Physically Meaningful Rendering using Tristimulus Colours

Johannes Meng†∗  Florian Simon‡∗  Johannes Hanika§  Carsten Dachsbacher¶

Karlsruhe Institute of Technology
∗joint first authors.

Abstract
In photorealistic image synthesis the radiative transfer equation is often not solved by simulating every wavelength of light, but instead by computing tristimulus transport, for instance using sRGB primaries as a basis. This choice is convenient, because input texture data is usually stored in RGB colour spaces. However, there are problems with this approach which are often overlooked or ignored. By comparing to spectral reference renderings, we show how rendering in tristimulus colour spaces introduces colour shifts in indirect light, violation of energy conservation, and unexpected behaviour in participating media.

Furthermore, we introduce a fast method to compute spectra from almost any given XYZ input colour. It creates spectra that match the input colour precisely. Additionally, like in natural reflectance spectra, their energy is smoothly distributed over wide wavelength bands. This method is both useful to upsample RGB input data when spectral transport is used and as an intermediate step for corrected tristimulus-based transport.

Finally, we show how energy conservation can be enforced in RGB by mapping colours to valid reflectances.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture;

1. Introduction
Physically-based rendering in computer graphics [PH10] solves the radiative transfer equation [Cha60]. To achieve this for applications such as architectural design, product visualisation, and special effects in the movie industry, Monte Carlo ray tracing [Erm75, Sob94] is often used.

The input to such rendering systems may consist of 3D meshes for the shapes, material properties, and textures for the colours. Following Grassmann’s law, these are usually stored in a tristimulus format, i.e. three values define a colour per texel [WS82]. The most well-known and most frequently used colour space for textures is sRGB [Rec02], which we
refer to as Rec.709 for the linear (non-gamma corrected) version.

With high-quality wide-gamut output devices being widely available, but also for lossless storage and post-processing, there is a need for wide-gamut colour spaces. Currently, we can observe an ever growing abundance of different colour spaces, such as AdobeRGB, ProPhotoRGB and lately the *academy colour encoding specification* (*ACES*) colour space used in the movie industry. These tristimulus-based colour spaces provide a very large gamut; in fact, ACES covers the whole range of visible colours. However, this comes at the cost that some values \( \begin{bmatrix} 0, 1 \end{bmatrix}^3 \) represent physically impossible colours, i.e. *imaginary stimuli* which lie outside the spectral locus in the chromaticity diagram.

While it is obvious that such values cannot result in physically meaningful rendering when used as reflectances (e.g. to define a bidirectional scattering distribution function, BSDF), it is less obvious that even values inside the spectral locus do not always represent physical reflectances [Sch19]. This is due to the fact that a colour can either have very saturated chromaticity or be very bright.

We will use the term *solid of valid reflectances* to describe the set of all visible colours for which energy conserving spectra (i.e. \( \rho(\lambda) \leq 1 \) for all wavelengths \( \lambda \)) can be found. Unlike emission spectra, naturally occurring reflectance spectra are generally smooth [Mal86]. Therefore, we similarly introduce the *solid of natural reflectances*. It contains all colours for which a smooth and energy conserving, or physically plausible, reflectance spectrum can be found.

We also thoroughly analyse the differences between spectral and RGB transport (assuming same-speed transport, i.e. no fluorescence) with regard to ensuring physically plausible reflectances for both cases. Physically, light-matter interaction can be modelled as a component-wise multiplication of the incoming spectral radiance with a reflectance spectrum (i.e. the BSDF depends on the wavelength \( \lambda \)). The results of tristimulus transport, however, heavily depend on the working colour space [Agi11]. Saturation, brightness and hue of indirect light depend on this “arbitrary” choice, since a component-wise spectrum multiplication is replaced by a component-wise tristimulus multiplication. Note that this cannot be corrected with a look-up table after rendering, since the colour shift will depend on the number of interactions of light before reaching the camera.

