TWO BIT/PIXEL FULL COLOR ENCODING

Graham Campbell* Thomas A. DeFanti** Jeff Frederiksen
Stephen A. Joyce Lawrence A. Leske John A. Lindberg
and Daniel J. Sandin**

ABSTRACT

Realism in computer graphics typically requires using 24 or more bits/pixel to generate an image. This paper describes a method developed by the authors called "Color Cell Compression" or "CCC" that preserves at least a limited animation and local update capability yet yields extraordinary-looking color images in approximately two bits/pixel independent of image complexity. Three intermediate methods of compressing images to six, four and three bits/pixel respectively are also described. The CCC encoding process for a 640 x 480 image averages 11 seconds on a VAX 11/750, however, the CCC method does permit real-time decoding of these images using software look-up tables and conventional display hardware. The three intermediate methods may also be decoded in real time but have the added advantage of requiring only 3-4 seconds for encoding on a VAX 11/750.

CR Categories: I.3 Computer Graphics; I.31 Hardware Architecture - - Raster display devices; I.32 Graphics Systems; I.4 Image Processing; I.42 Compression (coding).

I.0. INTRODUCTION

CCC was orignally developed to facilitate storing images in read-only memory (ROM) for systems that would have to pass serious ruggedness tests. This ruled out rotating memory sub-systems and since ROM is of course relatively expensive, compared to rotating memory media, nearly any amount of pre-processing costs are justified to reduce the chip count.

Run-length encoding is a popular method to use to store and transmit image data, easy to encode and decode. However it is efficient only if the image has reasonably long spans of identical pixels, not a feature of scanned-in and anti-aliased images. The encoding and decoding can be done by hardware permitting real-time animation yet the need for and presence of significant runs of same-color pixels severely limits the ability to do smooth shading, texture mapping, and anti-aliasing. See [10] for details on ANIMA II, a system that implements these ideas rather fully.

Aside from run-length encoding, compression of color images to 2 bits/pixel or better can be accomplished using transform coding [2]-[5]. Transform coding requires that (a) a two dimensional mathematical transform of an image block be performed, (b) discard some of the coefficients, (c) efficiently code the remainder of the coefficients, and (d) store/transmit. Decoding requires that the inverse transform be performed making this method inherently complex at both the compression and decompression stages. The Hadamard transform is the simplest transform method to implement since the matrix consists of ±1's thus only additions are required [5]. However since each of the compressed blocks represent a mathematical transform, this can introduce problems at the common edges of blocks which can complicate local updating.

The CCC algorithm for the compression of color images uses as a starting point the block truncation coding (BTC) algorithm for black and white images as developed by Delp and Mitchell [1]. A recent extension of BTC to color may be found in [12]. Equivalent work was carried out in Japan [7]-[9]. The BTC algorithm is presented and then the following color image compression

* Dept. of Computer Science, Illinois Institute of Technology.

** Electronic Visualization Laboratory, Univ. of Illinois at Chicago, m/c 154, Box 4348, Chicago, IL 60680.

Other authors are currently independent consultants.

techniques are described;

1) the BTC algorithm as extended to 24-bit color images (photos 3, 6, 10) which provides compression of images to 6 bits/pixel (photo 4),
2) an intermediate CCC technique which provides compression to 4 bits/pixel,
3) an intermediate CCC technique which provides compression to 3 bits/pixel and
4) the CCC technique which provides good quality images at approximately 2 bits/pixel (see photos 1,2,5,8,9,11).

2.0. THE BTC ALGORITHM

BTC uses a moment-preserving quantizer rather than depending upon underlying statistical distributions as do transform methods. An image is divided into n x n pixel blocks (n = 4 in our implementation), each of whose pixels are individually quantized to two levels, a and b such that the block sample mean η and variance σ^2 are preserved. If $X_1, X_2, \ldots, X_n$ are the original sampled values within a given block, then the sample mean is defined as

$$\eta = (1/N) \sum_{i=1}^{N} X_i = \bar{X} \qquad (1)$$

and the sample variance is defined as

$$\sigma^2 = (1/N) \sum_{i=1}^{N} X_i 2 \; - \; \eta^2 = \bar{X}^2 - \bar{X}^2 \qquad (2)$$

A threshold X_{th} is set to the sample mean $\bar{X}$ and a 16-bit bitplane M is created where a binary 1 indicates that the corresponding value in the original block is equal to or greater than X_{th} and a binary 0 indicates that the corresponding value is less than X_{th}. Another method for selecting X_{th} which offers slightly improved performance is given in [1] and [11].

