Progressive Refinement of Colormapped Image

KWONG Lap-ming

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Information Engineering

January 2001

Acknowledgement

I am grateful to Dr. Ng Wai-Yin for his kind assistance on my research work throughout these two years. I am also grateful to Mr. Chei Kwok Hung, Mr. Hung Wai and Mr. Ng Sau Koon for their kind help in the subjective test of the progressive image transmission. Lastly, I have to say thanks to my family for their supports and encouragement during these years.

Abstract

The thesis focuses on tackling the problem of image communication over a low bandwidth network connection. The new algorithm is a loss-less progressive transmission of colormapped image that allows early image recognition and provides flexible progression control of the spatial and the contrast resolution. The usage of colormapped image takes advantage of the characteristic of viewing device for efficient image communication.

The new method allows a colormapped image to be refined progressively in both spatial and contrast resolutions. For progressive fidelity transmission, a binary-tree likes color table is constructed. During color quantization, a color cluster is splitted into two child clusters and an approximate color of each child cluster is stored as a child node. The successive color approximations of each cluster are calculated and are stored in a binary tree until the desired number of quantization regions reached. With this data structure, the contrast resolution can be refined progressively during transmission by traversing the tree beginning with the root of the tree. Besides, the image pixels are reordered so that the distribution of pixels to be displayed can be more uniform for progressive spatial resolution as well.

On the other hand, it is believed that different image applications need to be transmitted in different transmission sequences for providing better viewing experience. Therefore, in the transmission algorithm, it provides flexibility in controlling the contrast and the spatial resolution progression so that it is possible to refine different images with their own optimal transmission sequences.

Both the server and the client programs were written in Java to implement the progressive refinement of colormapped image. The experiment results are given and are used to compare with some other techniques. It shows the new algorithm can give good performance for progressive image transmission.

簡介

此論文關於解決在窄頻網絡上圖象通訊的問題。 新的方法是將色彩配對圖象以漸進流送的方法傳送。 這種方法可令觀察者快速確認圖象的內容, 以及提供在平面解象度和色深解象度具彈性的控制方法。與此同時, 透過利用用戶的圖象顯視器顯象能力的限制, 傳送有不同色彩數量的色彩配對圖去提高圖象通訊的效率。

這一新方法容許色彩配對圖象在平面解象度和色深解象度同時地漸進優化。爲了達到色深解象度漸進優化的效果, 我們制造了一棵二元樹狀調色盤。在色彩量化過程中, 每一個色彩群會被分斥成兩組, 而每組會有自己一種代表顏色, 並將其作爲先前組群的子色群。 每一子色群的代表顏色會被計算出來並儲存在這二元樹中, 直到整個顏色群被分斥成所需要的數目。 利用這一排列, 每當從這二元樹的頂部流向底部, 圖象的色深解象度便能漸進優化。 另一方面, 爲了達至平均漸進優化圖象的平面解象度, 圖象象素的次序會被重新排列, 達到每一象素群都能平均分佈於二元空間之中。

再者， 由於不同圖象的應用需要不同的數據傳送次序來達到良好的圖象流送效果; 因此, 在新方法之中會提供具彈性的流送過程的控制, 籍著改變數據傳送的次序, 令圖象的平面解象度和色深解象度的增長速度能切合各種圖象應用的需要。

最後, 兩個用爪哇程式語言所編譯的伺服程式和用戶程式會用來去實踐整個圖象流送的過程。 實驗的結果會與其他方法作比較並加以分析。 在這可發現新的方法能夠達到良好的圖象流送效果。

Contents

Chapter 1

Introduction to Image Communication

Image communication is a very important application of information networks. There are many image applications used frequently over network such as still image transmission, interactive search of image database, etc. With limited network bandwidth, user often has to wait for a long time to receive the required images. The problem becomes more severe when the bandwidth of network is very low such as mobile network connection, which only supports 10k bps or lower transmission rate. For a color image with 256 by 256 24-bit pixels, its file size is $192K$ bytes. To transmit it over a 56k bps network connection, it needs about 27 seconds.

Image communication can have two kinds of usage. One of them is image file transfer. The whole image file is transmitted to the remote side for storage or other purposes without any processing during transmission. Therefore, the image file size is the only issue to be focused on for improving the efficiency of image transmission. Data compression can sufficiently solve the problem of transferring image file over a low bandwidth network.

Another usage is that still image is transmitted for browsing. A still image stored in a remote site is requested by a user for display on his viewport. The user expects that the image can be displayed on his device as soon as possible. Sometimes, the user may also want to control the whole communication process that allows him to save much time on finding the needed information. Interactive search of image database is a good example. In this application, user intends to browse the received image in a short period. A full resolution image may not be useful for this kind of application. The user needs a fast response on early image recognition. Therefore, it is more efficient if an intermediate image can be displayed during transmission instead of receiving all image data before display. Moreover, for supporting fast searching process, the user should be able to interrupt and control the transmission or initiate a new request at any time.

Unlike the process of image file transfer, the image transmitted for browsing is required to display on client's viewport as soon as possible. Therefore, the display algorithm and the characteristic of client's viewport are important issues in this kind of image communication. Display algorithm works with specified transmission algorithm to construct an intermediate image by using a subset of image data. It allows the viewer fast image recognition within a short period. Moreover, different client's viewing ports have their own displaying characteristic. They may only be able to display fewer numbers of colors than that in an image.

There are several issues that will affect the performance of image communication. The first issue is the file size of image. Under a limited-bandwidth network, the file size of the image is directly proportional to time needed for transmitting the image over a network.

The second one is the bandwidth of network connection. There are many kinds of networks like wireless network, wide-band network, mobile phone network and etc. They have different limitations on the network bandwidth.

The third one is image display. The straight forward approach is to use scanline order when an image is displayed from top to bottom and left to right. Alternatively, progressive scan displays an approximate image within a short period and refines it gradually during transmission. For the latter case, there are many variations on how to refine an image. The increasing rate of both spatial and contrast resolutions affects viewing experience significantly. The complexity of progressive scan method is higher than scanline order method because image data in the progressive scan method is not consecutively ordered and displayed one by one as in the scanline method. In the following figure, it shows the difference between these two image display methods. The progressive scan method allows faster image recognition than the scanline method.

Figure 1.1: Image display in scanline order and progressive scan (bytes received order: 505, 1126, 3789, 14746, 41472, 65536)

The characteristic of display device also influences the performance of image communication. Some devices may only be able to display 256 or fewer colors like handheld PC or palm pilot. Some other devices may be able to display 24-bit true colors. If a true-color image is transmitted to a display device which can only display 16 colors, it wastes much time. Ideally, images should be transmitted according to the display device's characteristic. Any viewport on a display device has an associated color table. It uses colormapping model to map image value to the colors in its color table. When a color inside an image is not included in the device's color table, the pixel will be mapped to the most similar color in the device's color table by color quantization. The computational cost of color quantization is relative large when the processing power of client's device is very low. Therefore, there will be a benefit if we perform the process of color quantization before image communication so that image can be displayed faster.

To improve the performance of image communication over a low-bandwidth network, we can focus on the three mentioned issues. However, as the network bandwidth cannot be increased dramatically, we only try to reduce the data traffic over network and transmit data in a more efficient way.

For an ideal image communication, an image should be compressed as much as possible without distinguishable quality degradation. Moreover, the compression algorithm should also consider the display capability of client's device in order to eliminate as much redundancy as possible. Afterward, the compressed image data is transmitted and an approximate image should be displayed early that allows fast image recognition. Then, there should be an optimal improvement of the degraded image during further data transmission. Other user controls are also desirable; for example, the user can stop the communication at any time, increasing the spatial or the contrast resolution of image, etc. With these kinds of functions, the image data are transmitted on demand. Moreover, as different images have different transmission preferences, there should be a flexibility to adjust the data transmission sequence; for a textual image, the spatial resolution should increase faster; for an art image, the contrast resolution needs to increase faster. Certainly, the whole process should be as simple as possible so that its computational cost can be kept low.

1.1 Existing Approach improving Image Communication

Two approaches improving image communication are image compression and progressive image transmission.

1.1.1 Data Compression

Compression coding is applied to a digital image to reduce its file size with no or little perceivable quality degradation. It works by removing statistical redundancy in the image representation as well as image details that the human visual system is insensitive to. JPEG and JPEG 2000 are the current and upcoming industrial image compression standards.

JPEG is a widely used image compression standard that its compression ratio of color images is between five and ten without causing perceivable quality degradation. In this approach, an image will be divided into 8×8 pixel non-overlapping blocks. Each block is then transformed from time domain into frequency domain by Discrete Cosine Transform (DCT). It results in DCT coefficients. Since a subset of DCT coefficients already contain a major part of image information so that some frequency coefficients can be quantized with fewer bits than others. Huffman coding is then applied to encode the coefficients for transmission.

JPEG 2000 [1] delivers extremely high compression ratios and guarantees high image quality. It will offer both lossy and lossless compression. JPEG 2000 abandons DCT compression in favor of wavelet compression. An image is coded wholly to form a continuous data stream instead of block by block, and so avoid the blocky artifacts associated with DCT compression. As an added benefit, the continuous wavelet stream can be uncompressed incrementally, so the same file can be viewed (or printed) at multi-resolutions. The wavelet stream can be truncated at virtually any point. It will let user download the same image at different resolutions depending upon the available bandwidth. A single small file, easy to store and transport, that when partially decompressed is suitable for relatively low-resolution screen display and local proofs, but when fully decompressed contains enough resolution for final output.

However, data compression is not a complete solution for image communication because the observer needs to wait for all compressed data to be received and decompressed before the image can be rendered. It benefits most for image file transfer. Moreover, image compression algorithms do not consider the characteristic of display device. Extra redundancy exists when the contrast resolution of display device is lower than that of the original image file.

1.1.2 Progressive Image Transmission

The concept of progressive image transmission is that a low-resolution image is sent to client first for immediate image recognition to viewer. Then extra bits are transmitted to the client to refine the image gradually. The goal is to shorten the time for the viewer to perceive the image content. The effectiveness of the image viewing process can be improved. When used in conjunction with control functions such as selecting a part of image for display, increasing or decreasing the spatial resolution of the image and so on, progressive image transmission makes the image viewing process interactive and efficient.

Progressive image transmission benefits most when the image file size is large with respect to the network bandwidth. If a full resolution image, with size 256 by 256 24-bits pixels, is transmitted via 56K bps network connection, it needs 27 seconds or more. However, if progressive image transmission is used, the user may only spend few seconds to receive a low-resolution image for early viewing. After the user recognized the image content, he can determine whether to wait for a higher quality image or to stop the transmission. So, the user can reserve much time for those images he really wants.

Progressive image transmission can provide greater flexibility in image communication. Map viewing is a good example. As the file size of a full resolution map image may be very large, it is inefficient to transmit over a network. The user may not need an entire map but only a part of it at a certain resolution. By means of progressive image transmission, a map image server may send a low-resolution map to the client first. Then, the user may choose which part of the map needs to be refined further to get more detailed information. The user may zoom in, zoom out and pan the map to find out required information inside the whole map. Such functions work alongside progressive image transmission naturally.

Compared with the scanline method, progressive image transmission needs more computation for encoding and decoding an image before it can be rendered on client side. A specified display algorithm is required to reallocate decoded data into its corresponding pixel for display. Therefore, the computational cost of progressive image transmission is relative large. As continuous refreshing of image during transmission, the latency time should increase as well. However, the latency caused by a low bandwidth network often is much more than that caused by extra computations in progressive image transmission. As mentioned before, it needs 27 seconds or more to transmit an image with size 256 by 256 24-bit pixels over a 56K bps network. Excluding the time for image encoding or preprocessing, the time spent for decoding and displaying the intermediate image during progressive image transmission is less than one second by using a PC with 200M Hz processing rate. With some increase in computational cost, progressive image transmission provides better viewing experience than the scanline method.

While data compression may reduce data traffic in image communication, progressive image transmission increases the efficacy of image communication.

Both should be combined in an algorithm that takes consideration on the characteristic of display device as well to improve image communication over a low-bandwidth network. Ideally, a color image should be compressed with the consideration of the characteristic of display device. The encoded image is then transmitted progressively and then an intermediate image should be able to display on the client's viewing port at any time. To fulfill different images' requirements, data transmission sequence should be adjusted to fit for their own transmission preferences as well.

Several techniques of progressive image transmission will be reviewed in the chapter 2. In chapters 3 and 4, the new method - progressive refinement of colormapped image is introduced. In the chapter 5, the performance of the new method is evaluated. Chapter 6 comprises the discussion and conclusion sections.

Chapter 2

Review of Progressive Image Transmission Methods

The aim of progressive image transmission is to improve the efficacy of image transmission that takes time. Image data are reordered according to their marginal contribution to the visual display that the client constructs and updates continually. Ideally, every bit transmitted is the best choice among those unsent in terms of improvement to the current display. Such improvement is often one of resolutions, spatial or contrast. Visual perception however requires certain minimum resolutions. Progressive image transmission should be initialized with a batch transmission of a visually meaningful, albeit low resolution, proxy image, when the principal concern is simply speed.

Three noteworthy approaches to progressive image transmission are pyramidal image coding, hierarchical data structure with bit plane transmission, and embedded transform coding.

2.1 Pyramidal Image Coding

An image may be recursively down-sampled by some fixed factor to form a set of new images with lower spatial resolutions. For an image X_n of size $2^n \times 2^n$, an image pyramid is defined as a sequence $\{X_k\}$. By a sequence $\{X_i\}_{i=1,\dots,r}$ is sufficient for reconstructing X_r, which is an image with level r spatial resolution. Progressive image transmission may proceed by scanning the pyramid from the lowest spatial resolution image on "top" towards the highest spatial resolution one at the "bottom". The pyramidal data structure of an image is shown as follows.