In summary, the contributions of our paper are:

- We show how tristimulus transport is not in general energy conserving in the physical sense, even though the methods converge mathematically (Sec. 2).
- We propose a fast method to compute smooth spectra from XYZ tristimulus input values which behave closely to natural spectra in light transport simulation (Sec. 3).
- We introduce a mapping technique for physically plausible reflectances. This is essential for tristimulus workflows (such as traditional RGB texture painting or RGB output of shading languages) to ensure that produced texture data results in energy conserving BSDFs (Sec. 4).
- We analyse three strategies to perform this mapping: a) trivial but fast clamping of the corresponding spectrum, b) preserving the exact chromaticity of the input, and c) with minimal CIE \( \Delta E \) colour distortion (Sec. 4).
- We show how to design a correctly colour managed rendering pipeline based on RGB transport (Sec. 5).

2. Background and Previous Work

*Colour Spaces* Tristimulus colour spaces are based on the colour matching functions [Wit28, SG31]. Our implementation relies on the recommendations given in [CIE04]: we use the 1931 2°.5mn steps colour matching functions \( x(\lambda), y(\lambda), z(\lambda) \) defined over the abridged interval \( \lambda = [380, 780] \text{nm} \), and integration by summation.

There is an abundance of colour spaces available today. We are interested in RGB-colour spaces (such as sRGB or Rec.709 [Rec02]), since these are commonly used to describe input colour to rendering systems (as opposed to CMYK-like spaces, which are commonly used for printing). A colour space is defined by the three *primaries*, which define what exactly red, green, and blue mean. This also fixes the location and shape of the gamut, as well as the *white point*, which is considered the neutral colour of the space. For the sake of describing reflectances, we consider only linear RGB colour spaces (i.e. scene-referred, no gamma correction or tone reproduction curves are applied) and use the equal-energy white point (CIE illuminant E, a constant spectrum over the visible wavelengths, to not bake any illumination tint into the surface properties).

To be able to compare these spaces, the *XYZ* colour space was introduced as a standard profile connection space [FBH98]. From there, \( (x, y) \) chromaticities are derived as \( x = X/(X + Y + Z) \) and \( y = Y/(X + Y + Z) \). An interesting special case is when \( X + Y + Z = 1 \) (unity brightness), since then \( X = x \) and \( Y = y \), which will simplify our reasoning in Sec. 3. The *XYZ* white point is fixed at illuminant E \( (x = 1/3, y = 1/3) \), which makes this space suitable for us to work with. For colour fundamentals, we refer to the textbook by Wyszecki and Stiles [WS82].

When converting between colour spaces, the different sizes of their domains in \( (x, y) \) space give rise to the problem of gamut mapping. There are very many different approaches in literature [Mor08]. Related to these, we describe techniques to map arbitrary *XYZ* colour input to the solid of valid reflectances.

Colour constancy methods [For90, FH00] try to avoid colour casts in output images due to coloured light sources, specifically light sources of different colour temperatures,
by ensuring the output lies within an a-priori known locus of valid reflectances. Similarly, Ward and Eydelberg-Vileshin [WEV02] propose rendering in a white-point adapted Sharp colour space to avoid perceived colour casts in the output image. In contrast, we strive to avoid such colour casts, since we believe that they can be an important part of the artistic process. This especially applies to physically based rendering, where multiple different illuminants may be chosen, and their effect should be faithfully reproduced.

Spectral Rendering When simulating light transport per wavelength, colour management simplifies substantially. The rendering equation [Kaj86] is solved per wavelength, either by transporting full spectra (discretised into bins or with basis functions [Mey88, Pee93, SFCD99]), continuous wavelength sampling, or transporting several wavelengths at the same time, combining the results via multiple importance sampling [VG95, WND’14]. The problem of upsampling RGB input textures to spectral data still needs to be solved in this case, including the issue that RGB input might not represent valid reflectances. We provide the theory and tools to perform both fast upsampling from RGB to spectral power distribution in almost the whole visible gamut, and mapping to the solid of valid reflectances (solid mapping).

