Visualization of the 3-D Histogram of Color Space Occupation: with an emphasis on color approximation algorithms for scanned images

D. P. Huijsmans and A. Son
University of Leiden, The Netherlands

Introduction

The need for a better visualization of color space occupation and partitioning evolved from a study into the exact behaviour of color approximation algorithms. We found the usual way to evaluate the results of approximation methods by presenting test images (originals and approximated versions) rather unsatisfactory: it is difficult to substantiate the differences with the original and the printing process itself may have distroyed most of those differences and introduced others.

Especially scanned color images, that contain a substantial amount of noise present a challenge to approximation algorithms; the expected frequencies of color triplets in any of the often used colorspaces (RGB, HSI, CMY,...) are very low, which greatly influences the way color approximation algorithms proceed in their rectangular or spheric decomposition of 3-D colorspaces.

Each of the proposed color approximation schemes is particularly well suited for a certain class of 3-D color occupation histograms. This 3-D histogram structure can not be obtained from the test images itself. We agree with (ref. 1) that 3-D visualization tools should be developed for color gamut transformations. In order to characterize the spatial distribution of occupied color triplet values we evaluated several volume visualization methods.

The cloudlike triplet density distribution of scanned color images are best visualized using a depth shaded volume imaging method on a resolution reduced 3-D histogram. The presence of clusters and the amount of noise is immediately evident from stereo pairs of such visualizations.

The visualization method developed was also used to display the color space subdivision structure of approximation algorithms and logic combinations of occupancy and subdivision.

Ways to visualize the occupation of color space

The goal of color approximation is to visualize the color information in the original image as accurate as possible given a restricted set of colors (we will tacitly assume that the number of pixels in the original and approximate image are equal). Since color images are meant for visual inspection by human beings, knowledge of the characteristics of the human visual system delivers important measures for judging the success of a specific approximation scheme. Important features of human color vision are:

- color perception highly depends upon local patterns and local changes and is less sensitive to shifts in absolute values (observable in effects like mach bands, simultaneous contrast and color constancy)
- a small amount of added random noise may mask contouring
- intensity resolution is much higher than hue or saturation resolution
- our own observations indicate that most color effects become undetectable (on color displays) when color resolution goes from 5 (32 values) to 6 (64 values) bits per RGB component; resolution for the green component is worse than the red and blue components (so about 17 bits (about 130.000 different colors) would be needed for an almost always acceptable reduced color image). Careful examination of just observable differences in the printing process (ref. 2) point to as high a value as 30 bits (3 times 10) for RGB values

When choosing an appropriate color approximation method one would like to know:

- how many colors can simultaneously be represented on the display (usually, with an 8 bit color code to 3 times 8 bit RGB lookuptable, 256 simultaneous out of 16.7 million possible colors can be displayed)
- how many different colors appear in the image: an image of n by m pixels can contain a maximum of n times m different colors (a typical 1024×768 frame may contain up to 786.432 different colors and as few as 1 color)
- how often a color triplet value occurs: the more often it and its immediate neighbours occur, the more important it is to keep that color in the reduced set; less often occuring and spatially isolated color triplets especially if they are due to quantization noise may be replaced by nearby colors
- whether the distribution of color triplets is homogeneous or not: in general the distribution of occupied color triplets can have several organisations, each favouring a specific approximation scheme; homogeneous, grouped in points, grouped around points, grouped along a line segment, grouped around a line segment, grouped in part of a plane, grouped around part of a plane, and even partly homogeneous and partly grouped. In 3-D space these line and plane segments can be either straight or curved
- how broad clusters of occuring color triplets are: if the width of the cluster is due to noise it can be used as a measure for the minimal width of the spatial subdivision structure
- how isolated different clusters of color triplets are: well separated clusters will behave differently in approximation schemes than partly overlapping clusters

Instead of using the 3-D histogram of the occurance of color triplet values, most algorithms only use 2-D or 1-D projections of this distribution to achieve a spatial subdivision along one of the axes; it is therefore necessary to have a visualization of both the 3-D histogram and its main projections (2-D and 1-D histograms) to understand why an algorithm produces a specific subdivision.

The usual way to visualize the occupation of 3-D color space is a generalization of the grey level histogram of black-and-white images. Instead of presenting the histogram of grey level frequencies, one now presents 3 histograms: one for the red component level frequencies, one for the green and one for the blue component. A lot of information about the structure of 3-D color space occupation is thus thrown away, because information about the spatial coherence of different color components is lost.

