Colour From Grey by Optimized Colour Ordering

Arash Vahdat and Mark S. Drew,
School of Computing Science, Simon Fraser University, Vancouver, British Columbia, Canada V5A 1S6, {***avahdat,mark***}***@cs.sfu.ca***

Abstract

What is the minimum amount of information we must transmit along with greyscale values in order to recover a colour original from a grey image? The problem of colour quantization is one of long standing with, for example, a GIF file consisting of single-byte grey values (appropriately compressed using entropy coding) plus a 256×3 colour lookup table in the file header. That lookup table can be generated in various ways (e.g., the standard median-cut algorithm). Here we take the viewpoint that we can generate greyscale values from an input, coder-side, colour image by traversing colour values in a particular order. In forming a grey value, we seek to replace an entire colour plane, orthogonal to the L^* in CIELAB colour space, by its grey value, and then reconstitute colour by visiting the byte-value L^* greylevels using a path in 3 dimensions such that the order of colour forms a parametric curve from grey plane to grey plane that traverses a rich sampling of colour space while optimally encompassing the gamut of the input image. In that way, we merely have to transmit the parameters for the curve itself along with the grey values in order to recover an approximation of colour. In particular, here we use a curve such that we need to transmit an additional 13 values only. Moreover we can use n-bit grey values with $n < 8$ for colour-reduced transmission to mobile devices. We find that we do better than the standard GIF file method in terms of CIELAB error for grey and comparably in recovered colour error, and with much less information transmitted. The method rests on an optimization is which grey is selected from nearby colour planes such that the overall grey error is minimized whilst also minimizing the colour error.

1. Introduction

Colour quantization consists of reducing the number of colour levels, usually in a lossy fashion. In particular, transforming a colour image to greyscale takes a 24-bit colour pixel into a 256-level (or fewer) greyscale value. As an intermediate representation, palettized colour consists of n-bit value indices into a colour lookup table that stores full-colour information for 2^n colour-cluster centers (see [1] for a short introduction to the well-known and efficient Median Cut algorithm for this purpose).

Here, we are concerned with the question of what is the minimum amount of auxiliary information necessary to transmit, along with greylevels, to be able to have the decoder side generate both an accurate grey image as well as an accurate colour representation. We concentrate here on colorimetric error [2], rather than PSNR say [3], and adopt CIELAB as a reasonable representation of perceptual colour difference. So we wish to utilize byte values or smaller of quantized L^* as our greyscale, and 24-bit CIELAB, displayed in sRGB colour space, as our representation of colour.

But we also wish to develop a colour-to-greyscale transformation which as best as possible also encodes colour information in the grey values, and we'll be willing to accept some greylevel error provided we also obtain a good level of colour accuracy. Here, we make use of a parametric curve, which visits planes of colour corresponding to greylevels in an orderly fashion. Then one can reconstruct an approximation of the input colour image from the greyscale image simply by also storing a small amount of knowledge about the mapping curve.

We make the parametric curve adaptive to the input image by optimizing its parameters such that the mapping produces a greyscale and a reconstructed colour image with minimum amount of error, balanced between the two. Here, we use a parametric expression of traversing CIELAB colour space that takes merely 13 parameters, and these would easily be included in a file header. The contribution here is the notion of utilizing a curve that visits grey value planes in colour space, as well as the novel mapping used in the parametric curve: The curve is designed so as to minimize colour error adaptively according to the data content of the input image. The intent of the paper is to examine whether one can indeed use a small characterization of the gamut of an image, encompassed by a simple curve, to represent colour using grey. Results are shown to be promising.

In §2 we introduce the method by describing in §2.1 the curve used for mapping from colour space to greylevel. We propose the transformation process in §2.2 and optimization procedure in §2.3. We demonstrate the efficiency of our algorithm by experimental results in §3, including showing substantive improvement of in CIELAB error over the standard GIF encoding, in §3. Finally, we conclude the paper and address future work in §4.

