Interactive Color Gamut Mapping

Ivar Farup and Jon Y. Hardeberg
ivar.farup@hig.no, jon.hardeberg@hig.no
Gjøvik University College
Gjøvik, Norway

Abstract

Gamut mapping is an important issue in cross-media publishing. Although much research and development has been performed, consensus on a single gamut mapping algorithm working for a broad range of images and devices has not yet been reached. The recent tendency in the literature suggests that image-dependent gamut mapping work best. To avoid the computational overhead associated with image-dependent gamut mapping algorithms, one solution is to use different rendering intents (absolute or relative colorimetric, perceptual, or saturation) for traditional device gamut based algorithms. The optimal solution, however, still turns out to be image dependent, leaving craftsmanship as the only real alternative. Unfortunately, no intuitive tools – neither software nor hardware – exists for this work, so one is left with trial-and-error based methods with no direct intuitive coupling between parameters adjusted and color corrections obtained. Hence, a software tool for interactive color gamut mapping in a device independent color space such as CIELAB is needed. Such a tool has been developed by the authors. The application allows for interactive manipulation of colors in the 3D color spaces CIELAB, CIEXYZ, and sRGB. Image and device gamuts can be visualized in various ways in the same figure. The view can be changed interactively, and points representing individual pixel colors, groups of pixels, or the image gamut boundary can be moved in color space using a pointing device. Already at the present stage, the application has become a useful tool for understanding mechanisms associated with color image reproduction, as well as for actually performing interactive image-dependent color gamut mapping.

1. Introduction

Color gamut mapping is an integral and important part of color management, and several gamut mapping algorithms have been proposed.[1] Most of the algorithms are formulated as geometric deformations of a color gamut – either the image's source or the image gamut directly – in some device independent color space. The transformations involved are often complex, and it is not straightforward to imagine how they influence the gamut and the resulting color image. In order to achieve a better understanding of the mechanisms involved, different methods for visualizing color gamuts have been developed in the past.

A short survey on existing techniques and tools for gamut visualization and interactive manipulation is presented in Section 2. Section 3 gives a brief introduction to common gamut mapping strategies. In Section 4, ICC3D – the newly developed tool for visualization of image and device gamuts and interactive manipulation of image gamuts, i.e., interactive color gamut mapping – is presented. The emphasis will be on application rather than implementation of the new tool. The tool combines many of the techniques which are found in the literature. Some early experimental results are presented in Section 5, and finally, some concluding remarks are drawn and suggestions for future work outlined.

2. Color Gamut Visualization

2.1. Visualization Strategies

Since color in tristimulus colorimetry can be represented by three values, color gamuts can be represented by objects in a 3D color space. Hence, 2D visualizations of gamuts must reduce the dimension either by 2D rendering of a 3D scene, or by simplifying the problem otherwise, e.g., by only showing certain cross sections.

The traditional way of visualizing a device gamut is to plot its outline in a 2D xy chromaticity diagram.[2] Alternative variants include plotting the gamut boundary in CIELAB as L^* vs. C^*_{ab} for given values of h_{ab},[3] or viewing projections of the gamut solid onto different principal planes, e.g., (a^*, b^*), in CIELAB.[4–6] Herzog[7] plotted C^*_{ab} as a function of L^* at constant hue along with a histogram (as circles) of an image, thus indicating out-of-gamut colors. Other instructive techniques include plotting gamut volume versus h_{ab}, L^*, and C^*_{ab},[5] or showing C^*_{ab} in grayscale as a function of h_{ab} and L^*.[3]

Already in 1935, MacAdam introduced the third dimension of the xyY chromaticity diagram by the use of contour lines.[8] Similar techniques have also been used later.[5] MacAdam also presented a stereoscopic photograph of a surface bounding all attainable colors in illuminant C in the xyY color space.[8] In 1988, Stone et al. presented orthographic views of idealized device color gamuts as "true" colored solids.[9] Following the increase in computing power, quasi-realistic perspective renderings of color gamuts represented as colored solids,[2,10,11] wireframe outlines,[6,12] or combinations of both,[5,13] have become common. Particularly appealing is the use of partially transparent solids.[14,15] When the gamut to be visualized is known from discrete measured colors or is an image gamut, it can also be visualized as a point cloud.[6,16] Another instructive technique is to depict the gamut as a leaf structure consisting of "true" colored planes of, e.g., constant hue or constant lightness, in a 3D scene.[4] Outline of the gamut boundary in such planes have also been used for visualization more recently.[17] A slightly different approach is to work directly in cylindrical coordinates, and plot the chroma as a function of lightness and hue, $C^*_{ab}(L^*, h_{ab})$, as a surface.[3,5–7,18]

Comparison of different color gamuts can be done by representing the gamuts as objects of different types (solid, wire-frame, etc.) in the same coordinate system,[6,11,14,15] or more directly by generating a solid from the difference between device gamuts.[10] These differences can become even clearer when only parts of the gamuts are shown, either as solids[15] or as color coded point clouds.[16]

Other quantities related to color can also gain from visualization. Displacement vectors associated with a gamut mapping algorithm can be shown as arrows or line segments,[2] errors associated with measurement uncertainty as spheres,[2] and color histograms for images as differently sized spheres.[19]

2.2. Gamut Boundary Determination

The determination of the color gamut boundary plays a crucial role in color gamut visualization. This can be performed in at least two principally different ways. First, it can be found from the measured data points (colors in some color space). Secondly, it can be derived from an analytic model. Some of the methods among the former set are directly applicable to other color sets such as images, whereas methods of the latter category necessarily depends strongly on the device.

2.2.1. Point Based Methods

The simplest way of obtaining the device gamut from measured colors, is to assume that the gamut boundary is preserved between device dependent and device independent color spaces. Stone et al.[9] measured the most extreme colors (gamut corners) of a printing system and connected them with planes. Bolte[10] obtained gamuts by direct triangulation of the measured printed colors laid out in a regular structure in the device color space. An improvement of this technique is to check for well-behavedness, i.e., that the gamut surface in the device color space corresponds to the surface in the device independent color space, and that no tetrahedra are mirrored.[20,21]

If there is no imposed structure in the measured data, a convex hull algorithm can be applied directly, resulting in a purely convex gamut approximation.[14,15] This strategy has recently been applied to gamuts derived from ICC profiles.[6] Guyler[12] recently argued that the data tend to be more convex in the CIEXYZ color space, and that the convex hull algorithm thus should be performed there and later transformed to the desired color space such as CIELAB. A similar solution which works for reasonably well-behaved printers is to compute the convex hull in a linearized dye density space.[22] Balasubramanian and Dalal[23] introduced a method for making the measured data more convex by means of a non-linear function before computing its convex hull. This method seems better suited to printer-like gamuts than to images since it requires the precise selection of an interior point.