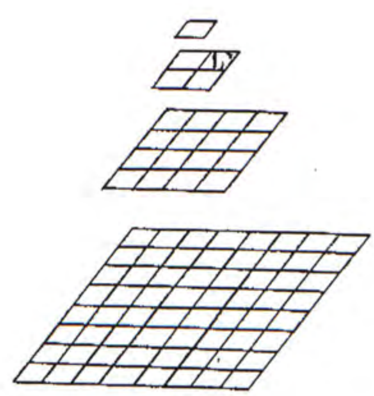


Figure 2.1: The pyramidal data structure of an image with four levels.

There are two types of image pyramid, namely, expansive pyramids that incur image size increase and non-expansive ones that do not.

2.1.1 Expansive Image Pyramid

An image pyramid is expansive when its size is greater than that of the original image. For example, an image X_n of size $2^n \times 2^n$ is down-sampled by 2 until producing an image with size 1x1. The size of the pyramid is larger than that of the original image size by one third. We examine two image formats below that use expansive pyramids, the Flashpix image file format and the FELICS image coding method.

2.1.1.1 Flashpix [3] and Internet Imaging Protocol(IIP) [2]

FlashPix [3] is an image file format. A Flashpix file comprises multiple versions of an image at different spatial resolutions, beginning with a largest base resolution. The version at each resolution level is sub-divided into 64x64 square tiles. Each tile may be independently accessed and individually displayed with no regard of other image data. Each tile is therefore a subimage, and may be JPEG compressed. A Flashpix file is larger than the corresponding JPEG file of the same JPEG quality level by one third.

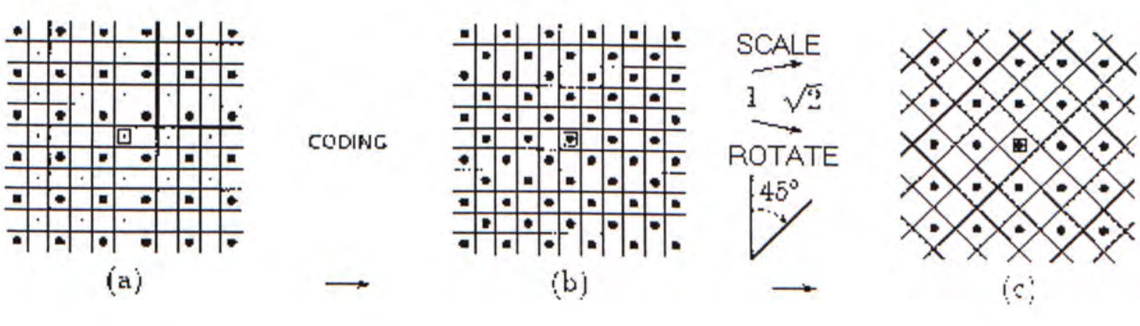
Progressive image transmission with user control is achieved using the Internet Imaging Protocol (IIP) in conjunction with a Flashpix file. An IIP server sends a low-resolution image to the client first. Then, the user specifies parts of image to be refined and sends the request to the server. The server receives the request and sends back needed image tiles in response. As a result, the user may zoom in, zoom out, pan the image to different regions and stop the transmission process at any time.

2.1.1.2 Progressive Fast, Efficient, Lossless Image Compression System (FELICS) [10]

Progressive FELICS uses expansive pyramid in conjunction with lossless image compression for progressive lossless compression of grayscale images. It combines Multi-Level Progressive image compression Method (MLP) with FELICS, which is a simple context-based image coding method.

Multi-Level Progressive Method (MLP) is based on a hierarchical pixel sequence, which encodes the image in different spatial resolution levels; the first level corresponds to a very highly compressed and very lossy encoding, and each successive level provides more detail, until the encoding is completely lossless.

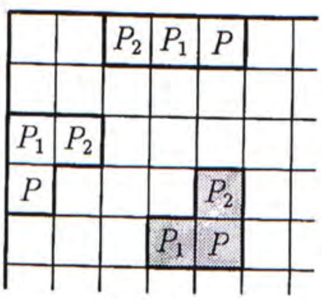
Figure 2.2: MLP coding process



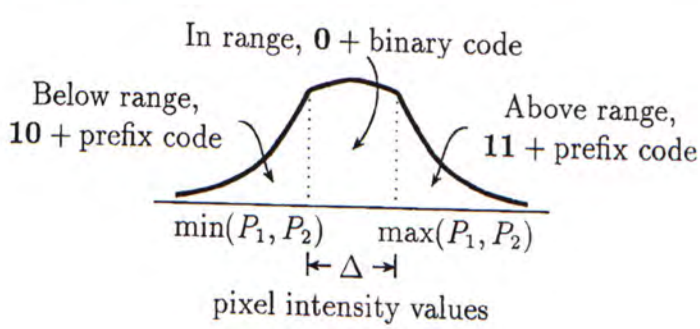
In each spatial level, the known pixels form a square grid as in figure 2.2(a). The midpoint of each grid square, which is represented by a small dot in figure 2.2(a), will be predicted. After coding the midpoints, the known pixels form a checkerboard pattern as in figure 2.2(b). For the next level, the coordinate system is scaled by $\sqrt{2}$ and rotated by 45^o. Then, the known pixels in figure 2.2(c) form the same pattern as in figure 2.2(a). In the next level, the midpoints are coded again. Arithmetic coding is applied to code the prediction error using the distribution supplied by the model.

FELICS is based on a simple function of four nearby pixels. Two of the four nearest known pixels are selected for coding, using single bits, adjusted binary codes, and simple prefix codes like Golomb codes. The coding parameter is estimated adaptively for each context, which is the absolute value of the difference of the predicting pixels. The adaptation statistics are adjusted at the beginning of each level in the progressive pixel sequence.

Figure 2.3: FELICS coding process. a) Coding context P with two nearest neighbors P_1 and P_2. b) Coding for different intensity ranges relative to the larger and smaller of the two context levels.



(a)



(b)

2.1.2 Non-Expansive Image Pyramid

A pyramidal data structure is non-expansive when the size of the pyramid is not greater than that of the original image size. The reduced-difference pyramid is a classic example of non-expansive pyramids used for progressive image transmission.

The reduced-difference pyramid [5] is derived from the truncated mean pyramid. The truncated mean pyramid is formed by successively averaging and truncating image blocks of 2x2 in size. The total number of pixels in the truncated mean pyramid is about one third more than the number of pixels in the original image.

For an image X_n of size $2^n \times 2^n$, the reduced-difference pyramid $\{D_k^*\}$ is constructed as follows:

1. Initialization: Let $k = n$
2. Compute the differences between the neighboring nodes for each spatially contiguous, non-overlapping block of size 2x2, for level k,

$$D^*_{k,i,j} = X^*_{k,i,j} - X^*_{k,i,j+1}$$
$$D^*_{k,i,j+1} = X^*_{k,i,j+1} - X^*_{k,i+1,j+1}$$
$$D^*_{k,i+1,j+1} = X^*_{k,i+1,j+1} - X^*_{k,i+1,j}$$
$$D^*_{k,i+1,j} = X^*_{k,i+1,j} - X^*_{k,i,j}$$

i,j = 1,3,...$2^k - 1$.

3. Formation of Level k-1: For each spatially contiguous, non-overlapping block of 2x2 pixels at level k, the truncated mean is calculated,

   $X^*_{k-1,[\frac{i+1}{2}],[\frac{j+1}{2}]} = [\frac{X^*_{k,i,j}+X^*_{k,i,j+1}+X^*_{k,i+1,j}+X^*_{k,i+1,j+1}}{4}].$

   i,j = 1,3,....$2^k - 1$.

   $[\alpha]$ is the truncation of $\alpha + 0.5$.

4. Let k = k-1 and if k $\neq$ 0, return to step 2; otherwise stop.

$D_{k,i,j}$ and $D_{k,i+1,j+1}$ are the differences of the neighboring nodes on the same row, and $D_{k,i,j+1}$ and $D_{k,i+1,j}$ are the differences of the nodes on the same column. In fact, only three out of four differences are needed to recover the truncated mean pyramid X^*_k. The reduced-difference pyramid is defined as a sequence $\{D_k\}$. A sequence $\{D_k\}_{k=1,\ldots,r}$ is sufficient for reconstructing $\{X_r\}$ that is an image with spatial resolution at level r. Therefore, the number of pixels equals to that of the original image. By this data structure, the original image can be exactly reconstructed starting from the lowest spatial resolution.

2.1.3 Pros and Cons of Pyramidal Data Structure

The main advantage of the pyramidal data structure is that the multi-spatial resolution transmission can be easily achieved. By this structure, the image is transmitted from the lowest resolution to the highest resolution with little computation. Moreover, the construction of the pyramidal data structure is simple.

Expansive pyramids require more storage space but less computation than non-expansive ones for displaying multi-resolution images. Besides, as the image data of each level in expansive data pyramid are coded independently, different parts of image in one spatial level can be easily and independently accessed.

2.2 Hierarchical Data Structure with Bit Plane Transmission

The method of hierarchical data structure with bit plane transmission may progress in both spatial and contrast resolutions of the image. It allows greater flexibility for changing data sequence and then the progression rate of both resolutions.

In this method, image data is stored in a tree-like data structure. Pixels are allocated in different nodes in different levels of the tree. The top of the tree represents the lowest spatial resolution and the bottom of that represents the highest spatial resolution. From the most significant bit to the least significant bit of image data, they are located from the top to the bottom. During transmission, the tree is traversed down from its top and so both spatial and contrast resolutions increase progressively at the same time. Some proposed methods also include data compression algorithm to eliminate redundancy. The Bitwise Condensed Quadtree method (BCQ) [6] and Progressive Transmission of Full-Search VQ [7] are two well-established examples.

2.2.1 Bitwise Condensed Quadtree method (BCQ)[6]

The Bitwise Condensed Quadtree algorithm is a method for progressive image transmission increasing both spatial and contrast resolutions at the same time. It combines gray scale and spatial hierarchies in a single tree. An image is first divided into four regions (four quadrants); each region represents the child node of the root of the tree. Then, starting from the most significant bit of pixels, whenever all the first bits of all the pixels of a region are the same, these bits are removed and added to their corresponding nodes of the tree. After all regions in this level had been examined, each quadrant is divided into four quadrants individually. The same process for finding any common bit of the pixels in each region is applied again. For example, we have a gray scale image with a binary map shown in Figure 2.4; its corresponding quadtree is shown in Figure 2.5.

000	001	011	011
001	001	011	011
010	011	100	100
010	011	101	110

Figure 2.4: Binary map of a gray scale image

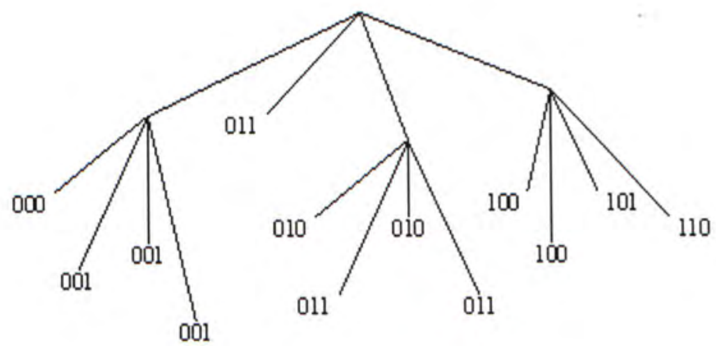


Figure 2.5: Quadtree for the image of figure 2.4.

The corresponding BCQ for the binary map of figure 2.4 is shown in figure 2.6.

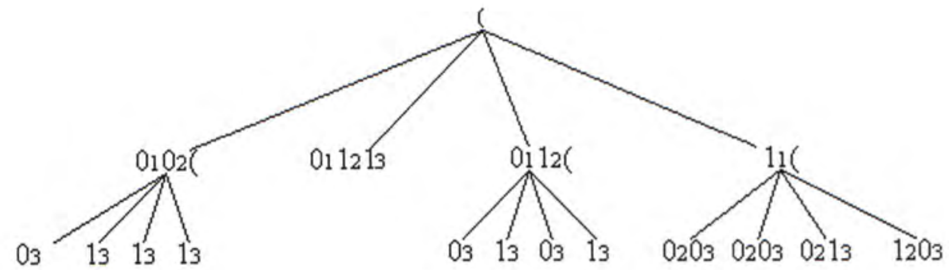


Figure 2.6: The BCQ for the image of figure 2.4.

The symbol "(" indicates there are still more bits needed before perfectly reconstruct the corresponding quad in the image. The components of figure 2.6 are shown in the Table 2.1.

Table 2.1: The components for the binary map of figure 2.4. (rows: spatial level s; columns: gray scale level c)

	1	2	3
0	(	-	-
1	0001	011(	(1(
2	-	0001	0111 0101 0010

The data stored in the BCQ have been sent staring from the top level of the tree. Any data sequence that can perfectly reconstruct the image is also allowed to use. The optimal sequence of the components to be transmitted varies with different images. The following image sequence has shown how the BCQ method refines an image progressively.

Figure 2.7: Color Image Sequence - "Lenna" (size: 256×256, 24 bits per pixel) produced by the BCQ method (bytes received: 505, 1126, 3789, 14746, 41472, 65536)

2.2.2 Progressive Transmission of Full-Search VQ[7]

Progressive Transmission of Full-Search Vector Quantization is based on fitting a progressive transmission tree to a full-search VQ codebook and uses the indexes defined by the tree as the codeword indexes and the intermediate codewords to display the intermediate image.

The progressive transmission tree is built by region-merging. An image is first divided into several Voronoi or encoding regions that each region has been quantized by vector quantization and represented by a codeword. The encoding regions are merged in pairs to form larger ones. Then, an intermediate codeword is assigned to the merged region, addressing the centroid of the encoding region that contains the original Voronoi region into which the input was first mapped. The intermediate codeword is assigned to the tree as the parent of the two merged regions. This merging process is done iteratively from the bottom level of the tree to the top level until there is only one encoding region. Finally a balanced binary tree is constructed as in figure 2.8.

In the balanced tree, each codeword C_i is assigned a new tree index, which is the bit string of the labels of the edges on the unique path from the root to the leaf labeled by C_i. In figure 2.8, the codeword C_6 is assigned the tree index 011, which labels the path from the root to the leaf labeled C_6. For searching the codeword in the leaf node, the bit string is transmitted and the client uses the bits to transverse the balanced tree from top to bottom. The intermediate codeword is then used to display an approximate image at any time during transmission.

