chainani, and W. E. Roper, "Hyperspectral imaging applied to medical diagnoses and food safety," in *Geo-Spatial and Temporal Image and Data Exploitation III*, N. L. Faust and W. E. Roper, Eds., vol. 5097. SPIE, August 18, 2003, pp. 215–221.

[20] Y. Garini, I. T. Young, and G. McNamara, "Spectral imaging: Principles and applications," *Cytometry Part A*, vol. 69A, no. 8, pp. 735–747, 2006.

[21] L. Zhou and W. S. El-Deiry, "Multispectral fluorescence imaging," *The Journal of Nuclear Medicine*, vol. 50, no. 10, pp. 1563–1566, October 1, 2009.

[22] W. J. Cukierski, X. Qi, and D. J. Foran, "Moving beyond color: The case for multispectral imaging in brightfield pathology," in *Biomedical Imaging: From Nano to Macro, 2009. ISBI '09. IEEE International Symposium on*, 2009, pp. 1111–1114.

[23] M. Yamaguchi, M. Mitsui, Y. Murakami, H. Fukuda, N. Ohyama, and Y. Kubota, "Multispectral color imaging for dermatology: application in inflammatory and immunologic diseases," in *Thirteenth Color Imaging Conference: Color Science and Engineering Systems, Technologies, and Applications*, November 2005, pp. 52–58.

[24] M. Yamaguchi, J. Kishimoto, Y. Komiya, Y. Kanno, Y. Murakami, H. Hiroyuki, R. Yamada, K. Miyajima, and H. Haneishi, "Video-based telemedicine with reliable color: field experiments of natural vision technology," in *Proceedings of the 3rd International Universal Communication Symposium*, ser. IUCS '09. New York, NY, USA: ACM, 2009, pp. 150–153.

[25] F. Vagni, "Survey of hyperspectral and multispectral imaging technologies," NATO Research and Technology Organization, Tech. Rep. TR-SET-065-P3, May 2007.

[26] J. Brauers and T. Aach, "A color filter array based multispectral camera," in *Workshop Farbbildverarbeitung*, German Color Group, Ed., Ilmenau, October 5-6, 2006.

[27] Y. M. Lu, C. Fredembach, M. Vetterli, and S. Susstrunk, "Designing color filter arrays for the joint capture of visible and near-infrared images," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, 2009, pp. 3797–3800.

[28] L. Miao and H. Qi, "The design and evaluation of a generic method for generating mosaicked multispectral filter arrays," *Image Processing, IEEE Transactions on*, vol. 15, no. 9, pp. 2780–2791, 2006.

[29] R. Shrestha, J. Y. Hardeberg, and R. Khan, "Spatial arrangement of color filter array for multispectral image acquisition," in *Sensors, Cameras, and Systems for Industrial, Scientific, and Consumer Applications XII*, R. Widenhorn and V. Nguyen, Eds., vol. 7875. SPIE, February 10, 2011, p. 787503.

[30] L. Condat, "A new color filter array with optimal properties for noiseless and noisy color image acquisition," in *Image Processing, 2009 IEEE International Conference on*, November 7-10, 2009.

[31] Y. Li, P. Hao, and Z. Lin, "Color filter arrays: A design methodology," Department of Computer Science, Queen Mary, University of London, Mile End Road, London E1 4NS, UK, Tech. Rep., May 2008.

[32] Y. M. Lu and M. Vetterli, "Optimal color filter array design: quantitative conditions and an efficient search procedure," in *Digital Photography V*, B. G. Rodricks and S. E. Susstrunk, Eds., vol. 7250. SPIE, January 18, 2009, p. 725009.

[33] R. Lukac and K. N. Plataniotis, "Color filter arrays: design and performance analysis," *Consumer Electronics, IEEE Transactions on*, vol. 51, no. 4, pp. 1260–1267, 2005.

[34] M. Parmar and S. J. Reeves, "A perceptually based design methodology for color filter arrays," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, vol. 3, 2004, pp. iii–473–6 vol.3.

[35] ——, "Selection of optimal spectral sensitivity functions for color filter arrays," in *Image Processing, 2006 IEEE International Conference on*, 2006, pp. 1005–1008.