From measured colors, the maximum chroma for cells in the (L^*, h_{ab}) plane can be found. These points can have an imposed regular structure, and can thus easily be plotted as a triangulated surface.[3,5] The segment maxima method is a similar method using polar instead of cylindrical coordinates.[24] It consists in subdividing the CIELAB color space into segments of uniform spherical angles (polar and azimuth). The center of the spherical coordinate system is taken either as the point having (L^*, a^*, b^*) coordinates $(50, 0, 0)$ or as the center of mass of the given points. For each segment, the point with the larger radius is stored, and together these points define the gamut boundary. Triangulation of the surface is straightforward due to the imposed regularity. This technique has also been used for image gamut volume determination, using the mass center as the center of the coordinate system.[25]

Recently, Cholewo and Love[13] introduced the use of alpha shapes[26] for gamut boundary determination of both images and devices. An alpha shape is a generalization of a convex hull applicable also to non-convex solids. It is obtained by performing a 3D Delaunay triangulation of the point set and subsequently removing the simplices, i.e., tetrahedra, triangles, and edges, for which there exists a sphere of radius α which does not contain any other points from the set. Cholewo and Love advised that the value of α is chosen at least large enough to make the resulting body consist in a single component. However, they also concluded that the optimum α value is best determined interactively by inspection.[13]

2.2.2. Model Based Methods

An early attempt at deriving the gamut from a physical model was to assume box-shaped reflectance of colorants and calculate CIEXYZ tristimulus values and thereby the gamut.[8] Meyer et al.[14,15] used the Kubelka–Munk[27] equations for determining the gamut of printing systems. The Neugebauer[28] equations were used by Mahy[29] for calculating the gamut of n-ink printing systems.

Inui[17] presented a novel four-step algorithm for computing printer color gamuts based on the assumed one-to-one correspondence between dye amount space and color space.

Herzog developed the concept of gamulyts – an analytical mathematical description of color gamuts – by modeling the gamut as a deformed cube.[7,18] The deformation was modeled by a set of *distortion functions*. He also discussed how this approach can be extended to systems with four or more colorants.

2.3. Interactivity

GamOpt by Kalra[16] is the only software tool known to the authors capable of interactive manipulation of gamut data. The interactivity consists in deformation of the unit circle in (a^*, b^*) with relative scaling of other colors, linear transformation of the L^* axis, linear transformation of the entire gamut, user specified functions for the whole gamut, and automatic homogenizing of gamut points.

3. Color Gamut Mapping

Recently, Morovic and Luo[1] published an extensive survey on different gamut mapping algorithms, so only a brief summary will be given here. According to MacDonald[30], the main aims of the majority of gamut mapping studies are to preserve the gray axis and aim for maximum luminance contrast, to reduce the number of out-of-gamut colors, to minimize hue shifts, and to increase rather than decrease the color saturation.

There are two principal kinds of gamut mapping strategies; gamut clipping and gamut compression. Gamut clipping algorithms only change colors which are outside the reproduction gamut, whereas gamut compression also affects colors inside the destination gamut. The compression can be performed in various ways such as linearly, piecewise linearly, or according to a more complex, preferably smooth function. Combination of compression and (soft) clipping is often used. So far, most gamut mapping algorithms have converted colors between different device gamuts, but several studies indicate that image-dependent gamut mapping works better.[31]

4. ICC3D: A Tool for Visualization and Interactive Manipulation of Color Gamuts

The aim of the authors is to make a visualization software by combining the most instructive techniques found in the literature with novel techniques for interactive altering of color gamuts and individual colors. The new software will thus allow for experimenting with color gamut mapping both for research and educational purposes as well as serving as a real usable tool for optimized color image reproduction.

In order to make the tool as platform independent as possible, the Java 2 SDK[32] was chosen for development. For convenience, the Java3D,[33] Java Advanced Imaging,[34] and Java Media Framework[35] APIs were also used. Unfortunately, these libraries are currently not available on the Macintosh platform, but this will hopefully change in the near future. So far, the current pre-release version of the application has been tested successfully on both Windows and Linux platforms. A screen-shot of the application is shown in Fig. 1. Information on how to obtain a beta version of the software for testing can be found on the web.[36]

Being based upon plug-ins which are loaded dynamically at start-up, the software architecture is extremely

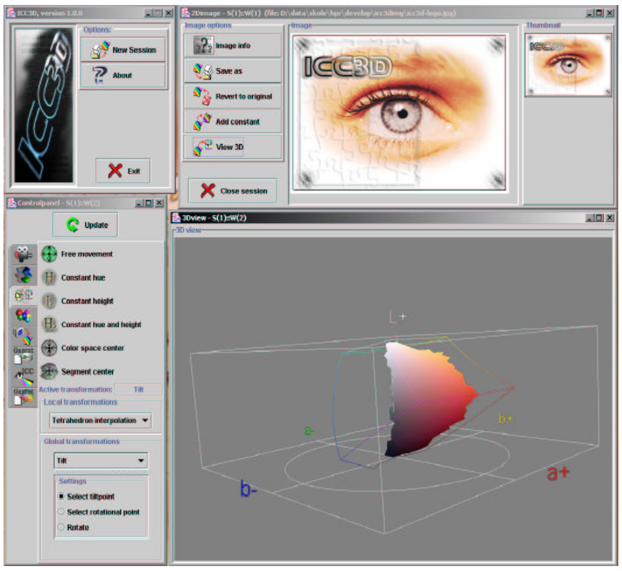


Figure 1: Screen-shot showing ICC3D at work in a Windows environment.

modular. The basic functionality of the software is the ability to read color data either as an image in some standard format, as measured colors in the IT8.7/2 file format, as extracted by the gamut tag of an ICC profile, or as color values in a regular file format identical to the one used by Color3D[19] for compatibility. The color data can then be displayed in different ways in various 3D color spaces. The user may interact with the visualization in several ways, possibly under some constraints, in order to change the color data as desired. The plug-ins thus fall into four different categories: color spaces, visualizations, interactions, and constraints.