A better but still incomplete picture is obtained when presenting three 2-D histograms, showing the frequencies of each combination of 2 of the 3 color components; red against green, red against blue and green against blue. These 2-D histograms are the orthogonal projections of the real 3-D density structure in the RG, RB and GB sides of the color cube. With our programs we can visualize these 2-D histograms with an added depth cue (see Figure 1a).

The only spatial coherent way to visualize the 3-D topology of the density structure present in the 3-D color space histogram, is a visualization showing the full 3-D structure using 2 stereo projections; using the unaided eye or a stereo viewer one can merge these stereo views into a perceived depth picture (Figures 1b and 1c).

Volume visualization tool

Before choosing a visualization method for our 3-D histogram, we have to say something about the minimum, average and maximum densities in our 3-D color space. A typical 1024×768 RGB scanned-in image (8 bits per channel) has a color space that consists of 16.7 million possible color triplet values. The expected occupation value is much smaller than 1, 0.045 to be precise, which means that a maximum of only 1 in 22 possible triplet values can be occupied. Any non uniform 3-D histogram will occupy less states. The original 3-D histogram therefore is a sparsely occupied space; at least 95% of the possible states are empty. One can therefore present the information in a maximum of 19 bits instead of 24 bits per pixel; however the space for all the entries of the coded RGB triplets can become as big as the original image! Scanned RGB images typically have 3-D histograms with very low frequencies; most states are empty, 1 or 2. The most important information is whether states are non-empty and how these non-empty states are distributed in color space.

An attractive 3-D color space seemed to us one which has an expected occupation value of about 1: such a color space histogram would immediately show whether the distribution of color triplets is homogeneous or not (completely filled or partly empty). An attractive visualization model would than be the binary voxel model, a data structure from volume visualization (ref. 3) registering which 3-D elements or voxels are part of an object and which are not.

In our case raising the expected frequency to about 1 amounts to lowering the color resolution so that the discrete number of states (k per side or k to the third for the total) becomes about equal to the number of pixels in the scanned color image (n times m); in the case of a 1024 by 768 image this would mean a 90×90×90 color space. For practical arguments (number of bits, powers of 2) this means choosing between a 64×64×64 or a 128×128×128 binary voxel cube. In our case we decided to use the 128 sided cube, because the color resolution is then still small enough (6 bit/channel) to allow for fair enough reduced color sets. A full screen color image of about 1640×1280 would be the optimal size for our 128 cubed color space; a 512 squared image the optimal size for a 64 cubed color space. Since these sizes represent high quality, high resolution displays on the one hand and video quality displays on the other hand, the range and quality of display images is well covered by these two reduced resolution 3-D color state spaces. Visualization of the states occupied by more than 1 triplet would give a fair indication of the non-uniformness of the 3-D histogram (Figure 1d).

Volume visualization

Due to former research efforts (ref. 4) our appartment has a volume visualization program called Exploview. It can display and subdivide 3-D scalar voxel models with sides up to 128 and covers binary, grey value and labeled voxel models with up to 256 discrete voxel values. A binary voxel model with only 2 discrete voxel values (1 bit/voxel) may be extracted from a grey value or labeled voxel model. The difference between grey value and labeled voxel models is in the meaning of the voxel value: in grey value models it represents a discrete sample from a continuous range; in labeled models each voxel value identifies a different substructure. The labeled voxel model can also be seen as the union of a series of non-overlapping binary voxel models.

Instead of producing just one binary voxel model from the 3-D histogram we mapped the frequencies of the 512×512×512 color space unto a 128×128×128 grey value voxel model; the frequencies above 255 were mapped to 255. Within Exploview we can then threshold this grey value voxel model of the reduced 3-D histogram at any value between 0 and 255. Because there is a substantial amount of noise in scanned color images, one can threshold the frequencies at a level higher than 1 to get a clearer view of non-uniformly filled regions (Figure 1d and 4).

The resulting color approximation comes down to a subdivision of color space into rectangular blocks or spheres: a maximum of 256 different color codes can be registered in the voxel values; all the voxels within a particular color coded region are then filled with that color code. This way the color coded spatial subdivision gives rise to a labeled voxel model. Just like the grey value voxel model, its structure can be visualized by selecting a particular color label or range of color labels (see Figure 2); unoccupied regions in the approximate color space get label 0 (background color).