[36] Sony Corporation Press Release, "Realization of natural color reproduction in digital still cameras, closer to the natural sight perception of the human eye," July 16, 2003.

[37] W. A. Zhu, K. J. Parker, and M. A. Kriss, "Color filter arrays based on mutually exclusive blue noise patterns," in *Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts III*, G. B. Beretta and R. Eschbach, Eds., vol. 3300. SPIE, January 2, 1998, pp. 207–218.

[38] R. Ramanath, W. E. Snyder, Y. Yoo, and M. S. Drew, "Color image processing pipeline," *Signal Processing Magazine, IEEE*, vol. 22, no. 1, pp. 34–43, 2005.

[39] S. H. Park, H. S. Kim, S. Lansel, M. Parmar, and B. A. Wandell, "A case for denoising before demosaicking color filter array data," in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, 2009, pp. 860–864.

[40] B. A. Wandell, *Foundations of Vision*. Sinauer Associates, May 1995.

[41] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta, "A standard default color space for the internet - sRGB," November 5, 1996.

[42] L. Condat, "A simple, fast and efficient approach to denoisaicking: Joint demosaicking and denoising," in *Image Processing, 2010 IEEE International Conference on*, 2010.

[43] K. Hirakawa, X.-L. Meng, and P. J. Wolfe, "A framework for wavelet-based analysis and processing of color filter array images with applications to denoising and demosaicing," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 1, 2007, pp. I–597–I–600.

[44] K. Hirakawa and T. W. Parks, "Joint demosaicing and denoising," *Image Processing, IEEE Transactions on*, vol. 15, no. 8, pp. 2146–2157, 2006.

[45] K. Hirakawa and X.-L. Meng, "An empirical Bayes EM-wavelet unification for simultaneous denoising, interpolation, and/or demosaicing," in *Image Processing, 2006 IEEE International Conference on*, 2006, pp. 1453–1456.

[46] D. Paliy, A. Foi, R. Bilcu, and V. Katkovnik, "Denoising and interpolation of noisy Bayer data with adaptive cross-color filters," in *Visual Communications and Image Processing 2008*, W. A. Pearlman, J. W. Woods, and L. Lu, Eds., vol. 6822. SPIE, January 27, 2008, p. 68221K.

[47] L. Zhang, X. Wu, and D. Zhang, "Color reproduction from noisy CFA data of single sensor digital cameras," *Image Processing, IEEE Transactions on*, vol. 16, no. 9, pp. 2184–2197, 2007.

[48] N. Joshi, C. L. Zitnick, R. Szeliski, and D. J. Kriegman, "Image deblurring and denoising using color priors," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 1550–1557.

[49] D. Paliy, A. Foi, R. Bilcu, V. Katkovnik, and K. Egiazarian, "Joint deblurring and demosaicing of Poissonian Bayer-data based on local adaptivity," in *Proceedings of 16th European Signal Processing Conference*, August 25-29, 2008.

[50] R. Ramanath and W. Snyder, "Adaptive demosaicking," *Journal of Electronic Imaging*, vol. 12, pp. 633–642, 2003.

[51] P. Chatterjee and P. Milanfar, "Is denoising dead?" *Image Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 895–911, 2010.

[52] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Color plane interpolation using alternating projections," *Image Processing, IEEE Transactions on*, vol. 11, no. 9, pp. 997–1013, 2002.

[53] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, R. W. Schafer, and R. M. Mersereau, "Demosaicking: color filter array interpolation," *Signal Processing Magazine, IEEE*, vol. 22, no. 1, pp. 44–54, 2005.

[54] X. Li, B. Gunturk, and L. Zhang, "Image demosaicing: a systematic survey," in *Visual Communications and Image Processing 2008*, W. A. Pearlman, J. W. Woods, and L. Lu, Eds., vol. 6822. SPIE, January 27, 2008 2008, p. 68221J.

[55] B. C. de Lavarène, D. Alleysson, and J. Hérault, "Practical implementation of LMMSE demosaicing using luminance and chrominance spaces," *Computer Vision and Image Understanding*, vol. 107, no. 1-2, pp. 3–13, August 2007.