In practice, a plug-in is implemented as a Java class which extends a given abstract base class. In addition to performing the required work, each plug-in is responsible for creating its own GUI.

4.1. Image Gamut Visualization

Images can be loaded in most standard image file formats and viewed on the monitor. Both the color data in the input files and the monitor are assumed to be sRGB. So far, the sRGB, CIEXYZ, and CIELAB color spaces have been implemented using the well-known analytical transformations.

The image colors can be visualized using either parallel or perspective rendering in one of the sRGB, CIEXYZ or CIELAB color spaces. In the chosen 3D color space, the user can rotate and zoom using the mouse. In addition, several predefined views such as straight from above or along the a^* axis with L^* directed upwards, can be chosen by means of push buttons.

Several modes of visualization are implemented. In the simplest mode, every pixel in the image is plotted as a point in the given color space, resulting in a cloud of colored points. With many different colors present in the im-

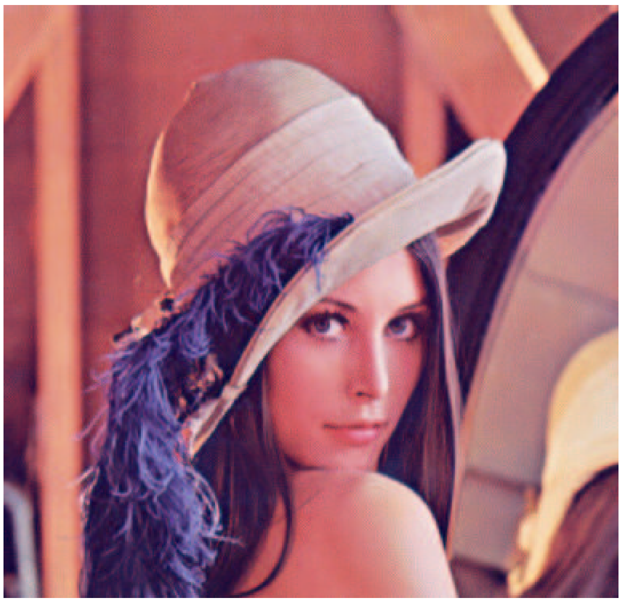


Figure 2: *The famous "Lena" image.*

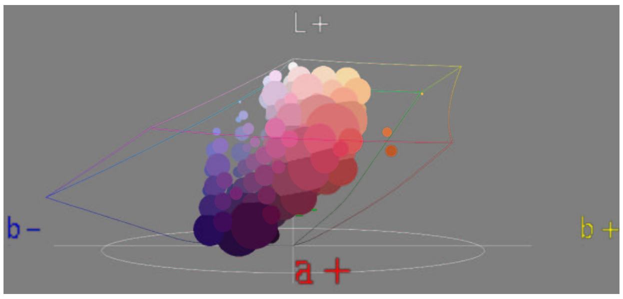


Figure 3: *3D histogram in CIELAB for the image shown in Fig. 2 (left). The deformed cube represents the sRGB gamut boundary.*

age, this results in a quite time-consuming rendering process. To cope with this problem, it is possible to quantize the color space into discrete regions and show a pixel for the color space region only if image pixels exist in that region. An extension of this approach is to count the number of pixels in each color space region and plot this as a histogram of differently sized colored spheres. Fig. 3 shows the histogram for the image shown in Fig. 2.

More interesting from the user's point of view, is to visualize the image gamut as a solid in the color space. In ICC3D, the gamut surface can be calculated by means of three different algorithms: the segment maxima algorithm, the convex hull algorithm, or by means of alpha shapes. For all of these methods, the gamut surface can be visualized as a colored wire-frame, colored solid, partially transparent colored solid. Fig. 4 shows gamut surfaces obtained by the segment maxima and the convex hull methods, whereas an alpha shape example is shown in Fig. 1.

4.2. Device Gamut Visualization

Device gamuts can be obtained in three ways. First, they can be extracted from the gamut tag of ICC profiles. Sec-

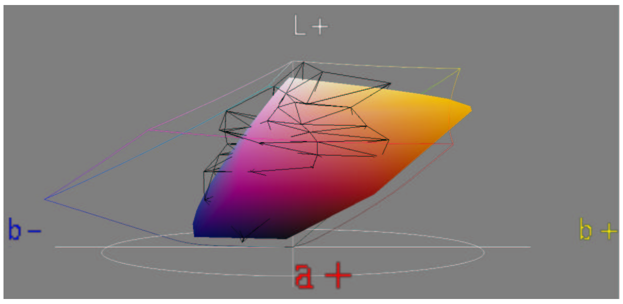


Figure 4: *Segment maxima method applied to the image (wire-frame) in Fig. 2 (left) along with a printer gamut visualized as a convex hull (solid).*

ondly, they can be obtained from standard IT8.7/2 measurement files containing CIELAB data, and, finally, they can be extracted from a regular file format identical to the one used by Color3D[37] for compatibility.

The device gamuts can be visualized in exactly the same ways as image gamuts, and several device gamuts can be shown together with an image gamut. An example of this is given in Fig. 4.

4.3. Interactions

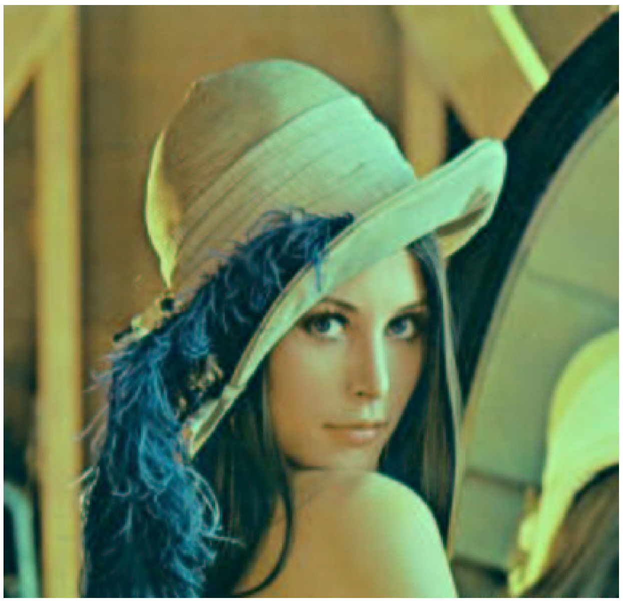
The probably most important strength of ICC3D is the possibility for interactive change of image colors in the 3D color space. ICC3D operates with two principally different kinds of interactions: global interactions which operate on the entire image or device gamut, and local interactions acting on a single color or a single group of colors. The effect of various interactive color changes to the image shown in Fig. 2 is shown in Fig. 5 and described below.

