Color Scrambling for Secure Digital Content Distribution

Peter Morovič, Ján Morovič, Michel Encrenaz, Jordi Vilar, Jordi Arnabat; Hewlett Packard; Sant Cugat del Vallés, Spain

Abstract

In order to distribute creative image content, authors go to great lengths to safeguard that it is used and reproduced in the way they want. In this paper we propose to use the framework of ICC color management in a novel way, in order to provide a means for determining two aspects of image reproduction: 1. on what devices and media it can be printed and 2. who authors want their content to print. We achieve this by creating a custom pair of ICC profiles, one part of which acts as a "public-key" and the other as a "private-key", while the color management engine acts as the encoder/decoder. The key to this approach is to depart from the pre-requisite of a common Profile Connection Space and instead generate a multitude of encrypted spaces. If the profile used to encode an image uses the same encrypted space as the profile used to decode it, the image is reconstructed without error, if this is not the case the reconstructed image is unusable as the colors are scrambled. The main benefit of this approach is that it requires no changes to the typical workflow of converting between color spaces using existing software, while affording control over content in a safe and easy way.

Introduction

With more and more image content being distributed over the internet by means of cloud–based solutions (such as www.snapfish.com or www.shutterfly.com), the problem of creative content management is becoming increasingly important. Creative customers such as professional photographers or fine artist are known to be very keen on keeping control over the way their content is presented and reproduced. In the words of Magnum photographer Guergui Pinkhassov: "*Sometimes I have not even recognized my own photographs. I have even hesitated to call them my own. […] Whoever controls the editing of a photographer, controls his fate.*" [1] (note that by 'editing' the process of developing and printing a photograph is referred to here).

In this paper we focus specifically on the problem of how images are handled in the printing context and present a method that gives some control to the author over how their content is reproduced, specifically, what device/paper it is printed/viewed on and what rendering intent is employed.

There are numerous approaches that address the broader topic of safeguarding creative content, for example Nikon's Image Authentication Software [2] that enables detecting the alteration of images (for high-end Nikon DSLRs only), or digital watermarking solutions such as Digimarc [3], shipping with Adobe Photoshop as a plug-in, that embed information in the image for author identification or copyright purposes. Such solutions however, do not prevent uncontrolled reproduction and in the worst case can even be content destructive. Another alternative used is to add visual watermarking with a copyright message, but this is counterproductive in the case of distributing image content with the aim to enable a determined type of reproduction.

Our approach instead is one that is non-destructive and focuses on key aspects that can be chosen at the point of printing an image. Furthermore, it is straightforward to use, as it doesn't require custom tools to be employed by the end-user. We achieve this by departing from a key principle of ICC Color Management in a controlled way, by creating a custom pair of profiles that give desired results when matching, and unusable output in any other case, mimicking the mechanisms of public-key, private-key encryption. In doing so we are providing authors all the control associated with ICC profiles that are both device and media specific (in case of printers).

The following section provides some background on ICC color management and specifically the elements that we employ for out method, the next two sections then detail our new approach showing results. Finally we conclude the paper by outlining the main benefits of the proposed method.

Background

Color management is a ubiquitous part of reproducing color content and at least for purposes of fine art and professional photography its employment can largely be taken for granted. Print service providers as well as authors themselves have been exposed for years to the inevitable necessity of controlled color in order to achieve repeatable, faithful and desired reproductions of photographic or fine-art color content. By far the most popular means to exercise control over color is that of the International Color Consortium's (ICC) color management framework. Operating systems, color devices (cameras, printers, scanners), imaging software as well as some web browsers are now well equipped to handle content tagged with ICC profiles throughout.

The ICC framework proposes the use of profiles associated with devices and/or content. It provides the ability to communicate color via a Profile Connection Space (PCS), representing colorimetry (e.g. CIE XYZ or LAB), the *lingua franca* among all proprietary device representations of color. Thus an image's color is interpreted thanks to an associated source profile (e.g. sRGB) and employing a color management engine it can be transformed to a destination color space (e.g. some device CMYK) via the intermediate PCS. A fundamental principle of this workflow is that a device's profiles are independent and agnostic of other devices and a transformation between any two is defined. [4] The key to this mechanism is thus the intermediate, common PCS.