Spectral Energy Conservation While upsampling RGB or XYZ input to spectral data representing radiance (or emission) is possible without considering energy conservation, it needs to be taken into account for surface reflection (BSDF). Energy conservation holds per wavelength, e.g. in the case of a diffuse BSDF we have for every \( \lambda \) individually:

\[
1 \geq \int_{\Omega} \rho(\lambda) / \pi d\omega \downarrow = \rho(\lambda) \int_{\Omega} 1 / \pi d\omega \downarrow = \rho(\lambda). \tag{1}
\]

This bounds the spectral reflectance distribution \( \rho(\lambda) \). Similarly, the coefficients of other BSDF models can be bound to ensure energy conservation.

Spectral Upsampling There are several approaches to convert XYZ values to spectral power distributions in literature.

MacAdam [Mac35a, Mac35b] implicitly defines an XYZ to spectrum conversion in his work on achieving the highest possible brightness for a given colour saturation in printing. The method is not limited to a specific input, but produces spectra which are only suitable for rendering to a limited extent. MacAdam’s spectra consist of a single box function wrapping around the interval of definition \( \Lambda \) (i.e. only one rising and one falling edge). Fig. 2 shows a comparison of the shapes of the corresponding spectra, and Fig. 3 shows an unintuitive colour shift in the indirect illumination when using box spectra.

Glassner [Gla95, p.104] gives a family of conversions which take a set of three basis functions as input, for instance the monitor response curves or the first three Fourier bases.

These methods can produce negative spectral power, which is a severe problem in physically based light transport simulation. Negative power will have to be clamped, resulting in uncontrolled colour shift.

The method by Smits [Smi99] is practical and widely adopted. It is based on a small set of ten precomputed box bases. They are optimised for smoothness and to match Rec.709 primaries adapted to white point E, while obeying spectral energy conservation. Since these contradicting goals can only partly be met [Sch19], the resulting spectral output can be slightly larger than unity. Also, the method does not support round trips: converting a colour to spectrum and then back will result in a small, but noticeable colour cast. For instance the input \( XYZ=(1,1,1) \) results in a maximum spectrum value of 1.00012, and output \( XYZ=(99945, 99942, 99982) \). This is not the worst case, and the difference seems small. However, with multiple interreflections and in the presence of bright light sources, colour shifts can be perceived. Furthermore, Smits’ method is not designed to work with colours outside the Rec.709 gamut. See Fig. 9 for an analysis of the perceptual error in- and outside this gamut. Finally, the optimisation performed to
obtain the set of box bases intrinsically depends on an RGB working space and does not work well for wide-gamut working spaces. This is discussed further in supplemental material. In our approach, we also use a non-linear optimisation to find basis spectra, but change the interpolation scheme to be independent of the input colour space. Thus, our method can be seen as a generalisation of Smit’s work.

Wang et al. [WXS04] measure a set of reflectance spectra, and extend this set using Beer’s Law. For upsampling, their technique linearly interpolates the basis spectra in three-dimensional XYZ space. The values for the measured spectra are not provided in [WXS04], and so reproducing the results is a challenge since measuring spectra requires special hardware. Also, it is not clear how well the solid of natural reflectances is covered by the extrapolation technique used in [WXS04]. Our method is conceptually similar, but instead of a three-dimensional search, we employ a very simple and efficient two-dimensional scheme.

All the works mentioned so far (including ours) ignore fluorescence at light interaction points, i.e. there is no crosstalk between wavelengths. Fluorescence has been considered in the print context [Cou07], but in the global illumination context, simulating light transport becomes significantly more involved. There have been some works to simulate fluorescence using Monte Carlo techniques [Gla94, WGRK’97, WWLP06, SN07, BHD’08, HHA’10], but none of these implement full-featured bidirectional path tracing yet. We consider extending our work to fluorescence for future work.