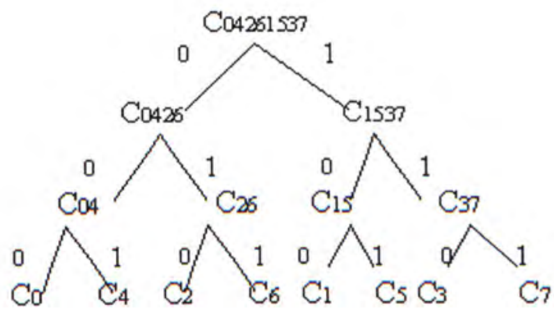


Figure 2.8: Progressive transmission tree

2.2.3 Pros and Cons of Hierarchical Data Structure with Bit Plane Transmission

The main advantage of hierarchical data structure with bit plane transmission is that both spatial and contrast resolutions are processed in the progressive image transmission. The data in the nodes of tree can produce an embedded bit stream. The reception of code bits can be stopped at any point and an intermediate image can be decoded and reconstructed at any time.

2.3 Embedded Transform Coding

Transform coding makes use of image representations in special domains. An ideal domain is one which analyzes an image into components that correspond to the way human visual system (HVS) processes it. Then, components may be ordered with respect to the importance to visual perception, in terms of visibility for instance. Wavelet transform and Discrete Cosine Transform are two popular transforms. Embedded code represents a sequence of binary decisions in which bits are generated in order of importance. Using embedded coding, an encoder or decoder can terminate the encoding or decoding at any point thereby allowing a target rate or target distortion metric to be exactly met. The SPIHT [16] method and the embedded DCT method [13] are two well-established examples used embedded transform coding for progressive image transmission.

2.3.1 SPIHT method [16]

The method of set partitioning in hierarchical trees (SPIHT) is an innovative implementation of Embedded zerotree wavelet (EZW) coding [17]. By the property of the spatial self-similarity between subbands, some redundancies are eliminated and then the rest significant coefficients are transmitted to refine the degraded image gradually. SPIHT is based on the following principles:

Partial ordering by magnitude

Image data is transformed by a unitary hierarchical subband transformation (e.g. Orthogonal pyramid transformation). The coefficients are reordered according to the minimum number of bits required for its magnitude in binary representation.

Set partitioning sorting algorithm

The sorting algorithm divides the set of pixels into partitioning subsets and performs magnitude tests on the set of pixels that compared with a threshold and determine whether they are significant or not. The bits of significant coefficients will be transmitted to client first.

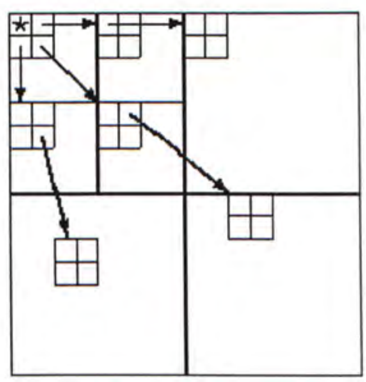
Ordered bit plane transmission

For the binary representation of a list of magnitude-ordered coefficients, bits of all coefficient in the same bit plane are transmitted sequentially starting from the most significant bit and end with the least significant bit.

Self-similarity and the Spatial orientation trees

Normally, most energy of an image is concentrated in the low frequency components and the variance decreases with the levels of the subband pyramid. It is believed that there is spatial self-similarities between subbands. For instance, large low-activity areas are expected to be identified in the highest levels of the pyramid, and replicated in the lower levels at the same spatial locations. Therefore a tree structure - spatial orientation tree is defined. The tree is constructed by the recursive four-subband splitting. Each node of the tree corresponds to a pixel and is identified by the pixel coordinate. Its direct descendants correspond to the pixels of the same spatial orientation in the next finer level of the pyramid. Each node has either no offspring or four offsprings that form a group of 2x2 adjacent pixels. The structure is illustrated in figure 2.9. By this tree, coefficients are reordered and grouped according to their importance that defined by comparing with a threshold. The set of significant coefficient will be sent to the client first that followed by less significant coefficient. The set of coefficient considered insignificant may be treated as redundancy and they will be eliminated for data compression.

Figure 2.9: Examples of parent-offspring dependencies in the spatial orientation tree



By the mentioned mechanism, the SPIHT method successfully applies wavelet transform into progressive image transmission. Together with the compression technique, it improves the efficiency of image transmission. The image coding results in most cases surpass some previous techniques on the same images, such as the method developed by Shapiro [17] and the image subband coding [18]. The following image sequence shows how the SPIHT method refines a color image in both spatial and contrast resolutions progressively.

Figure 2.10: A color image sequence(size: 256×256, 24 bits per pixel)produced by the SPIHT method (bytes received: 505, 1126, 3789, 14746, 41472, 65536).

2.3.2 Embedded DCT Method [13]

The embedded DCT method is based on layered discrete cosine transform (DCT) image compression scheme, which generates an embedded bit stream for DCT coefficients according to their importance. The layered DCT coder groups all bit B_i of coefficients C_j, $1 \leq j \leq 63$, into layer L_i. The coder first encodes the most significant layer L_1, then layer L_2 and so on. Within each layer L_i, the coder first encodes the most significant coefficient C_1, then C_2 and so on. The resulting bit stream has the embedded property and rate-control can be achieved by truncating the bit stream according to the desired coding budget. It is suitable and useful for progressive image transmission.

The method applies successive quantization of AC coefficients other than one-step quantization that is used in JPEG, which maps each DCT coefficient to a value in a finite index set. At each layer i, the DCT coefficients is quantized up to the precision of a significant threshold T_i. Then, the quantization result of layer i is refined with a smaller significant threshold T_{i+1} at layer $i+1$. It results in a more efficient quantization procedure and it finally provides good performance with respect to the JPEG Huffman coder and the JPEG arithmetic coder.

2.3.3 Pros and Cons of Embedded Transform Coding

The main advantage of the embedded transform coding is that it takes into account the characteristic of the human visual system (HVS) to design the algorithm to transmit first data to which the HVS is most sensitive. Therefore, the viewer can get meaningful and visible information earlier and then he/she recognizes the image faster. With the embedded property, rate-control can be easily achieved and the reception of code bits can be stopped at any point. An approximate image can be decompressed and reconstructed at any time during transmission.

However, as the information related to the spatial and the contrast resolution has been coded together to form codes, it is hard to control the increasing rate between the spatial and the contrast resolution during transmission. Hence, it lacks flexibility to allow user to change the data transmission sequence to fulfill different requirements of the image applications.

2.4 Summary

The previous techniques provide good solutions to handle the problem of image communication over a low bandwidth network. They mainly focus on data compression and transmission algorithm to reduce the latency time of image recognition. Some methods like FELICS, SPIHT and BCQ even successfully combine both data compression and progressive transmission algorithms together to solve the problem.

However, all of them have no consideration on the display capability of client's viewport. Viewer may waste time on useless data that can not increase image quality further for display. It is because different kinds of viewports have their own display capabilities. Some of them can display 32 bits true color image such as a desktop PC with a super VGA monitor. Some of them may only display 8 bits color image such as handheld PC in the current market. The same image will have different qualities when they are displayed on different viewports. Obviously, the viewport with lower display capability can only display a lower quality image (fewer colors) even though it received a full resolution image.

To avoid wastage of network bandwidth, image data should be processed according to the characteristic of client's viewport before transmission. In the above example, a 24-bits color image should be quantized into an 8-bits color image before it is transmitted and displayed on the viewport that can only display 256 colors. To ensure all received image data is useful for image display, the characteristic of client's viewport should be considered.

In the next chapters, a new method for progressive refinement of colormapped image that takes advantage of the characteristic of client's viewport will be introduced and analysed. State-of-art techniques such as SPIHT and BCQ will be used as benchmarks for comparison.

Chapter 3

Progressive Refinement of Colormapped Image

In display systems, image data is loaded into frame buffer for display, which is a computer memory organized into an m x n rectangular grid of dots, called pixels. Each pixel requires a certain number of bits, varying from 1 in bit-map displays to 24 or more in high-quality color displays. We call the number of bits per pixel the "depth" of the frame buffer. There are two different frame buffer architectures, namely, direct model and indirect model. In direct model, there are three independent memories for the red, green, and blue components of an image. Typically 8 bits are used per component. A color lookup table (or color table for short) for each component is inserted between the picture memory and display device in order to allow contrast correction.

On the other hand, the indirect frame buffer model stores a single color number at each pixel rather than three separate components. These color numbers are used as addresses into a single color lookup table, which contains a set of colors with three color components (red, green and blue). The color numbers of pixels form a colormap. This colormap provides a level of indirection between the data in the picture memory and the actual displayed image. In general, the indirect frame buffer occupies only about 8 bits per pixel. Also, it should be realized that the order of colors in a colormap is arbitrary. For those shallow, low-resolution graphic terminals (or viewports) such as mobile phone or hand-held PC, the indirect frame buffer model is an efficient way to render an image.

Whether the direct or indirect frame buffer model is used, color quantization is applied to an image if its contrast resolution is higher than what the viewport supports. By color quantization, the colors in an image will be approximated and replaced by a small set of colors for display. In display system, the small set of colors is those colors inside the color table of the display terminal. If the processing power of a viewport is relatively low, the latency time for image display will significantly increase by the extra process and it directly affects the viewing experience. Handheld PC and PDA (private data assistance) in the current market are good examples. The workload of client's viewport is another issue we concerned.

Aiming these concerns about network bandwidth and computational cost in client side, we are proposing a progressive transmission method, which includes consideration on the characteristic of viewport by preprocessing color images before transmission. Color quantization will be first applied into each image with a specified contrast resolution such that all transmitted data will be useful for display in the specified viewport. Moreover, if a color image can be quantized with respect to the colors of the color table of the target viewport, color quantization is not required before image rendering. Hence, both client's workload and therefore latency can be reduced. Besides, as quantized images may only contain a small number of colors, say 256 colors, the colormapped image model becomes an efficient method to represent them.

3.1 Colormapped Image

Many methods reviewed in the previous chapter use direct color model. For a grayscale image, the intensity of each pixel is usually represented by 8-bit values. For a color image, each pixel has three color values in certain color space such as RGB or YC_rC_b. If each color value needs 8 bits, each pixel has 24 bit data to represent its color value. However, this is not the unique model to represent an image. The colormapped image model is another effective and efficient way to represent an image.

Colormapped image model is a method to represent an image, which uses similar colormapping process that adopted in some display systems using indirect frame buffer model to render an image. It consists of an index map and a color table. The index map is the whole set of pixels in which their values is the indices of the colors in the color table. By referring to the color table, the color value of each pixel can be obtained and then the image can be rendered on screen.

3.1.1 Pros and Cons of the Usage of Colormapped Image

The advantage of using the colormapped model is that it can represent an image efficiently that a color image can be simply compressed without distinguishable quality degradation. For a 256x256 color image with 16 colors, if using the direct color model to represent, it needs 24 bits x 256 x 256 = 196,608 bytes. If using the colormapped image model to represent, assuming using 8-bit values for each color plane and 4 bits (16 colors) for the index, it only needs ($256 \times 256 \times 4$ bits) + ($16 \times 3 \times 8$ bits) $= 32,816$ bytes. It is about 17% of the data used by the direct color model. The comparison of the memory size between using the direct color model and the colormapped model in different cases is illustrated in the following table. For the colormapped model, we assume the color table contains 256 colors and each color has 24 bit data.

Table 3.1: Comparison of memory size (in bytes)between the direct color model and the colormapped model.

Spatial resolution	**Direct model**	**Colormapping model**	**ratio**
512x512	768K	257K	3.0
256x256	192K	64.75K	3.0
64x64	12K	4.75K	2.5
32x32	3K	1.75K	1.7

However, the colormapped model is not superior to the direct color model in all cases. If the size of the color table is too large, the memory size of a colormapped image will become larger than that represented in the direct color model. For example, a 256x256 color image with 50K colors, if using colormapped model to represent, it needs $256 \times 256 + 50K \times 3$ bytes = 214K bytes. If using the direct color model with 24-bit data for each pixel to represent the image, it only needs $256 \times 256 \times 3$ bytes = 192K bytes. Therefore, the advantage of using colormapped model depends on the color table size and the image size.

Generally speaking, the usage of the colormapped model can probably reduce the data traffic for image transmission, especially when the image contains a small number of colors. If the colors of colormapped image are limited inside the color table used in a viewport before transmission, it can reduce the workload of the client for color quantization as well. It really ensures all transmitted data are useful for display.

The usage of colormapped image model has already achieved the effect of image compression. The last issue needed for handling the problem of image communication is developing an efficient image transmission algorithm. As shown before, progressive scan provides better viewing experience than the display method in scanline order. Therefore, progressive image transmission will be used as the transmission model that works in conjunction with the colormapped image model.

3.2 Progressive Refinement in both Spatial and Contrast Resolutions

Before starting to apply a colormapped image into progressive image transmission, another issue is worthy of consideration. It is the transmission algorithm that supports progressive refinement in both spatial and contrast resolutions such as the BCQ method. It is believed that different image applications need different progression rate allocation between these two resolutions during progressive image transmission for an optimal viewing experience. For example, an image dominated by the textual information needs the spatial resolution to increase faster because this arrangement can let the viewer recognize the textual information faster.

In the following figure, both images are reconstructed after received 3789 bytes of data from the server side. The left image has a higher spatial resolution and a lower contrast resolution than the right one. Obviously, the left image can show a rough textual information but the right image cannot. We can conclude that the spatial resolution is more important than the contrast resolution in this image application. Moreover, the relative growth rate between the spatial resolution and the contrast resolution will influence the performance of progressive transmission significantly. Therefore, it is necessary that the transmission algorithm should allow controlling the degree of bias towards the spatial resolution and the contrast resolution for the requirement of different kinds of images.