2. Colour to Grey and Back

The mapping from colour image to greyscale typically deletes colour information by the assignment of the same grey intensity to different colours. For example, consider the standard NTSC multimedia-standard mapping from RGB to greyscale luma, $Y' = 0.299R' + 0.587G' + 0.114B'$, where typically primed quantities, meaning gamma-corrected values, are used (although in fact the transform was derived for linear-light values!). Practically speaking, for quantized greylevels this means that in RGB colour space points on planes with normal vector $u = (0.299, 0.587, 0.114)$ all are assigned the same greyscale value. As an extreme case, in Fig. 1 a colour image and its corresponding, *single* grey level in this case, image are shown: Although the image consists of different colours, all coloured points on it have the same greyscale value.

In order to encode colour information in an L^* greyscale image, we assume that in a small colour neighbourhood each grey intensity corresponds to a fixed colour point, representing all neighbouring colours. The task at hand, then, is to decide which representative colour to use for each greylevel, adaptive to the input image, and also allow greylevels to migrate to nearby planes if that will decrease colour error at the expense of some greyscale error.

In Fig. 2 this assignment for a small region in colour space is shown, using solid points for the representative colour in each plane. That is, suppose we have created a curve traversing colours in colour space (we use CIELAB triples) in an orderly fashion so as to move steadily upward from L* greylevel 0 to greylevel 255, visiting a rich colour sampling in both hue and saturation. In the illustrative figure, suppose that for quantized greyscale g, the colour assigned is a green, and for the next quantized grey value $(g-1)$, the colour is a pink and for $(g-2)$ is a blue — filled points on a spiralling curve are the *initial* representative colours that correspond to each of the quantized greylevels.

In this case to encode an input colour, that happens to correspond to greylevel g, we use the greyscale of the *closest* filled point in 3D; e.g., for input-image point A, which happens to be a red and has quantized L^* value g, and is not one of our representative colours on the curve, we find that we should actually use a greyscale of $(g-1)$ instead of g, because (1): $(g-1)$ is within a small search range in quantized greyscale of g, and (2): the representative colour for $(g-1)$ is closer to A's colour. So we assign colour B to this pixel because it is closer to the actual colour.

That is, upon reconstruction this pixel will be assigned colour B. All that is necessary at the decoder side is (1) the gray level actually assigned to the pixel and (2) the parametric curve used, so that the colour pertaining to that greylevel is known. Note that the search procedure in fact adds error to grey, but on the other hand it also produces a better colour for the reconstructed colour image.

We assume that input-image colours are in standard, non-linear sRGB colour space [4]. We use CIELAB as our searching algorithm's colour space, but display both grey and colours in sRGB. With Lightness L^* used as our internal greyscale, the quantization planes will be parallel to the a^*, b^* plane.

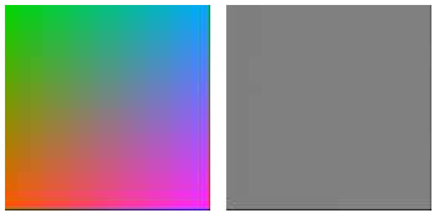


Figure 1. *Left: an image containing different colours, all corresponding to the same grey. Right: Greyscale image of left image.*

2.1. Parametric curve

As illustrated in Fig. 2, we need an efficient way to store and represent the mapping between greyscale and colour values. The standard approach is to use a palette to store RGB values in a lookup table. The image data itself then consists of lookup table index values, which cannot of course provide a sensible greyscale image itself – instead, grey is produced by regenerating an approximate colour image and then transforming that to gray. Several approaches have been used for producing the lookup table, both adaptive to the image data and non-adaptive.

Here we take the tack of instead using a 3-space curve $C(g)$: $\Re \mapsto \Re^3$ which maps greyscale values to points in colour space. By choosing an appropriate parametric function we can optimize its shape for each input image, and attach its parameters instead of a large lookup table. Not only do we thus save on bandwidth, but in fact we show below in §3 that we do better than the standard adaptive GIF file in the accuracy of grey images produced and similarly for recovered colour.