3. Converting XYZ to Spectral Data

We are interested in upsampling arbitrary given XYZ input to a spectral power distribution, such that the result is physically meaningful in the sense that it resembles natural spectra as much as possible. In particular, we do not want to be limited by the Rec.709 gamut. Also, the theoretical limit of MacAdam’s spectra are too sharp for our purposes, since spectra derived this way produce unnaturally saturated indirect light with unexpected colour shifts (see Fig. 3). Lastly, the method should be efficient and suitable for evaluation at run-time.

First we consider finding a spectrum \( s(x, y, \lambda) \) for certain chromaticity coordinates \((x, y)\) with unity brightness (i.e. \( b = X + Y + Z = 1 \)). For arbitrary brightness \( b \), the spectrum \( s(X, Y, Z, \lambda) \) can be obtained by simple scaling, since the integral operator converting the spectrum to XYZ is linear. In a one-time precomputation we find the smoothest possible such spectra by performing a brute-force optimisation, described next, over sample points in the \((x, y)\) chromaticity space.

3.1. Optimisation

We search for discretised smooth spectra with 5nm steps in \( \lambda \) which match the input \((x, y)\) chromaticity. More precisely, we minimise \( f(x) = \sum_{i=1}^{N} (x^{i+1} - x)^2 \) subject to \( x^i \in [0, 1000] \) and \( \sum_i 5nm \cdot x^i = (x, y, 1 - x - y) \), where \( x^i \) is the spectral value of bin \( i \) and \((x^i, y^i, \lambda^i)\) are the tabulated colour matching functions. We use Python’s scipy package to perform the optimisation (scipy.optimize.minimize with SLSQP, based on [Kra88]), and always start with an equal energy white spectrum.

The resulting spectra are smooth and mostly uni-modal, except for samples near the purple line, where two modes can be observed. There is some slight noise in the output due to the numerical minimisation, which also depends on the starting point, but overall we found the process to be stable (cf. Fig. 6, in which the maximum brightness was determined by per-pixel optimisation, and supplemental material, where we show our precomputed spectra). As opposed to MacAdam’s optimal spectra [Mac35b], which consist of a simple box function, our spectra are more like natural spectra in terms of smoothness (see Fig. 2).

3.2. Fast Implementation: Grid-based Upsampling

Since the optimisation process is slow, we precompute spectra for a set of points \((x, y)\) offline (see Fig. 4) and store the resulting spectra \( s_i(\lambda) = s(x_i, y_i, \lambda) \). Spectra \( s(x, y, \lambda) \) are then interpolated linearly from this precomputed set:

\[ s(x, y, \lambda) = \sum a_i(x, y) s_i(\lambda). \] 

(2)

To determine the weights \( a_i(x, y) \) quickly, we mostly use a regular grid topology with bilinear interpolation for inner points, and barycentric interpolation of triangle fans at the boundary of the spectral locus.

To better fit a grid to the spectral locus, we first rotate the \((x, y)\) coordinate system into a new space \( x^*y^*z^* \), which first aligns the \( x \) axis with the purple line and then places the origin at the illuminant E white point \((1/3, 1/3)\).

The location of the grid points is then determined by regular sampling inside the spectral locus, where we make sure the most saturated red (lower right corner) lies exactly on a grid point. This makes sure the white point and the purple line can be sampled appropriately. The set is extended by a few points on the spectral line which form triangle fans together with the inner grid points. To smooth out interpolation artifacts, we also insert one more point at the center of gravity of all points generated in each boundary cell (see Fig. 4 for a grid with \( 4 \times 6 \) cells between white and red).

Finally, all grid locations are scaled towards the white point by a small adjustment factor \( 1 - \varepsilon \). This step avoids numerical instabilities in the optimisation for colours near the spectral locus, but it also means that we cannot upsamp some very saturated and monochromatic colours. This is further explained in supplemental material.