Figure 3.1: Images with different spatial resolutions and contrast resolutions

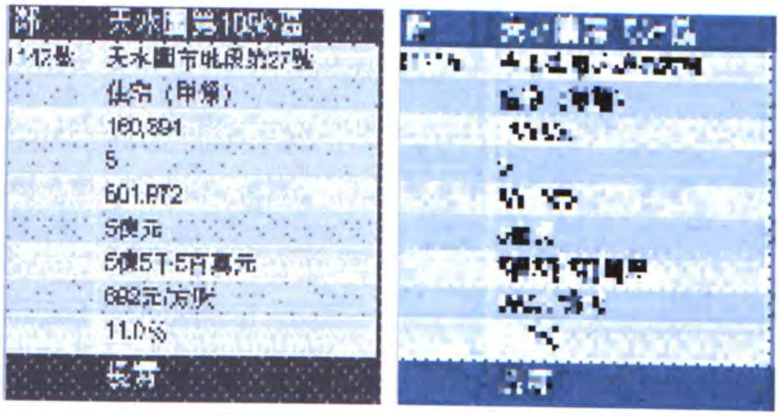


Figure 3.2: The original image with textual information

For progressive refinement in the contrast resolution, an approximate color of each pixel should be obtained when data is received. Then, there should be an improvement in the contrast resolution when extra data has been received. In the colormapped model, the index map only works as a reference of color table to obtain corresponding color of each pixel. It lacks the ability of showing approximate color for each pixel during progressive transmission of the indices. To make this possible, a new data structure would be required so that an approximate color index may represent an approximate color with respect to the given color table. It means that a new finer approximate color for a pixel should be obtained when an extra bit of the color index has received. Each bit of a color index can be treated as a refinement level. In each level, there is an approximate color used for image display when the corresponding bit of data received. To find out those approximate colors in each refinement level, the process of color quantization should be divided into several levels and then the color set of image will be quantized into several smaller color clusters. The approximate color of each cluster in each quantization level (or refinement level) will be calculated and stored for image display during transmission. We shall show that the Sequential Scalar Quantization method (SSQ) [14] and a binary tree like data structure can be used to achieve the above purposes.

For progressive refinement in the spatial resolution, pixels will be reordered in a pattern such that any subset of pixels is distributed uniformly over the 2-d space. Hence, there is no bias in any region inside the image during image display. Moreover, the image should be able to display at any time. To do so, the pixels without receiving any data in a specified region will be replaced by those pixels that have received data for display; each specified region should only have a pixel received data at any time. The details of the algorithm will be presented in the next chapter.

Chapter 4

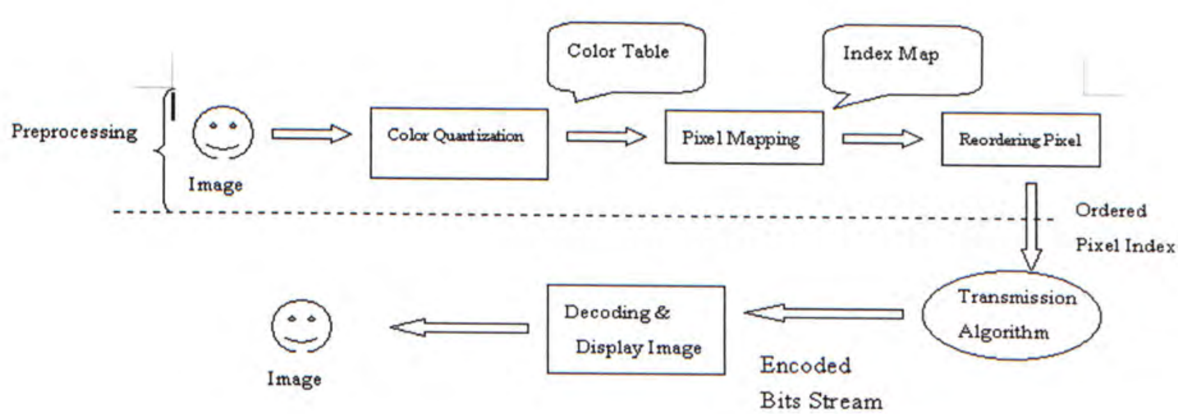
The Design of Progressive Refinement of Colormapped Image

For progressive refinement of colormapped image, there are several different processes included. They are color quantization, pixel reordering for uniform display over 2-d space, the transmission algorithm and the display algorithm.

Firstly, a full resolution image represented in the direct color model is quantized to a finite numbers of colors and a color table is constructed to store the approximate colors. Then, each pixel is assigned an index that is the index of the nearest color in the color table. Secondly, for ensuring the image can be refined uniformly in the 2-D space during transmission, the pixels are reordered such that any partially received sequence of pixels is distributed uniformly over the 2-D image space.

At the beginning of the transmission, a binary-tree like color table is transmitted to the client first. Currently, the value of pixel is an index of the color table instead of the direct color value. For progressive refinement in the contrast resolution, we expect each pixel can be assigned a new approximate color for display after receiving extra bit of its index. Therefore, the index of color for a pixel will be the binary path for traversing the binary color table from the root to the color. Moreover, each node of the binary color table stores an approximate color of the pixel for display at that point. After transmitted the color table, the bits of the index of each ordered pixel are transmitted to the client side. When the client received the data stream from the server, the data will be decoded and assigned to the corresponding pixel for display at any time. The whole model of this process is shown in the following block diagram and the details of each process will be covered in the following sections.

Figure 4.1: Block diagram of the process for progressive refinement of colormapped image.



4.1 The Scalar Quantization in the YC_rC_b color space

Firstly, a true-color image needs to be transformed to the colormapped image. To do so, color quantization is applied. The color space for the quantization, the quantization process and the order of quantization among different color planes should be concerned carefully.

4.1.1 The Color Space for Color Quantization

It is well known that the human visual system (HVS) perceives a color stimulus in terms of the luminance and the chrominance attributes, rather than in terms of R, G, and B color values. Hence, we have to transform the image to a luminance-chrominance space prior to performing the color quantization. Therefore, we pick YC_rC_b color space as the basis. By a simple linear transform, an image can be easily transformed from the RGB color space to the YC_rC_b space. The transform is performed by the following equations:

$$Y = 0.299R + 0.587G + 0.114B \tag{4.1}$$

$$C_r = 0.713(R - Y) + 128 \tag{4.2}$$

$$C_b = 0.564(B - Y) + 128 \tag{4.3}$$

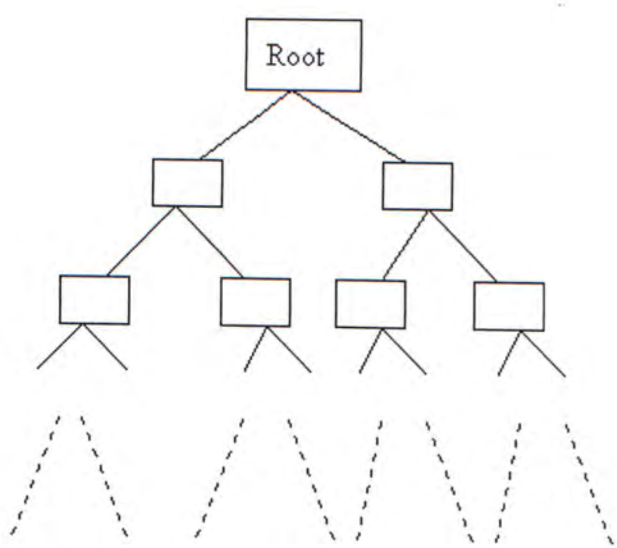
4.1.2 Color Quantization

During color quantization, it may not be necessary to include all pixels in the image for quantization. It is because pixel neighborhoods may have similar color values. So, we can downsample the image by some factors to get a set of sample colors for color quantization.

There are two goals for the process of color quantization. The first one is to transform the true-color image from the direct color model to the colormapped model. The second is to construct a data structure that allows progressive contrast resolution refinement of colormapped image. It is also called progressive fidelity transmission. During the progressive fidelity transmission, the color value of each pixel should be closer and closer to the original one; An approximate color should be assigned to each pixel for display at any time. To achieve the above effect, the quantization process should be divided into several levels and an approximate color of each color cluster (a set of colors) should be obtained during each quantization level.

In the implementation, each color cluster will be quantized into two smaller clusters. Then the approximate color in each cluster will be calculated and stored. The quantization process performs in each cluster until the desired number of the quantization regions is reached. By this process, we can construct a binary-tree like data structure, in which each node stores the approximate color value of each cluster and each of them has splitted into two child nodes after the next quantization process, except the nodes at the bottom of the tree which represent colors assigned to the pixels for final display. The rest is used for image display during transmission. The index values of the nodes in the bottom level are the binary path from the root and they will be assigned the corresponding pixels to form the index map. The approximate color of each pixel in each level can be obtained by using the received bits of index to traverse down the tree during transmission. Hence, the progressive contrast resolution refinement is achieved. The binary-tree data structure of the color table is shown in following figure.

Figure 4.2: Binary-tree data structure



By the primary design of the color quantization process, the method of the sequential scalar quantization (SSQ)[14] is adopted because it can divide the whole color quantization process into several levels. Its algorithm allows to quantize each color cluster sequentially and then provides an approximate color of each color cluster in each quantization level. We can build up a hierarchical structure, such as binary tree, to link up the color clusters and store the approximate color of each color cluster in each level as the above figure for progressive fidelity transmission. Moreover, SSQ provides good performance in image quality with a small computational cost. In SSQ , only one color component X_i, i=1,2,..., where $X_i = Y$, C_r or C_b in the YC_rC_b color space is considered during each time of quantization. By this arrangement, the colors can be easily grouped by the particular color property of the luminance or the chrominance component.

From an image, we take a set of the sample colors for the color quantization. The sample colors can be obtained by down-sampling the image by some factors such as two. Then, we will sort all sample colors along X_i dimension and use the mean of the X_i component among the sample colors as the boundary to split the cluster into two. Then we calculate the mean value along Y, C_r and C_b dimension in each cluster and use it as the approximate color of the new cluster. Then, the quantization process applies to each cluster continuously until the specified number of quantization regions is reached.

4.1.3 The Order of Quantization

During the process of the sequential scalar quantization, only one color component of the colors is considered. Therefore, the order of the color component for the color quantization can be determined arbitrarily. There should be a lot of combinations of the quantization order and they result in different image quality and transmission performance. It is necessary to limit the number of combinations and find out the good ones for color quantization.

It is obvious that the error will be very large when using an approximate color to represent a large color cluster. So, the approximate color in the first few quantization levels becomes less meaning. Nevertheless, the human visual system (HVS) is more sensitive to the luminance component. It can provide more information to the observer to recognize the image content. The luminance information is especially important when there are only few colors used to represent a true-color image. Therefore, it is reasonable that we quantize a color image along Y dimension first so that the contrast of the luminance between colors can be larger and so the observer can recognize the image content easier and faster at the beginning of the transmission. Moreover, for providing better visual performance, the effect of the chrominance component is eliminated in the first few quantization levels by setting the value of the C_r and C_b component of the approximate color to 128. Hence, the approximate colors become monochromatic and it lets the image clearer. In the following figure, we have shown the images quantized first along the Y, C_r and C_b dimension respectively.

Figure 4.3: Images quantized along different color dimension in YC_rC_b space; Left image: quantized along Y, Middle image: quantized along C_r, Right image: quantized along C_b)

From the above figure, the image quantized along Y dimension first provides better performance than others do. To allow the user to have an earlier recognition of the image content, it is a good attempt to use the monochromic image for display at the beginning of the transmission. Therefore, the image will be quantized along Y dimension first.

After that, the Y, C_r or C_b components of the image will be quantized sequentially until the desired number of the quantization regions is reached. The next step is how to order the color components for the color quantization in the following quantization levels. In the implementation, the number of the quantization regions is 256. The image is quantized for Y component three times first so as to construct a smaller cluster before the quantization is performed along the chrominance dimensions. Then, as there is no obvious reason for any bias between the C_r and the C_b components, the number of times for the quantization along these two dimensions should be equal. An experiment is performed to find out a general quantization order for color quantization. In the experiment, color images are quantizated in different quantization order. By the experimental result, the quantization order: $Y-Y-Y-C_r-C_b-Y-C_r-C_b$ and $Y-Y-Y-C_b-C_r-Y-C_b-C_r$ provide good performance during progressive transmission. The following figure illustrates the results of the images quantized by different quantization orders.

Figure 4.4: Image quantized by different quantization orders

Quantization Order:

(left image :Y-Y-Y-C_r-C_b-Y-C_r-C_b) (right image:Y-Y-C_r-C_b-Y-Y-C_r-C_b)

(left image :C_r-C_b-Y-C_r-C_b-Y-C_r-C_b) (right image:C_r-C_b-C_r-C_b-C_r-C_b-Y-Y)

Figure 4.5: Fine art image quantized by different quantization orders

Quantization Order:

(left image :Y-Y-Y-C_r-C_b-Y-C_r-C_b) (right image:Y-Y-C_r-C_b-Y-Y-C_r-C_b)

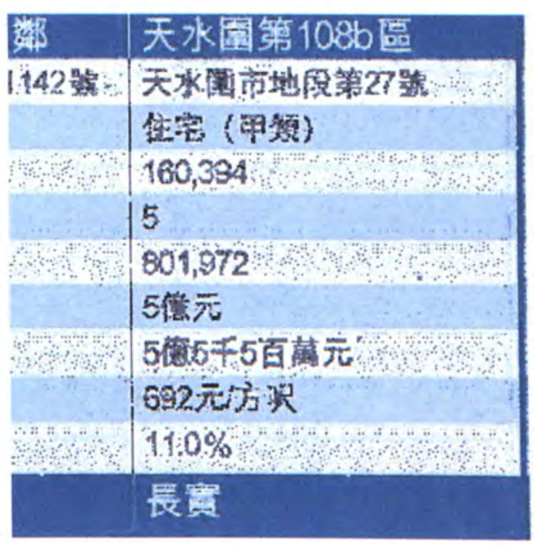

(left image :C_r-C_b-Y-C_r-C_b-Y-C_r-C_b) (right image:C_r-C_b-C_r-C_b-C_r-C_b-Y-Y)


Figure 4.6: Textual image quantized by different quantization orders

Quantization Order:

(left image :Y-Y-Y-C_r-C_b-Y-C_r-C_b) (right image:Y-Y-C_r-C_b-Y-Y-C_r-C_b)

(left image :C_r-C_b-Y-C_r-C_b-Y-C_r-C_b) (right image:C_r-C_b-C_r-C_b-C_r-C_b-Y-Y)



From the above images, we can see that different quantization orders and different number of times to quantize along Y dimension will affect the image quality significantly. Different images may need different quantization orders to provide better visual performance. For example, fine art images like the one in figure 4.5 may need to quantize along C_r and C_b more to provide a more details in the chrominance information. On the other hand, textual images like the one in figure 4.6 may need to quantize along Y dimension more to provide a clearer and sharper image for better content recognition.

