https://doi.org/10.2352/ISSN.2169-2629.2018.26.38

Color color processing

!"#$ &'&()*+, -.-'$ &'&()*+,!&'/)$0 '#121.+3)4.&'$5)-6&+, -'-$7 1891')#+, 1()-'$;1'><1+=-'6)&$> .4?-2-?-'-+@, $?#4B+C1.1D&#)1+=91)#$

!

" #$%&' (%!

Color imaging involves a variety of processing operations, from interpolation, via matrix transformation, to smoothing and predictive modeling. Since colors can be represented as coordinates in color space, the general methods of mathematics can be applied to them. However, if color coordinates are treated simply as generic spatial coordinates, their processing can have undesirable consequences, deriving from a disconnect between the coordinates representing a color and the color formation properties resulting in it. E.g., interpolating among colors with very different lightnesses may lead to a grainy result in print, or varying the interpolation support when processing a transition may lead to unwanted cross-contamination of colorants. To address such challenges, the present paper proposes two color processing algorithms that do take the color properties of processed coordinates into account. They can therefore, in some sense, be thought of as "color color" processing algorithms rather than as geometric or mathematical color processing ones. The consequences of making color-native choices when processing color data then are improved transitions, "purity" and grain.

) ' (*+&, -./ !!

Smoothly transitioning between colors is challenging in print, because of the non-linear way in which inks interact among themselves and with the substrate they are printed on. The more inks an ink set has, the greater this challenge becomes, since more and more disparate ink combination variations need to be well behaved. As a result, artifacts often appear in transitions, which in turn can make prints unsaleable. Such artifacts also reduce the level of predictability that a printer has, including its goodness of soft-proofing, all of which impacts customer satisfaction and total cost of operation. Transition smoothness is an important component of overall print image quality, particularly critical in applications like professional photography, fine art, high-end point of sale and interior decoration.

There are several factors that determine the goodness of color transitions, including ink spectra, ink-substrate material interactions, color separation and ICC profiling. An important constraint here is that key computations that are needed for building color separations from RGB/CMYK to ink channels / NPacs are performed on a purely geometric basis. Interpolation first selects coordinates of points in 3D/4D on the basis of which interpolation proceeds on a geometric basis either involving volumetric relationships in a tessellation or distances.

Such methods are "blind" to the colors that color space coordinates represent and to the preferences there may be in how to combine them, and while they may work well under some circumstances, they may not be robust in general. An example of where generic geometric methods work well is interpolation in regular, cubic grids, especially when these are of sufficient density (e.g., 17^3 RGB samplings or 9^4 CMYK ones). Here a variety of tessellation methods have been proposed and tested previously, including tri-linear, prism, pyramidal and a variety of tetrahedral ones [1-5, 12].

Here, a good, early example of a move away from purely geometric approaches is interpolation in a regular grid in RGB where sub-cubes are analytically tessellated so that their long diagonals, shared by sub-tetrahedra are parallel with the black-white long diagonal of the RGB encoding cube [6]. Other examples are in the context of smoothing, where, e.g., weighted combinations of a look-up table (LUT) node's neighborhood only consider lightness neighbors and not full, 3D neighborhoods, resulting in greater accuracy at a given level of smoothness [7]. Alternatively, in the context of CMYK LUTs, separate smoothing of K while preserving colorimetry by appropriately choosing CMY values has also been used successfully [13]. Finally, there are also examples of hue-planes being considered and preserved while performing camera characterization [14-15]. In spite of such exceptions, current solutions remain either wholly or at least predominantly geometric in their approach.