Simple calculus shows that each interpolated spectrum...
Figure 4: Visualisation of our grid of optimised spectra, overlaid on a chromaticity diagram, and the Rec.709 gamut. We use precomputed spectra at 186 sample points (green/white) to interpolate spectra at arbitrary \((x, y)\) locations. There are \(4 \times 6\) grid cells between white and red.

\[
s(x, y, \lambda) \text{ will result in the correct chromaticity } (x, y):
X = \int_{\lambda} \bar{\chi}(\lambda) \cdot \sum_{i} a_{i} s_{i}(\lambda) \, d\lambda
= \sum_{i} a_{i} \int_{\lambda} \bar{\chi}(\lambda) \cdot s_{i}(\lambda) \, d\lambda.
\]

if the weights \(a_{i}\) have been chosen such that \(x = \sum_{i} a_{i} x_{i}\) and \(y = \sum_{i} a_{i} y_{i}\). From this we can see that if we interpolate the \(x, y\) chromaticities linearly, we will also interpolate the spectra linearly and vice versa. These spectra also satisfy \(X + Y + Z = 1\) if the input spectra \(s_{i}(\lambda)\) satisfy this condition:

\[
X + Y + Z = \sum_{i} a_{i} x_{i} + \sum_{i} a_{i} y_{i} + \sum_{i} a_{i} z_{i}
= \sum_{i} a_{i} (X_{i} + Y_{i} + Z_{i}).
\]

Note that we can always choose the brightness \(X_{i} + Y_{i} + Z_{i} = 1\) for all the sample spectra \(s_{i}(\lambda)\) by simple uniform scaling. Visually this means that we stay on the same 2D-slice through the spectral gamut, on the plane of \((x, y)\) chromaticities which is used to visualise gamuts in 2D such as in Figs. 4 and 9.

4. Solid of Natural Reflectances

Now that we have methods at hand to create smooth spectra by upsampling arbitrary tristimulus color spaces, we can use these methods to define the solid of natural reflectances, and introduce mapping techniques to map arbitrary input to this solid (solid mapping), thus ensuring energy conservation.

Following MacAdam \[Mac35a\], we can define the solid of valid reflectances using any method to upsample \(XYZ\) to spectra: For every chromaticity \((x, y)\) we compute a spectrum, and rescale it by a brightness \(b(x, y)\) such that \(b \cdot s(x, y, \lambda) \leq 1\) for all \(\lambda\). This value \(b(x, y)\) is the maximum brightness that a colour of this chromaticity can achieve to obey

Implementing MacAdam’s Spectra

For comparison, we also implemented an \(XYZ\) upscaling algorithm based on the theoretical spectra derived by MacAdam \[Mac35a\]. In a pre-pass, we store a 1024 \times 1024 texture as look-up table, where every pixel represents one \((x, y)\) chromaticity pair. In a simple nested loop, we enumerate all \((\lambda_{1}, \lambda_{2})\) pairs which define all box functions: \(\lambda_{1} \in \Lambda\) is the location of the raising edge, \(\lambda_{2} \in \Lambda_{2} = (\lambda_{1}, 730 + (730 - 380))\) the falling one, defined on the extended domain \(\Lambda_{2}\) to simplify wrapping around the interval of definition.

For each such pair, we use precomputed cumulative distribution functions of the colour matching functions to quickly compute the integrals to obtain \((X, Y, Z)\). From that we derive the chromaticity coordinates of the sample and bin the output data (brightness \(X + Y + Z\), \(\lambda_{1}, \lambda_{2}\)) into the three colour channels of the texture.

During run-time, we do a simple look-up based on the desired chromaticity and reconstruct the spectrum by testing each wavelength \(\lambda\) against the box in the extended domain: \(s(\lambda) = 1\), if \(\lambda_{1} < \lambda \leq \lambda_{2}\) or \(\lambda_{1} < \lambda + (730 - 380) \leq \lambda_{2}\), otherwise \(s(\lambda) = 0\).
energy conservation, assuming this underlying model for spectral distributions.

Using MacAdam’s approach and inserting the theoretical optimum (box spectra) results in the largest possible solid, but this will not be a tight bound around the colours of smooth spectra of real materials, and thus look unnatural (see Fig. 3). If we instead make use of our spectra optimised for smoothness, we obtain a smaller solid which we refer to as the solid of natural reflectances. See Fig. 5 for a comparison of the two solid shapes.