It has been shown [1] that sample moments $\bar{X}$ and $\bar{X}^2$ are preserved by using the quantizer levels

$$a = \eta - \sigma \sqrt{(q/p)}$$

$$b = \eta + \sigma \sqrt{(p/q)} \qquad (3)$$

where q and p are the number of pixels above and below the sample mean, respectively. Pixels below the sample mean are quantized to level a and the others to level b. For each block, a, b, and a bitplane M (one bit per pixel to select quantizer level a or b) are transmitted. Using 4 x 4 pixel blocks and 8 bits for each of a and b, and 16 bits for the bitplane M, the data rate is 2 bits/pixel. The value η and σ could be transmitted instead of a and b but this requires that (3) be computed at the receiver. There is advantage in transmitting η and σ in that different quantization levels could be used for η and σ. If 6 bits and 4 bits were assigned to η and σ respectively then the date rate is reduced to 1.675 bits/pixel at the cost of computing (3) at the receiver and some decrease in quality [1].

3.0. SIX BITS/PIXEL COMPRESSION USING BTC

The BTC method as described above can be applied to color images by applying the BTC technique to each of the R,G, and B color planes. This results in three 32-bit vectors for each corresponding cell and provides good quality images at six bits/pixel. (See photos 1 and 2, 3 and 4.) All the images in this paper (except photo 9) are taken from videotape; whatever benefits of seeing the images on a high quality monitor are reduced to those antifacts that still come through as normally viewed by humans. The difference between 24-bit and 6-bit images can be discerned on a high quality monitor but the viewing distance must be less that for normal television viewing.

4.0. CCC ALGORITHM

CCC encoding provides color images equivalant to the quality of the 6/bit pixel technique described in section 3.0 but requires only 2 bits/pixel plus a fixed size look-up of 256 x 3 x 8 = 6144 bits. CCC encoding uses a BTC type binary bitmap of the luminance values of a cell to partition the corresponding cells of the R, G, and B components into two areas. Subsequently, after selecting 256 appropriate colors, each cell is represented by this 16-bit bitmap and two 8-bit pointers into a table containing 256 entries, each entry representing a 24-bit (R,G,B) color. The detailed steps are presented below.

4.1 An image is decomposed into R,G,B, and Y (NTSC luminance) planes and then each plane is divided into cells of 4 x 4 pixels. The luminance values are obtained from

$$y(i) = .30 \times r(i) + .59 \; g(i) + .11 \; b(i) \qquad (4)$$

4.2 A sample mean as per (1) is obtained for each cell of the Y plane and this value is used to partition each Y cell into two areas of high and low luminance. Each pixel with a luminance value greater than the sample mean is represented by a binary 1 and each pixel with a luminance less than or equal to the average is represented by a binary 0. This results in a 16-bit binary bitmap M for each cell. The sample variance (2) is not used.

4.3 The bitmap M is used to partition the pixels of each of the corresponding R,G, and B cells into two distinct groups. A color is then chosen for each partition that best represents the colors within that partition. This is accomplished with equations (5) and (6).

The set of equations for the 0's partition:

$$r_o = 1/m \sum_{i=1}^{m} r(i)$$

$$g_o = 1/m \sum_{i=1}^{m} g(i) \qquad (5)$$

$$b_o = 1/m \sum_{i=1}^{m} b(i)$$

where m is the number of pixels (colors) in the 0's partition.