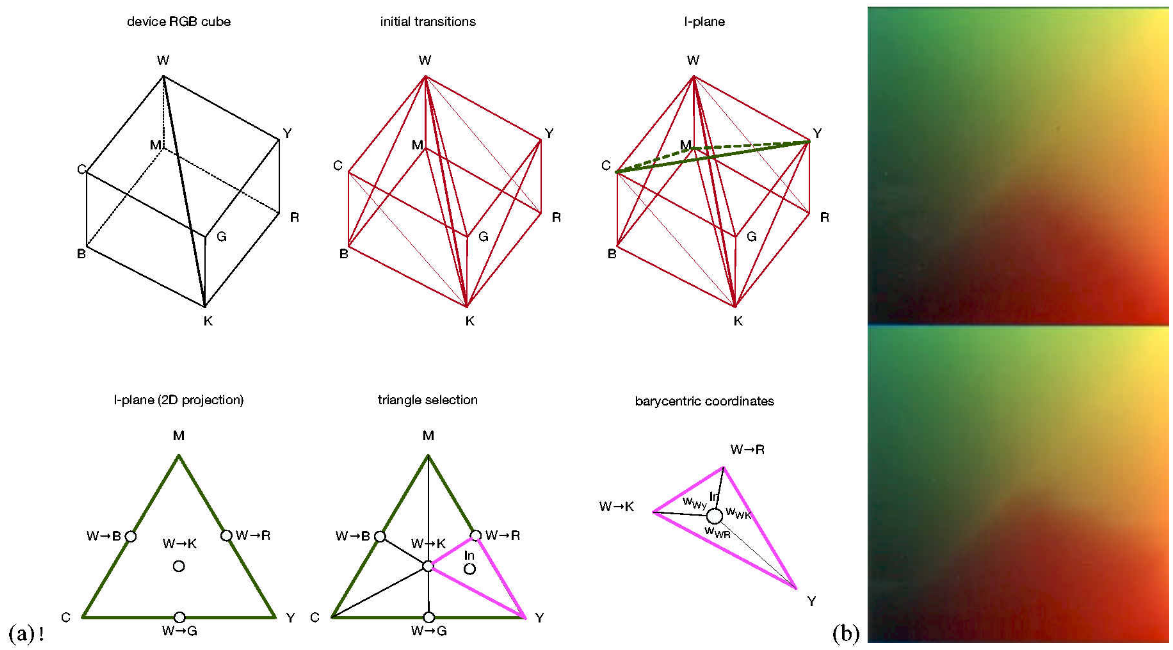
Finally, and before proceeding to the new approaches proposed in this paper, it is worth noting that the context where they are applied is that of building color LUTs that map from a device color space (RGB, CMYK) to a printing system's colorant (ink, toner, …) or Neugebauer Primary area coverage (NPac) space. This has two consequences: first, that there is no concept of color accuracy here (neither of the spaces being colorimetric) and neither is computational cost a consideration (since the approaches described here are applied only off-line; at print time, existing interpolation techniques, often with dedicated hardware acceleration, are applied).

0 ,1, &!(,1, &2.%3&4 ,1'%2, .!

The thinking presented here grows out of the insight that computations performed on the basis of color space coordinates ought to take into account the fact they these coordinates represent colors. Instead of "geometric color" processing, we therefore propose a native "color color" processing.

A particular challenge in this context is the interpolation that needs to be performed when sparse, partially-regular data is the basis for a full, regular LUT. E.g., when ramps are defined in a color space and the entire space needs to be "filled in" on their basis. An example starting point here could be a series of ramps from white to CMYRGB and from there to black, plus ramps between hue-ring neighbors and the black-white ramp. The question then arises of how the remaining nodes of a regular grid can be computed. Today a typical approach would be to use a Delaunay tessellation [8-9], where tetrahedra are formed without their vertices being considered from the perspective of their colors. The result can be transitions issues, e.g., when tetrahedron edges and faces cut through more salient directions of color change (e.g., from a primary or secondary to white or black), and also an impact on grain (e.g., when large lightness differences are present between the vertices among which interpolation then takes place).

To address such challenges, a new lightness-plane, hue-neighbor interpolation scheme is presented here, instead of the previous volumetrically-smallest circumsphere approach (i.e., Delaunay). Rather than considering tessellation geometry in the



! "#$%&$%'()*%+, -(./0%12/+./"#1345%".6/5, 4-(6%4.0%3)%/7(8 ,-/%9":/5/.;/%3/6< //.=%'6%4,)%+, -(./0%12/+./"#1345%(.9%3466%48)%>/-(2.(?%,54;/@@".#$%

abstract, choices are made based on the lightnesses of an input ramp's nodes and on their hues.

