Automatically designing an image processing pipeline for a five-band camera prototype using the Local, Linear, Learned ($\boldsymbol{L^3}$) method

Qiyuan Tian[a], Henryk Blasinski[a], Steven Lansel[b], Haomiao Jiang[a], Munenori Fukunishi[b], Joyce E. Farrell[a], and Brian A. Wandell[a,c]

[a]Electrical Engineering Department, Stanford University, Stanford, CA-94305, USA;
[b]Olympus Corporation of the Americas, Sunnyvale, CA-94085, USA;
[c]Psychology Department, Stanford University, Stanford, CA-94305, USA.

ABSTRACT

The development of an image processing pipeline for each new camera design can be time-consuming. To speed camera development, we developed a method named L^3 (Local, Linear, Learned) that automatically creates an image processing pipeline for any design. In this paper, we describe how we used the L^3 method to design and implement an image processing pipeline for a prototype camera with five color channels. The process includes calibrating and simulating the prototype, learning local linear transforms and accelerating the pipeline using graphics processing units (GPUs).

Keywords: local linear learned (L^3), image processing pipeline, camera calibration and simulation, parallel computing, graphics processing units

1. INTRODUCTION

Advances in sensor technologies and the related imaging infrastructure have enabled innovation in camera designs. The first generation of consumer digital cameras and computer vision algorithms were based on the classic Bayer color filter array (CFA).[1] In the next decade we expect that new sensor designs will include different CFAs.[2–4] By increasing the number and type of color filters, it is possible to improve sensor low-light sensitivity,[2,5] dynamic range,[6,7] color accuracy, additional multispectral,[4] IR,[8] polarization[3] and light field information. Commercial product sensors adopting clear pixels for low light photography are already available in the market, such as Aptina's Clarity+ sensor, OmniVision's Clear Pixel sensor and Sony's Exmor RS RGB/W sensor.

Each new sensor design must be accompanied by new image processing algorithms, the series of calculations that transform the raw output from the sensor into a desirable image. These computations normally include demosaicking, denoising, sensor and illuminant correction. The time and effort needed to develop algorithms that are necessary for a new sensor slows the development and implementation of novel designs.[3,4,9] The Local, Linear, Learned (L^3) method automates the development of image processing pipelines for cameras with any novel architectures, such as containing sensors with new CFAs and using new multi-capture or multi-sensor architectures.[5,10–12]

The L^3 method consists of two parts: a L^3 processing pipeline and a L^3 training module. The L^3 processing pipeline is a new imaging architecture based on look-up tables that store pre-computed linear transforms for different classes of sensor pixels. A pixel class is defined based on the pixel type (e.g., red, green, blue), the saturation pattern (e.g. non-saturation, green-saturation), mean response level (e.g. low, middle, high) and spatial contrast (e.g. uniform, texture) in the neighboring pixels. The L^3 processing pipeline applies pre-computed local linear transforms that are associated with each class to translate the pixel values into the target output representation (typically CIE XYZ). This methodology integrates multiple processing steps (e.g. demosaicking, denoising and color balancing) into the transform, reduces runtime computation and allows for parallel computing acceleration.

Send correspondence to Qiyuan Tian. E-mail: qytian@stanford.edu.