The set of equations for the 1's partition.

$$r_1 = 1/(16-m) \sum_{i=m+1}^{16} r(i)$$

$$g_1 = 1/(16-m) \sum_{i=m+1}^{16} g(i) \qquad (6)$$

$$b_1 = 1/(16-m) \sum_{i=m+1}^{16} b(i)$$

Each cell of the original image has now been reduced to a 16-bit binary bitmap M and two 24-bit vectors, each vector representing three 8-bit color values. The 16-bit binary bitmap M specifies where the two representative colors are located within the cell. This information is written to a temporary file TEMP.
At this point the image has been compressed to 4 bits/pixel.

4.4 The 8-bit values for R, G, and B are quantized to 5 bits. Three 5-bit values can be stored in two bytes, thus at this point each 4 x 4 cell of the original image is described by a 16-bit binary bitmap and two 15-bit vectors for a total of 46 bits. The image has now been compressed to slightly under 3 bits/pixel. From a practical point of view the 16th bit in each of the two bytes of storage used for the three 5-bit color values would be used to enhance one of the colors or could be used as a mode bit. The image at this point has been compressed to 3 bits/pixel.

4.5 The following steps are carried out to obtain 256 colors most representative of the original image.

(a) A color histogram which is indexed by color and contains corresponding color frequency of occurence values is created and is sorted into ascending order by color.
(b) All frequency information for identical colors are combined so that the histogram contains only unique color frequency pairs. The ordering by color is maintained.
(c) The histogram data is processed to select 256 colors which best represent the range of colors present in the original image. Heckbert [6] devised a method called "Median Cut" which accomplishes this in an elegant manner. (Simply by choosing the most frequently occuring colors results in images as shown in photo 7.) A look-up table is created consisting of the 256 colors represented by 8 bits for each of the R, G, and B values.

4.6 The file FINAL is created by replacing each of the 24-bit vectors representing the R,G,B components in the file TEMP with an 8-bit value pointing at the color in the look-up table which best matches the 24 bit color in each partition of each cell. The look-up table is appended to the file FINAL. A possible option is to use the bitmap values, 0 or 1, to each index one of two 256 entry tables providing a choice of 512 values. There would be some duplication of colors in the two tables since a color in a 1's partition in a given cell could be the same color in the 0's partition in another cell. At this point the image has been compressed to 2 bits/pixel plus the look-up table. The encoding process is complete.

5.0. DISCUSSION OF CCC ALGORITHM

The CCC algorithm is based on the observation that in normal images change in chrominance almost always indicates a change in luminance as well. A practical reason for this is that most computer (and video) images are meant to be viewable on black and white television (e.g. blue lettering on a red background of same luminance would not be readable on a black and white television). The NTSC signal allocates far more of its bandwidth to luminance than to chrominance so that adjacent colors of the same luminance do not have sharp vertical edges.

The color space of camera-digitized or synthetic images is often as high as 24 bits/pixel, or 8 bits for each of red, green and blue. For natural images digitized at a spatial resolution of 640 by 480 pixels, the number of distinct colors produced is typically 30,000 or less, which is about ten pixels per color for this image resolution. Many synthetic images, and certain natural images such as rainbows will produce a distinct color for nearly every pixel of the ditital image. This level of color resolution, however, is seldom discernable by the human eye. For the vast majority of images encountered, 5 bits each of red, green, and blue (15 bits total) is sufficient both computationally and aesthetically.

The reduction of the R, G, and B values from 8 bits to 5 bits reduces the size of the histogram

in 4.4, this has important consequences for the production of large numbers of CCC encoded images, since the number of distinct colors to be processed is reduced by up to an order of magnitude, typically to around 5000 colors.

A wide range of 24-bit RGB images and 15-bit RGB quantizations of the same images have been approximated to 256 colors or less with look-up tables generated by the median cut process. The results have been acceptable, with no difference in quality using either the 24-bit or the 15-bit histogram as the source. This same conclusion applies to the CCC encoded versions of the very same images.