4.1.4 Pixel Mapping

After a color table is constructed from the color quantization, the next step is to map each pixel to the approximate color in the color table. The index of the mapped color is assigned to the pixel to form an index map. In the binary-tree like color table, the quantized colors used for mapping are only those colors at the bottom of the binary tree. The ancestor colors in the binary tree are used for progressive fidelity transmission. Moreover, the assigned index of each pixel is actually the path for traversing down the binary tree to find out the approximate color for the pixel.

To simplify the mapping process, a look up table is prepared while the color quantization that indicates the sample color should belong to which quantization region. Each quantization region has its own look up table to indicate which new quantization region a color value should belong to. Hence, an input color can be mapped to the final corresponding quantization region (or the approximate color) without any computation by referencing the look up tables continuously.

For example, an input color $c1$ has the color value, $[y,c_r,c_b]$, in the YC_rC_b color space. With a particular order of quantization, the corresponding color component value of c_1 is addressed in the look up table and the new index of the quantization region included c_1 is found. Then, the look up table marked with the same index is used in the next stage to find out the next quantization region contains $c1$. The process progresses continuously and then the approximate color of c_1 will be eventually found. This process is shown in Figure 4.7. In the figure,

$X_i = y, c_r$ or c_b, where $i = 1, 2, ...k$. The value of k depends on the desired number of quantization regions; the value of X_i depends on the order of quantization.

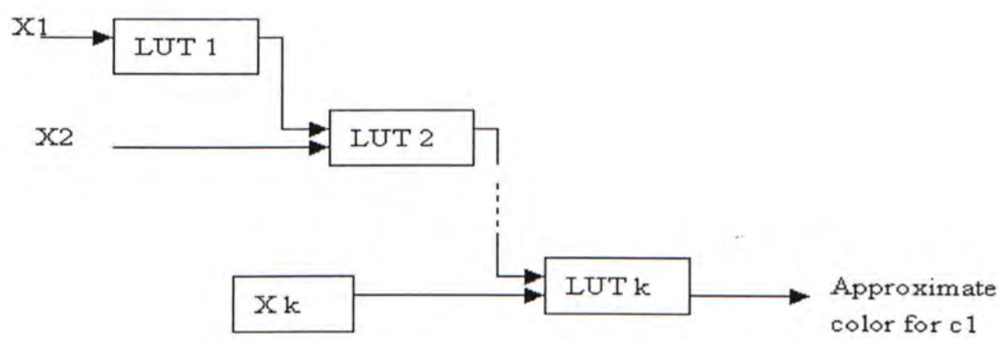


Figure 4.7: Pixel mapping by the Look Up Tables(LUT)

In the algebraic form, for the color c1=[y,c_r,c_b] and the look up table array LUT$_i$,where i=1,2,...,8 for 256 colors, the index of the approximate color mapped for $c1$ can be obtained as follows:

Assuming the quantization order is Y-Y-Y-C_r-C_b-Y-C_r-C_b. To prepare an array stored the color value c_1 with the mentioned order of quantization likes this: ary[1]=y, ary[2]=y, ary[3]=y, ary[4]=c_r, etc. The mapping process is as follows.

1. Initialize i = 1, region = LUTi[ary[i]], set i = 2;
2. Starting, region = LUTi[region][ary[i]];
3. i $\leftarrow$ i + 1;

4. If i $<$ 8, go to step2, otherwise stop;

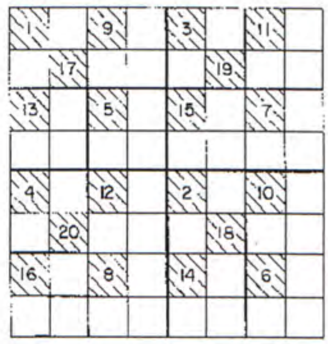
Finally, the value of 'region' equals to the index of the approximate color for c_1.

4.2 Reordering Pixels

After the preprocessing of the progressive fidelity transmission, next comes progressive transmission of the spatial resolution. During transmission, the bits of the pixels' indices are transmitted sequentially. On the client side, those bits are decoded and used to construct an approximate image for display. With no previous acknowledgment of the location of the image content inside the image and no reason for biasing on any particular region inside an image, the image should be refined evenly during transmission.

The goal is that any partially received consecutive reordered pixels should be evenly distributed over the image, with broader detail being represented first, followed by increasingly fine detail. To do so, we use a method introduced by Lloyd-Williams and Andrew [15] for reordering the pixel map of digitized image. This method reorders the pixel map of a multidimensional image by representing each pixel address as a distinct set of coordinates in binary representation. By this method, the re-ordered pixels are uniformly distributed in 2-d space as shown in Figure 4.8.

Figure 4.8: The distribution of the reordered pixels.



To obtain the new sequence order of pixels, bitwise exclusive or (XOR) is applied to the x and y coordinate of the pixel and we get a new number z in the binary form. Then the bits of y and z are reversed, and we interleave the binary digits of z and y alternatively. The composite binary number formed is used as the new pixel sequence number.

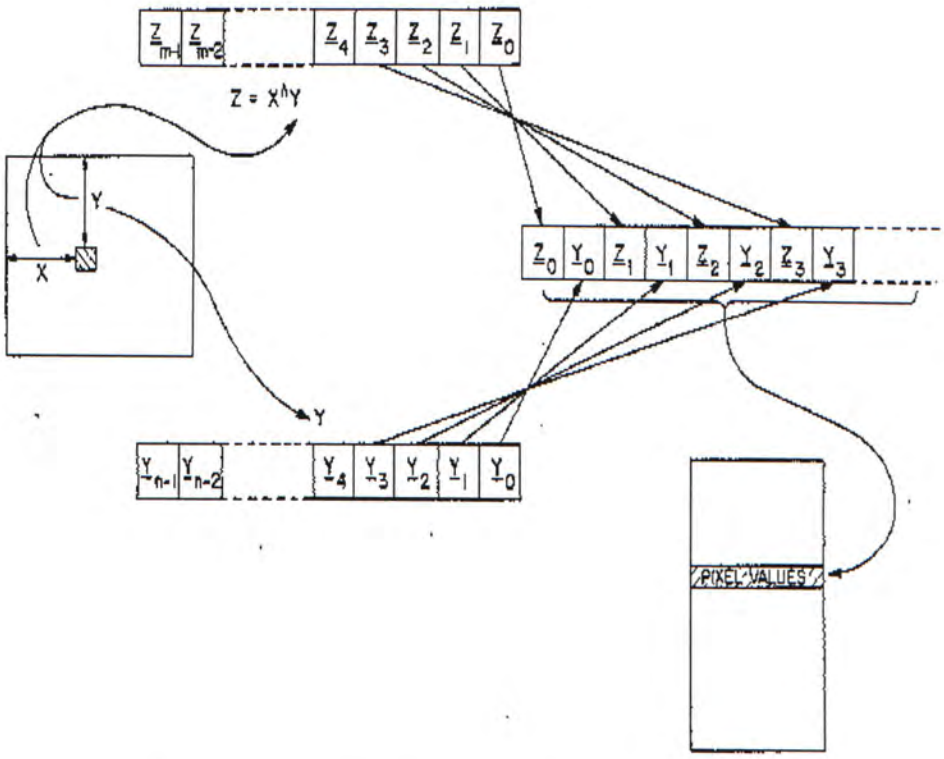
For example, x-coordinate X = $X_2X_1X_0$ (binary form) and y-coordinate Y = $Y_2Y_1Y_0$.

Z = $Z_2Z_1Z_0$ = X XOR Y.

Then, the new sequence number = $Z_0Y_0Z_1Y_1Z_2Y_2$

The method mentioned is illustrated in Figure 4.9.

Figure 4.9: The method for calculating the new sequence number of the pixel.



For example, there is a pixel with coordinate $X = 4$ and $Y = 0$ in a 8×8 pixel map. In binary form, $X = 100$ and $Y = 000$.

Then Z = X XOR Y = 100. Then the new sequence number = 000010 = 2. Note: The sequence number starts from zero and ends with 63 in this example.

4.3 Transmission Sequence

Now, the pixels in an image are reordered with their new sequence numbers and the indices of the pixels are saved in a file. The binary form of the indices stored in the file is shown in Figure 4.10. Together with the binary-tree like color table, the image transmission may now start.

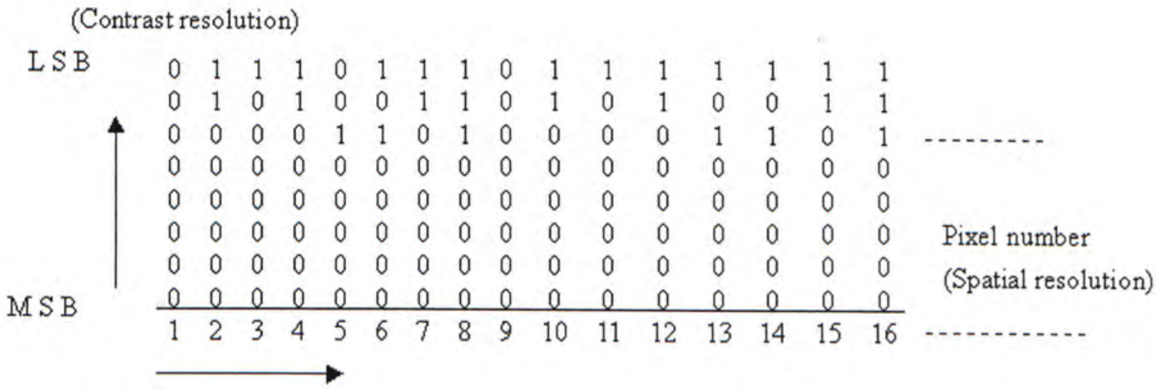


Figure 4.10: The binary representation of the reordered pixel's indices

The data transmission starts from the pixel with the smallest sequence number 1 and ends at the pixel with the largest sequence number $2^n \times 2^n$ for an image with size 2^n by 2^n. Moreover, for every pixel, the most significant bit (msb) must be transmitted first. The less significant bits will be transmitted later and the least significant bit (lsb) is transmitted last. When the most significant bit of a pixel is transmitted, the spatial resolution of the image increases. When the less significant bits of a pixel are transmitted, the contrast resolution of the image increases. Hence, by changing the data transmission sequence, both spatial and contrast resolutions of the image can increase progressively.

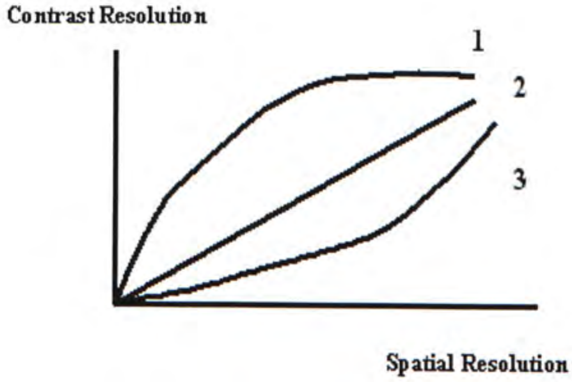
Besides, for ensuring both spatial and contrast resolutions can increase uniformly in all regions inside the image, the number of bits of a pixel that has already been transmitted should not be greater than the number of bits of those previous pixels had been transmitted. For example, if the pixel with the sequence number '6' has been transmitted 3 bits, then the pixel with the sequence number '7' must not have more than 3 bits to be transmitted. Or else, if the pixel with the sequence number '6' has not been transmitted any bit, the bits of the seventh pixel must not been transmitted. Otherwise, the image will be refined unevenly because some regions inside the image got much more bits than others. Apart from the above rule, there is not any other limitation on the transmission sequence among the data. So, we can change the data transmission sequence to control the increasing rate of the spatial resolution and the contrast resolution during the transmission. By this flexibility, we can adjust the progression rate of these two resolutions. In the following paragraph, the details of the transmission algorithm is presented.

4.4 Changing the progression rate of the spatial and the contrast resolution

The new method has provided greater flexibility to change the transmission sequence to control the progression rate of the spatial and the contrast resolution. In the algorithm, a function is used to accept a variable to determine the degree of bias between the spatial and the contrast resolutions during transmission.

The degree of bias between these two resolutions during transmission can be presented graphically in the following figure.

Figure 4.11: Degree of bias between the spatial and the contrast resolution



In fact, the degree of bias forms a locus in the graph. The locus can be represented by a function $f(x)$. User can arbitrarily use different functions to determine the bias behavior between two resolutions, but those functions should be monotonic increase and are as smooth as possible for better viewing experience. In the function, the number of pixels received bits is the function input. We call those pixels have received one or more bits to be 'filled pixel'. The average number of bits per 'filled pixel' is the function output. Then, the algorithm will use this information to determine which bit of the pixels should be transmitted to achieve the desired spatial and the contrast resolution.

In the implementation, exponential function is used to represent the resolutions bias behavior because it can provide smooth and monotonic increase continuous curve as shown in figure 4.11. Moreover, it provides saturation region that fits for represent the upper limits of the resolutions. In the function, the number of pixels received bits, x, is the function input. The average number of bits per 'filled pixel', y, is the function output. So, for the contrast resolution increase faster likes the curve '1' in figure 4.11, $y = n(1 - e^{-bx/z})$, where n is the total number of bits per pixel in the colormapped image, z is the total number of pixel in the image and b is a positive integer constant called 'bias factor', which used to control the degree of bias between the resolutions. When b increase, the degree of bias to the contrast resolution increase.