4.3.1. Global interactions

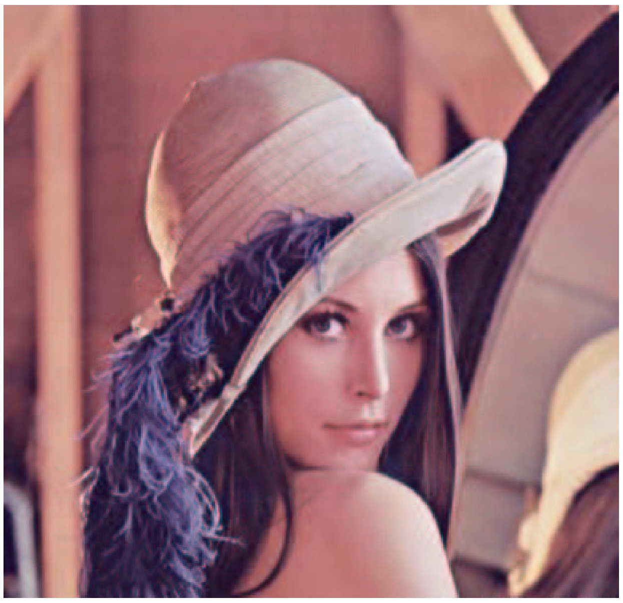
Using the global interactions, the user can perform actions on the entire gamut object, thereby influencing all colors, similar to the effects obtainable by familiar image manipulation programs. The global interactions include moving the gamut in the (a^*, b^*) plane (corresponding to hue shifts, cf. Fig. 5(a)), scaling the L^* axis (contrast), moving along the L^* axis (brightness), radial expansion/compression (corresponding to saturation and desaturation, correspondingly; cf. Fig, 5(b)), and tilting of the entire gamut (white-point correction). The global interactions can be performed with any gamut visualization mode.

4.3.2. Local interactions

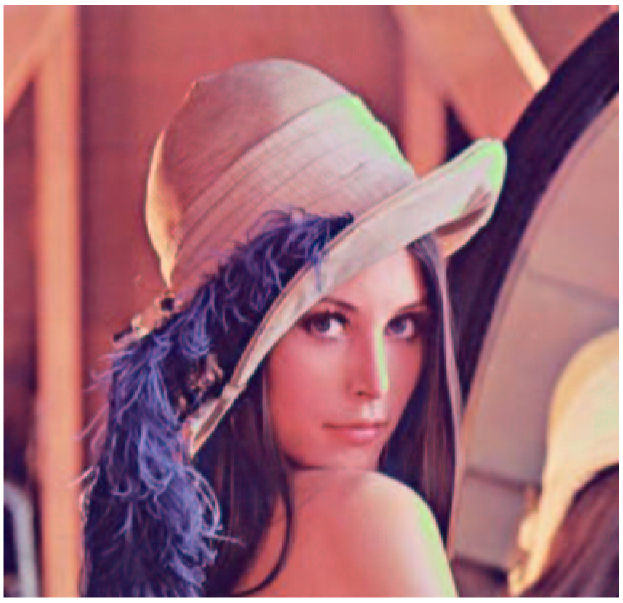
When the image is visualized as a point cloud in color space, individual pixels can be moved using the mouse. The image preview is updated after each move. In the quantized view, it is possible to drag objects representing collections of pixels. All colors represented by the object can be moved an equal distance in color space. However, since this inevitably leads to banding effects, a better solution can be chosen by the user: A tetrahedra structure is



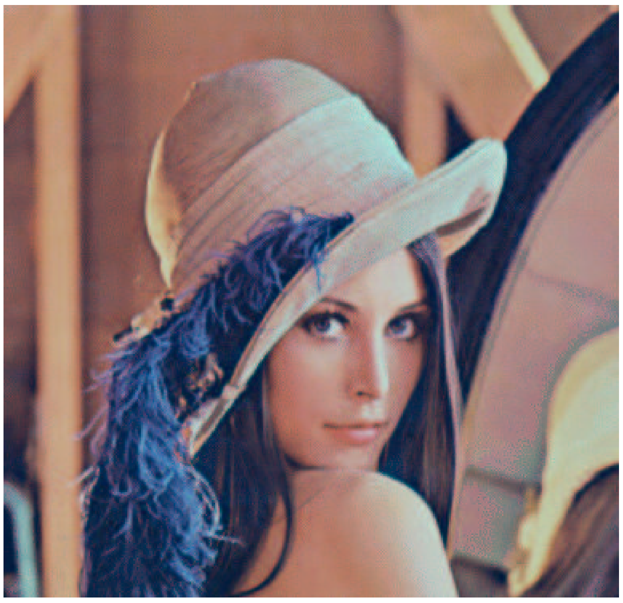
(a) Effect of moving the entire gamut in the (a^*, b^*) plane.



(b) Effect of decreasing the radial extension of the entire gamut (corresponding to a desaturation).



(c) The effect of moving a single group of colors.



(d) The result of transforming the test image (Fig. 2) using the gamut mapping shown in Fig. 6.

Figure 5: *Various operations performed on the famous "Lena" image.*

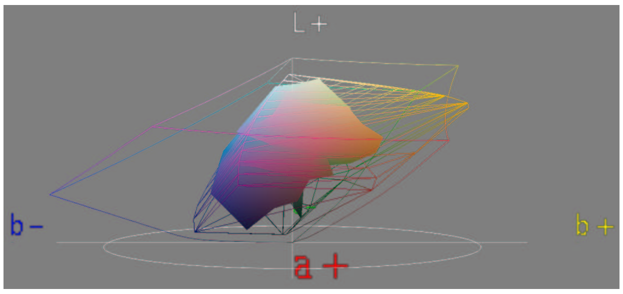


Figure 6: The image gamut from Fig. 4 (now as solid) interactively manipulated to fit within the device gamut from the same figure (now as wire-frame).

built for the entire color space using the histogram points as nodes. This is straightforward since the histogram cells have been chosen as a regular structure. Tetrahedral interpolation is then used for all pixels not lying exactly at the nodal positions. For an example of the result of moving a single group of colors, cf. Fig. 5(c).