Figure 1. *ICC Color Management communication via a common PCS.*

Each profile then provides a means to transform device color content into the PCS – the forward, *AToB* transform, and back

into the device's own color space – the reverse, *BToA* transforms. In order to encode these transformations, there are four mechanisms that can be employed: a 3x3 matrix (if the PCS is in CIE XYZ space), two linearization tables (or a gamma curve) that prefix and postfix a Color Look-Up Table (CLUT)[1]. The order in which they are applied both in the forward and reverse direction are shown in Figure 2 below.

These elements are used by Color Management Engines (CMMs) such as Adobe's ACE, the Apple CMM, littleCMS [5] and others, to transform colors on the fly. While there are differences among CMMs (such as interpolation schemes employed within the CLUTs), so long as in the context of any one chain of transformations the engine is the same, the results are consistent.

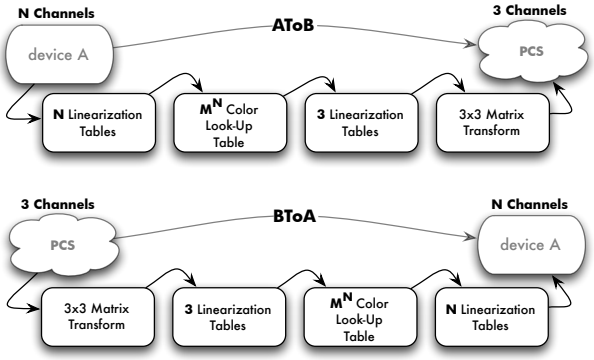

Figure 2. *Internal mechanism of transforming device color co-ordinates (e.g. RGB, CMYK, …) to PCS color co-ordinates: input and output linearization tables , a Color Look-Up Table and a transformation matrix.*

Color Scrambling – Cryptochrome

Our method exploits the principle of a common PCS and turns it around: if in order to reproduce an image correctly the image's embedded (input) ICC profile and a device's (output) destination ICC profile communicate via a common PCS, then if they don't share a common PCS, the image cannot be reproduced correctly and is subsequently useless. The way we propose to exploit this is by creating a multitude of custom, non-standard connection spaces that still encode colorimetry, albeit in an encrypted, scrambled way.

The idea takes its inspiration from the public–private key encryption principle whereby content is encoded using a public key that is freely available, while to decode it, a private (limited access) key is needed. [6] By designing custom ICC *abstract profile* pairs (where abstract profiles embody transformations to and from a colorimetric space), one of which is used to *scramble*, deform the colorimetric space[2] (i.e., turn XYZ to XYZ'), and the second to decode, *unscramble* it. Pre-fixing a chosen device profile with a XYZ → XYZ' scrambling transform (altering the BToA tags) results in a standard ICC profile that can easily be applied to an image e.g. in Photoshop, while post–fixing it with a XYZ' → XYZ unscrambling (altering the AToB tags) is the counterpart decoding profile. The former is then the equivalent of a public key or image-key (i.e. freely available), while the latter, the private key or printer-media-key is private (i.e. lives inside a print service providers printer alone) – we refer to such a pair of profiles as a *Cryptochrome ICC profile pair*.

Hence the key to this method is a shift from a common, perceptually relevant PCS to a multitude of new, custom and perceptually scrambled spaces by means of the XYZ to XYZ' (or LAB to LAB') transforms. Figure 3 illustrates how the proposed method differs from that of the standard ICC approach shown in Figure 1.

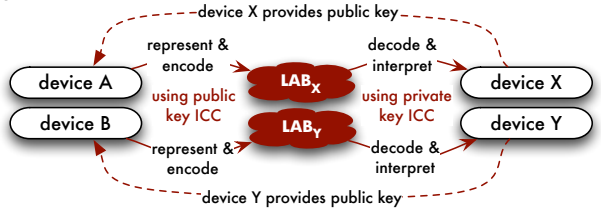

Figure 3. *The proposed, Cryptochrome workflow where devices on the left need to have the public keys of the destination devices (right).*

A typical workflow is to obtain the public Cryptochrome ICC profile generated and provided for example by a print service provider and convert images into the scrambled domain using the profile in much the same way as soft-proofing. Since the Cryptochrome profiles need to be generated in a non-standard way, it is not possible to circumvent this method easily. Consequently, if the scrambled images were printed or viewed without the correct private Cryptochrome profile counterpart, the outcome would be unusable because of the color space deformation effected by the scrambling transform.