For both resolutions increase equally likes the curve '2' in figure 4.11, we use $b = 0$ to indicate and the function is $y/n = x/z$.

For the spatial resolution increase faster likes the curve '3' in figure 4.11, we use $b =$ negative integer to indicate and the function is $y = n(1 - e^{x/bz})$. When the absolute value of b increase, the degree of bias to the spatial resolution increase.

In each time, user needs to enter the bias factor b before transmission. Positive integer means the transmission biasing to the contrast resolution. Zero means no bias between the resolutions. Negative integer means the transmission biasing to the spatial resolution. Then the algorithm uses the corresponding function to calculate the average number of bits per 'filled pixel' to determine the data transmission sequence. Afterwards, the number of filled pixel x will be increased by one and then the new average number of bits per 'filled pixel', y, will be obtained. Then, the algorithm uses the new y to arrange the corresponding data to transmit to the client.

For ensuring image can be refined uniformly in both resolutions, there are some rules to order bits among those 'filled pixels' to be transmitted. As mentioned before, pixels are reordered in a particular sequence and x can be treated as the sequence number of pixels. With a new value y, the algorithm arranges bits among those x pixels by the following rules:

1. For the pixel x, the most significant k bits will be transmitted first, where k is the minimum number of bits had been transmitted among the previous $x - 1$ 'filled pixels';
2. Then, bits in the $k + 1$ bit plane among the x pixels follows. The order always starts from the pixel with the smallest sequence number and ends at the pixel with the sequence number x;
3. If the current average number of bits per 'filled pixel' is still smaller than y, bits in the next bit plane will also be transmitted at that time.
4. This process continues until the current average number of bits per 'filled pixel' reaches y.

Hence, a bit sequence can be treated as two sets of bits. The first set is the k bits of the pixel x. The second one is those bits among the x pixels in which bits are located in the $k + 1$ or higher bit plane. The first bit in the second bit set comes from the pixel with the smallest sequence number in where its $k + 1$ bit has not been transmitted yet.

The details of the algorithm are as follows:

Let k be the number of bits of the pixel x to be transmitted, x' be the first pixel in which its $k+1^{th}$ msb will be transmitted, and y' be the current average number of bits per 'filled pixel'.

<u>Encoding Algorithm</u>

1. Obtain bias factor b from user input and use the corresponding function $f(x)$ for encoding; $y = n(1-e^{-bx/z})$ when b is a positive integer; $y/n = x/z$ when $b = 0$; $y = n(1 - e^{x/bz})$ when b is a negative integer; (Note: n is maximum number of bits per pixel, z is total number of pixels in an image.)

2. Initialization: $x = 1, x' = 1, y' = 1$ and $k = 1$;

3. Obtain y where $y = f(x)$; If $x = z$, set $y = n$. For pixel "x", output first k msb; Calculate y'. If $y' \geq y$, go to step 6, else go to step 4.

4. Calculate y'; if $y' < y$, then output the $k+1^{th}$ msb of the pixel x'; Calculate the current y',

   Case 1: if $y' < y$ and $x' < x$, $x' \leftarrow x' + 1$ and go to step 4 again;

   Case 2: if $y' < y$ and $x' = x$, go to step 5;

   Case 3: if $y' \geq y$ and $x' < x$, $x' \leftarrow x' + 1$ go to step 6;

   Case 4: if $y' \geq y$ and $x' = x$, go to step 7;

5. Set $k = k + 1$, $x' = 1$ and go to Step 4 again;

6. if $x \leq z$, $x \leftarrow x + 1$; Go to step 8;

7. if $x \leq z$ and $k < n$, set $x' = 1$, $k \leftarrow k + 1$ and $x \leftarrow x + 1$; Go to step 8;

8. Go to Step 3 if $x \leq z$ and $y' \leq n$. Otherwise, Stop.

The following example gives a clearer explanation of the encoding algorithm: Assuming there is a locus function $y = f(x)$ has the following properties:
$1 = f(1)$: 1 bit per pixel when only pixel '1' receiving bit;
$2.5 = f(2)$: 2.5 bits per pixel when there are two pixels receiving bit;

Now, we refer to Figure 4.10: Binary representation of reordered pixel index and the above encoding algorithm.

Steps (First cycle):

1. We have the function characteristic for x = 1 and 2;
2. Let $y = f(x)$;
3. Initialize: $x = 1$, $x' = 1$, $y = f(1) = 1$ and $k = 1$;
4. For pixel '1', we output the first msb as k = 1(i.e the msb of the pixel'1' in figure 4.10). **Output: 0**.
5. As current bpp y' = 1 = y, go to step 6 in the encoding algorithm;
6. $x \leftarrow x + 1 (= 2)$, go to step 8 in the encoding algorithm;
7. Then, as $x \leq z$ go to step 3 of the encoding algorithm.

Steps (Second cycle):

1. Currently, $k = 1$ and $x = 2$;
2. $y = f(2) = 2.5$, For pixel '2', we output the first k (=1) bit (i.e. the msb of pixel '2' in figure 4.10). **Output: 0**;

3. As current bpp = 1 < 2.5, so we have to output the bits in the 2nd msb-plane for pixel '1' and '2'; **Output: 0,0;**

4. As current bpp = 2 < 2.5 and x' = 2 = x, go to step 5 in the encoding algorithm. set $k \leftarrow k+1(=2)$, set $x' = 1$ and go to step 4 of the encoding algorithm again;

5. Now, Output the 3rd msb bit for pixel '1'(**Output: 0**), bpp now becomes 2.5; set $x' \leftarrow x' + 1 = 2$ and go to step 6 of the encoding algorithm.

6. set $x \leftarrow x + 1$ and go to step 8 of the encoding algorithm;

7. As $x < z$ and $y' < n$, go to step 3 of the encoding algorithm again.

4.5 Data Transmission

On the client side, the decoding algorithm is almost the same as the encoding algorithm to assign received bits to their corresponding pixels. The difference is that the client needs to find out the coordinates of the pixels in the image since the pixels have been reordered as mentioned in section 4.2. The details of the procedures to find out the coordinates of the reordered pixels is as follows:

1. For a sequence number s, we can reconstruct the y coordinate and a number z by getting bits alternatively from the sequence number expressed in the binary form;

   For getting z, obtain bits alternatively starting from the most significant bit of s and they are located starting from the least significant bit of z;

   For y, obtain bits alternatively starting from the second most significant bit of s and they are located starting from the least significant bit of y;

2. For x coordinate, we perform bitwise exclusive or, XOR, between the z and y.

For example, there is a sequence number S = $S_5S_4S_3S_2S_1S_0$. $Z = S_1S_3S_5$. $Y = S_0S_2S_4$. Then X = Z XOR Y.

After the received bits were assigned to their corresponding pixels, the approximate color of each pixel can be obtained by using the received bits of the pixel to traverse the binary tree sequentially starting from the root. Then an approximate image can be displayed during transmission.

4.6 Displaying the image

During the transmission, an approximate image should be able to display at any time. To do so, those pixels that have not ever received bits, should be assigned a temporary color value. By the method of the pixel reordering mentioned in the section 4.2, the pixels are ordered in the pattern shown in figure 4.8. As the pixels are ordered uniformly in the 2-d space and the top left corner of the pixel in a quad region of the image must received bits first, those pixels that have not received any bit in this region may be assigned the color value of the top left corner pixel for display. The size of the region dominated with the top left corner pixel is calculated with the number of pixels that already received one or more bits. The concept is that the image is initially divided into 4 quadra. When each quadra has pixel received bits, each of them is individually divided into 4 smaller quadra. When all quadra have pixel received bits, they are divided again. Therefore, we can use the number of pixel received bits to determine the region size that dominant by the pixel. For an image $2^r \times 2^r$ in size, the region size can be calculated as follows:

Assuming there are p pixels with one or more bits, the smallest integer k can be found for which $2^{2k} \geq p$.

Let $r_x(k) = \frac{2^r}{2^k}$, $r_y(k) = \frac{2^r}{2^k}$

For those pixels with the sequence number $< 2^{2(k-1)}$, where $k \geq 1$ and the sequence number ≥ 0, the region size equals $r_x(k-1) \times r_y(k-1)$.
Therefore, the pixel with the sequence number $< 2^{2(k-1)}$ can dominant the whole region with size $\frac{2^r}{2^{k-1}} \times \frac{2^r}{2^{k-1}}$ for display

Besides, for those pixels with the sequence number $\geq 2^{2(k-1)}$, where $k \geq 1$ and the sequence number ≥ 0, the region size $= r_x(k) \times r_y(k)$. Therefore, the pixel with the sequence number $\geq 2^{2(k-1)}$ can dominant the whole region with size $\frac{2^r}{2^k} \times \frac{2^r}{2^k}$ for display.

The details of the new algorithm for progressive refinement of colormapped image have already been explained. In the following section, the performance of the new algorithm will be evaluated and analyzed to show how it benefits for the image transmission.

Chapter 5

Results Analysis & Performance Evaluation

5.1 Traffic overhead

For a 256×256 color image, which is quantized to 256 colors, its size equals to the sum of the size of the index map and the size of the color table. That is, 256×256 bytes + 256×3 bytes = 66304 bytes. In the implementation, the overhead includes the extra approximate colors in the upper level of the binary-tree like color table and 4 bytes used to indicate the image size. For 256 colors, the total colors in the binary-tree like color table is 510 ($2^1 + 2^2 + ... + 2^8$). Hence, the overhead equals to 254×3 bytes + 4 bytes = 766 bytes. It is only about 1.16% of the original file size of the colormapped image.

Although the coding algorithm has not included any data compression algorithm, there is still reduction on the data traffic by the characteristic of the colormapped image. For an image with size 256 by 256 and with 24 bits per pixel, the file size equals to 192K bytes that is about 3 times of the file size of the colormapped image.

5.2 Performance Evaluation

5.2.1 Experiment

To evaluate the performance of the progressive transmission algorithm, the viewer's judgment is important because he is the end-user of the communication system. A computer simulation is useful to demonstrate the algorithm in real time and it allows the subjective test on the algorithm.

The subjective test aims to evaluate three things. The first is a threshold test. It determines the threshold of the spatial and the contrast resolution for meaningful recognition. Moreover, it can find out the relative importance of the spatial resolution and the contrast resolution for the image. The second is the comparative test. The images are processed by the new approach and other proposed approaches to compare with each other. The third is the investigation of factors causing quality variation.

The preprocessing steps including the color quantization, the index mapping and the pixel reordering were done by the MatLab programs. Then, both server and client programs were written in Java to simulate the progressive transmission of colormapped image. The Java programs can simulate the progressive transmission of an $n \times n$ colormapped image with 256 or fewer colors.

In the client program, the user needs to enter a 'bias factor' b before transmission. The bias factor controls the degree of bias between the spatial resolution and the contrast resolution in the whole transmission process. With it, an image may be transmitted in different transmission sequences and then providing different viewing experience. Hence, we can find out what kind of transmission sequence is suitable for what kinds of images.

The 'bias factor' b only accepts integer. Positive integer indicates the transmission bias towards the contrast resolution. Zero value indicates the contrast resolution and the spatial resolution grows directly proportional. Negative integer indicates the transmission bias towards the spatial resolution.

Color images used for the experiment are quantized to 256 colors with 8 bits per pixel. The first result is shown in Figure 5.1. The image - "Lenna" has a spatial resolution of r=8 ($2^8 \times 2^8$ pixels) and a contrast resolution of 8 bits per pixel (256 colors).

Figure 5.1: Increasing the spatial resolution faster with bias factor $b = -20$(bytes received: 505, 1126, 3789, 14746, 41472, 65536)

Similarly, the progressive image transmission can increase the contrast resolution faster than the spatial resolution by transmitting more bits for a pixel before transmitting bits to other pixels. The result is shown in Figure 5.2.

Figure 5.2: Increasing the contrast resolution faster with bias factor $b = 20$(bytes received: 505, 1126, 3789, 14746, 41472, 65536)

Certainly, the image also can be refined fairly between the spatial and the contrast resolution as shown in Figure 5.3.

Figure 5.3: Increasing both the spatial and the contrast resolution equally with bias factor $b = 0$(bytes received: 505, 1126, 3789, 14746, 41472, 65536)

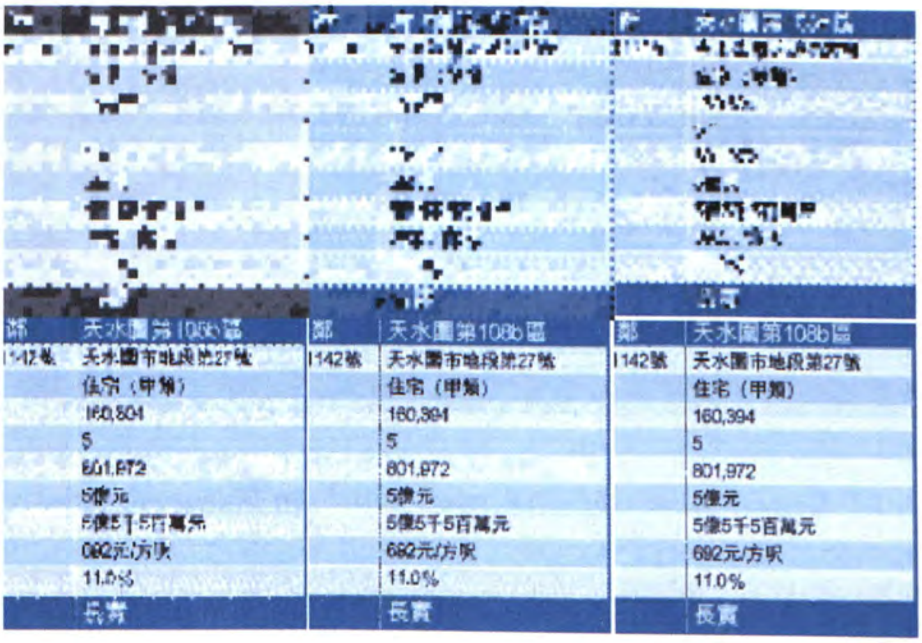
For the image "Lenna", the image sequence in Figure 5.2 that the contrast resolution increasing faster gives a better viewing experience on the subjective test. The client needs to receive about 3.7K bytes of data before recognizing the image's subject and the surrounding environment, which consumes about 0.5 second by using the 56K bps modem network connection. It is only about 6% of the whole. As the received data is distributed uniformly over the 2-d image space, it lets user to have a better acknowledgment of the whole image instead of a part of the image. It is very important when the viewer do not know where the concerned information is located in the image.