The CCC encoding method essentially introduces large color "jaggies" but retains the luminance information fairly well. This is true for the 6, 4, 3 and 2 bit cases. It rather poorly encodes the case where more than two colors exist in a cell, as when three different color vectors intersect. Nevertheless, the images produced look much better than we originally predicted given the limitations of the CCC method.

CCC has the same desirable property of frame buffers, that is, each area on the screen has a unique area in memory containing the information describing that area. Thus to update a CCC sub-image, one only needs to change the corresponding blocks. Colors for the replacement blocks must be in the look-up table. When using CCC in a dynamic environment, i.e. modifying parts of the image, it is desirable to median cut to less than 256 colors thus allowing for the creation of new colors or to reserve certain values for matting purposes.

6.0. IMPLEMENTATION CONSIDERATIONS

The implementation of CCC is straightforward in that the algorithm is computationally simple. CCC can be implemented using integer arithmetic and still generate acceptable images. The histogram building and the median cut algorithm in the compression phrase are time-consuming but not inordinately so. A 640 by 480 image may be processed in 11 seconds on a VAX 750. The 6 bit/pixel BTC method, and the 4 and 3 bit/pixel intermediate CCC methods do not require the histogram construction or median cut algorithm and therefore will run in 3-4 seconds on the VAX 11/750. This includes the time for loading the 640x 480x3 bytes from disk storage. CCC offers a distinct economic advantage over the compression techniques for many applications in that the decompression phase consists solely of a table look-up operation. Software decoding could be implemented on any microcomputer which has display hardware with a capability of 256 colors chosen from a larger palette.

The CCC encoding algorithm as implemented requires slightly over one megabyte of memory for code and data, most of which is used for the median cut tree, the initial 640 x 480 x 24 bit image and the associated blocks which are generated. The first stage of compression, 4.3, reduces memory requirements by 75% but approximately 270K of this freed up memory is used by the median cut algorithm.

7.0. APPLICATIONS

7.1 Animation Considerations

Standard film animation for cartoons is "done on two's", that is, twelve frames per second. Real time playback of 320 x 480 CCC images requires a transfer rate of 12 x 320 x 480 x 2 = 3,686,400 bits/second. Adding 73,728 bits for twelve 768 byte lookup tables results in a required transfer rate of 3,760,128 bits or 470,000 bytes per second. Current microcomputer hard disks have a transfer rate of 5,000,000 bits or 625,000 bytes per second. Allowing for "no data" space on the disk tracks and head movement it means that a 30 megabyte hard disk, now standard on the IBM AT, could store and display in real time approximately 60 seconds of animation.

Full size 640 x 480 CCC images require a transfer rate of 940,000 bytes per second for animation purposes. This is well within the transfer rate of conventional hard disks on mini and larger computers.

Much of the interest in developing CCC came from the idea of storing digital images (and program information) on videodisks, essentially a giant ROM, albeit analog. The teletext industry provides access to encoding and decoding chips which store 256 bits/line of video or 15,360 bytes/frame, assuming 480 active lines. This is 460,800 bytes per second. The calculations above indicate that the real-time play-back of 320 x 480 pixel images is possible using CCC.

Digital encoding provides some advantages over conventional NTSC encoding;

1) Analog video loses 3 db per copy made, digital nothing.
2) Panning an analog image requires that an entire frame be retransmitted each update; digital panning requires that only the edge(s) in the direction of the pan be transmitted.
3) High resolution images, say 1024 x 1024, may be encoded; not possible with analog NTSC.

These advantages apply to any digital encoding technique; CCC requires one-twelfth the storage of conventional 24-bit quantized R, G, B, images.

7.2 Electronic Shopping and Image Transmission in General

A potential application of CCC is electronic shopping. In such a system the editing and coding of the images takes place at a central facility, and the images are then transmitted to possibly thousands or tens of thousands of viewers for display,

preferably with the addition of spoken comment, via cable. The advantage of CCC in this type of application, aside from the reduction in storage requirements and data rates, is the minimal cost of the decoding hardware, which must be supplied in large quantities. The same hardware could be used to provide "slide and sound" shows on a variety of subjects on a demand basis.