Scrambling Mechanisms

The next consideration is to be made about mechanisms to devise scrambling transforms. All elements used in ICC profiles as shown in Figure 2 can be used: matrix transformation, linearization tables and CLUTs. However, the overall scrambling transform has to satisfy two conditions, it has to be *invertible*, and *symmetrical*. The first condition is needed in order to ensure the non-destructive nature of the scrambling – if original content is scrambled and then unscrambled, it should match the initial original without error. The second condition, linked with the first, means that concatenating a pre-fixed profile and a post-fixed one the same profile results in no effect. Strictly speaking, the transformation has to be *bijective*, meaning that for every color co-ordinate in XYZ, there is exactly one corresponding color co-ordinate in XYZ' and no unmapped co-ordinate exists in either XYZ or XYZ'. Bijective transforms from a set onto itself are also called *permutations*.

Denoting $l_2(x)$ a function representing the second linearization table, $c(x)$ a function representing the CLUT, $l_1(x)$ the first interpolation and $m(x)$ the matrix transform, the overall scrambling of a color value x into a scrambled value y using all these mechanisms can be written as:

$$y = l_2(c(l_1(m(x))) \quad (1)$$

[1] For simplicity sake we refer to ICC v2 profiles throughout.
[2] Without loss of generality we use CIE XYZ as the PCS.

And the conditions above translate to the existence of the inverses of each component, $m^{-1}(y)$, $l^{-1}{}_1(y)$, $c^{-1}(y)$, $l^{-1}{}_2(y)$ such that when applied in the reverse order (according to Figure 2), they satisfy:

$$x = m^{-1}(l^{-1}{}_1(c^{-1}(l^{-1}{}_2(y)))) \quad (2)$$

If not all mechanisms are employed in the scrambling transforms, they still need to be taken care of (matrices, linearization tables and CLUTs set to map identity) in order not to affect the bijectivity.

Without loss of generality, let us consider the case of having an abstract profile that uses only CLUTs for the transformation between a one dimensional PCS and that these tables are at full resolution, such that for example a 3 bit, single channel device's ICC profile CLUT has 8 nodes (a 16 bit three channel device's ICC would be 65536^3 size). Under these circumstances, any scrambling is valid, including random permutations, such as the example in Table 1 below.

Input	Scrambled Output
0	1
1	3
2	5
3	7
4	0
5	2
6	4
7	6

Scrambled Input	Output
0	4
1	0
2	5
3	1
4	6
5	2
6	7
7	3

Table 1: *A simple scrambling (permutation) on a 3 bit single channel LUT showing the encoding, scrambling direction (left) and decoding, unscrambling direction (right).*

This scrambling is bijective and satisfies the conditions we outlined earlier, it is both invertible (swapping the columns and sorting them according to the indexing space defines the unscrambling) and symmetrical as it is the only element of the transformation. The reason for this is that in this case the CLUT acts as a full dictionary and if value x is encoded as y in the forward direction then using the reverse table for the unscrambling direction, y will give x. Hence, in the case of full resolution tables, any scrambling is valid as all permutations are bijective.

The number of such scramblings depends on the table(s) used and their resolutions, so that in the above example there are 8! (=40,320) possibilities while for an 8 bit full resolution table with a single channel there would be 256! ($=8.6\text{x}10^{506}$) permutations, so the space of possible encodings is sufficient in order to make it difficult to decode. Note that for practical purposes, many of these permutations would not scramble the image sufficiently or noticeably, however, since the mapping is in a colorimetric space (CIE LAB or XYZ) it is straightforward to determine whether a scrambling would produce noticeable departures from the original, using metrics such as CIE DE 2000 [7] or the spatial S-CIELAB [8]. A caveat to take into account in this case is that an 8 bit full resolution three channel CLUT occupies 16.8MB uncompressed, making it costly in terms of space, albeit secure in terms of the number of permutations. Another implicit assumption made here is that the CMM module dealing with a full resolution CLUT doesn't interpolate at any stage (true for littleCMS v2 [5]).