Although the histogram strategy reduces the number of points somewhat, there are still far to many points to move in order to obtain a reasonable gamut mapping. To cope with this, interactivity directly with the gamut surfaces has been introduced. An interactively altered image gamut surface is shown in Fig. 6. Upon movement of a single surface point, all of the interior points are updated according one out of two schemes. The simplest solution is to scale the translation vector of the boundary point linearly with the radius for all pixels in the original segment, inevitably leading to in banding. The improved solution consists in using tetrahedral interpolation, the tetrahedra being constructed from the surface triangles and the central point. For pixels lying outside the segment maxima surface (they may exist), extrapolation using the same equations as for the tetrahedral interpolation is applied after selecting the nearest tetrahedron. The resulting image is shown in Fig. 5(d).

For the convex hull and alpha shapes visualizations, no local interaction is available yet.

4.4. Constraints

Moving points in a 3D color space through a 2D user interface can be quite challenging. To help the user, it is possible to constrain the movement to certain planes or certain lines. These can be chosen as lines toward the nearest point on the gray axis, lines toward color space or gamut center, planes of constant lightness, and planes of constant hue.

5. Experimental Results

In order to test the principle of interactive gamut mapping, we constructed the test image shown in Fig. 7 in sRGB. The image is designed as a typical presentation slide. The background image contains a texture with extremely saturated and bright colors, the image contains much softer color tones, and the text contains an intermediate color. The aim of the study is to compare interactive and simple automatic gamut mapping of this image onto the Euroscale Uncoated CMYK gamut defined by the standard ICC profile. Containing important textures both outside and inside this quite small gamut, the image is designed such that it is difficult to gamut map nicely.

Gamut clipping was obtained by first converting the sRGB image to CMYK by means of the absolute colorimetric rendering intent in the ICC profile, and secondly converting it back to sRGB, using the absolute colorimetric rendering intent. The resulting image shown in Fig. 8 shows that the image is reproduced nicely, since most of it is inside the Euroscale gamut. The background, however, has lost its texture, since is is completely out of gamut. The text has more or less kept its original color.

By converting the original sRGB test image to CMYK by means of the perceptual rendering intent in the ICC profile, and secondly converting it back to sRGB, using the absolute colorimetric rendering intent, a sort of gamut compression was obtained. The resulting image shown in Fig. 9 shows that all colors have been desaturated, and that the contrast of the image has been reduced. In this way, the out-of-gamut textures in the background has survived, whereas the colors of the photograph and the text has changed far too much. Such color changes would be crucial if the text were, e.g., a corporate logo.

Finally, interactive color gamut mapping with ICC3D was applied to the test image. First, the contrast was reduced slightly. Secondly, the blue part of the color space was compressed using the tetrahedral interpolation on the segment maxima surface in order to keep as much as possible of the background textures. Only slight changes had do be performed in other parts of the color space in order to make the entire image fit into the Euroscale Uncoated gamut. The resulting image shown in Fig. 10 reveals that this image keeps the best of both the compression and clipping regime. The image is not too desaturated, the text color is fairly correct, and the background texture is partly preserved.

6. Conclusion and Future Work

A novel tool for visualization and interactive manipulation of color gamuts has been presented. Although far from complete, it is already at the present stage useful for experimenting with different gamut mapping strategies and for gaining an increased understanding of the mechanisms involved. Since the tool is based upon a plug-in structure, it is extremely well-suited for future extensions. In this paper, the usefulness of the tool has hopefully been clearly demonstrated.

For the future, there are several possibilities for adding features which will make tool even more interesting:

- ICC profiles for both input files and monitor in order to increase realism and usefulness. Also, a color appearance model should be included.