The previous paragraph describes one-to-many applications; CCC would also be useful on a one-to-one basis. For instance a graphics arts shop could transmit a compressed version of a sample color picture to a customer for review and approval of changes in layout and content. A typical 640 x 480 color image can be compressed to 620,000 bits thus the image could be transmitted in approximately one minute over a 9600 bit-per-second phone circuit. Again in such an application the major computational and memory requirements are at the producer site, the consumer site only requires implementation of a look-up table aside from the conventional display electronics.

Many-to-many applications are also conceivable in teleconferencing where an image representing each speaker's face could be encoded and distributed to all participants for display. (See photo 9). Only the image representing the person talking would remain "live" and be transmitted to the other conferees. The entire image could of course be utilized to display a drawing requiring higher resolution.

8.0. CONCLUSIONS

The CCC algorithm for compressing color images has been described. The potential saving of 22/24 or approximately 90% of storage or transmission costs is particularly attractive in consumer or personal graphics applications. Furthermore CCC encoding of an image guarantees that the encoding results will not change in size with image complexity. Also any change to the image not requiring a change of the color lookup tables can be effected on a purely local basis, an important consideration when updating parts of encoded images. A further advantage of the CCC algorithm is that it is easily implemented with conventional display hardware and microcomputers, and the decoding is particularly easy to implement.

ACKNOWLEDGEMENTS

The authors wish to thank the Bally Corporation for several years of support, Ed Catmull and Pat Cole at Lucasfilm Ltd. for some early help in demonstrating the CCC ideas, and John Whitney Jr. of Digital Productions, Inc. for supplying some of the test images.

REFERENCES

[1] E.J. Delp and O.R. Mitchell, "Image compression using block truncation coding,", IEEE Trans. Commun., vol. COM-27, Sept. 1979.

[2] W.K. Pratt and H.C. Andrews, "Fourier transform coding of images, "Proc. Hawaii Inter. Conf. System Sciences, Jan. 1968.

[3] P.A. Wintz, "Transform picture coding," Proc. IEEE, vol. 60, no. 7, pp. 809-820, July 1972.

[4] W. Chen, C.H. Smith, and S. Fralick, "A fast computational algorithm for the discrete cosine transform, "IEEE Trans. Commun., pp. 1004-1009, Sept. 1977.

[5] H.J. Landau and D. Slepian, "Some computer experiments in picture processing for bandwidth reduction, "Bell Syst. Tech. Journal, vol. 50, pp. 1525-1540, May-June 1971.

[6] Paul Heckbert, "Color image quantization for frame buffer display," SIGGRAPH 1982 Proceedings, pp. 297-307.

[7] T. Kishimoto, E. Mitsuya, and K. Hoshida, "An experiment of still picture coding by block process" (in Japanese), Nat. Conf. of the Inst. of Elec. and Commun. Eng. of Japan, March 1975, no. 974.

[8] T. Kishimoto, E. Mitsuya, and K. Hoshida, "An experiment of still picture coding by block processing" (in Japanese), Nat. Conf. of the Inst. of Elec. and Commun. Eng. of Japan, March 1978, no. 975.

[9] T. Kishimoto, E. Mitsuya, and K. Hoshida, tech. group of commun. systems of the Inst. of Elec. and Commun. Eng. of Japan, pp. 63-69, July 1978.

[10] R.J. Hackathorn, "ANIMA II: A 3-D color animation system," 1977 SIGGRAPH Proceedings, pp. 54-64.

[11] D.R. Halverson, "On the implementation of a block truncation coding algorithm, IEEE Trans. Commun., Vol Com-30, No. 11, pp. 2482-2484, Nov. 1982.

[12] M. Lena and O.R. Mitchell, "Absolute Moment Block Truncation Coding and its Application to Color Images," IEEE Trans. Commun., Vol.Com-32, No. 10, October 1984, pp. 1148-1157.