The mapping function can be thought as a curve with greyscale value as its parameter. Its main characteristic is that it should traverse different regions of colour space as g increases. Hence in CIELAB space if we assume a curve with a vertical axis, equal to lightness component L^*, a helical curve will satisfy this requirement. As shown in Fig. 2, as g increases the curve travels through different colours on planes parallel to the a^*, b^* plane. However, the requisite curve should also cover different saturation values. Thus, instead of using a fixed radius, we assume an alternating radius with a sliding Gaussian peak:

$$\text{radius} = r(g) = c_1 \exp\left(\frac{-(g-c_2)^2}{(c_3)^2}\right) \sin(c_4\, g + c_5) + c_6 \quad (1)$$

where g is greylevel value.The mapping curve is then taken to be

$$\begin{aligned} a^* &= r(g)\sin(c_7\, g) + c_a \\ b^* &= r(g)\cos(c_7\, g) + c_b \\ L^* &= g \end{aligned} \quad (2)$$

where L^*, a^*, b^* are the colour components and $c_1, c_2, \ldots, c_7$ are the coefficients which will be optimized later. c_a and c_b are the center of the curve in the a^*, b^* plane, which can be assigned to the mid-value 0 (128 in Matlab). However to allow the main L^* axis to bend to suit the actual gamut of the image, instead of fixing c_a and c_b we add two further 3-vector parameters, w_1, w_2 expressing a polynomial fit of L^* to a^* and b^*:

$$\begin{aligned} \left[1,\, L^*,\, (L^*)^2\right] w_1 &= a^* \\ \left[1,\, L^*,\, (L^*)^2\right] w_2 &= b^* \end{aligned} \quad (3)$$

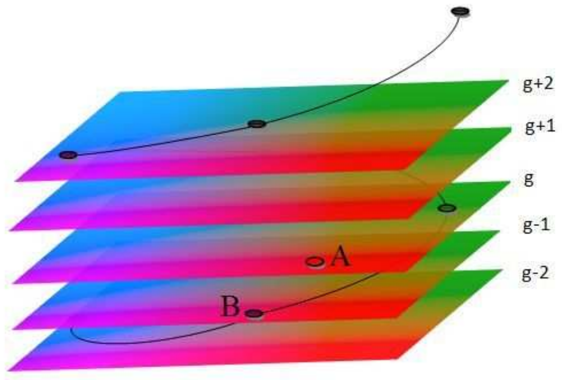


Figure 2. *Colour planes with uniform greyscale values. On each plane we fix a colour point on the intersecting curve and during transformation from colour image to greyscale value we use greyscale of the closest fixed point. So point* A *is transformed to greyscale* $g-1$ *and in reconstruction colour point* B *is used.*

Hence our final expression of curve C is as follows:

$$\begin{aligned} a^* &= r(g)\sin(c_7 g) + w_1 \cdot [1, g, g^2] \\ b^* &= r(g)\cos(c_7 g) + w_2 \cdot [1, g, g^2] \\ L^* &= g \end{aligned} \quad (4)$$

The radius and curve with initial parameters used in optimization are shown in Figs. 3 and 4.