[56] E. Dubois, "Filter design for adaptive frequency-domain Bayer demosaicking," in *Image Processing, 2006 IEEE International Conference on*, 2006, pp. 2705–2708.

[57] J. Portilla, D. Otaduy, and C. Dorronsoro, "Low-complexity linear demosaicing using joint spatial-chromatic image statistics," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 1, 2005, pp. I–61–4.

[58] D. Taubman, "Generalized Wiener reconstruction of images from colour sensor data using a scale invariant prior," in *Image Processing, 2000. Proceedings. 2000 International Conference on*, vol. 3, 2000, pp. 801–804 vol.3.

[59] H. J. Trussell and R. E. Hartwig, "Mathematics for demosaicking," *Image Processing, IEEE Transactions on*, vol. 11, no. 4, pp. 485–492, 2002.

[60] L. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation," *Image Processing, IEEE Transactions on*, vol. 14, no. 12, pp. 2167–2178, 2005.

[61] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *Image Processing, IEEE Transactions on*, vol. 17, no. 1, pp. 53–69, 2008.

[62] J. Gu, P. J. Wolfe, and K. Hirakawa, "Filterbank-based universal demosaicking," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, 2010, pp. 1981–1984.

[63] R. Lukac and K. N. Plataniotis, "Universal demosaicking for imaging pipelines with an RGB color filter array," *Pattern Recognition*, vol. 38, no. 11, pp. 2208–2212, November 2005.

[64] D. L. Donoho and J. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, September 1, 1994.

[65] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand, "A gentle introduction to bilateral filtering and its applications," in *ACM SIGGRAPH 2007 courses*, ser. SIGGRAPH '07. New York, NY, USA: ACM, 2007.

[66] R. R. Coifman and D. L. Donoho, "Translation-invariant de-noising," in *Wavelets and Statistics*. Springer-Verlag, 1995, pp. 125–150.

[67] P. Chatterjee and P. Milanfar, "Image denoising using locally learned dictionaries," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ser. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 7246, February 2009, provided by the SAO/NASA Astrophysics Data System.

[68] M. Motwani, M. Gadiya, R. Motwani, and F. Harris, "A survey of image denoising techniques," in *Proceedings of GSPx 2004*, September 27-30, 2004.

[69] C. Gatta, A. Rizzi, and D. Marini, "Local linear LUT method for spatial colour-correction algorithm speed-up," *Vision, Image and Signal Processing, IEE Proceedings* -, vol. 153, no. 3, pp. 357–363, 2006.

[70] M. R. Gupta, E. K. Garcia, and E. Chin, "Adaptive local linear regression with application to printer color management," *Image Processing, IEEE Transactions on*, vol. 17, no. 6, pp. 936–945, 2008.

[71] N. Hrustemovic and M. R. Gupta, "Multiresolutional regularization of local linear regression over adaptive neighborhoods for color management," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, 2008, pp. 497–500.

[72] J. Farrell, P. Catrysse, and B. Wandell, "The digital camera is an imaging system," in *Imaging Systems*. Optical Society of America, 2010, p. JTuA26.

[73] J. E. Farrell, F. Xiao, P. B. Catrysse, and B. A. Wandell, "A simulation tool for evaluating digital camera image quality," in *Image Quality and System Performance*, Y. Miyake and D. R. Rasmussen, Eds., vol. 5294. SPIE, December 19, 2003, pp. 124–131.

[74] J. Farrell, M. Parmar, P. Catrysse, and B. Wandell, *Digital Camera Simulation*, ser. Handbook of Digital Imaging. Wiley Press, (to appear).

[75] International Electrotechnical Commission, "Multimedia systems and equipment - colour measurement and management - part 2-1: Colour management - default RGB colour space - sRGB," Tech. Rep. IEC 61966-2-1, 1999.

[76] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? A new look at signal fidelity measures," *Signal Processing Magazine, IEEE*, vol. 26, no. 1, pp. 98–117, 2009.

[77] W. E. K. Middleton, *Vision through the atmosphere*. University of Toronto Press, 1955.