In fact, different images with different kinds of contents or with different application purposes may need different transmission sequences to provide better viewing. The new method allows the transmission sequence to be controlled. For shape-oriented queries, the spatial resolution is increasing with a higher priority than the contrast resolution. On the other hand, if color information is more important than the object's shape, the contrast resolution should increase faster. For example, some images such as natural images or fine art images, the color information inside the image is more important than the object's shape. So, such kind of images may need the contrast resolution increase faster during transmission.

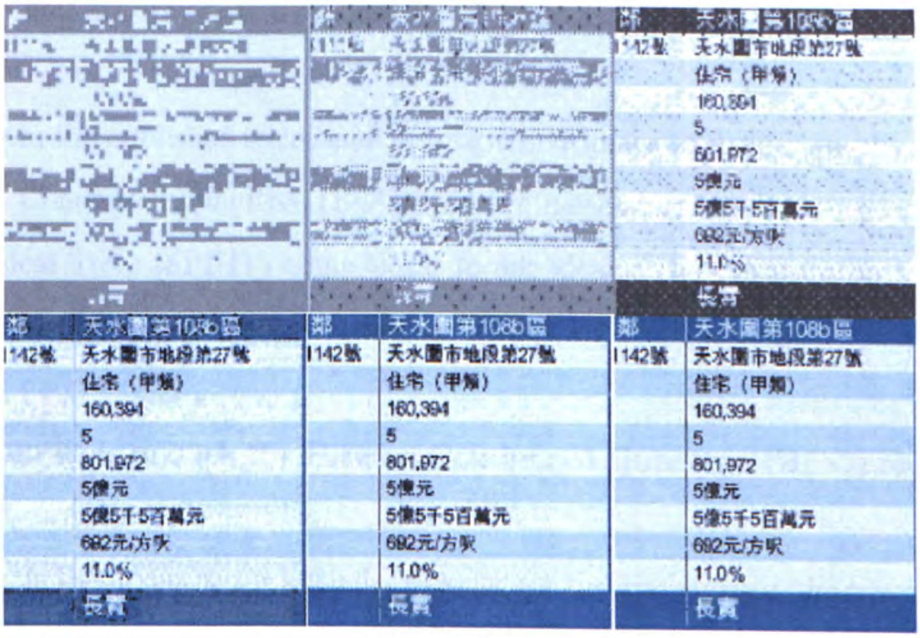
In Figure 5.4, the image with textual information has a spatial resolution of r=8 ($2^8 \times 2^8$ pixels) and has 256 colors (8 bits per pixel). We have seen that the image sequence with the spatial resolution increasing faster provides better result on the basis of fast textual information recognition than others do.

Figure 5.4: Image sequences: Increasing the contrast resolution faster, increasing both contrast and spatial resolution equally, and increasing the spatial resolution faster. (bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536)

Increasing the contrast resolution faster with bias factor $b = 20$



Increasing both contrast resolution and spatial resolution equally with bias factor $b = 0$



Increasing the spatial resolution faster with bias factor $b = -20$

5.2.2 Comparison with other Methods

It is necessary and useful to compare the new method with some proposed methods to find out its relative strengths and weaknesses.

From the published techniques of progressive image transmission, the method of Bit Condensed Quadtree (BCQ) and the method of Set Partitioning in Hierarchical Trees (SPIHT) come closer to our idea as they concern both spatial and contrast resolutions in progressive transmission.

Comparison with the Bitwise Condensed Quadtree (BCQ) method[6]

As the BCQ method is initially designed for the gray images, the new method has reconstructed a sequence of gray images to compare with it. The BCQ method is also modified to produce a sequence of color images by processing each color plane individually. In Figures 5.5 and 5.6, they show the color image sequence and the gray image sequence refined by the BCQ method and the new method respectively. The image has spatial resolution r=8 ($2^8 \times 2^8$pixels) and contrast resolution c=8 (8 bits/pixel). As both methods can change the transmission sequence freely, we have to limit this variation. Therefore, in Figures 5.5 and 5.6, the transmission sequence of the images are fixed to increase both spatial and contrast resolutions at the same rate.

Figure 5.5: The color image sequence produced by the BCQ method and the new method. (bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536) The image sequence produced by the BCQ method

The image sequence produced by the new method (bias factor $b = 0$)

Figure 5.6: The gray image sequence produced by the BCQ method and the new method. (bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536)

The image sequence produced by the BCQ method

The image sequence produced by the new method (bias factor $b = 0$)

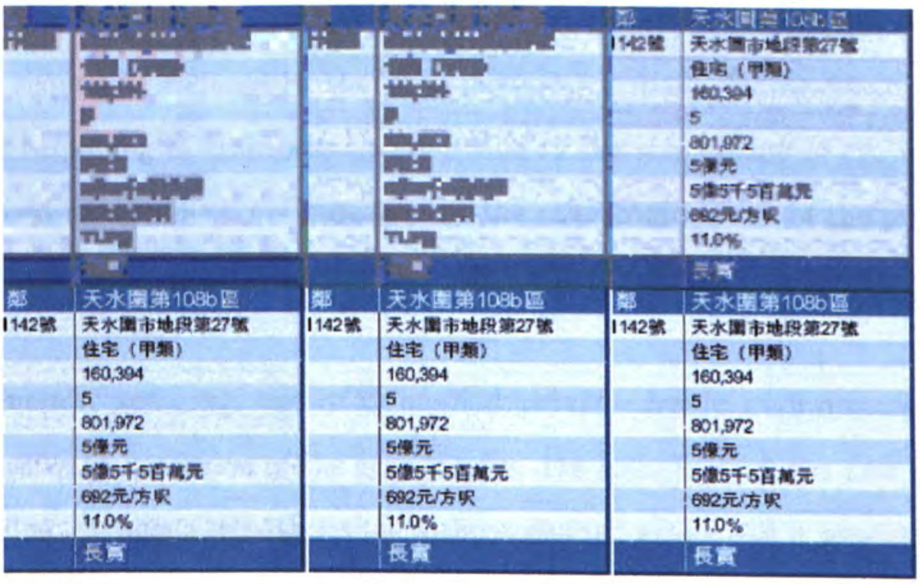
By comparing the two color image sequences in figure 5.5, we found that the BCQ method provides a better sequence that allows the viewer faster acknowledgment of the image content than the new method. It is because the new method has not included the data compression algorithm but the BCQ method does. Moreover, the pixel value in the image sequence produced by the BCQ method is further processed. It is that the pixels' values have been enlarged before display by bitwise shifting. For example, a pixel has received 3 bits '011', which will be shifted towards left-hand side to obtain '0110 0000'. So, the pixel value becomes larger for display. As the human visual system is more sensitive to luminance, the sequence produced by the BCQ method looks better than that produced by the new method. However, it is easy to find that there are some obvious errors in the image produced by the BCQ method. It is because the BCQ method has processed the three-color planes (in RGB or YC_rC_b space) individually. The process of refining the color value of different color planes becomes asynchronizes. So, sometimes a color component of a pixel becomes outstanding. This phenomenon has not happened in the image sequence produced by the new method because the pixel value will be refined in the three planes simultaneously in the color table. This is an advantage of the new method.

On the contrary, in figure 5.6, the above problem does not happen in the image sequence produced by the BCQ method because the gray image has only one color plane.

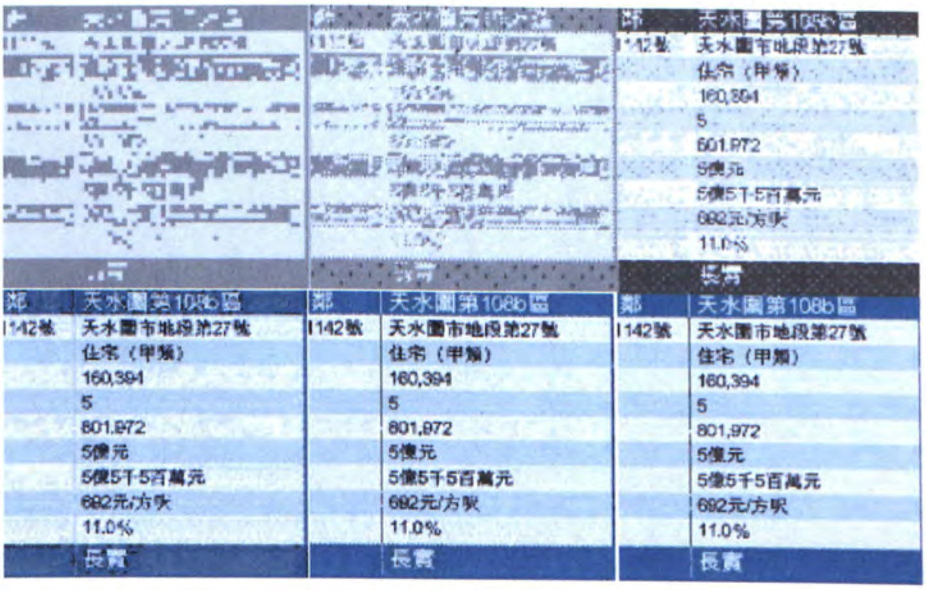
Apart from using some classic images for comparison, image with textual information are also worth testing. In Figure 5.7, both image sequences increase the spatial and the contrast resolution in the same rate.

Figure 5.7: The image sequence produced by the BCQ method and the new method.(bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536)

The image sequence produced by the BCQ method



The image sequence produced by the new method (bias factor $b = 0$)



In the above image sequences with textual information, the BCQ method provides smoother growth in the contrast resolution than the new method. It is due to the compression algorithm has included in the BCQ method but the new method has only focused on the transmission algorithm. About the recognition of the textual information, both methods can give good performance and allow user to recognize the content of the textual information quickly. However, the image sequence produced by the BCQ method has many distinguishable errors at the beginning of transmission. It is due to the color plane of the image is processed independently. Consequently, the color components of the pixels are refined in asynchronize mode and it results in those errors.

Generally speaking, the BCQ method provides better performance in the progressive transmission of the gray images. For color images, the new method can refine the image smoother that without distinct errors and it provide better viewing experience for the user.

Comparison with Set Partitioning in Hierarchical Trees (SPIHT) [16]

The Set Partitioning in Hierarchical Trees ($SPIHT$) algorithm uses the principles of the partial ordering by magnitude, the set partitioning by the significance of magnitudes with respect to a sequence of thresholds decreasing by octaves, the ordered bit plane transmission, and the self-similarity across scale in an image wavelet transform. Progressive transmission is achieved by transmitting the most important coefficients to the client first. The refinement of the image during transmission can be processed in both the spatial resolution and the contrast resolution.

However, there is no way to control the relatives rates between the spatial and the contrast resolution because both information are encoded in the coefficients. Therefore, for comparison, we are only able to limit the number of bits received during transmission and fix the image file size compressed by the SPIHT method equal to the file size of the colormapped image.

For the new method, the transmission sequence can be varied freely. To limit this variation, we will only use the image sequence produced by a transmission sequence that gives almost the best result for the image to compare with that produced by the SPIHT method. By this arrangement, we can easier to show the differences of them. The result is shown in Figure 5.8.

Figure 5.8: The image sequences produced by the SPIHT method and the new method. (bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536)

The image sequence produced by the SPIHT method

The image sequence produced by the new method with bias factor $b = 20$

It is obvious that the image sequence produced by the SPIHT algorithm can provide faster image recognition than that produced by the new method. This is because SPIHT encoder transmits the most important coefficient first which actually contains information among set of pixels. The new method only transmits data of pixels one by one. However, the test image produced by the SPIHT method becomes blur in some regions at the beginning of transmission. It is because the transmission priority of high frequency signal is lower than that of low frequency signal. Moreover, the SPIHT method is currently used a lossy image compression algorithm that some high frequency signals are sacrificed. The image can not be perfectly reconstructed as we have fixed the compressed image file size equal to the file size of the colormapped image. On the contrary, the new method can perfectly reconstruct the image using any transmission sequences.

For the gray image, as the total image file size in the direct color model almost equals to that of the colormapped image, the SPIHT method is not necessary to perform lossy compression on the image. Therefore, the gray image sequence produced by the SPIHT method can be perfectly reconstructed and so it can give a better result than the new method. The result is shown in the following figure.

Figure 5.9: The gray image sequences produced by the SPIHT method and the new method. (bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536)

The image sequence produced by the SPIHT method

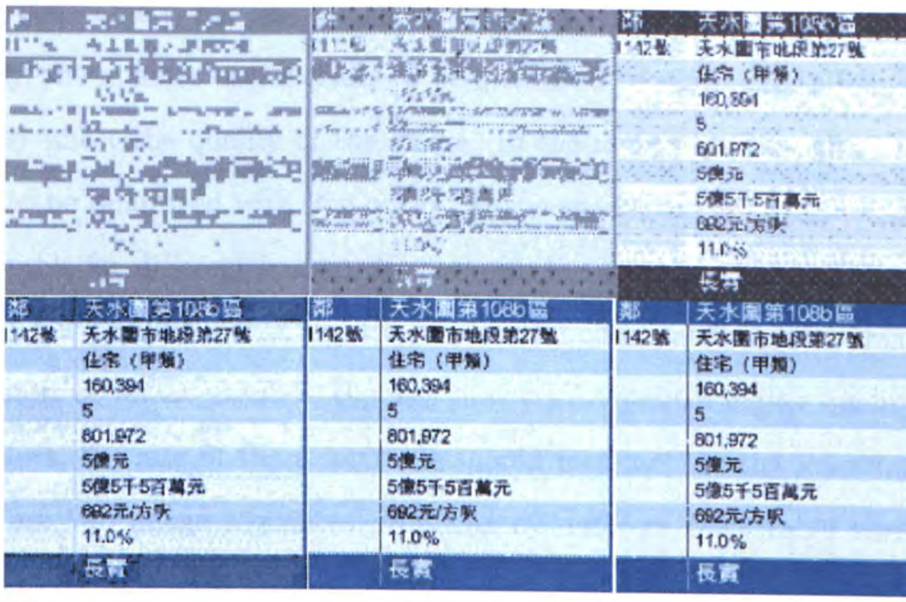
The image sequence produced by the new method (bias factor $b = 20$)

For the color image with textual information, the viewer will prefer to have a high spatial resolution and a clear image to allow him recognizing the textual content inside the image. Therefore, a lossy compression algorithm may not be suitable in this application as it causes the image unclear in some regions. In such kind of application requiring lossless transmission algorithm, the new method may provide better performance. The result of the image sequences produced by the new method and the SPIHT method are shown as follows.

