Bull. Nov. Comp. Center, Comp. Science, 1 (1993)

Colour roundoff via octree algorithm

O.E. Baklanova and V.A. Vasilenko

Special colour roundoff problem arises in the visualization of the colour images in the computer screen under restricted palette after real valued treatment of the initial image, connected for example with the compression of colour components, and also in the scanning of colour images, palette exchanges and so on. An effective algorithm of colour roundoff based on octrees is presented in this paper.

1. Introduction

The various complicated numerical algorithms in digital colour image processing are used in recent time, and often they require the calculation with the real values. In this situation the initial colour image described by integers transforms to the few real arrays, corresponding for example to the red-green-blue components. We have here some problem for the visualization of the output image in the computer screen because its palette is restricted (under IBM standard only 256 colour combinations in the palette are possible). So, the colour roundoff problem arises in the natural way. Other reasons for the colour roundoff are also possible: an effective scanning of the colour images, the exchange of palette (Dali's picture in Gogen's palette?), data compression and so on. However in every case we need to treat all pixels, and for the huge image we need to minimize the computational expenses.

An effective colour roundoff algorithm, based on the octrees technology, is presented in this paper. Three-dimensional bisections provide the fast search of the "nearest colour" from the given restricted palette.

2. Geometric approach to the colour roundoff process and octree codes

Let us connect the three-dimensional Cartesian coordinate system with the intensities of red-green-blue components in the colour image and consider

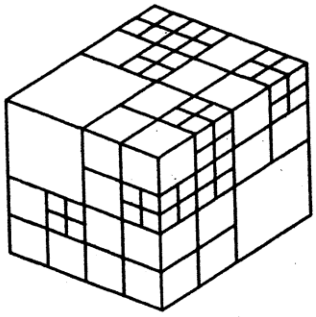

Figure. 3D-bisections

three-dimensional cube

$$\Omega = [0,63] \times [0,63] \times [0,63].$$

Every integer valued vector $P = (r,g,b)$ corresponds to the colour red-green-blue combination. Let

$$\Pi = \{P_i = (r_i, g_i, b_i), i = 1,2,\ldots,256\}$$

be some palette, and each point P_i is situated inside cube Ω, r_i, g_i, b_i are integers. The palette Π forms in Ω some scattered set, and our aim is to organize the fast algorithm to find the nearest point $P_i \in \Pi$ with respect to arbitrary real valued point $Q = (x,y,z)$ that lies inside cube Ω or outside of it.

Let us produce three-dimensional bisection of the cube Ω, i.e., cut it into eight equal cubes Ω_i, $i = 0,1,\ldots,7$. If some cube Ω_i does not contain any points of the palette Π, we call this cube "white", if it contains one point from Π then this cube is "black", in other situation this cube is "grey". In the following step we produce $3D$-bisections of the grey cubes $\Omega_{i_1}, \Omega_{i_2}, \ldots, \Omega_{i_s}$ and obtain white-black-grey cubes $\Omega_{i_k j}$, $k = 1,2,\ldots,s$, $j = 0,1,\ldots,7$. We take again only grey cubes from them and repeat this procedure. As a result of this process we obtain white-black structure of cubes without grey colour. In every black cube one point of palette Π is situated, and white cubes are empty.

For every palette point P_i we have some multi-number of the suitable black cube $\left(q_{k_1}^{(i)}, q_{k_2}^{(i)}, \ldots, q_{k_i}^{(i)}\right)$, where $0 \le q_j^{(i)} \le 7$, $j = 1,2,\ldots,k_i$, $1 \le k_i \le 6$ (because $64 = 2^6$), $i = 1,2,\ldots,256$. This integer vector we call octocode of the point $P_i \in \Pi$.

It is possible to cut the computational expenses for the construction of the octocode table by the successive calculations of palette point octocodes

with the memorizing of previous results. In every case the octocode tabulation for 256 colours in the palette is not expensive procedure.

3. Restructuring of the octocode table and fast search

First of all we need to provide the fast search in the octocode table. By these means some restructuring of the table is necessary. It looks like the fairly organizing content in the book, or more exactly in the dictionary. It is natural to organize at first the lexicographical reordering of the octocodes in the table and in addition the special recursive structure like "content in content in content...". After the construction of this modified table of octocodes we can begin the colour roundoff process.

Let $P = (x, y, z)$ be real valued vector. Transform it to the vector $P' = (x', y', z')$ by the rule

$$\begin{aligned} x' &= \min(63, \max(0, x)), \\ y' &= \min(63, \max(0, y)), \\ z' &= \min(63, \max(0, z)). \end{aligned}$$

It is clear that P' lies inside cube Ω. After that we calculate the full octocode $(q_1, q_2, \ldots, q_6)$ of the length six for the point P', $0 \leq q_i \leq 7$, $i = 1, 2, \ldots, 6$. If there are no octocodes in the table with q_1 in the first position (rare but possible case!), we need to compare the distances (for example in the uniform norm) between P' and every palette point and select the minimum (expensive but rare operation!). If the octocodes with q_1 in the first position exist in the table, we look for octocodes with q_2 in the second position only among them and so on. Any number s does exist such that octocodes of the type $(q_1, q_2, \ldots, q_s, q_{s+1}, *, \ldots, *)$ are in the table, but there are no octocodes of the type $(q_1, q_2, \ldots, q_s, q_{s+1}, *, \ldots, *)$. Then the search process is over and we determine the nearest point in palette among the points with $q_1, q_2, \ldots, q_s$ in the positions $1, 2, \ldots, s$. The particular case is $s = 6$, and in this situation we have the full identification because $(q_1, q_2, \ldots, q_6)$ is exactly in the table.

If we use some reasonable processing of the initial image, then the palette Π is not very "far" from the initial colouring of the picture and we obtain only few comparisons of distances in the last step of the octocode identification.

This colour roundoff algorithm was realized in the computer IBM PC AT 286. The execution time for the construction of modified octocode

table was 3.39 seconds for 256-colour palette. The roundoff process of the standard computer screen with $320 \times 200 = 64000$ colour pixels requires about 39 seconds.

References

[1] H. Samet, The quadtree and related hierarchical data structures, ASM Comput. Surveys, 16, No.2, 1984, 187-260.

[2] H. Samet, Implementing ray tracing with octrees and neighbor finding, Computer and Graphics, 13, No.4, 1989, 445-460.

[3] K. Yamaguchi, T.L. Kuni, K. Fujimura, Octree-related data structures and algorithms, IEEE Computer Graphics and Applications, January, 1984, 53-59.