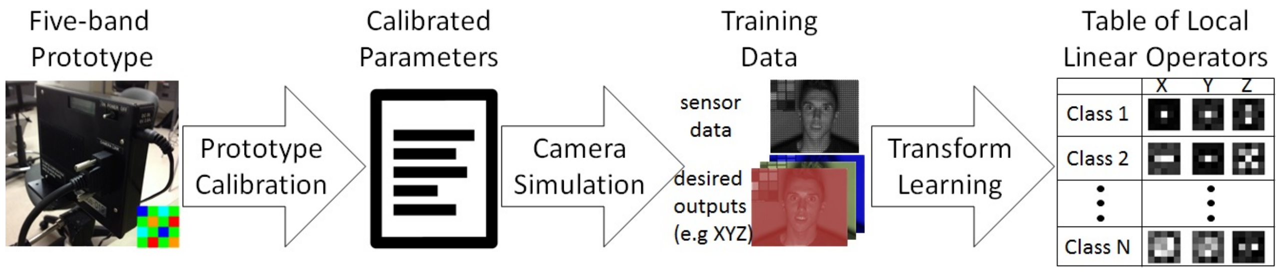


Figure 1. **Overview of the L^3 training module.** The five-band prototype camera is calibrated to estimate the ISET camera simulation parameters. The sensor simulation is applied to a collection of multispectral scenes to estimate the sensor images and corresponding XYZ values. The L^3 training process classifies sensor pixels into classes based on the pixel and surrounding pixel pattern. Three linear operators that map the sensor data into the corresponding XYZ values are estimated for each cluster.

The L^3 training module calculates the linear transforms that are stored for each class. The optimal local linear transforms, i.e. Wiener filters, are learned in a data-driven way from a training set of images produced by camera simulation. These transforms adapt to a specific training scene, camera design and desired outputs.

Tian et al. created an L^3 processing pipeline for a simulated 2×2 RGB/W CFA design that was designed to produce superior low light performance.[5] In this paper, we used the L^3 method to design and implement an image processing pipeline for a prototype camera with a single image sensor containing five color channels arranged in a 4×4 super-pixel (Figure 2). In the following section, we describe the five-band camera prototype (Section 2.1.1), calibrating and simulating the camera prototype (Section 2.1.2) and learning local linear transforms (Section 2.1.3) in the L^3 training module (Figure 1). In Section 2.2, we show how we accelerated the created L^3 processing pipeline using graphics processing units (GPUs) (Figure 3). Section 3 presents experimental results and Section 4 concludes the paper.

2. METHOD

2.1 L^3 Training Module

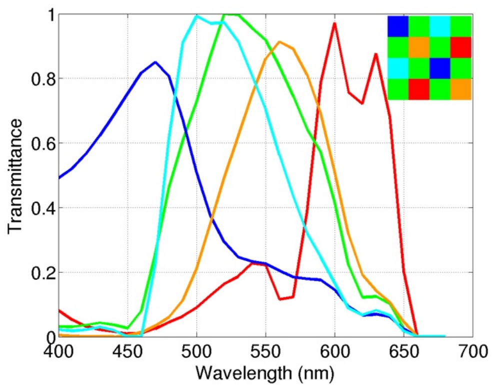
The L^3 method derives the transform from the sensor image to the desired output in a data-driven fashion. The nature and quality of the learned transforms depends on the training data. Therefore the training scenes should span the large variety of scenes the camera may encounter. By using a simulation of the prototype, it is possible to create a variety of illuminants, luminance levels, reflectances and spatial statistics that are necessary to build a table of linear transforms. Simulation also enables computing the target outputs that must be registered with the sensor images from a unconventional CFA. It is labor intensive and impractical to acquire training data experimentally with a physical camera.

2.1.1 Five-band Camera Prototype

The five-band camera prototype includes an optical lens, an image sensor, a frame grabber board and software. The optical system is compliant with OM standard interchangeable lenses. In our experiment we used a lens that had a fixed focal length of 50 mm and an F number of 2.8. The image sensor is a CMOS detector with 2048 (H) × 1164 (V) pixels, 5 μm × 5 μm pixel size and 24,000e well capacity. The sensor has an IR cut filter above the surface and no microlens array. The CFA spectral responsivity and spatial arrangement are shown in Figure 2 (left). The frame grabber board and capture software were developed in house. The prototype has been used to improve remote cardio-pulmonary measurements.[13]

2.1.2 Calibrating and Simulating the Camera Prototype

Camera calibration can be divided into two steps. First, the combined spectral characteristics of the lens, pixel stack and the photodetector are estimated. This step characterizes the wavelength dependent responsivities of each of the camera channels. Once these responsivities are known, we determine whether there is a multiplicative