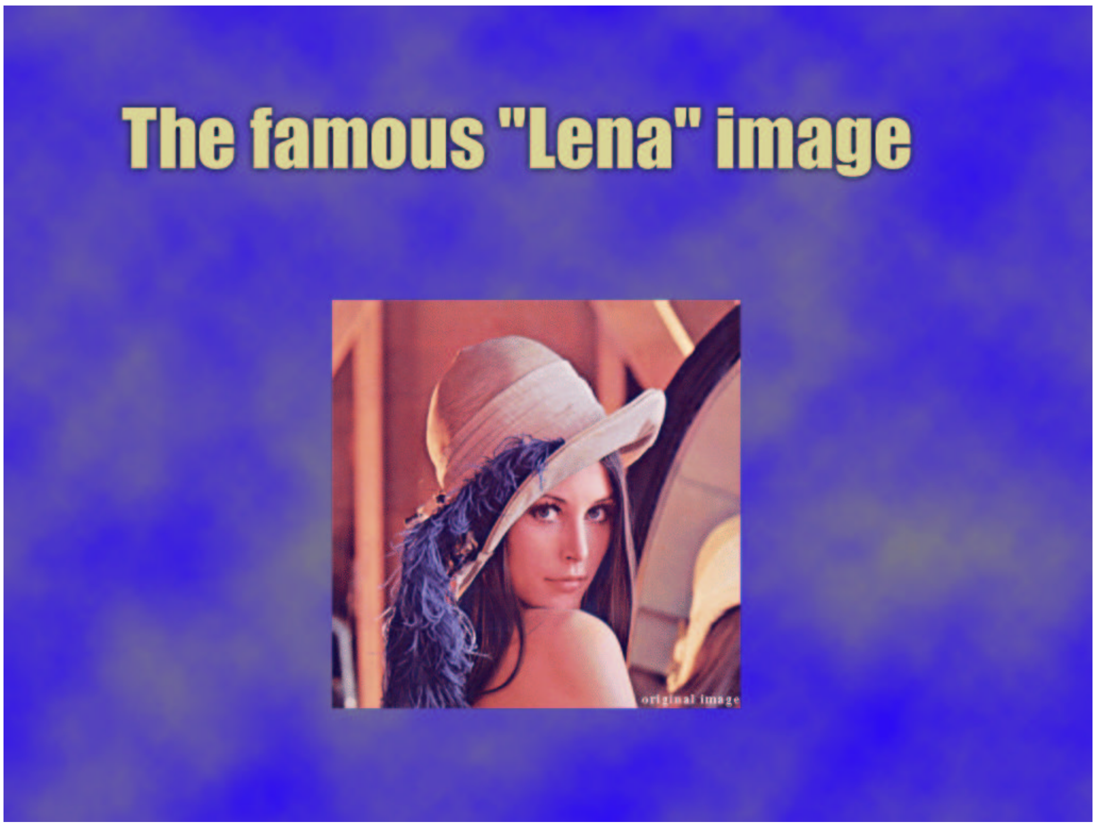


Figure 7: *The sRGB test image used for comparison between interactive and automatic gamut mapping.*

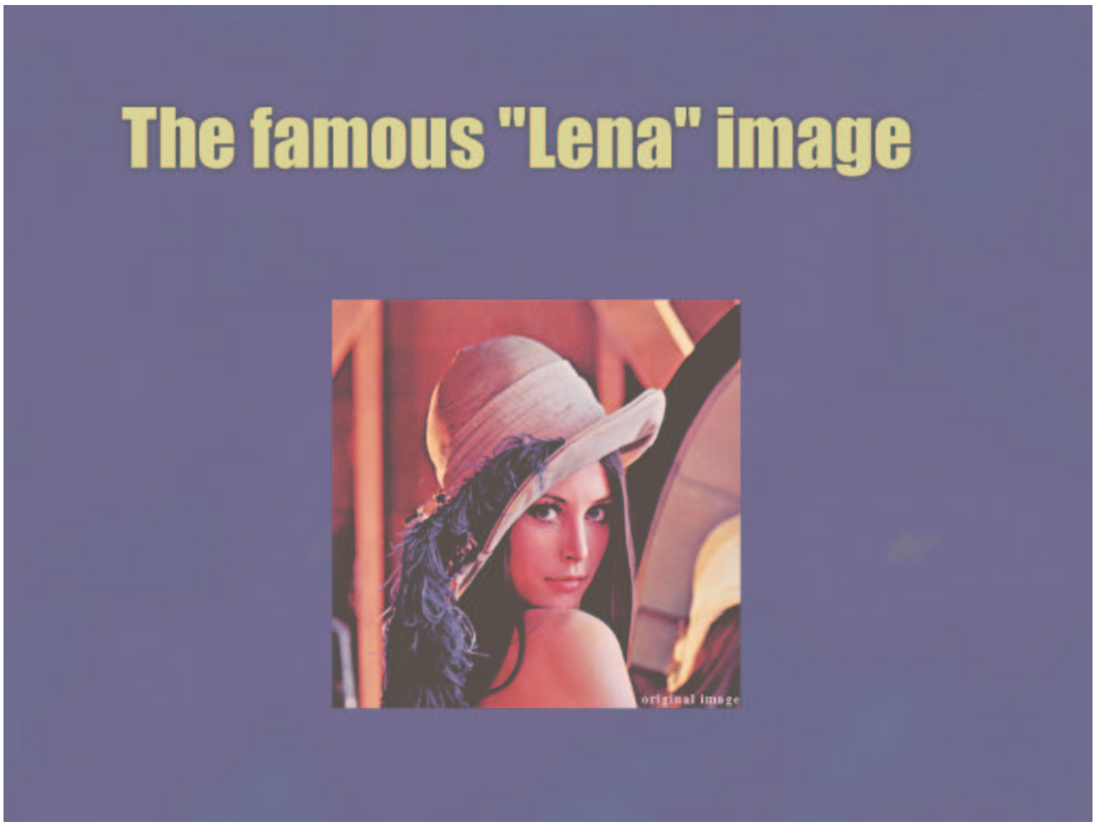


Figure 8: *The test image from Fig. 7 gamut mapped into the Euroscale Uncoated gamut by means of gamut clipping.*