The basic idea here is to start with a set of ramps as set out above and to first use them to interpolate nodes that are in the same plane orthogonal to the black to white diagonal of the RGB cube (Fig. 1a). This is effectively a pseudo-lightness "l-plane", with all points interpolated in it being equally far up along lines parallel to the black-white axis of an RGB device color space and sharing the same "l" coordinate, computed from device RGB (dRGB) as follows:

$$l = R + G + B \tag{1}$$

The result then, for the set of ramps used as an example here, will be one point from the black to white ramp (i.e., the neutral axis) plus six or 12 points from the RGB cube's surface. There will be six such points for l-levels between black and each of R, G and B and then again between C, M and Y and W. In the region of l-values between those of R, G and B and C, M and Y there will be 12 points from RGB cube surface ramps – six from RGBCMY ramps and six from ramps connecting hue-neighbors among them (i.e., RY, YG, GC, CB, BM and MR).

Next, the points in a constant-l-plane are projected into a plane parallel to that defined by the three vertices of the RGB cube having maximum values in one of the three dimensions (i.e., RGB being [0,0,1], [0,1,0] and [1,0,0] respectively). Here it is advantageous to project into the plane L defined by the following point and pair of vectors:

!" # $%' & (%' & (%' &) (2)

*+" # ,- (%' .7 (0 %' .71 (3)

*2" # 34 /' & (0 %' .5 (0 %' .56 (4)

Projecting RGB coordinates into P then yields [r,g] coordinates where the neutral point is mapped onto the [r,g]=[0,0] origin, which is obtained as follows:

7 # 89 : ; < => ?@AB> / CAB> C (5)

D # 89 : ; < => ?@AE> / CAE> C (6)

Radial coordinates are computed next from [r,g] as follows:

$$h = atan2(r,g) \tag{7}$$

F # GH2 I J2 (8)

where h and c is pseudo-hue and pseudo-chroma and *atan2*() is the four–quadrant inverse tangent. Two pseudo-hue-neighboring colors – N_1 and N_2 (i.e., being either side of the color to be interpolated) are identified next and a triangle is then formed by them and the neutral axis point N_0 in the same plane. Barycentric weights are computed for N_0, N_1 and N_2 on the basis of their and the input point I's [r,g] coordinates, all of which are in the same pseudo-lightness plane and are close in hue:

KLM L+N # KHM 0 H2 JM 0 J2 H+ 0 H2 J+ 0 J2N <+ O KHP 0 H2 JP 0 J2N (9)

$$w_2 = 1 - w_0 - w_1 \tag{10}$$

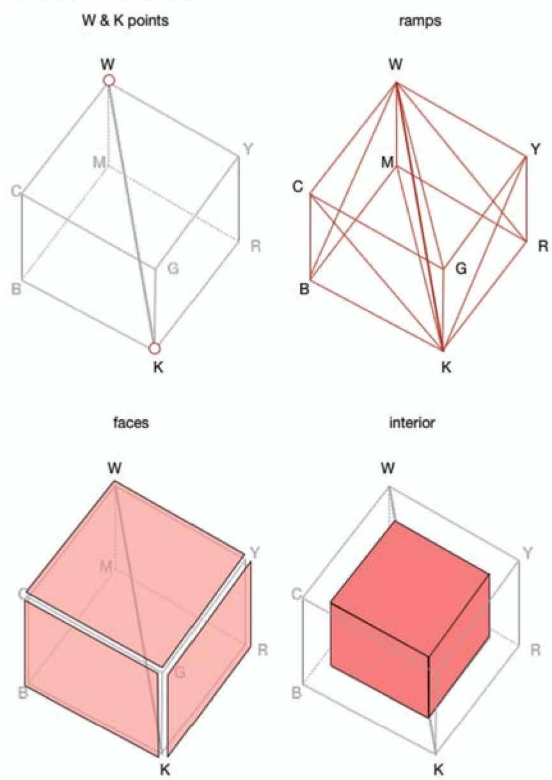
Finally, the output colorant or NPac vector O is obtained as the weighted sum (with weights w_1, w_2 and w_3) of the colorant or NPac vectors O_0, O_1 and O_2 corresponding to N_0, N_1 and N_2 respectively:

$$Q \# R^{2}_{TVM} S_T U L_T \quad (11)$$

The end result is both that interpolation is among ramp nodes of similar lightness, that it is performed in a way where the "purity" of ramps is preserved and that the cross-combination of multiple nodes is minimized. All of this leads to clean, simple transitions (Fig. 1b) that can reduce grain too.

Note also that this approach extends beyond color, to print or printing system properties in general. E.g., in 3D printing, instead of operating in l-planes, it could be subspaces that have more similar fusing requirements, or that are similar in terms of tensile strength, etc., that are the criterion for interpolation.

0 , 1, &!(, 1, &!$5 , , %2. + !



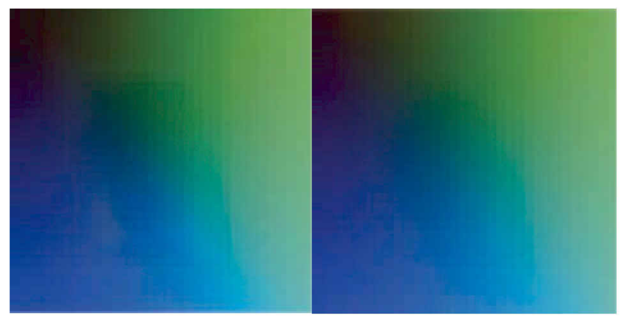
! '#$%A=%B 4 4-5%; 4-45%@8 44 6l ". #%5/ # '4 . @$%

To complement interpolation, smoothing too can be done in a color-aware way. The key here is to realize that some parts of a LUT are part of the gamut boundary, others control the printing of neutrals, etc. Here the underlying insight is that colors are best smoothed with a support that is like them in gamut location terms. Smoothing neutrals based on other neutrals, colors along gamut edges based on other gamut-edge colors, those from gamut faces based on those faces alone, and interior colors based on their full neighborhoods (Fig. 2) leads to a result that is both smooth and that preserves color gamut and neutral color choices.

This translates into smoothing along edges and the neutral axis being a weighted sum of ±x nodes along a given edge or axis. Then, within faces the process is applied to a [±x,±x] window, but excluding edges, and in the interior the operation is performed over a [±x,±x,±x] neighborhood, again excluding nodes that belong either to faces, edges or the neutral axis. In the case of interior nodes, smoothing is obtained as follows (and for other node types it is achieved analogously):

$$W \# \frac{R_{X]} \hat{\ } R_{Y]} \hat{\ } R_{Z]} \hat{\ } A_{XYZ} U [_{XYZ}}{R_{X]} \hat{\ } R_{Y]} \hat{\ } R_{Z]} \hat{\ } A_{XYZ}} \quad (12)$$

where the coordinates of the node being smoothed are [i,j,k]=[0,0,0], N are the nodes of the look-up table before smoothing, S is the smoothed node contents, and *v* are the smoothing weights, which can simply be distance in the LUT's indexing space (RGB, CMYK) from the node being smoothed, or some function of these. Stronger smoothing can be achieved both by large values of x, by applying a broader weighting function and by giving the node that is being smoothed a weight of zero. Fig. 3 here shows the effect of the above smoothing on the KGCB face of the RGB cube.