Pixel Width/ Height (μm)	**5**
Fill Factor	**0.5**
Dark Voltage (V/sec)	**0.00048**
Read Noise (V)	**0.0017**
Dark Signal Non-uniformity (V)	**9.07e-5**
Photo Response Non-uniformity (%)	**0.017**
Conversion Gain (V/e-)	**0.0001**
Voltage Swing (V)	**1**
Well Capacity (e-)	**24,000**
Analog Gain	**2.12**
Analog Offset (V)	**0.056**
F Number	**2.8**
Focal Length (m)	**0.05**

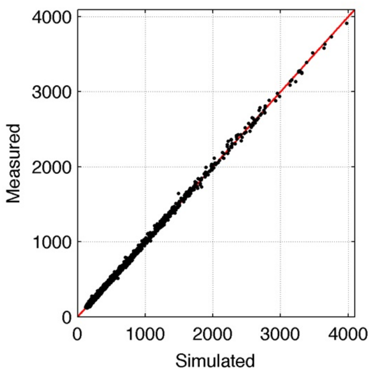


Figure 2. **Five-band camera specifications and simulation results.** Left: sensor CFA layout (inset) and spectral responsivity of the color filters; middle: sensor and optics parameters; right: the scatter plot of measured versus simulated sensor data intensities (quantized to 12 bits) of the test scene (Macbeth Color Checker) for a variety of radiances.

gain factor and offset applied to the predicted camera output. When parameters such as exposure duration and camera's aperture are known, this gain and offset jointly represent the internal analog amplifiers settings and pixel's quantuum efficiency.

Camera spectral responsivity curves were estimated by taking a sequence of reference white target images illuminated with monochromatic lights of different wavelengths. Monochromatic light was generated with an Oriel Cornerstone 130 monochromator, and its spectral power distribution was measured with a Photo Research PR715 spectrophotometer. The spectral responsivity, r, of each channel was found by solving the optimization problem:

$$\begin{aligned} \underset{r}{\text{minimize}} \quad & \|m - L^T r\|_2^2 + \delta \|Rr\|_2^2 \\ \text{subject to} \quad & r \geq 0 \end{aligned} \tag{1}$$

where m are the measured pixel intensities, columns of L represent the spectral power distributions of the monochromatic lights (photons), and $|Rr|$ is a measure of the variance in the derivative and acts as a roughness penalty. The amount of smoothing is controlled by the parameter δ, whose value was selected via cross-validation on a set of Macbeth chart images. The calibrated spectral responsivity curves are shown in Figure 2 (left).

Camera gain and offset parameters were evaluated by taking images of a Macbeth color target under four different illuminants: D65, tungsten, fluorescent and illuminant A. The radiance of each patch-illuminant combination was measured with the Photo Research PR715 spectrophotometer. The optimal values of the gain α and the offset β were estimated by solving a least-squares problem:

$$\begin{aligned} \underset{\alpha,\beta}{\text{minimize}} \quad & \|m - \alpha E^T r - \beta\|_2^2 \\ \text{subject to} \quad & \alpha \geq 0 \end{aligned} \tag{2}$$

where m represents the measured pixel intensities, r is the camera spectral responsivity given by Equation 1, and E is a matrix whose columns are formed by scene radiances measured in photons.

To complete the characterization of the physical camera, we estimated the sensor noise properties (dark voltage, read noise, dark signal non-uniformity (DSNU) and photo response non-uniformity (PRNU)) from multiple images captured at several different exposure durations.[14] A uniform white target was used to estimate read noise and PRNU. For every exposure duration, the average response of each pixel was computed. This average was then subtracted from pixel responses to compute the noise offset. The read noise was estimated as the standard deviation of these offsets across all exposure durations. The influence of the read noise was removed by estimating the remaining parameters from sensor responses averaged across trials.