[78] A. Ben-Israel and T. Greville, *Generalized Inverses: Theory and Applications*. Springer, 2003.

[79] X. Zhang and B. A. Wandell, "A spatial extension of CIELAB for digital color image reproduction," in *Proceedings of the SID Symposiums*, 1996, pp. 731–734.

[80] International Organization for Standardization, "Photography - electronic still-picture cameras - resolution measurements," Tech. Rep. ISO 12233:2000, 2000.

[81] ——, "Graphic technology - standard object colour spectra database for colour reproduction evaluation (SOCS)," Tech. Rep. ISO/TR 16066:2003(E), 2003.

[82] M. Parmar, F. Imai, S. H. Park, and J. Farrell, "A database of high dynamic range visible and near-infrared multispectral images," in *Digital Photography IV*,

J. M. DiCarlo and B. G. Rodricks, Eds., vol. 6817. SPIE, February 14, 2008, p. 68170N.

[83] H. J. Trusseli and M. S. Kulkarni, "Sampling and processing of color signals," *Image Processing, IEEE Transactions on*, vol. 5, no. 4, pp. 677–681, 1996.

[84] J. P. S. Parkkinen, J. Hallikainen, and T. Jääskeläinen, "Characteristic spectra of Munsell colors," *Journal of the Optical Society of America A*, vol. 6, no. 2, pp. 318–322, February 1989.

[85] J. Lehtonen, J. Parkkinen, and T. Jääskeläinen, "Optimal sampling of color spectra," *Journal of the Optical Society of America A*, vol. 23, no. 12, pp. 2983–2988, December 2006.

[86] V. Heikkinen, R. Lenz, T. Jetsu, J. Parkkinen, M. Hauta-Kasari, and T. Jääskeläinen, "Evaluation and unification of some methods for estimating reflectance spectra from RGB images," *Journal of the Optical Society of America A*, vol. 25, no. 10, pp. 2444–2458, October 2008.

[87] D. H. Brainard and W. T. Freeman, "Bayesian color constancy," *Journal of the Optical Society of America A*, vol. 14, no. 7, pp. 1393–1411, July 1997.

[88] B. A. Wandell, "The synthesis and analysis of color images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-9, no. 1, pp. 2–13, 1987.

[89] D. H. Marimont and B. A. Wandell, "Linear models of surface and illuminant spectra," *Journal of the Optical Society of America A*, vol. 9, no. 11, pp. 1905–1913, November 1992.

[90] J. M. DiCarlo and B. A. Wandell, "Spectral estimation theory: beyond linear but before Bayesian," *Journal of the Optical Society of America A*, vol. 20, no. 7, pp. 1261–1270, July 2003.

[91] O. Kohonen, J. Parkkinen, and T. Jääskeläinen, "Databases for spectral color science," *Color Research & Application*, vol. 31, no. 5, pp. 381–390, 2006.

[92] K. Barnard, L. Martin, B. Funt, and A. Coath, "A data set for color research," *Color Research & Application*, vol. 27, no. 3, pp. 147–151, 2002.

[93] U. of Joensuu Color Group, "Spectral database."

[94] M. Storring, H. J. Andersen, and E. Granum, "Estimation of the illuminant colour from human skin colour," in *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, 2000, pp. 64–69.

[95] J. Long and A. A. Gooch, "One-click white balance using human skin reflectance," in *Proceedings of Graphics Interface 2009*, ser. GI '09. Toronto, Ont., Canada: Canadian Information Processing Society, 2009, pp. 55–62.

[96] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," October 2008, version 20081110. [Online]. Available: http://www2.imm.dtu.dk/pubdb/p.php?3274

[97] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proceedings of the twenty-first international conference on Machine learning*, ser. ICML '04. New York, NY, USA: ACM, 2004, p. 29.

[98] D.-Y. Tzeng and R. S. Berns, "A review of principal component analysis and its applications to color technology," *Color Research & Application*, vol. 30, no. 2, pp. 84–98, 2005.

[99] J. A. Worthey and M. H. Brill, "Principal components applied to modeling: Dealing with the mean vector," *Color Research & Application*, vol. 29, no. 4, pp. 261–266, 2004.