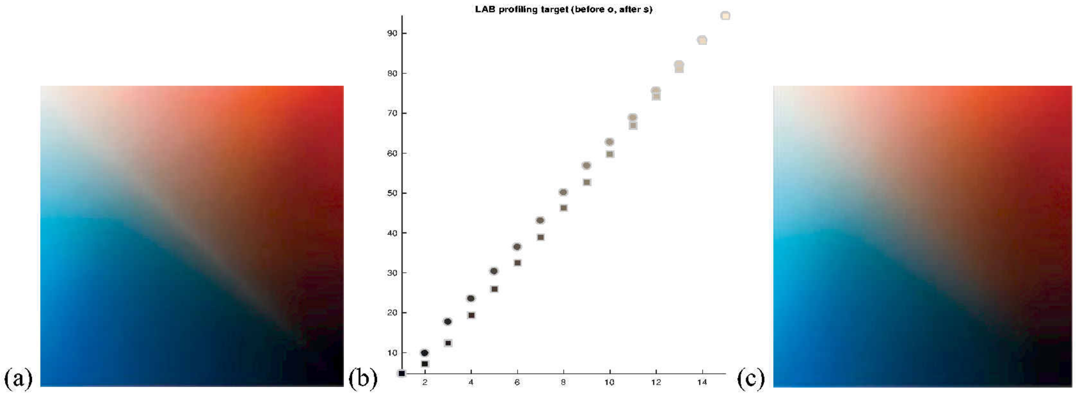
! '#$%C=%D1 / %E F B G %:(; / %4 :%6l / %H F G %; 2 3 / %4 3 6(". / 9 %:5 4 8 %(%* I D %3 2 "'9 ". #%, 5 4 ; / @@%
'-/ :6)%(. 9 %@8 44 6l / 9 %2@". #%6l / %; 4 4-5%; 4 -45%@8 44 6l ". #%@6 5(6/ #? %'5 '#1 6)$%

An additional challenge here is the preservation of neutral colors. Smoothing them with their full neighborhoods would give the smoothest result overall, but would lose their being strictly on the R=G=B axis of the RGB LUT, which is a mechanism used for then mapping them to specific ink combinations or NPacs in a color separation (e.g., ensuring that they are printed only using black or neutral inks). Conversely, just smoothing them among themselves – which ensures that they remain on the R=G=B axis – runs the risk of a lightness difference between the neutral axis and its neighborhood. Fig. 4a shows an example where smoothing has been applied to an entire LUT, but where neutrals were smoothed only among themselves and where a clear lightening of the neutrals can be seen. This is because of smoothing neutrals only along the neutral axis disconnects them from the colors obtained by smoothing the neighborhood of the neutral axis.

Here the solution is to re-interpolate the neutral axis to match the lightness of what would be obtained if its nodes were interpolated on their full neighborhood and not only based on neutrals (Fig. 4c).

75 ' + 2. + !4 2 4 3 12. 3 12. %3 &' (%2, . $!

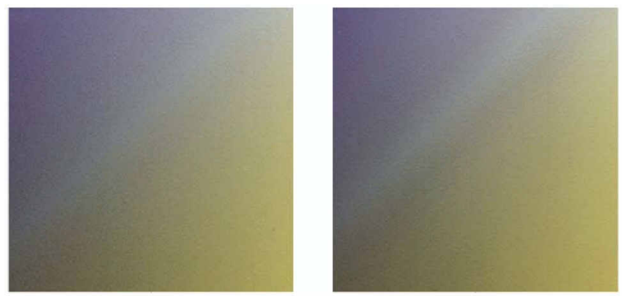
Even though the color color processing examples given above were in the context of RGB-indexed color separation LUTs, and therefore applicable to either conventional colorant-channel



! "#$%$%' ()%N /265(+@%.46%@8 446l /9%4%,5/@/5T/%l /'5%;4-45%@/, (5(6'4 .%;14";/@%'43@/5T/%l /'5%'#16/ . ". #)U%'3)%*V%8 / (@25/8 / .6@%4 :%45'#". (-%. /265(+@%' ; '5; -/@)%(.9%
. /265(+@%@8 446l /9%4 . %2 -%C> %. /'#13451449%'@W2 (5/@)U%' ;)%. /265(+@%5/+-". / (5%X/9%4%8 (6;1%'#16. /@@%4 :%2 -?+@8 446l /9%. /265(+@$%