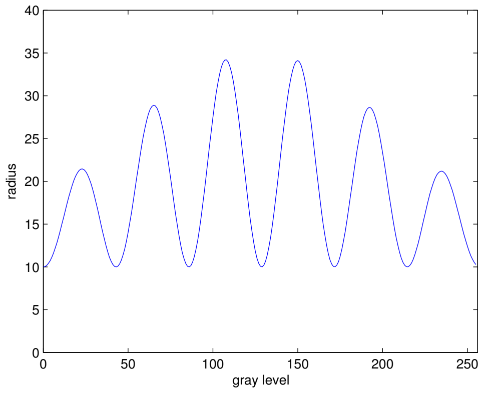


Figure 3. *Radius function.*

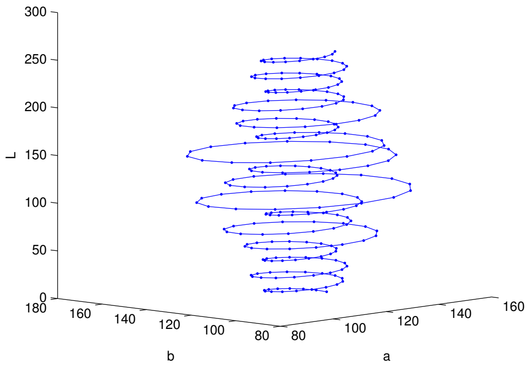


Figure 4. *Parameteric curve tranversing colour space.*

2.2. Transform to greyscale image and colour image reconstruction

Each pixel in the input image can be thought as a point in 3-dimensional CIELAB space. In order to transfer it to greyscale, first the closest point on the curve in 3D is found: then its greylevel is used as that pixel's greylevel. Due to the special shape of the curve that closest colour point will also have similar greyscale value and the amount of error will be small — and we can enforce this by using only greyscale values ...$(g-2)$, $(g-1)$, g, $(g+1)$. $(g+2)$,.... in a small search range. Hence, the greyscale value for each pixel p with colour $\rho(p)$ is obtained by:

$$grey_{appx} = \arg\min_g (\|C(g) - \rho(p)\|) \quad (5)$$

where $g = 0..255$. In order to reconstruct the colour image from the corresponding greyscale image constructed in this way, we simply use the corresponding colour point on the curve for grey value $grey_{appx}$:

$$Lab_{recon} = C(grey_{appx}) \quad (6)$$

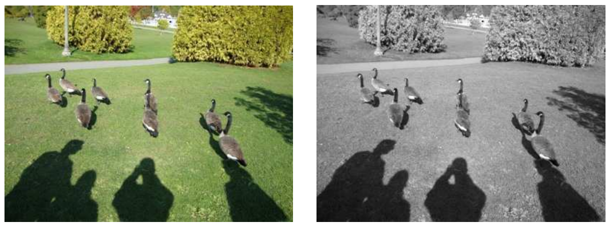


Figure 5. *(a): Input image; (b): grey for input.*

2.3. Optimization

Each colour image can consist of points in different regions of CIELAB space. So a fixed curve cannot always reconstruct the image very well. The main advantage of the proposed curve is that its shape can be changed by a few parameters. Thus, for each input image we adaptively optimize the parameters such that the error in both the grey image and the reconstructed colour image is low. Suppose we limit the search over nearby greyscale planes to $g \to g \pm k$, $k = 0..K$ (we use K=5 here). Then the appropriate objective function is as follows:

$$\min_{c1..c7} Cost(C, \rho) = \sum_{p \in \Omega} \left\{ \min_{k=-K..K} \|C(g+k) - \rho\| \right\}^2 \quad (7)$$

with $g = L^*(\rho)$ and where Ω is the image domain. We use Matlab's built-in implementation of the Trust Region Approach [5] to minimize the cost function in (7).

3. Experiments

Let us compare the performance of our algorithm with the standard GIF file format colour quantization method, especially concentrating on low-bitrate representations for mobile-device applications. For fewer bits, we quantize to only 2^n greylevels and also search in colour space only on those quantized planes. Of course, we could compare to JPEG and JPEG-2000 but those are considerably more time-complex methods and the GIF standard is the closest model to which to compare.

Consider the input image I_1 shown in Fig. 5(a): its corresponding grey is shown in Fig. 5(b). Now, Figs. 6(a-d) show the grey images for the present method, for bpp (bits per pixel) values 3,4,6,8; and Figs. 6(e-h) show the corresponding grey images using GIF. Figs. 7(a-d) show the colour versions for the present method, over these bpp values, and Figs. 7(e-h) show GIF colour. Fig. 8 shows that, typically, the present method almost always generates a better grey representation, especially for low bitrates, and can also generate a better colour as well, although for the full 8 bits often the GIF is better.

