Harvey A. Cohen, *Effective Colour Quantization for Image Display,* Proceedings, Australian Pattern Recognition Society Workshop on Colour Imaging and Applications, Canberra 5-7 December, 1994, pp37-42. Preprint.

EFFECTIVE COLOUR QUANTIZATION FOR IMAGE DISPLAY

Harvey A. Cohen
Computer Science and Computing EngineeringLa Trobe University, Bundoora Victoria
Australia 3083 Email: H.Cohen@latcs1.lat.oz.edu

ABSTRACT
Because of the limitations of contemporary display technology, images captured or synthesised with far higher broader spectrum, must be reduced to a small spectrum, usually offering just 256 distinct colours. Colour quantization achieved by using a fixed palette is very cheap but give poor results on particular pictures, whereas near ideal colour quantization has very high computational cost. A simple practical algorithm, the median or centroid cut algorithm of Heckbert, is described. This algorithm can be easily improved to take into account some basic pyschology of colour perception. Dithering will give further improvement to image display.

INTRODUCTION
In the display technology that currently predominates, only a limited number of colours can be displayed at the one time. Typically for VGA and most SVGA the number of such colours is 256, out of a palette of 2^{18} colours for six bit DACs, or 2^{24} for 8-bit DACs. Images as produced by contemporary imaging, such as CCD cameras with 3 separate sensors, do indeed produce images with a range of colour, (almost) attuned to human colour acuity, which on digitisation is usually 8 bits per primary colour for RGB format, or the equivalent colour specifically for other colour formats.

Colour quantization attempts to select a set of display colours so that a particular image will on display at a workstation be a satisfactory approximation of the "true-colour" image. Insofar as colour is a three dimensional quantity, colour quantization is a clustering problem, where one is seeking to find the distribution of image true-colours in 3D colour space, and to locate the centres of colour groups. This simple description is complicated by the reality that the sensitivity of the eye, and the ability of the eye to differentiate colours is not uniform.

As discussed below, the ahievement of colour quantization of high fidelity is computationally expensive. In this paper we first descibe two practical algorithms for colour quantization for the display of 24 bit and 15 bit RGB images on the IBM PC display adapter, the VGA,, for which only 6-bit DAC's are available. The algorithms are firstly what we term Cheap Colour, and secondly, Heckebert's Algorithm.

PICTURE INDEPENDENT CONVERSION - CHEAP COLOUR

The simplest and fastest scheme for conversion of 24bit and 15 bit colour images is to use a fixed conversion scheme:

For each pixel with red, green and blue components (r,g,b) which are
(a) byte quantities for 24 bit colour
(b) 5 bit quantities for 15 bit colour

determine an 8-bit rgb colour

rgb = (unsigned char) FIXED(r,g,b)

and use an associated look-up table with fixed entries for

LUT_RED(rgb), LUT_GREEN(rgb), LUT_BLUE(rgb).

The simplest fixed colour scheme is here called Cheap Colour.

It uses an 8-bit value, for byte r,g and b input values given by

rgb = (byte) ((r & 0xe0) | ((g & 0xe0)>>3 | (b & 0xe0) >> 5)

That is the byte rgb has just the 3 most significant bits of the red and green input components, and the two most significant bits for the blue component, consistent with the eyes lesser sensitivity to blue. The corresponding entries in the look-up table for this rgb are the 6-bit quantities:

```
LUT_RED(rgb)    =  (byte) ((rgb&0x30)>>2);
LUT_GREEN(rgb)  =  (byte) ((rgb & 0x2c) << 1);
LUT_BLUE(rgb)   =   (byte) (( rgb & 3) <<4) ;
```

Using Cheap Colour the reduction of a 24-bit interleaved colour to rgb values is lighting fast; for image data in three consecitive blocks the process takes a few seconds.

The disdvantage of Cheap Colour are that
(a) rgb spectrum may be wasted by colours that are not present or barely present in the 24-bit image
(b) subtle differences in colour - as in the graduation of skin tones may be lost
(c) false edge effects as adjacent pixels may be encoded visually markedly differently.

Other fixed picture independent image quantization schemes will suffer from these same defects. However a fixed translation (quantization) scheme may be well adapted to a certain range of images, containing, for instance, facial skin tones, and produce visually satisfactory results over this section of the input spectrum, at the cost of poorer translation in other regions of the spectrum.

MEDIAN-CUT QUANTIZATION

A practical approach to colour quantization was given by Heckert [1] in the form of the 'median-cut' colour quantization. This algorithm involves surrounding the true colour values (in 3D colour space) by a rectangular box that is the tightest possible fit,

then 'cutting' the longest dimension of the box by division at the median value, then repeating the process until as many quantization boxes as required (e.g.., 256) are available. As the iteration proceeds, the boxes shrink. The display colours to be used are then the centroids of each of these quantization boxes. For display, each colour in the image is replaced by either the centroid of the box it lies in, or, for better results, the closest such centroid (ie, possibly in a different box).. This algorithm requires the consideration of some special cases when it proves difficult to effect the 'cut'.

An implementation of Heckbert's algorithm in MSC is described in [5]. This writer's implementation in Turbo-C has used the same iterative approach, using a dynamic list of cubes that is iteratively extended until 256 cubes are available. As each cube is split, one half is first shrunk then replaces the cube in the list, the other after shrinking is added to the end of the list.