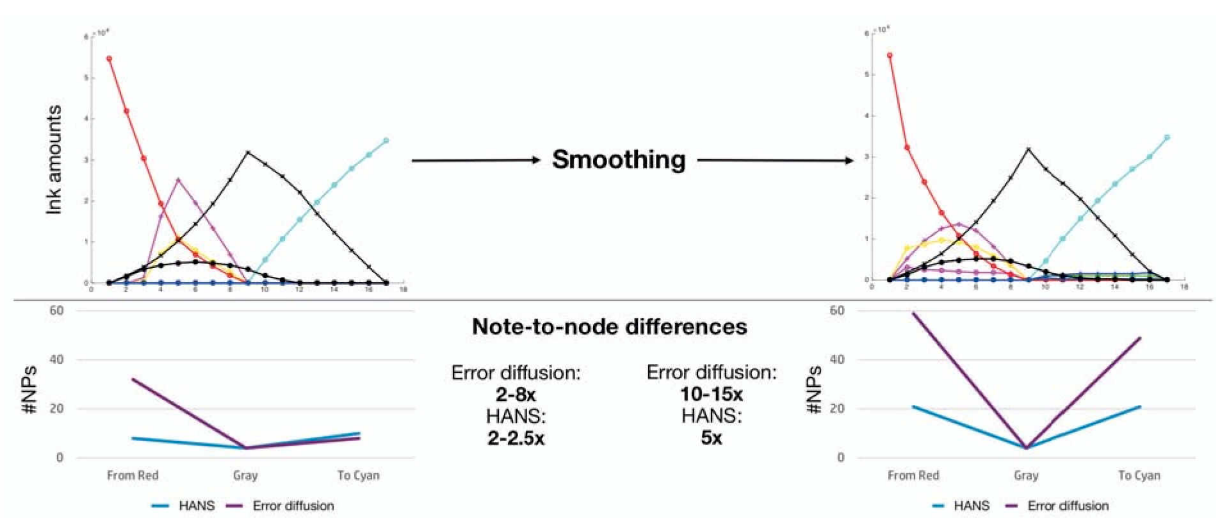
imaging pipelines, or the HANS pipeline [10] using PARAWACS halftoning [11] (branded as HP Pixel Control), their resulting level of performance has distinct differences between the two cases.

While the color color interpolation algorithm tends to work well for both imaging pipeline types, the smoothing performs distinctly better when applied to HANS. E.g., Fig. 5 shows a challenging purple-gray-yellow transition printed with a color separation that has been smoothed. Here the image on the left shows the result for a HANS pipeline, while the result on the right was obtained using an ink-channel pipeline using error diffusion halftoning. Note that both separations use the same ink amounts at the LUT nodes, yet the former specifies NP area coverages while the latter deals only with ink amounts. Looking at the smoothness of the two prints, it can be seen that the HANS transition is much smoother, while in the ink-channel case dark bands appear either side of the neutral axis.



! "#$%J=%K146l %4 :%,5". 6%8 (9/%< '6l %@8 446l /9%, ", / -". /%2@". #%LMNO%' -/ :6)%(.9%". P+
;1 (. . / -%@/ , (5(6'4 . %Q%/ 5545%9 ":2@'4 . %, ", / -". /@%'5'#16)%R%. 46/%l /%9 (5P/ . ". #%
(542 .9%l /%. /265(+@%". %l /%". P%;1 (. . / - ; (@/$%

To understand the reason behind this disparity, Fig. 6 shows what happens both in the ink-channel and HANS domains as a result of smoothing. For a fair comparison, color separations were set up



! "#$%Y$%Z. P%(.9%N K %T"/< @%4 :%l /%/ ::/ ;6%4 :%@8 446l ". #%4 . %,5". 6@%8 (9/%< '6l %". P+;1 (. . / -%T/5@2@%LMNO%, ", / -". /@$%

where the ink use at each node is the same in the ink-channel and HANS cases. The top of the figure therefore shows ink amounts used to transition from red, via gray to cyan, i.e., along one of the RGB LUT's diagonals, both before (left) and after (right) smoothing. While the picture is the same here for the two pipelines, the differences become apparent when the two are analyzed in the Neugebauer Primary (NP) domain. In the case of the HANS pipeline this is directly the control domain (i.e., NP area coverages are specified at LUT nodes and transitions between LUT nodes are computed in area coverage terms), while in the ink-channel pipeline's case NP usage is an indirect consequence of using error diffusion on ink-channel data.