The volume visualization program contains several shading methods, some of them can be used for binary or labeled voxel models. For binary voxel models we can use either depth shading or gradient shading. Depth shading (or atmospheric perspective) displays foreground voxels with an intensity value that depends upon the distance of

the foreground voxel to the viewer; usually the closer the lighter, the further away the darker. Gradient shading tries to estimate a surface normal direction for foreground voxels; given a voxels surface normal, a direction of view and a lighting direction, one can calculate a reflexion intensity value for the local surface voxel. These shading methods however are very sensitive to noise: only compact, slowly varying surfaces can be displayed that way; in our case due to digitization noise, the density distribution is usually cloudlike, irregular and non-compact (see for instance Figure 4). For scanned color images therefore only depth shading gives understandable results, which is why we adopted this simplest shading method.

Color approximation algorithms

All color approximation algorithms have to take care of two things: first find a specified maximum number of representative colors, and second map the unrepresented colors unto the selected ones.

Most color approximation algorithms (ref 5, 6, 7 and 8) combine the two steps by subdividing color space into local regions, usually rectangular blocks, and taking the centroid as the representative color for each block (uniform, median cut, octree quantization). For an example see Figure 5 which shows the occupation and octree subdivision of a color test image, made up of the colors in three sides of the cube and 4 trajectories; one diagonal from black to white and 3 CMY trajectories.

Some algorithms determine points in color space as representative color values (ref. 5) and map the rest of the color triplets to the nearest representative color, in which case the spatial subdivision is more like a spheric subdivision with possibly overlapping spheres centered at the representative color triplets (popularity and clustering algorithms).

One of the color approximation algorithms is not steered by the density distribution, uniform quantization, but all the others are.

Some of the approximation algorithms subdivide clusters (median cut), while others keep clusters together (clustering algorithm).

Most algorithms proceed in an iterative way; subdividing one of the already formed blocks along one of its three axes at a certain position according to characteristics of the projected density distributions. After subdividing a shrink phase may be added to create smallest enclosing blocks; our visualisation in Figure 3 shows that in the presence of noise this shrink phase is hardly ever worth the trouble.

Almost no color approximation scheme is concerned about keeping local image changes the way they are (for an exeption see ref 9), since only color coherence is represented in the 3-D histogram, and coherence in color space not necessarily coincides with coherence in image space. One way to faithfully approximate local color changes in the color image would be to develop algorithms that are steered by the 3-D histogram of pixel-to-pixel changes; for instance the color changes between 4- or 8-connected neighbour pixels. Such an approach is the subject of ungoing research within our department.

References

1 M. C. Stone, W. B. Cowan and J. C. Beatty, Color Gamut Mapping and the Printing of Digital Color Images, *ACM Trans. Graph.*, Vol. **7-4**, Oct. 1988, pp. 249-292.

2. M. Stokes, M. D. Fairchild and R. S. Berns, Precision Requirements for Digital Color Reproduction, *ACM Trans-Graph.*, Vol. **11-4**, Oct. 1992, pp. 406-422.
3. A. Kaufman, Volume visualization, IEEE Computer Science Press, Washington, D. C., 1991.
4. D. P. Huijsmans and G. J. Jense, Recent Advances in 3D Display, in P. W. Hawkes (ed.) *Advances in Electronics and Electron Physics*, Vol. **85**, Academic Press, Boston, 1993, pp. 77-229.
5. P. Heckbert, Color Image Quantization for Frame Buffer Display, *ACM Comp. Graph.*, Vol. **16-3**, 1982, pp. 297-307.
6. M. Gervautz and W. Purgathofer, Octree Quantization, in A. S. Glassner (ed.), *Graphic Gems*, Academic Press, 1990, pp. 287-293.
7. S. J. Wan, S. K. M. Wong and P. Prusinkiewicz, An Algorithm for Multidimensional Data Clustering, *ACM Trans. Math. Softw.*, Vol. **14-2**, June 1988, pp. 153-162.
8. X. Wu, Color Quantization by Dynamic Programming and Principal Analysis, *ACM Trans. Graph.*, Vol. **11-4**, Oct. 1992, pp. 348-372.
9. S. S. Dixit, Quantization of Color Images for Display/ Printing on limited Color Output Devices, *Computers and Graphics*, Vol. **15-4**, 1991, pp. 561-567.