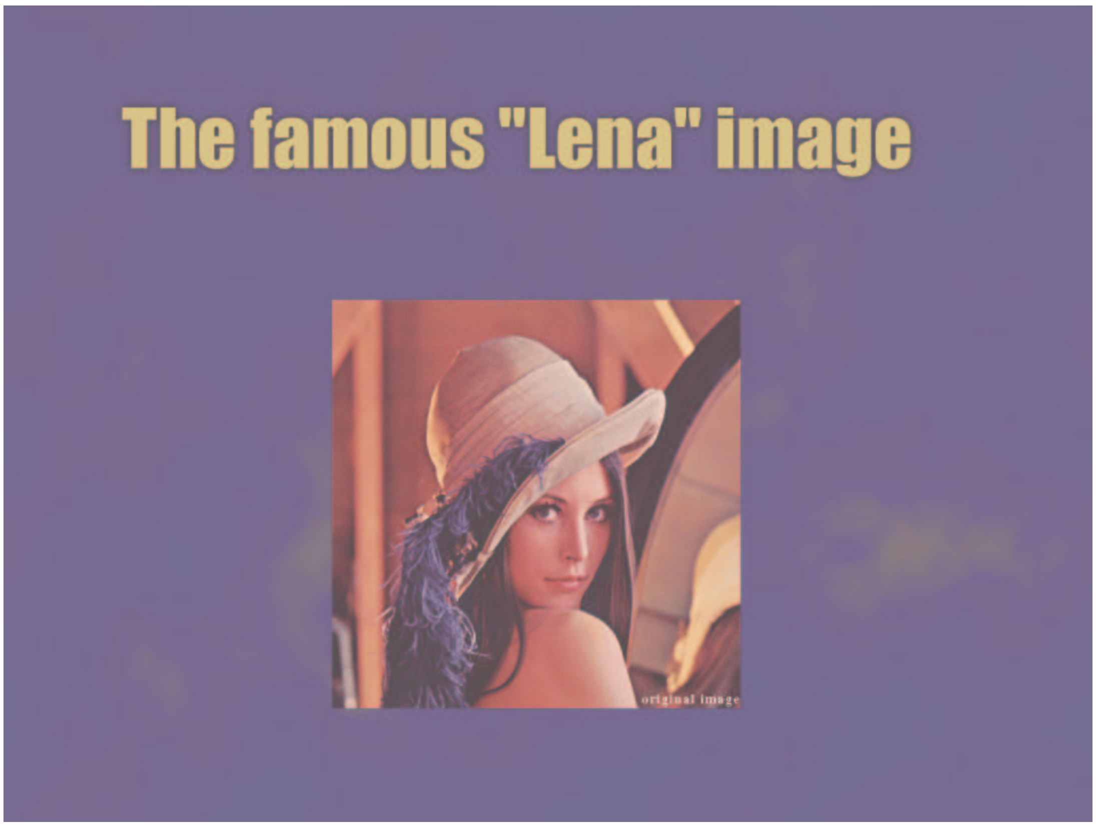


Figure 9: *The test image from Fig. 7 gamut mapped into the Euroscale Uncoated gamut by means of gamut compression.*

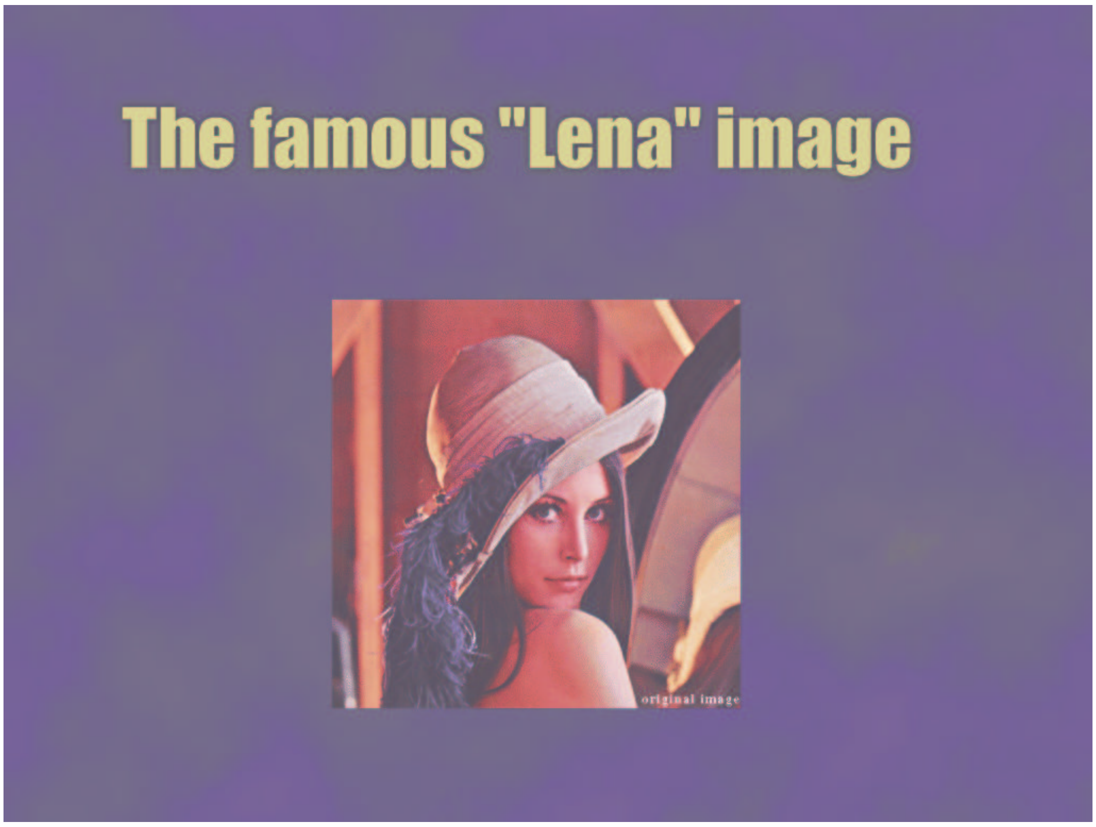


Figure 10: *The test image from Fig. 7 gamut mapped into the Euroscale Uncoated gamut by means of interactive gamut mapping.*

- Other forms of constraints when performing interactive color adjustments. For example, it should be impossible to mirror a tetrahedron while moving. This can be solved either by constraining the movement, or by making the movement influence also the movement of neighboring tetrahedra.

- Introduce other algorithms for gamut boundary determination, e.g., convex hulls computed by "convexification" of the image data, cf. the survey above.

- Possibility of selecting regions in the color space and highlighting the corresponding pixels in the image and vice versa. Selections could also be made automatically based on specific properties, e.g., pixels outside given device gamut. It should then also be possible to perform operations (translations etc.) on such groups of points.

- More advanced interpolation functions, e.g., S-curves and clipping.

- Operations on a device gamut as well as on an image gamut, and save the result as a general gamut mapping algorithm in the form of, e.g., an abstract ICC profile.[38]

- Experiment somewhat with the user interface. In the present version, confusingly many parameters are to be controlled by the use of a single mouse. Alternatives include the use of track balls, joy sticks or simply the keyboard.

Acknowledgments

The authors would like to thank Mr. Arne M. Bakke, Mr. Ståle Kopperud and Mr. Anders Rindal for implementing ICC3D, and Mr. Rune Lossius and the library at Gjøvik University College for assisting the literature survey.