For both pipelines though it is possible to count how many NPs end up being used once color separation and halftoning are applied, and this is shown in the bottom half of Fig. 6. Looking at the state before smoothing, it can be seen that the HANS pipeline uses far fewer NPs per node (les that 10 at most), while the ink-channel, error-diffused pipeline uses many more (over 30). What also differs is how the number of NPs changes from one node to the next. Here HANS has a factor of 2-2.5, while in the case of the ink-channel pipeline changes are up to 8x in the number of NPs used from one node to the next. This means that the ink-channel pipeline needs to transition between patterns of greater difference than the HANS pipeline.

When smoothing is then applied to both HANS and ink-channel LUTs (resulting in the ink use at the top right of the figure), the difference between the two pipelines becomes even greater. While the HANS pipeline now needs to cope with node-to-node NP differences of 5x, the ink-channel pipeline's nodes differ by a factor of up to 15x. Another way to look at this is to note that that HANS pipeline's nodes barely exceed 20 NPs per color, while the same ink amounts, when halftoned using error diffusion applied to ink channels, result in close to 60 NPs.

0 , . (1-$2, .$!' ./ !8-%-&3!9 , &*!

In summary, the aim of this paper has been to present two color processing methods that are fundamentally derived from it being color coordinates that are processed instead of generic spatial ones. This allows for choices to be made on the basis of lightness and hue when performing color interpolation, instead of analytical cube tessellations or the volumes of circumspheres. For smoothing, a color color processing approach allows for locations within a color gamut and the question of whether a node represents a neutral color or not to be key factors. Such color considerations result in look-up tables that are better behaved, where the purity of certain transitions is preserved and where smoothness is delivered without a gamut sacrifice.

The methods presented here are used in the HP Designjet Z6 and Z9+ printer series as part of their HP Pixel Control (HANS) imaging pipeline. In terms of future work there is potential both to apply "color color" processing to other color workflows and pipeline processes and to adapt it to controlling properties other than color.

" (*.,9 13/+35 3.%!

The authors wish to thank their colleagues at HP Inc. for their support and advice, in particular Africa Real, Albert Serra, Hector Gomez, Javier Maestro, Joan Enric Garcia, Konstantinos Kontonikolas, Tanausu Ramirez.

: 3&3. (3$!

[1]! Kasson J. M., Nin S. I., Plouffe W., Hafner J. L. (1993) A Tetrahedral Interpolation Technique for Color Space Conversion, *SPIE Proc.*, 1909:127–138.

[2]! Balasubramanian R. (1994) Color Transformations for Printer Color Correction, *IS&T/SID 2nd Color Imaging Conference*, 62–65.

[3]! Kasson J. M., Nin S. I., Plouffe W., Hafner J. L. (1995) Performing Color Space Conversions with Three–Dimensional Linear Interpolation, *Journal of Electronic Imaging*, 4/3:226–250.

[4]! Bala R. (2003) Device characterization, *Digital Color Imaging Handbook*, Sharma G. (ed.), CRC Press, 269–384.

[5]! Hu J. L., Deng J. B., Zou S. S. (2010) A Novel Algorithm for Color Space Conversion Model from CMYK to LAB, Journal of Multimedia, 5/2:159–166.

[6]! Clark D. A., Strong D. C., White T. Q. (1984) *Method for Color Conversion with Improved Interpolation*, U. S. Patent 4,477,833.

[7]! Morovic J., Albarran A., Arnabat J., Richard Y., Maria M. (2008) Accuracy-Preserving Smoothing of Color Transformation LUTs, *IS&T/SID 16th Color Imaging Conference*, 243–246.