Algorithms to generate random permutations of a finite set in linear time exist, such as Knuth's "P algorithm" [9], a variation of the Yates–Fisher shuffle [10]. Since these algorithms generate the permutations with uniform likelihood and randomly, as well as and due to the space of permutations being very large (i.e. on the order of 10^{506}), the likelihood of generating encryption keys by different users that coincide is negligible.

Figure 4 below shows an original image and two examples of scrambled counterparts using random permutations of an 8 bit LAB to LAB' 256^3 CLUT.

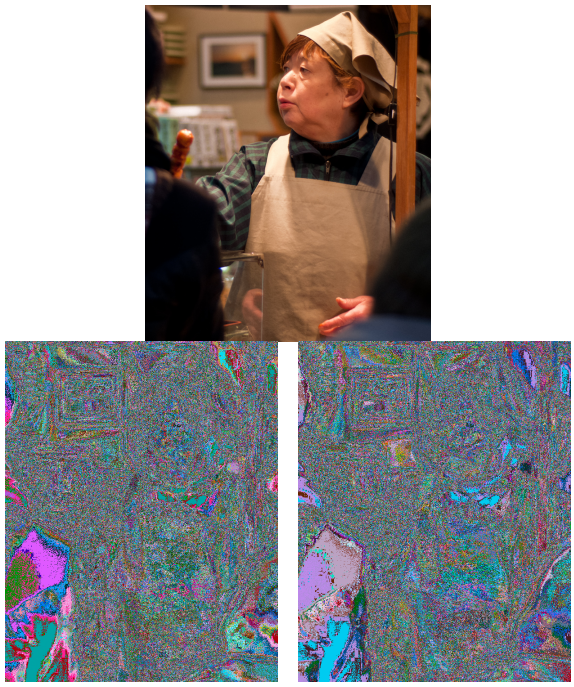

Figure 4. *The original image in sRGB (top) and two scrambled counterparts using full resolution random permutation scramblings shown as sRGB previews (bottom).*

The above approach has a technical caveat in that according to the ICC v2 specifications, the size of the CLUT (i.e. the number of nodes per dimension) is encoded by a single 8 bit value, making it's maximum size just short of the needed 256 nodes. Consequently, in practice it is not possible to use such full resolution scrambling CLUTs at the moment. Instead, what can be done is to use the input and output linearization tables at full resolution and define random scramblings in that domain (much the same way as shown in Table 1). The space of permutations and hence of the scrambling complexity is reduced, as the scrambling transform is performed using three linear vectors instead of a full 3D matrix.

An alternative to the case of using a full dictionary is to stick to common ICC profiles which do not tend to have full resolution tables and rely on interpolation applied by CMMs within lower sampled CLUTs to transform device color to PCS and back. For example, a typical printer profile might use between 17 to 33 nodes per channel in 16 bits. While the constraints outlined above apply equally in this context, their implementation has new

constraints since, for example the above approach of a random permutation would not satisfy bijectivity due to interpolation. In fact, in the case of sampled CLUTs, invertibility means the scrambling transforms have to be *strictly monotonic*. The space of possible monotonic transformations encoded as a CLUT is a continuum and hence of infinite cardinality. Key to this approach is the necessity to change the XYZ' values, instead of permuting them. Any kind of monotonic function can be used in this context, such as polynomials, exponentials, inverses, etc… Furthermore, the complexity of transforms can be increased by using the full sequence of possible mechanisms as outlined in Figure 2 above: matrices, per-channel transformations encoded in the linearization tables and the CLUT itself. The resulting scrambling will not have the random appearance of the example in Figure 4 but instead will be smooth, because of interpolation over a monotonic function.

Figure 5 shows a scrambling transform for the Perceptual rendering intent, post-fixed with an embedded sRGB profile (the public key) and pre-fixed with an output printer profile (the private key).

Figure 5. *A scrambled image (sRGB preview) and the re-constructed image in the target device's RGB space (sRGB preview) with a simple scrambling over a CLUT of size* 33^3.