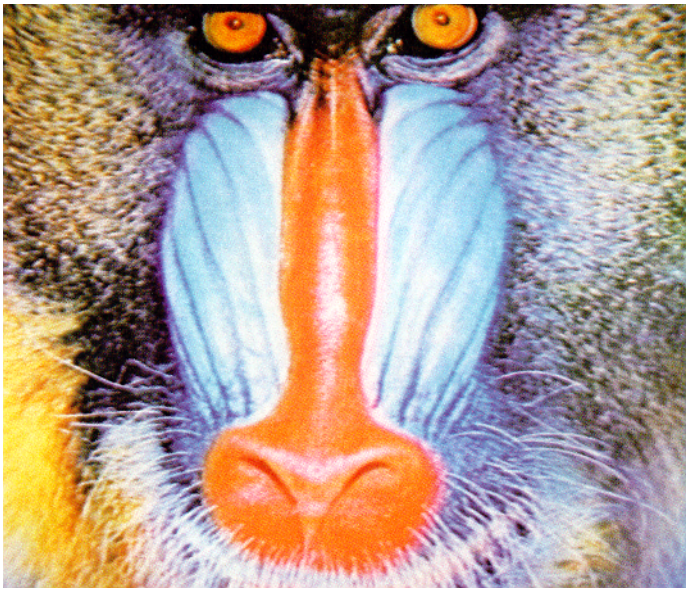

Photo 1: 512x480x2 bits/pixel CCC Image

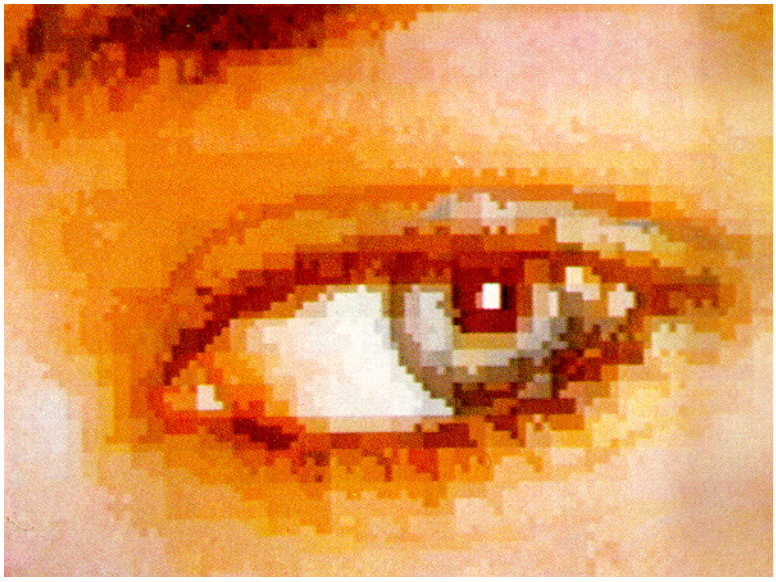

Photo 2: Detail of Photo 8. 2 bits/pixel CCC Image

Photo 3: 512x480x24 bits/pixel Image

Photo 4: 512x480x6 bits/pixel 3-color BTC Image

Photo 5: 512x480x2 bits/pixel CCC Image

Photo 6: 512x480x24 bits/pixel Image

Photo 7: 512x480x2 bits/pixel CCC Image showing the most 'popular' 256 colors

Photo 8: 512x480x2 bits/pixel CCC Image colors selected with median cut algorithm