We can now use this solid to map a given colour in \(XYZ\) to a colour within the solid of natural reflectances (solid mapping).

**Scaling** Scaling the spectrum uniformly such that its maximum is mapped to one preserves smoothness and chromaticity exactly, but may alter brightness drastically. We can use the precomputed map in Fig. 6 as a scaling factor directly:

\[
S(s(x, y, \lambda)) = s(x, y, \lambda) \cdot b(x, y), \quad \text{or} \quad S(X, Y, Z) = (X, Y, Z) \cdot b(x, y).
\]  

Note that using the precomputed \(b(x, y)\) allows to apply scaling directly to the \(XYZ\) data without explicitly computing the corresponding spectrum.

**Clipping** Clipping the spectrum at a value of 1 yields a valid reflectance spectrum, however, potentially introducing kinks into the spectrum:

\[
C(s(x, y, \lambda)) = \min\{1, s(x, y, \lambda)\}. \quad (7)
\]

This is particularly useful for renderers which implement transport based on a single wavelength, continuously chosen by Monte Carlo, because it can be performed for each \(\lambda\) individually.

**Minimal Distortion** This method is more involved than the other two and useful as a reference. It achieves minimum colour shift (in terms of CIE 1976 \(\Delta E\) error) by changing both brightness and chromaticity.

\[
M(s(x, y, \lambda)) = s(x', y', \lambda), \quad \text{with} \quad (x', y') = \arg\min \left\| (X', Y', Z') - (X, Y, Z) \right\|_{\Delta E} \quad (8)
\]

Since CIE 1976 \(\Delta E\) is the Cartesian distance in the Lab colour space [CIE08], we are geometrically searching for the shortest distance by finding an orthogonal line from the surface of the 3D solid of valid reflectances to the original point \((X, Y, Z)\) (in Lab space).

We do this using a predictor/corrector method: The predictor projects the current Lab point onto the surface of the solid by converting it to \(XYZ\), applying scaling by \(b(x, y)\) and converting back to Lab. The corrector estimates the normal of the solid surface in Lab space via finite differences and predicts a new point where the distance between the input point and the plane defined by this normal is the smallest (see Fig. 6). The process repeats and usually converges after 3-10 iterations.

Note that only clipping needs to go through spectral representation explicitly, if tristimulus values are to be transformed. All three strategies can be performed for MacAdam’s limit as well as for smooth spectra, by using the corresponding dense map \(b(x, y)\) (Fig. 5 and 6, left). Smits’ method already includes implicit energy conservation and the respective dense map would be almost constant.

### 5. A Colour-Managed Pipeline

**Input** Often input textures are given as sRGB, or linear Rec709, i.e. relative to a D65 white point. Baking a white point into a reflectance is meaningless, since it is a property of the illuminant. Instead, we split up the informa-
With this setup, an input colour given as \( \text{Rec.709} \) \((1, 1, 1)\) will result in an equal energy reflectance spectrum, and look like the input texture did on screen while painting it, when illuminated with a D65 light source (see Fig. 7).

**Input Solid Mapping**
This input texture data then may need to be mapped into the solid of natural reflectances. Any method will work, but yield slightly different results. In particular, one has to decide whether colour differences or absolute values of correct input are more important to a specific application (see Fig. 11). Note that for parameters of volume transport, this solid mapping needs special attention, since scattering cross sections and mean free paths do not directly describe colours. See Sec. 6 for a discussion.

**Spectral Rendering**
When simulating light transport per wavelength, the tristimulus input needs to be upsampled to spectral representation during runtime. When using our grid-based method, we need to re-clamp to \( s(x,y,\lambda) \leq 1 \) due to the approximate nature of the interpolation between grid points and because the solid shape is not convex. We still recommend mapping the input up front with scaling or using the \( \min \Delta \varepsilon \) method, to avoid grid artefacts.