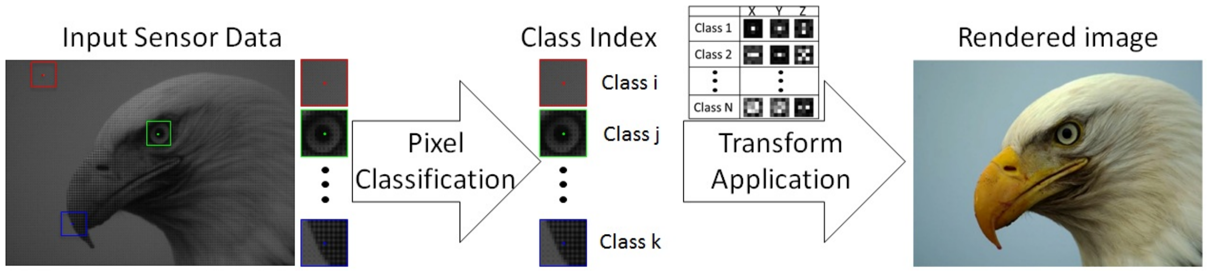


Figure 3. **Overview of the L^3 processing pipeline.** Class-specific linear transforms are pre-computed and stored in a table. Each captured sensor pixel is classified into one of many possible classes, and the appropriate linear transform is applied to the pixel and its neighborhood to render the data.

The dark voltage was estimated from the multiple exposures captured with the camera lens cap on. The dark voltage component, measured in volts per second, is the average slope of the linear relationship between sensor values and exposure duration. DSNU is the standard deviation of the intercept of this linear function. Similarly, PRNU was estimated by the standard deviation of the slope of the best-fitting linear function relating sensor values and exposure duration in the camera images for the white target. To account for possible non-uniformities in illumination, the slopes were locally normalized by the mean slope value computed over some neighborhood.

We simulated the five-band camera prototype with the calibrated parameters (Figure 2, Table) using the Image Systems Engineering Toolbox (ISET).[14,15] ISET simulations use a phenomenological model to account for the key properties of a camera, including the optics, pixel stack, photodetector and associated sensor circuits. We validated our ISET simulations by comparing the measured and predicted sensor voltages to a set of test scenes (Figure 2, right). After validation, we used the simulation model to predict the responses to a large number of other scenes that were used to derive the local linear transforms.

2.1.3 Learning Local Linear Transforms

We used Wiener estimation to select the optimal local linear transforms in the presence of noise in the pixel data.[5,11] Using ISET, we calculated the predicted sensor images and the target outputs for the training scenes. For a consumer photography application, the multispectral training scenes are from a public database that includes faces and objects[16,17] and the target outputs are the scene CIE XYZ values. Each pixel was classified based on the pixel color, the saturation pattern, mean response level and spatial contrast of the pixel neighborhood.[5] For each class, three linear operators are derived to optimize the transformation from the pixel to CIE XYZ values.

2.2 L^3 Processing Pipeline Acceleration

Figure 3 provides an overview of the L^3 image processing pipeline. Each pixel in the input image is classified into a proper class using the same criteria as in the training module (Section 2.1.3). The pre-computed linear operators for the classes are stored in a table, and they are applied to the sensor data to produce the rendered values. This rendering process can be parallelized pixel-wise and performed relatively quickly. By using hundreds of processing units simultaneously, the rendering speed can be substantially accelerated (by orders of magnitude) compared to serial CPU computing. Fast rendering is important to many applications that utilize unconventional CFAs, such as rendering high dynamic range videos captured in a single shot using novel CFAs.[2,6,7]

We implemented the L^3 processing on GPUs using CUDA, a parallel computing platform and programming model created by NVIDIA and implemented for their GPUs.[18] Each pixel of sensor data is assigned to one processing unit (CUDA kernel) that calculates the pixel statistics necessary for classification, and then retrieves and applies the appropriate linear transforms to compute the output values. To highlight the fast rendering feature of L^3 processing pipeline, we also processed videos captured by the five-band camera.