References

1. Ján Morovič and M. Ronnier Luo. The fundamentals of gamut mapping: A survey. *Journal of Imaging Science and Technology*, **45**(3), 283–290 (2001).
2. Robert Rolleston. Visualization of colorimetric calibration. In *Color Hard Copy and Graphic Arts II*, volume 1912 of *Proc. SPIE*, pp. 299–309 (1993).
3. Gustav J. Braun and Mark D. Fairchild. Techniques for gamut surface definition and visualization. In *Proceedings of IS&T and SID's 5th Color Imaging Conference: Color Science, Systems and Applications*, pp. 147–152, Scottsdale, Arizona (1997).
4. Philip K. Robertson. Visualizing color gamuts: A user interface for the effective use of perceptual color spaces in data displays. *IEEE Computer Graphics & Applications*, **8**(5), 50–64 (1988).
5. Richard L. Reel and Michael A. Penrod. Gamut visualization tools and metrics. In *Proceedings of IS&T and SID's 7th Color Imaging Conference: Color Science, Systems and Applications*, pp. 247–251, Scottsdale, Arizona (1999).
6. Xianfeng Zhao. Implementing an ICC printer profile visualization software. Master's thesis, School of Printing Manamement and Sciences in the College of Imaging Arts and Sciences of the Rochester Institute of Technology (2001).
7. Patrick G. Herzog. Specifying and visualizing colour gamut boundaries. In L. W. MacDonald and M. R. Luo, editors, *Colour Imaging: Vision and Technology*. John Wiley & Sons Ltd. (1998).
8. D. L. MacAdam. Maximum visual efficiency of colored materials. *J. Opt. Soc. Amer.*, **25**, 361–367 (1935).
9. M. C. Stone, W. B. Cowan, and J. C. Beatty. Color gamut mapping and the printing of digital color images. *ACM Transactions on Graphics*, **7**(4), 249–292 (1988).
10. S. B. Bolte. A perspective on non-impact printing in color. In *Color Hardcopy and Graphic Arts*, volume 1670 of *Proc. SPIE*, pp. 1–11 (1992).
11. Peter A. Crean and Robert R. Buckley. Device independent color – who wants it? In *Proc. SPIE*, volume 2171, pp. 267–273 (1994).
12. Karl Guyler. Visualization of expanded printing gamuts using 3-dimensional convex hulls. *American Ink Maker*, **79**(9), 36–56 (2001).
13. Tomasz J. Cholewo and Shaun Love. Gamut boundary determination using alpha-shapes. In *Proceedings of IS&T and SID's 7th Color Imaging Conference: Color Science, Systems and Applications*, pp. 200–204 (1999).
14. Gary W. Meyer, Linda S. Peting, and Ferenc Rakoczi. A color gamut visualization tool. In *Proceedings of IS&T and SID's Color Imaging Conference: Transforms and Transportability of Color*, pp. 197–201, Scottsdale, Arizona (1993).
15. Gary W. Meyer and Chad A. Robertson. A data flow approach to color gamut visualization. In *Proceedings of IS&T and SID's 5th Color Imaging Conference: Color Science, Systems and Applications*, pp. 209–214, Scottsdale, Arizona (1997).
16. Devendra Kalra. GamOpt: A tool for visualization and optimization of gamuts. In *Proc. SPIE*, volume 2171, pp. 297–304 (1994).
17. Masao Inui. Fast algorithm for computing color gamuts. *Color Research and Applications*, **18**(5), 341–348 (1993).
18. Patrick G. Herzog. Analytical color gamut representations. *Journal of Imaging Science and Technology*, **40**, 516–521 (1996).
19. Gabriel Marcu and Satoshi Abe. Three-dimensional histogram visualization in different color spaces and applications. *Journal of Electronic Imaging*, **4**(4), 330–346 (1995).
20. Jon Y. Hardeberg and Francis Schmitt. Color printer characterization using a computational geometry approach. In *Proceedings of IS&T and SID's 5th Color Imaging Conference: Color Science, Systems and Applications*, pp. 96–99, Scottsdale, Arizona (1997). Also in R. Buckley, ed., *Recent Progress in Color Management and Communications*, IS&T, pages 88-91, 1998.
21. Jon Y. Hardeberg. *Acquisition and Reproduction of Color Images: Colorimetric and Multispectral Approaches*. Dissertation.com, Parkland, Florida, USA (2001).
22. J. A. Stephen Viggiano and William J. Hoagland. Colorant selection for six-color lithographic printing. In *Proceedings of IS&T and SID's 6th Color Imaging Conference: Color Science, Systems and Applications*, volume 6, pp. 112–115, Scottsdale, Arizona (1998).
23. Raja Balasubramanian and Edul Dalal. A method for quantifying the color gamut of an output device. In *Color Imaging: Device-Independent Color, Color Hard Copy, and Graphic Arts II*, volume 3018 of *Proc. SPIE*, San Jose, CA (1997).
24. Ján Morovič and M. R. Luo. Gamut mapping algorithms

based on psychophysical experiment. In *Proceedings of IS&T and SID's 5th Color Imaging Conference: Color Science, Systems and Applications*, pp. 44–49, Scottsdale, Arizona (1997).

25. Ryoichi Saito and Hiroaki Kotera. Extraction of image gamut surface and calculation of its volume. In *Proceedings of IS&T and SID's 8th Color Imaging Conference: Color Science and Engineering*, pp. 330–334, Scottsdale, Arizona (2000).
26. Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, **13**(1), 43–72 (1994).
27. P. G. Engeldrum. Computing color gamuts of ink-jet printing systems. In *SID Proceedings*, volume 27, pp. 25–30 (1986).
28. H. E. J. Neugebauer. Die theoretischen Grundlagen des Mehrfarbenbuchdrucks. *Zeitschrift für wissenschaftliche Photographie, Photophysik und Photochemie*, **36**(4), 73–89 (1937).
29. M. Mahy. Calculation of color gamuts based on the Neugebauer model. *Color Research and Application*, **22**, 365–374 (1997).
30. L. W. MacDonald. Gamut mapping in perceptual color space. In *Proceedings of IS&T and SID's Color Imaging Conference: Transforms and Transportability of Color*, pp. 193–196, Scottsdale, Arizona (1993).
31. E. D. Montag and M. D. Fairchild. Phychophysical evaluation of gamut mapping techniques using simple rendered images and artificial gamut boundaries. *IEEE Trans. Image Proc.*, **6**, 977–989 (1997).
32. Sun Micro Systems. Java. *http://java.sun.com* (Visited 2002).
33. Sun Micro Systems. Java3D. *http://java.sun.com/products/java-media/3D* (Visited 2002).
34. Sun Micro Systems. Java Advanced Imaging. *http://java.sun.com/products/java-media/jai* (Visited 2002).
35. Sun Micro Systems. Java Media Framework. *http://java.sun.com/products/java-media/jmf* (Visited 2002).
36. The Color Lab at Gjøvik Universtiy College. *http://www.fargelab.no* (Visited 2002).
37. ExactC. Color3D version 2.0. *http://www.color3d.com* (Visited 2002).
38. International Color Concortium. Specification ICC.1:2001-12. File Format for Color Profiles (Version 4.0.0) (2001).