**Tristimulus Rendering**
When not using spectral transport, it is most physically meaningful to perform transport in colour spaces such as CIE RGB or wide-gamut RGB, since the primaries are mono-wavelength spectral intensities (at 435.8, 546.1, 700nm and 450, 525, 700nm, respectively). This corresponds to simulating three times single-wavelength spectral transport. Unfortunately this produces results which are very different from simulating the full spectrum, as will any other RGB working space [MMC15] (see Fig. 8 and Fig. 1 in our supplemental material). This is an inherent limitation of tristimulus transport, but we can now provide values which correspond to natural reflectances.

**Output**
Converting to the output colour space is straightforward, using the dot product with the colour matching functions for spectral data, or the colour matrix to convert RGB to \( \text{XYZ} \). This can be stored in the framebuffer without clamping for further processing and handed over to display colour management.

### 6. Results
All images in our figures are encoded in sRGB for display on commodity monitors, even though the issues with tristimulus reflectances arise especially for colour spaces with wider gamut. We recommend viewing them on screen, some detail may be lost in print.

<table>
<thead>
<tr>
<th>method</th>
<th>MacAdam</th>
<th>Smits [Smi99]</th>
<th>grid (ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>time for 10k runs</td>
<td>1700 ms</td>
<td>3342 ms</td>
<td>3314 ms</td>
</tr>
</tbody>
</table>

**Table 1:** Speed comparison of the upsampling methods, fastest of three runs on an Intel i7-2630QM CPU @ 2.00GHz.

**Upsampling Speed**
We implemented all three \( \text{XYZ} \) to spectrum upsampling methods (MacAdam [Mac35a], Smits [Smi99] and the grid-based approach described in Sec. 3) in C. A performance comparison can be seen in Tab. 1. There is no noticeable speed impact of the grid-based approach as compared to Smits’ method. Our implementation of MacAdam’s spectra is very fast, but suffers from quantisation artifacts (see Fig. 9).

**Upsampling Precision**
Fig. 9 analyses the precision of the three upsampling techniques, converting \( \text{XYZ} \) to spectral power distribution and back. The grid uses only a moderately fine \( 6 \times 4 \)-cell quantisation between white and red here, but the linear interpolation stage still matches the colours exactly (see Eq. (3)).

Smits’ method is not designed to work outside the \( \text{Rec.709} \) gamut, and trades round-trip precision for validity of reflectances even within this gamut. This practical algorithm combines the two goals, optimisation to match colour and energy conservation, in only one step, meeting both goals only partly in the end.

The MacAdam-processed image in Fig. 9 is computed by dividing out the brightness \( b(x,y) \) which we store in the pre-computed map, to undo the baked-in solid mapping by scaling. MacAdam’s construct is of high theoretical interest, but using it in practice is limited by the precision of the colour...
matching function tables (even at 1nm bin size). The smallest possible step is to include (or not include) a wavelength bin, incurring substantial steps in \((x,y)\) chromaticity (hence the quantisation artifacts in Fig. 9). Improving those by linear interpolation would require a tessellation of the \((\lambda_1, \lambda_2)\) in \((x,y)\)-space, essentially resulting in the same algorithm as we use for our smooth spectra on a grid, but with more complex topology.

**Mapping to Valid Reflectances** Fig. 10 shows the importance of solid mapping for reflectances. Even at moderate brightness of \(X+Y+Z = 1\) the red corner of the Rec.709 gamut does not define valid reflectances and needs to be adjusted. This can also be observed in 3D renderings (Fig. 1). Fig. 10 also shows the steep falloff in brightness for colours with \(X+Y+Z = 3\) near the white point \((1,1,1)\). The more complex method optimising for \(\Delta E\) pays off especially for very saturated tones close to the spectral locus. The grid-based solid mapping methods show some grid interpolation artifacts. We propose that even in a spectral renderer, the input should be solid mapped before upsampling.