The error measure used here is CIELAB error, for both grey images and colour, and one should note that since this is a pixelwise measure, it does not correctly take into account mechanisms of spatial integration, as in e.g. the s-CIELAB measure [6]. Since for lower bitrates GIF tends to produce speckled images, a better error metric might tend to discount these speckles in favour of the overall fidelity of the image and thus numerical results might be different.

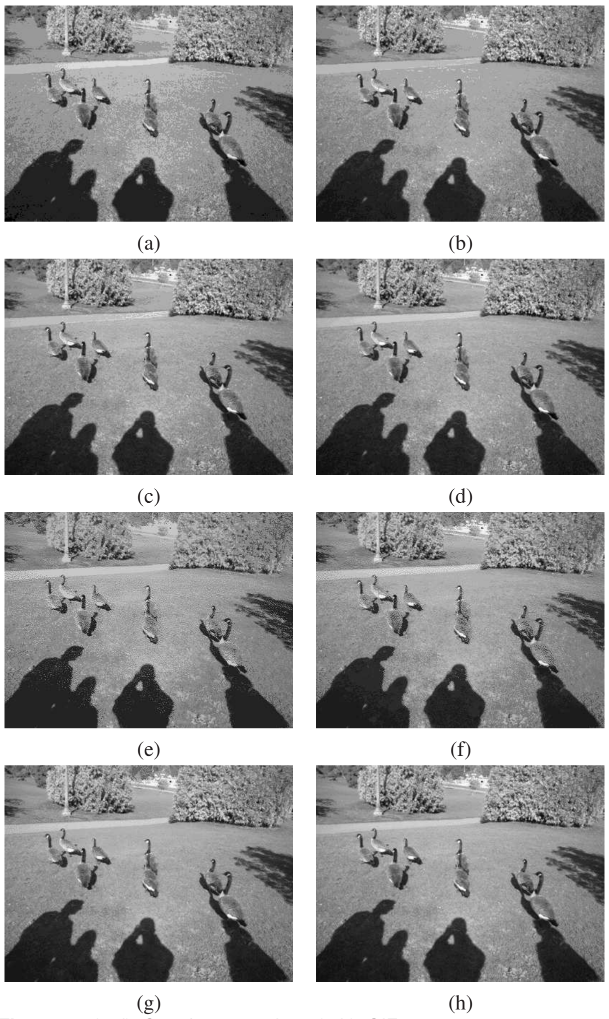


Figure 6. *(a-d): Grey for 3,4,6,8 bpp; (e-h): GIF grey.*

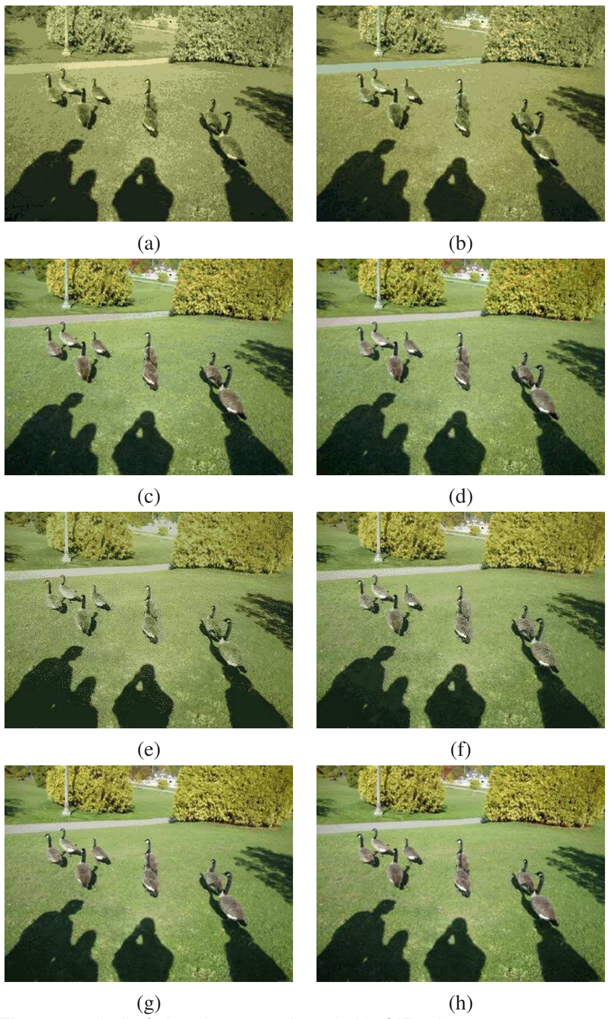


Figure 7. *(a-d): Colour for 3,4,6,8 bpp; (e-h): GIF colour.*

Nevertheless we see that the method performs remarkably well given the small number of curve parameters needed to be stored on top of the greyscale values.

Fig. 9(a) shows how the fit to the gamut for Fig. 5 (shown in red, subsampled) is indeed well fit by the (blue) parameteric curve. The GIF palette is shown in Fig. 9(b), as opposed to that for the proposed method in Fig. 9((c). For our method, the palette is of course well-ordered from L^*=0 to 100. Further results are shown in Fig. 10.

4. Conclusions

We have proposed a novel method to reconstruct colour from a greyscale image, by optimizing a mapping from greyscale colour using a parametric curve. Remarkably, even for low bitrate results show that the method reconstructs both greyscale and colour information with surprisingly small error. No catastrophic failure was observed for any image, and almost always, the grey