[8]! Delaunay B. (1934) Sur la sphère vide, *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh I Estestvennykh Nauk*, 7:793–800.

[9]! Hardeberg J. Y., Schmitt F. (1997) Color Printer Characterization Using a Computational Geometry Approach, *IS&T/SID 5th Color Imaging Conference*, 96–99.

[10]! Morovič J., Morovič P., Arnabat J. (2011) "HANS – Controlling Inkjet Print Attributes Via Neugebauer Primary Area Coverages," *IEEE Transactions on Image Processing*, 21/2:688–696.

[11]! Morovic P., Morovic J., Gondek J., Ulichney R. (2017) Direct Pattern Control Halftoning of Neugebauer Primaries, *IEEE Transactions on Image Processing*, 26/9:4404-4413.

[12]! Kanamori K., Kawakami H., Kotera H. (1990) Novel color transformation algorithm and its applications, *Proc. SPIE*, 1244.

[13]! Hung P. C. (1994) Smooth colorimetric calibration technique utilizing the entire color gamut of CMYK printers, *Journal of Electronic Imaging*, 3(4).

[14]! Andersen C. F., Connah D. (2016) Weighted Constrained Hue-Plane Preserving Camera Characterization. *IEEE Trans. Image Processing* 25(9): 4329-4339.

[15]! Andersen C. F., Hardeberg J. Y. (2005) Colorimetric Characterization of Digital Cameras Preserving Hue Planes, *IS&T Color Imaging Conference*, 141-146.

!

" -%, &!) 2, +&'4623$!

Ján Morovič received his Ph.D. in color science from the University of Derby (UK) in 1998, where he then worked as a lecturer. Since 2003 he has been at Hewlett-Packard in Barcelona as a senior color scientist and later master technologist. He has also served as the director of CIE Division 8 on Image Technology and Wiley and Sons have published his 'Color Gamut Mapping' book. He is the author of over 100 papers and has filed 100+ US patents (36 granted).

Peter Morovič received his Ph.D. in computer science from the University of East Anglia (UK) in 2002 and holds a B.Sc. in theoretical computer science from Comenius University (Slovakia). He has been a senior color

and imaging scientist at HP Inc. since 2007, has published 50+ scientific articles and has 100+ US patents filed (33 granted) to date. His interests include 2D/3D image processing, color vision, computational photography, computational geometry and his Erdős number is 4.

Jordi Arnabat received his M.Sc. in computer vision in 2001, holds a B.Sc. degree in physics from the University Autonoma of Barcelona, Catalonia. He is currently a Master Technologist at HP Inc in Barcelona. He published over 20 journal and conference articles; his research interests include color reproduction, image segmentation, 3D reconstruction and computational geometry.

Víctor Diego, born in Entrambasaguas (Spain). Graduated in mathematics in 2012 by University of Cantabria. Received the Master degree in Advanced Mathematics and Mathematical Engineering by Polytechnic University of Catalunya in 2013 in Barcelona and a Ph.D. in discrete mathematics and graph theory in 2017 by the same university. Currently, and since a year ago, he is R&D developer in H.P. in the Sant Cugat site (Barcelona).

Pere Gasparin holds a bachelor's degree in Indrustrial Engineering in Electronics from the Universitat Politecnica de Catalunya, and a bachelor's degree in Computer Science from the Universitat Oberta de Catalunya. He has been working at HP in Barcelona since 2002, involved in Writing Systems and Color and Imaging teams for the development of large format inkjet printers.

Xavier Fariña was born in Barcelona in 1976. He holds a Master Degree in Mathematics and a Master in Big Data and Business Intelligence. Before joining HP in 2011 he built his professional career in several companies related to the software development, performing roles like Analyst, DBA, Web and mobile developer or UX designer. He loves playing board games, reading and spending time with his wife, his two little children and his two cats.

Sergio Etchebehere received his MSc in color science from University of Eastern Finland and Universidad de Granada in 2016, where then he later worked as a researcher. Currently he is a Color and Imaging Scientist at HP Inc. His research interests are color science, image processing, human vision and computer vision.