A visual difference between he original (in Figure 4 on the top) and the reconstructed image (Figure 5 on the right) can be seen here as the original is in sRGB space while the bottom-left, reconstructed image is a soft-proofed sRGB preview of a printer profile's colors, hence gamut mapping took place in this sequence of transformations, resulting in equivalent colors to simply converting the original sRGB image to the destination printer profile. If the destination profile were the same as the source profile (i.e. sRGB to sRGB via LAB ➔ LAB' scrambling) the output would be identical to the input – the Cryptochrome transforms are non-destructive. Note that even though this is a trivial case of scrambling, unless the specific scrambling transform is known, the original image cannot be reconstructed and thus cannot be reproduced faithfully.

Conclusions

Cryptochrome is a mechanism to create ICC profile pairs that match with each other and mismatch with any other profile. This principle enables an individual to determine how their content is dealt with and especially how it is reproduced, while not having to use custom tools and benefiting from the existing processing mechanisms of Color Management Engines. The complexity of this encryption varies depending on the mechanisms employed to define the scrambling (which in turn are constrained to those provided by the ICC profile specs), however in any case it is sufficiently complex to make it difficult to decrypt. As with house locks, all encryptions ultimately buy time, not absolute security.

Acknowledgements

The authors would like to thank Angel Albarrán and Martí Maria for fruitful conversations and comments as well as Andrés Gonzalez, Eduard García and Ana Heredero for their support.

References

[1] *Magmum Stories*, Phaidon Press, Dec. 2004, Chris Boot ed.
[2] *Nikon Image Authentication Software*, see http://goo.gl/QXxL
[3] *Digimarc - Digital Watermarking for Images*, see https://www.digimarc.com/solutions/dwm.asp
[4] Specification *ICC.1:2001-04 File Format for Color Profiles*, available at http://www.color.org/
[5] Maria, M., *LittleCMS: A free color management engine in 100K*
[6] *Public-key cryptography*, see http://en.wikipedia.org/wiki/Public-key_cryptography
[7] Luo, M. R., Cui, G., Rigg, B., *The Development of the CIE 2000 Color Difference Formula: CIEDE2000*, Col. Res. Appl. **26**, pp 340-350, 2001
[8] Zhang, X., Wandell, B., *A spatial Extension of CIELAB for digital color image reproduction*, SID Journal, 1997
[9] Knuth, D., *The Art of Computer Programming: Fundamental Algorithms, (3rd ed.), Reading (MA): Addison-Wesley, 1997*
[10] Fisher, R.A., Yates, F., *Statistical tables for biological, agricultural and medical research (3rd ed.)*. London: Oliver & Boyd. pp. 26–27, 1948

Author Biography

Peter Morovič received his Ph.D. in computing sciences from University of East Anglia, UK in 2002 and holds a B.Sc. degree in theoretical computer science from Comenius University, Slovakia. He has been working as Color Imaging Scientist at Hewlett–Packard Barcelona since 2007. His interests include computer vision, color reproduction, image processing, computational geometry and parallelization.

Ján Morovič received his Ph.D. in color science from the Colour & Imaging Institute (CII) of the University of Derby (UK) in 1998. After working there as a lecturer in digital color reproduction, he joined Hewlett–Packard Barcelona in 2003 as senior color scientist and later master technologist. He served as chairman of the CIE's Technical Committee 8–03 on Gamut Mapping and Wiley and Sons have published his book entitled 'Color Gamut Mapping.'

Michel Encrenaz was born in 1970 in Paris. He got an engineering degree from Ecole Nationale d'Ingénieurs in Tarbes in 1994, and a PhD from the University Bordeaux 1 in 1998. He joined Hewlett–Packard Barcelona in 1999 where he has been working in the R&D Labs of the large format printer division in various positions related to writing system integration, as well as color and imaging technologies.

Jordi Vilar received his B.Sc. degree in Physics from the Universitat de Barcelona in 1999. He has been working as a Software Engineer and Color Imaging Scientist at Hewlett Packard S.L. Spain since 1999. He has published 5 journal and conference articles, and holds several patents in the fields of color science and printing workflows.

Jordi Arnabat received his M.Sc. in computer vision in 2001, holds a B.Sc. in physics from the Universitat Autonoma of Barcelona, Catalonia and an Advanced Project Manager degree from Stanford University, USA. He has been working as a Color Imaging Expert at Hewlett–Packard S.L. Spain since 2008. His interests include color reproduction, image segmentation, 3D reconstruction and computational geometry.