**Continuity of the Mapping** Fig. 11 schematically illustrates three mapping strategies (\(\min \Delta E\), scaling, and matrix transforms such as [For90]). All three are \(C_1\) continuous, but \(\min \Delta E\) and scaling introduce kinks in the derivative. Both these map multiple colours outside the solid of valid reflectances to the same output, which might be undesirable. The matrix transform does not introduce these drawbacks, but on the other hand might result in a much smaller output solid and also change perfectly valid input reflectances. For maximum colour accuracy, we think it is best to leave valid reflectances exactly as they were and inform the artist instantly about invalid colours during texture painting.

**Surface Transport** The scene in Fig. 1 consists of four colourful chairs, with diffuse reflectance defined as Rec.709 \((1.00, 0.1), (1.0, 1.1), (0.1, 0.1, 1)\) and \((0.1, 1.0, 1)\). We show spectral and RGB renders of all combinations of upsampling methods and solid mapping techniques in supplemental material.

**Volumetric Transport** In a render with participating media (Fig. 12), an improvement in tonality can clearly be observed. We rendered the Buddha statue, which is 50cm in height, with a fixed mean free path of 15mm, constant for all wavelengths. It is unclear how to convert mean free paths given as RGB to spectral, since the effect on colour is quite different to a simple multiplication. Thus, we only analyse the effect of defining the scattering albedo \((\sigma_t/\sigma_s)\) of a homogeneous medium as a colour. The input
### Figure 11: Side view of the solid of natural reflectances, with three mapping strategies, illustrating the smoothness of the mapping with respect to albedo. Every line represents input of constant chromaticity coordinate \((x, y)\) at different albedo and how the mapping affects these colours.

The small insets represent the four original images without solid mapping, using CIE RGB transport or spectral upsampled with one of the three methods. Note how the original RGB render looks very flat and shows almost no tonal detail. The grid-upsampled original shows a black image: due to numbers larger than the 32-bit floating point range, the render diverged. Smits’ version shows a strong purple colour cast in the blue statue, and MacAdam’s spectra diverge as well, albeit more slowly.

Spectra generated with Smits’ method are by definition almost energy conserving, and so neither clamping nor scaling have a big effect here. Solid mapping via min \(\Delta E\) uses again the MacAdam spectra as a basis, and the smooth spectra for the grid-based method.

The greatest effect of solid mapping is visible in spectral transport using the grid method, which is to be expected. Scaling the spectra works best here, both clamping and the min \(\Delta E\) method exhibit a colour cast. In participating media this is much more severe, since colour saturation accumulates over many more light bounces than in the surface transport case. Note that the RGB version shows almost no detail on the Buddha shape except for what is caused by the Fresnel layer on the interface (this remains true even when stopping down and applying a contrast curve).

### 7. Conclusion and Future Work

We showed that today’s RGB-based rendering pipelines are in general not energy conserving in a physical sense, even though no energy is produced such that the values per RGB channel would increase. That is, the images converge but have no physical counterpart: no real material could produce such colour. We introduced a set of tools and methods to perform solid mapping to the solid of valid reflectances.

Among these tools is a practical XYZ to spectrum up-sampling algorithm which enables fast, per-wavelength access to the spectral power distribution, accepts almost the whole space of visible colours as input, and allows for precise round-trips back to XYZ.

To define the space of natural reflectances, we searched for smooth spectra, assuming no cross-wavelength interactions. Materials can appear brighter and more saturated with fluorescent dyes. While this has been considered in the corresponding literature concerned with printing, light transport becomes significantly more complicated to implement in this case and the effect on colour in indirect light is not easily described in closed form.

The choice of our smoothness term determines the shape of the solid of natural reflectances. While it is motivated by [Smi99], it is arbitrary and there may be a term that produces brighter, yet physically plausible, reflectance spectra.

The grid quantisation of our method comes with slight artifacts when solid mapping has to be performed after upsampling. If this is an issue for an application, it is simple enough to increase the grid resolution.

Source code for optimisation and upsampling can be found along with the supplemental material.
References


[Sob94] SOBOL’ I.: A Primer for the Monte Carlo Method. CRC Press, 1994. 1


© 2015 The Author(s)

Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.