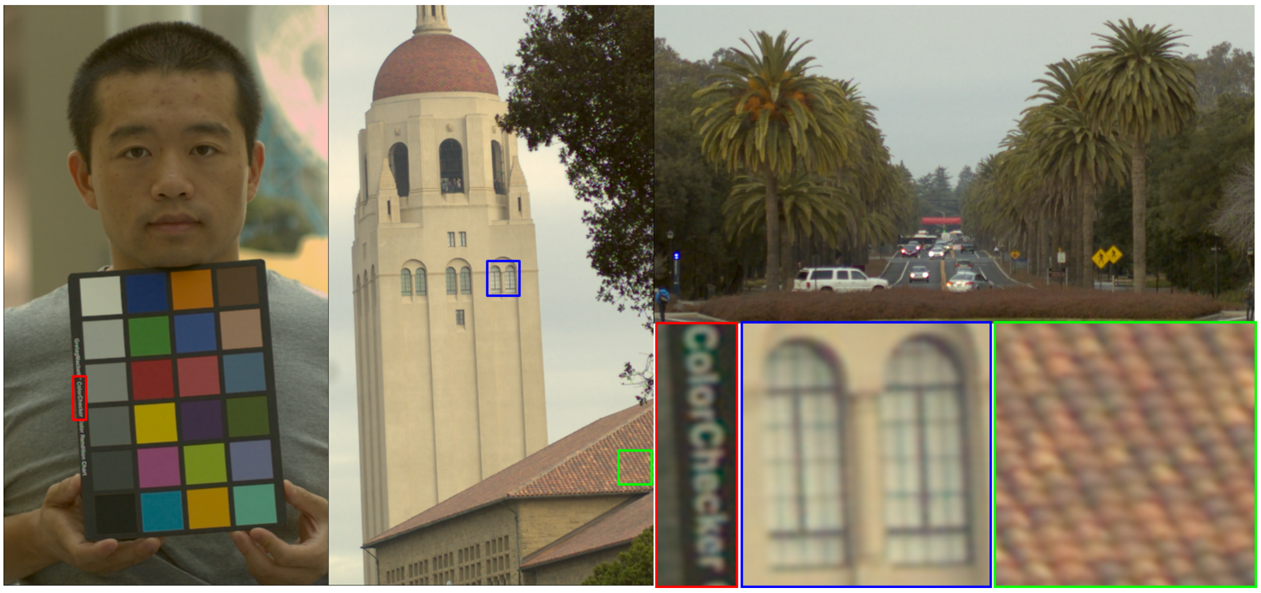


Figure 4. L^3 **rendered results and enlarged regions of interest.**

3. RESULTS

3.1 Rendering Results

Figure 4 shows sRGB images rendered using the L^3 image processing pipeline from the sensor data captured by the five-band camera prototype. The sensor data were first converted to CIE XYZ values based on L^3 pixel classes and linear transforms. Then, the XYZ values were converted to sRGB for the final display. The resultant images of several example scenes reproduce accurate color and preserve spatial details, using the automatically generated L^3 image processing pipeline.

3.2 Rendering Speed

We implemented both optimized CPU serial rendering (Intel Core i7-4770K processors, 3.5 GHz clock rate) and GPU parallel rendering (NVIDIA GEFORCE GTX 770 graphics card, 1536 CUDA cores and 1.085 GHz base clock rate). The raw image and video data are stored in main memory while linear operators are stored in the GPU global memory. The local linear operators are 9×9 pixels. For 720p (1280×720) color images, the mean processing time of CPU and GPU rendering is 12.4 seconds and 60.2 milliseconds (ms), respectively. The GPU times include delays for copying the raw image from main memory to GPU global memory (1.5 ms), computing the rendered image (55.3 ms), and returning it to CPU memory (3.4 ms). The GPU rendering achieved about $200\times$ acceleration compared to CPU rendering, corresponding to 16 video frames per second (FPS). For a 720p color video with 1800 frames, the total computational time is 163.2 seconds (11 FPS).

The rendering times scale with the number of CUDA cores, the number of pixels in an image and the size of linear operators. An image with VGA resolution (640×480), which could meet the demand for many applications, would require one third of the processing time needed for a 720p image (i.e. 25 ms and 40 FPS). Using 5×5 pixel linear transforms require only two thirds of the computations of the 9×9 pixel transforms. But to preserve image quality, one should only reduce the operator size for CFAs with fewer number of pixels in the repeating pattern.