version generated is better than GIF. The colour version has comparable or better error especially for low bitrate. However if there are not many colours in the image, the method does not do as well because we may use some bits for other colours which do not exist in the image. We used a constant quantization rate to sample from our curve. This results in dense samples at small radius or low saturation but sparse samples at large radius or saturated colours. So, the method works better for images with low saturation, and reconstruction error may increase as saturation increases.

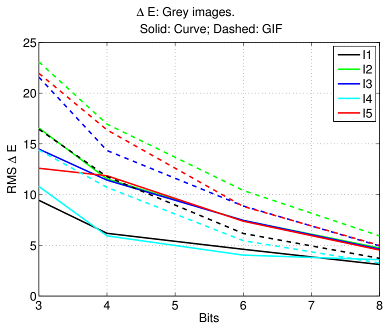
Overall, we have succeeded in the intent of the paper, namely showing that a very low-complexity curve (just 13 parameters, here) may indeed describe the gamut of the input image sufficiently well. Clearly it is likely possible to improve the precise form of the curve, and as well user studies are called for to study the efficacy of the method. These are the subject of future work, but the main proposal is indeed justified by the current results.

References

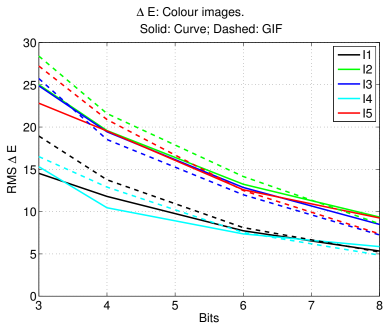
[1] Z.-N. Li and M.S. Drew. *Fundamentals of Multimedia*. Prentice-Hall, 2004.

[2] G. Wyszecki and W.S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulas*. Wiley, New York, 2nd edition, 1982.

[3] S.J. Daly. The visible difference predictor: an algorithm for the assessment of image fidelity. In Allebach Rogowitz and Klein, editors, *SPIE: Human Vision, Visual Processing, and Digital Display III*, pages 1666:2–15, 1992.