Figure 5.10: The image sequences produced by the SPIHT method and the new method. (bytes received in each sequence: 505, 1126, 3789, 14746, 41472, 65536)

The image sequence produced by the SPIHT method

The image sequence produced by the new method with increasing both spatial and contrast resolutions in the same rate (bias factor $b = 0$)



5.2.3 Image quality variation

Apart from the transmission sequence, there are still other factors need to be concerned. One is the size of the color table for the colormapped image. Obviously, the number of the colors to be quantized to represent an image should directly affect the quality of the image. In the above sections, the images are fixed to be quantized with 256 colors. However, we can change the size of the color table for different images. Sometimes, the user may need a fast response with a low quality image; so, the size of the color table can be smaller, say 128 colors or fewer. Besides, the user may also request a higher quality image. Therefore, the size of the color table should increase. In the following figure, there are two image sequences produced by different color table sizes. They have 128 colors and 512 colors respectively. Both sequences increase the spatial and the contrast resolution in the same rate. The quality of the image sequence with 128 colors is lower than that of the image sequence with 512 colors, but the former can use fewer bits to complete the progressive transmission.

Figure 5.11: The image sequences produced by different color table sizes

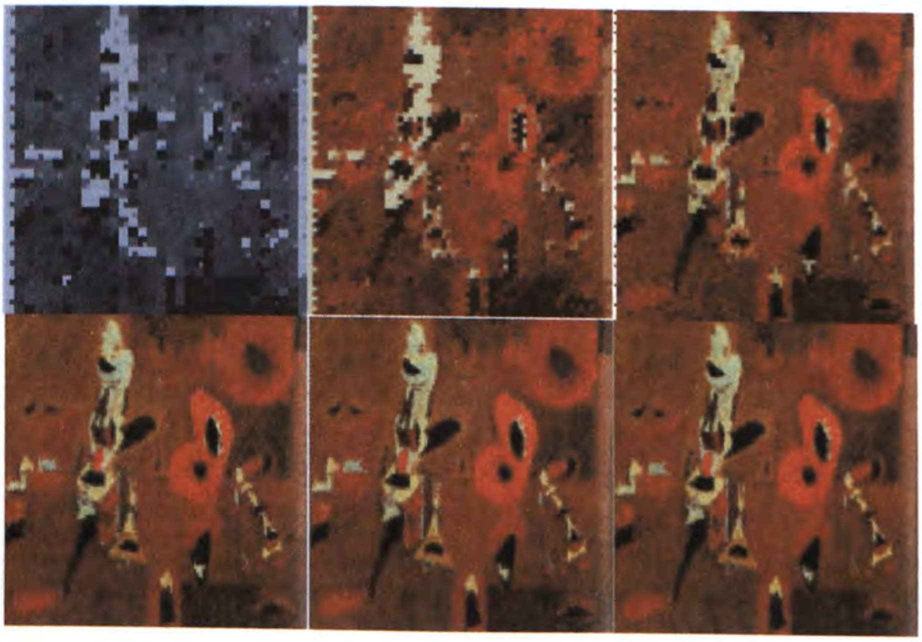
The image sequence with 128 colors (byte received: 505, 1126, 3789, 14746, 41472, 57344)

The image sequence with 512 colors (byte received: 505, 1126, 3789, 14746, 41472, 65536)

Another factor affecting the performance of the progressive transmission is the type of the image. As mentioned before, different images need different increasing rates between the spatial resolution and the contrast resolution. For the image "Lenna", the image sequence with increasing the contrast resolution faster can provide a better result than other transmission sequences do. However, the image with textual information needs to increase the spatial resolution faster to provide a better result. It allows the user to have a clearer view of the textual content. Some images may almost totally bias to the contrast resolution. For example, some fine art images do not contain shaped object inside the image. The image consists of a large set of colors like the picture in Figure 5.12. So, the transmission sequence used should be able to increase the contrast resolution as fast as possible to provide a better viewing experience. We can see that such kind of images increasing the contrast resolution faster during transmission can provide good insight to the viewer very soon as the importance of the spatial resolution is relatively low.

Figure 5.12: The art image sequence produced by the new method with increasing the contrast resolution faster (bias factor $b = 20$).

(byte received: 505, 1126, 3789, 14746, 41472, 65536 (256 colors))



Chapter 6

Discussion and Conclusion

6.1 Discussion

The colormapped image model is successfully applied in the progressive image transmission. By simply constructing a binary-tree like color table, an image can be refined in the contrast resolution and the spatial resolution as well. The encoding and decoding process involve only simple calculations in time domain.

The overhead for the progressive fidelity transmission is the extra spaces for storing the approximate colors of each cluster in the binary-tree like color table. In general, the overhead is small with respect to the file size of an image. For example, a colormapped image with size 256 by 256 pixels has 256 colors. The size of the overhead equals to 254 colors $(2^1 + 2^2 + ... + 2^7)$ $\times 3$ bytes $= 762$ bytes. It is only about 1.1% of the colormapped image size. For progressive spatial transmission, the overhead is only the small computation effort for assigning bits to their corresponding pixels.

The usage of colormapped image not only achieves image compression but also takes advantage of the displaying capability of client's view port. It can ensure the image data transmitted is useful for display in the client side and reduce the computational cost in the client for color quantization.

The transmission algorithm is a good algorithm because it provides flexibility to change the data transmission sequence to suit different kinds of the images. It allows user to have quick recognition of the image content and provides a good viewing experience to the user during the whole transmission process.

To enhance the performance of the progressive image transmission, more works are needed. Including the compression algorithm in the new method may be a good starting point for further research. It is possible to find out more efficient ways to represent the colormapped image. On the other hand, the investigation of the optimal transmission sequence for different kinds of the images is another key work has to be done to improve the transmission process. As different kinds of the images or different image applications have their own transmission preferences, further research is required to find out the optimal transmission sequences for them.

6.2 Conclusion

The problem concerned in this thesis is image communication over a low bandwidth network. The new method for progressive refinement of colormapped image was introduced and implemented successfully. It provides an alternative way using progressive image transmission with increasing both spatial and contrast resolutions simultaneously in the image communication. With the characteristics of the colormapped image and the new transmission algorithm, it provides efficient and effective way for displaying image progressively in the indirect color model instead of using the direct color model that widely used in the previous proposed methods.

The use of colormapped image takes advantage of the characteristic of client's view port to eliminate redundancy. It can ensure the image data transmitted is useful for display in the client side and reduce the computational cost in the client for color quantization.

In the implementation, a true color image is quantized into a finite number of colors by the method of sequential scalar quantization. The binary-tree like color table and the index map are constructed. The contrast resolution of the colormapped image can increase progressively by traversing the binary-tree like color table from its root to the bottom. Besides, for ensuring the image can be refined uniformly over the 2-d space during transmission, the indices of the pixels are reordered such that any subset of the consecutive pixels is distributed uniformly. Together with the bit plane transmission of the index for each pixel, the colormapped image can now be refined progressively in both spatial and contrast resolutions. Moreover, the transmission algorithm allows changing the transmission sequence freely because any transmission sequence that can perfectly reconstruct the colormapped image can be used. Therefore, the progression rate of the spatial and the contrast resolution can be changed for fitting different requirements of the applications.

As a conclusion, we showed that the new method for progressive refinement of colormapped image provides competitive performance compared with the previous techniques. It realizes the use of colormapped images in progressive image transmission. It includes data compression and efficient data transmission at the same time. Besides, the transmission algorithm provides greater flexibility to change the transmission sequence to fit for different requirements of image applications.

However, there are still some points that worth studying in more details. Firstly, it is necessary to study how to include the compression algorithm for colormapped image. If so, the quality of the progressive transmission should be improved further more. Besides, optimal transmission sequence for different kinds of the images is another issue needs to be studied more. It can make the transmission process of different kinds of the images smoother and more efficient.

Bibliography

[1] Luisa Simone, "JPEG 2000: The Next Wave(let)", PC Magazine, April 21, 1999.

[2] Hewlett Packard Co., Live Picture Inc. and Eastman Kodak Co., "Internet Imaging Protocol Specification - Version 1.0.5", 1997. (http://members.digitalimaging.org/publicspecs/).

[3] Digital Imaging Group, Inc., "Flashpix Format Specification - Version 1.0.2", 1997. (http://members.digitalimaging.org/publicspecs/).

[4] Ho John Lee, "High-resolution Graphics with FlashPix and the PsiFi Internet Imaging Protocol", URL: (http://www.webtechniques.com/features/1996/12/lee/lee.shtml).

[5] Morris Goldberg and Limin Wang, "Comparative Performance of Pyramid Data Structures for Progressive Image Transmission", IEEE Transactions, vol. 39, no. 4, April 1991, pp.540-548.

[6] Martin J. Durst & Tosiyasu L. Kunii, " Progressive Transmission Increasing both Spatial and Gray Scale Resolution", International Conference on Multimedia Information Systems 91, pp.175-186.

[7] Eve A. Riskin, Richard Ladner, Ren-Yuh Wang, and Les E. Atlas, "Index Assignment for Progressive Transmission of Full-Search Vector Quantization", IEEE Transaction on Image Processing, Vol. 3, No. 3, May 1994, pp.307-312.

[8] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design.", IEEE Trans. Commun., vol. COM-28, pp. 84-95, Jan, 1980.

[9] A. Buzo, A. H. Gray Jr., R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization," IEEE Trans. Acoust. Speech Signal Processing, vol. ASSP-28, pp. 562-574, Oct, 1980.

[10] Paul G. Howard and Jeffrey Scott Vitter, "Fast progressive lossless image compression", Proceedings of the 1994 IST/SPIC International Symposium on Electronic Imaging Science and Technology, San Jose, CA, February 1994. (www.cs.duke.edu/ jsv/Papers/catalog/)

[11] P. G. Howard and J. S. Vitter, "Fast and Efficient Lossless Image Compression", in Proc. Data Compression Conference, J.A. Storer and M. Cohn, eds., Snowbird, Utah, March 30, April, 1993, pp. 351-360.

[12] P.G. Howard and J.S. Vitter, "New Methods for Lossless Image Compression Using Arithmetic Coding", Information Processing and Management, 28:6, 1992, pp. 765-779.

[13] Jiankun Li, Jin Li and C.-C. Jay Kuo,"An embedded DCT approach to Progressive Image Compression", International Conference on Image Processing 96, vol.1, Sept. 1996, pp. 201-204.

[14] Raja Balasubramanian, Charles A. Bouman & Jan P. Allebach, "Sequential Scalar Quantization of Color Images", Journal of Electronic Imaging, vol. 3, no. 1, Jan. 1994, pp.45-59.

[15] Lloyd-Williams; Andrew, "Method for reordering the pixel map of a digitized image", April 9, 1991, in web:"patent.womplex.ibm.com" patent:US5007001.

[16] Amir Said & William A. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Tress", IEEE Transactions on Circuits and Systems for Video Technology, Vol.6, June 1996.

[17] Jerome M. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients", IEEE Transactions on Signal Processing. Vol. 41. No.12, December 1993, pp. 3445-3462.

[18] R.L. Joshi, V.J. Crump, and T.R. Fischer, "Image subband coding using arithmetic and trellis coded quantization," IEEE Trans. Circ. & Syst. Video Tech., vol. 5, pp. 515-523, Dec., 1995.

[19] S. Tanimoto and T. Pavlidis, "A Hierarchical Data Structure for Picture Processing", Computer Graphics and Image Processing 4, 1975, pp. 104-119.

[20] F.S. Hill, Sheldon Walker & Fuwen Gao, " Interactive Image Query System Using Progressive Transmission", Computer Graphics, Vol.17, No.3, July 1983, pp.323-330.

[21] Ken Knowlton, "Progressive Transmission of Grey-Scale and Binary Pictures by Simple, Efficient, and Lossless Encoding Schemes", Proceedings of the IEEE, Vol.68, No.7, July 1980, pp.885-896.

[22] Kenneth R. Sloan, Jr., and Steven L. Tanimoto, " Progressive Refinement of Raster Images", IEEE Transactions on Computers, vol., c-28, No. 11, November 1979, pp.871-874.

[23] Dinei A. F. Florencio and Ronald W. Schafer, " A Non-Expansive Pyramidal Morphological Image Coder", IEEE Transactions Communications, 1994, pp.331-335.

[24] Uwe Rauschenbach, "Progressive Image Transmission using Levels of Detail and Regions of Interest", Proceeding IASTED Conference on Computer Graphics and Imaging - CGIM'98, Halifax, Nova Scotia, Canada, June 1-4, 1998.

[25] Amir Said and William A. Pearlman, "An Image Multiresolution Representation for Lossless and Lossy Compression", IEEE Transactions on Image Processing, vol., 5, No. 9 September 1996, pp. 1303-1310.

[26] Yunming Huang, Howard M. Dreizen and Nikolas P. Galatsanos, "Prioritized DCT for Compression and Progressive Transmission of Images", IEEE Transactions on Image Processing, vol. 1, No.4, October 1992, pp. 477-487.