4. CONCLUSIONS

We used the L^3 method to develop a customized image processing pipeline for a new, five-band camera prototype. The implementation involves (1) calibrating the camera and building a camera model with the calibrated parameters, (2) generating simulated camera responses and target outputs, (3) learning the local linear transforms from the training data, and (4) implementing an efficient GPU-accelerated rendering process. The automatically generated L^3 image processing pipeline provides satisfying image quality and efficient rendering speed for the novel five-band camera prototype.

REFERENCES

[1] Bayer, B. E., "Color imaging array," (July 20 1976). US Patent 3,971,065.

[2] Parmar, M. and Wandell, B. A., "Interleaved imaging: an imaging system design inspired by rod-cone vision," in [*IS&T/SPIE Electronic Imaging*], 725008–725008, International Society for Optics and Photonics (2009).

[3] Narasimhan, S. G. and Nayar, S. K., "Enhancing resolution along multiple imaging dimensions using assorted pixels," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **27**(4), 518–530 (2005).

[4] Yasuma, F., Mitsunaga, T., Iso, D., and Nayar, S. K., "Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum," *Image Processing, IEEE Transactions on* **19**(9), 2241–2253 (2010).

[5] Tian, Q., Lansel, S., Farrell, J. E., and Wandell, B. A., "Automating the design of image processing pipelines for novel color filter arrays: Local, Linear, Learned (L^3) method," in [*IS&T/SPIE Electronic Imaging*], 90230K–90230K, International Society for Optics and Photonics (2014).

[6] Nayar, S. K. and Mitsunaga, T., "High dynamic range imaging: Spatially varying pixel exposures," in [*Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*], **1**, 472–479, IEEE (2000).

[7] Cheng, C.-H., Au, O. C., Cheung, N.-M., Liu, C.-H., and Yip, K.-Y., "High dynamic range image capturing by spatial varying exposed color filter array with specific demosaicking algorithm," in [*Communications, Computers and Signal Processing, 2009. PacRim 2009. IEEE Pacific Rim Conference on*], 648–653, IEEE (2009).

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

[9] Heide, F., Steinberger, M., Tsai, Y.-T., Rouf, M., Pajak, D., Reddy, D., Gallo, O., Liu, J., Heidrich, W., Egiazarian, K., et al., "Flexisp: A flexible camera image processing framework," *ACM Transactions on Graphics* **33**(6) (2014).

[10] Lansel, S. and Wandell, B., "Local linear learned image processing pipeline," in [*Imaging Systems and Applications*], Optical Society of America (2011).

[11] Lansel, S. P., *Local Linear Learned Method for Image and Reflectance Estimation*, PhD thesis, Stanford University (2011).

[12] Lansel, S., Wandell, B., et al., "Learning of image processing pipeline for digital imaging devices," (Dec. 7 2012). WO Patent 2,012,166,840.

[13] McDuff, D., Gontarek, S., and Picard, R., "Improvements in remote cardiopulmonary measurement using a five band digital camera," *Biomedical Engineering, IEEE Transactions on* **61**, 2593–2601 (Oct 2014).

[14] Farrell, J. E., Catrysse, P. B., and Wandell, B. A., "Digital camera simulation," *Applied Optics* **51**(4), A80–A90 (2012).

[15] Farrell, J. E., Xiao, F., Catrysse, P. B., and Wandell, B. A., "A simulation tool for evaluating digital camera image quality," in [*Electronic Imaging 2004*], 124–131, International Society for Optics and Photonics (2003).

[16] Parmar, M., Lansel, S., and Farrell, J., "An led-based lighting system for acquiring multispectral scenes," in [*IS&T/SPIE Electronic Imaging*], 82990P–82990P, International Society for Optics and Photonics (2012).

[17] `http://imageval.com/scene-database/`.

[18] `http://www.nvidia.com/object/cuda_home_new.html`.