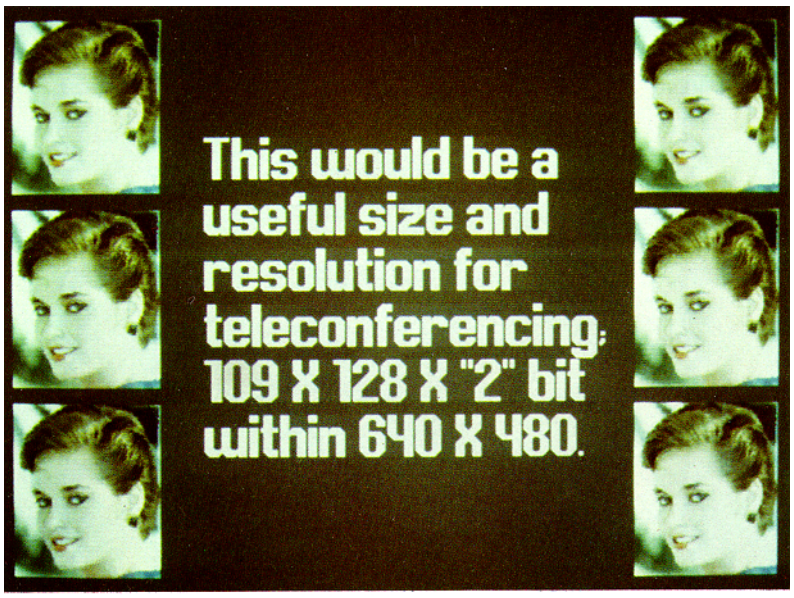

Photo 9: Potential Application of CCC
(Assume faces are different)

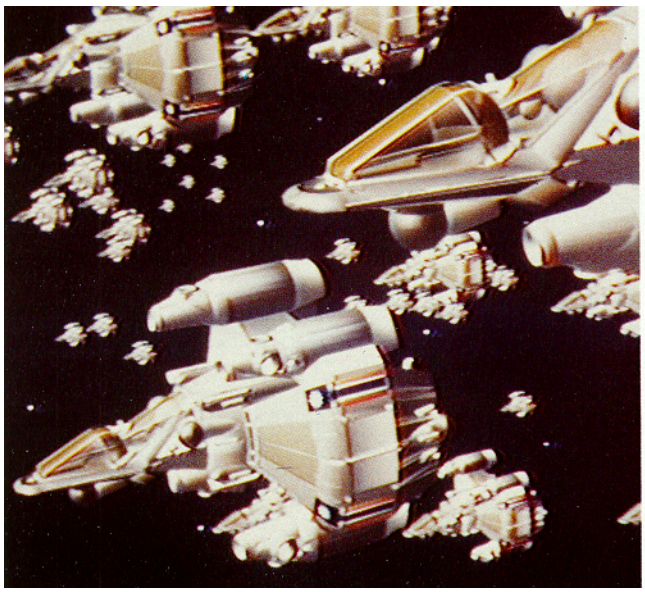

Photo 10: 512x480x24 bits/pixel Image
(Courtesy of Digital Productions, copyright 1984, Digital Scene Simulation(SM). All rights reserved.)

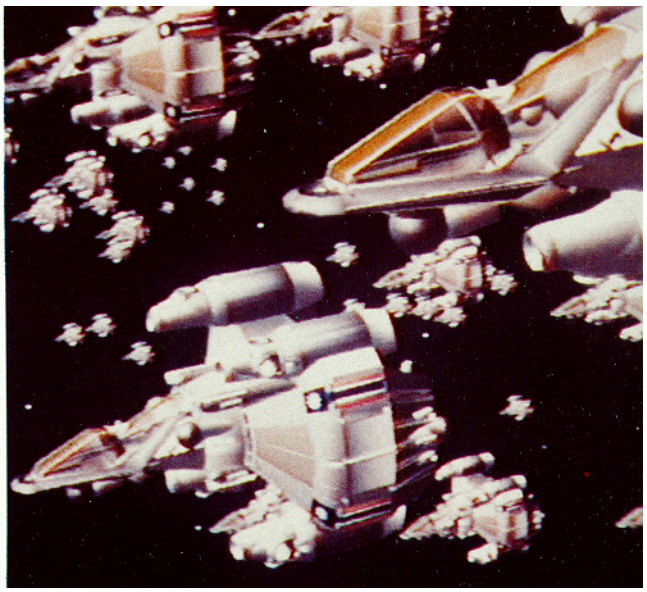

Photo 11: 512x480x2 bits/pixel CCC Image
Note the re-introduction of jaggies on sharp edges.
(Courtesy of Digital Productions, copyright 1984. Digital Scene Simulation(SM). All rights reserved.)