Central to the algorithm is the preparation of a list of cubes, and a 3D colour histogram.. Using 24-bit data directly, would require a 3D colour histogram that contains two bytes (if not more) per each colour, which is 2^{25} bytes, or 32 Meg, an impractical size, of no actual benefit to the algorithm. In that the output actually required involves the determination of 6-bit look-up table (LUT) values, the most apprropriate size of histogram with two byte values per entry would be 2^{19} bytes, or 0.5 Meg. However, although this amount of RAM is easily available on today's microcomputers, the compiler available, Borland Turbo-C could not readily handle more than 1Meg of data overall, so that a further compromise was made. The actual implementation used a histogram for 15-bit colour, packed into 2 bytes, with just 5 bits for each of red, blue and green. Corresponding to this choice, using just 2^{15} (truncated) colours, the 3D colour histogram, with 2 bytes available for storing each byte, was just 64K long, while the other major data structure, the list of cubes, as it was not required to hold more than 16 bytes of data for a maximum of 256 cubes, was only 4K long. By operating in a batch manner, and making use of hard disk for intermediate storage, only small buffers were required, of less than 32K in all, enabling use of a Large Data Model.

An Improved Median-Cut Algorithm

There is an obvious improvement that can be made to Heckbert's algorithm, that involves only trivial alteration to the algorithm code. In the improved algorithm box dimensions are 'weighted' before determining the (weighted) maximum dimension for applying the median cut. With a weight of 0.5 for Blue, and 1.0 for Red and 1.0 for Green, the weighted algorithm tends to increase the Blue dimension of quantization boxes, while shrinking the Red and Green dimensions, in line with the eyes greater sensitivity to red and green compared to blue. This produces a visually significant alteration to the displayed image.

CONCLUSION

Harvey A. Cohen, *Effective Colour Quantization for Image Display,* Proceedings, Australian Pattern Recognition Society Workshop on Colour Imaging and Applications, Canberra 5-7 December, 1994, pp37-42

Colour quantization is required for the credible display of image that contain by a large margin more colours, or potentially more colours, than the display device can produce simultaneously. This is a very live area of research, with various new approaches being trialed, one such, 'agglomerative clustering', recently presented by Xiang et al. [3].

For the quick inspection of images Cheap Colour gives credible results, but such display images often contain significant defects.

Heckbert's Median Cut algorithm , although far more costly than Cheap Colour, is a reasonably fast algorithm for colour quantization, taking of the order of 60 secs on a 486 DX-2 processor at 66 Mhz and theoretically give far better quality results than Cheap Colour.

However, in experimentation, the problem of inadequate specification of digital images became painfully apparent. The images data available does not include representations for standard colours, which should ideally be white and two "standard" colours.

In attempting to obtain hard copy of the experimental images, using commercial printing routines, the limitations of these printer drivers became painfully apparent. No utility was available so that the printed output could be made to match the image on the VGA display in actual colour. In addition, colour printing, being based on the further limitation of the use of 3/4 inks, uses dithering as well as unspecified quantization for reproduction. Nevertheless, in Fig 3, which reproduces 320x200 fragments from a 512x512 (True Colour (15 bit/pixel) Mandril image the qualitative difference between Cheap Colour and Median-Cut Quantization can be appreciated.

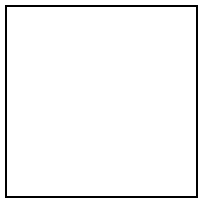
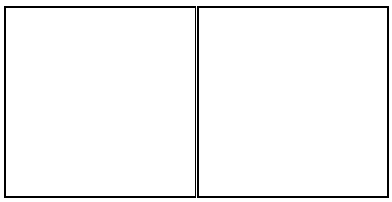


Fig 1 2D Colour Space, with just 14 pixels (•), There are visually four clusters as marked. .Minimal enclosing rectangle shown, with longest dimension along the blue axis. For Heckbert's algorithm, the first median cut of For Heckbert's algorithm the first cut would be along AA. Using the weighting scheme, with (1.0, 1.0 ,0.5) RGB weighting, the first cut would be along BB.



(a) Unweighted (b) Weighted median

Fig (2) Quantization using Heckbert Algorithm applied to same set of 14 pixels in 2D colour space as in Fig 1. Shown at the stage that there are 4 quantization rectangles .
(a) Heckbert's algorithm. (b) Modified algorithm with (1.0, 1.0 ,0.5) RGB weighting, leading to better discrimination of red, at expense of blue, in line with visual acuity. The possibility of either median cut algorithm splitting "true clusters" is somewhat exaggerated by the low pixel number in these diagrams.

REFERENCES

[1] Heckbert, P *Color Image Quantization for Frame Buffer Display* Computer Graphics Vol 16, No 3, (July 1982) pp 297-307 (Proc SIGGRAPH'82)

[2] Pratt, W.K., *Digital Image Processing*, John Wiley and Sons, New York, 1978.

[3] Xiang, Z., and G. Joy, *Color Image Quantization by Agglomerative Clustering,* IEEE Computer Graphics and Applications, Vol 14 No 3 May 1994, pp 44-48.

[4] Gervautz, M. and W. Purgatholer, A Simple Method for Color Quantization: Octree Quantization, in "New Trends in Computer Graphics" N. Magnetat-Thalmann and D. Thalmann, eds, Springer-Verlag, Berlin, 1988, pp 219-231.

[5] Kruger, A *Median-Cut Color Quantization* Dr. Dobb's Journal, September 1994, pp 46+.

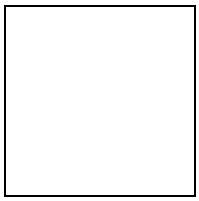


Fig 5 Histogram for 15 bit colour for public domain Mandril image, displayed as 2D image. Zero entries in the histogram are displayed as white (blank), while non-zero entries display as black. The extensive clear areas in this histogram are especially indicativ\e of the disavantages of using a fixed colour quantization scheme such as Cheap Colour.