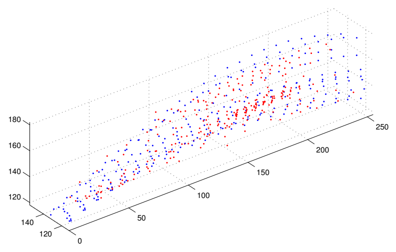


(a)

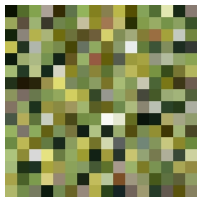
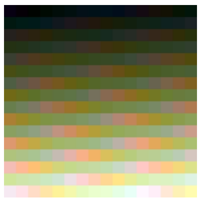


(b)

Figure 8. *(a): CIELAB errors for grey images, images* I_1*-*I_5*; (b): Colour error.*



(a)

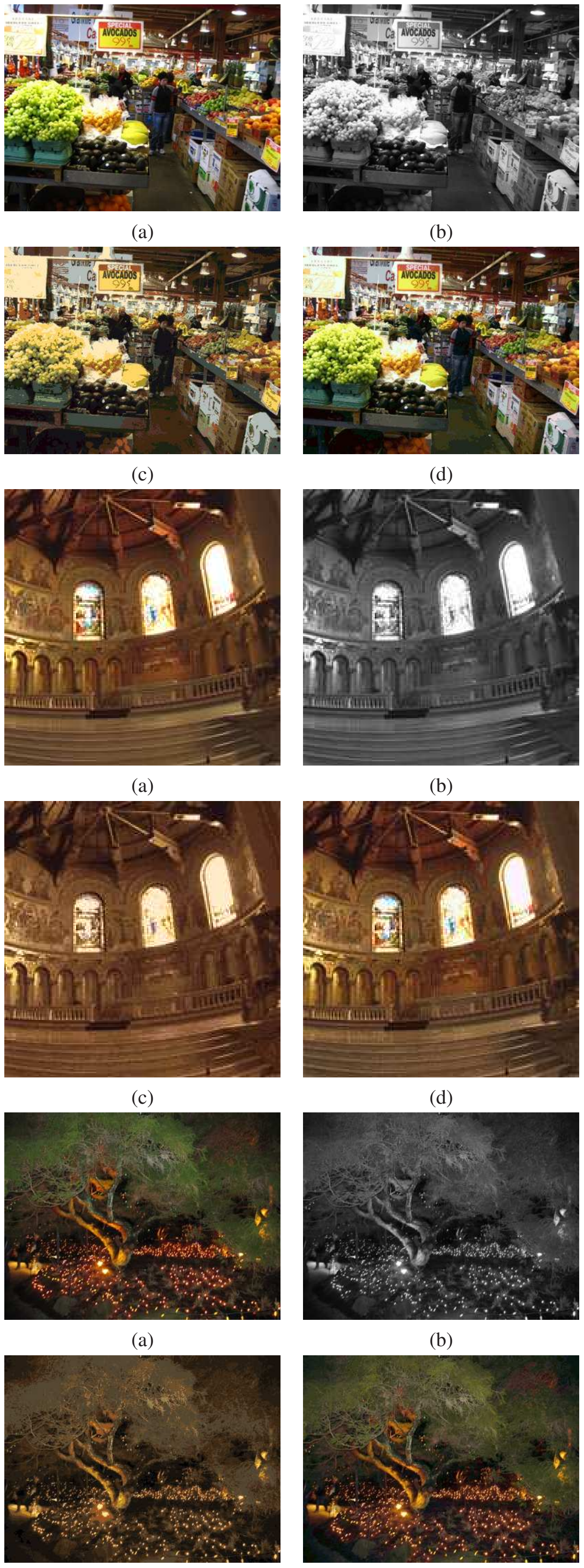
(b) (c)

Figure 9. *(a): Gamut encompassed by parameteric curve; (b): GIF palette; (c): Ordered colours along curve.*

[4] International Electrotechnical Commission. Multimedia systems and equipment – colour measurement and management – part 2-1: Colour management – default RGB colour space – sRGB. IEC 61966-2-1:1999.

[5] T.F. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM J. Optim.*, 6:418–445, 1996.

[6] X. Zhang and B.A. Wandell. A spatial extension of CIELAB for digital color image reproduction. *SID Journal*, 5:61–63, 1997.



(a) (b)

(c) (d)

(a) (b)

(c) (d)

(a) (b)

(c) (d)

Figure 10. *(a): Input image; (b): Grey for input; (c,d): Colour output at 4bpp,8bpp.*