Description of the Image Compression Toolbox

1.0 Introduction

The image compression toolbox, written by Satish Kumar, located on the internet at http://www.debugmode.com/imagecmp/icomptbx.htm, is a wavelet based image compression tool. The toolbox contains a collection of functions that are commonly used in wavelet based image compression techniques. The following section gives a description of each function that comprises the image compression toolbox.

The image toolbox follows the following steps to compress/encode a raw image [7].

1. Specifying the Rate (bits available) and Distortion (tolerable error) parameters for the target image.
2. Classification of the image data based on their importance (wavelet transformation).
3. Dividing the available bit budget among these classes, such that the distortion is a minimum (optimal bit allocation).
4. Quantizing each class separately using the bit allocation information (quantization).
5. Encoding each class separately using an entropy coder and writing to the file.

The reverse operation of reconstruction the image from the encoded data follows these steps [7].

1. Read in the quantized data from the file, using an entropy decoder. (reverse of step 5).
2. Dequantize the data. (reverse of step 4).
3. Rebuild the image. (reverse of step 2).

The RAW image reader/writer will be replaced with a Tiff image reader/writer. Each one of the functions except for the image reader and writer and the entropy encoder and decoder, will be rewritten into parallel program.

2.0 Description of the Image Toolbox
2.1 Reading and Writing RAW image file

In the compression process a RAW image file is read and the image data is stored in an array. In the decompression process, a RAW image file is written. A RAW image is a fixed size grey scale image. To allow the ability to work with variable image sizes, the RAW image reader/writer will be replaced with a Tiff image reader/writer. This is the input and output part of the program that remains sequential.

2.2 Classification of the image data using two dimesional Discrete Watelet Transform

In this step the more important information of an image, or the base, is separated from the less important information, or the detail, using wavelet transformation. Wavelets are functions that satisfy certain mathematical requirements and can be used to represent data or other functions [3]. In digital image processing, wavelets are used to divide data into different frequency components. The wavelet analysis adopts a wavelet prototype function, called an analyzing wavelet or mother wavelet. Frequency analysis is performed with a dilated or low frequency version of the prototype wavelet.

An image is a positive function on a plane. The value of this function at each point specifies the luminance or brightness of the image at that point [4]. Digital images are samples versions of such images where the value of the function is specified at discrete locations called pixels. For the most part, still images have smooth luminance or color variation with the fine details being represented as sharp edges between the smooth variations. For example, looking at an image of a human face, slight color variation between neighboring coefficients can be observed within the face, and much larger color difference can be observed towards the edge of the face, where the color of the face contrasts with the background color. The smooth (small) variation in color can be termed as low frequency variation and the sharp variations as high frequency variation.

The low frequency components constitute the base of the image and contain more important information than the high frequency components, which constitute the details of the image. The low frequency components can be separated from the high frequency components using a Discrete Wavelet Transform (DWT).
DTW uses a high pass filter and a low pass filter to separate the low frequency signal from the high frequency signal. A low pass filter and a high pass filter are selected such that they divide the frequency range of the image equally into two. These filter pair are called the analysis filter pair. The analysis filter pair are applied on the image data such that they divide the frequency range of the data between themselves. Digital filtering operations is defined as:

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$$

where $h(k)$, $k = 0, 1, ..., N-1$ are the coefficients of the filter, and $x(n)$ and $y(n)$, respectively the input and output of the filter.

The process of applying DTW to the 2D image plane goes as follows:

Low pass filter is applied to each row of image data separating low frequency components up to half the frequency range. The filter is applied to half of the image data in a row by sampling every other data in the image plane such that the output data contains only half the original number of samples. Then high pass filter is applied to the data in each row that has not been sampled by low pass filter. The output is half the original number of samples.

Then the low pass and high pass filtering is done to each column of the intermediate data. That results in the image plane being divided into four sections as shown in the figure below: LL, HL, LH, HH.

The LL section contains the most important information of the image. Then low pass and high pass filtering is done to the LL section dividing the section into four parts as shown in the figure below. The LL section at the top left corner contains the most important information of the image. This process continues on creating a pyramid. The top of the pyramid contains the most important information of the image and detail information are captured by the rest of the pyramid.
2.3 Optimal bit allocation for various classes of the image data

The process of optimal bit allocation follows the wavelet transformation process, where image data is divided into different classes based on the importance of the data. The procedure where each class is allocated a portion of the total bit budget, such that the compressed image has the minimum possible distortion, is called bit allocation [6]. The aim of bit allocation using rate-distortion techniques is meeting the requirement of overflow prevention while maximizing the image/video quality.

The method for bit allocation used in the image compression tool kit is called the generalized BFOS algorithm, which is an extension of an algorithm for optimal pruning in the tree-structured classification and regression to coding. The generalized BFOS algorithm determines the optimal bit rates by allocating a maximum number of bits to each class, and then deallocating or “pruning” bits optimally [6].

Using the method of wavelet transformation, source data is segregated into classes or subbands. Initially, each class has equal number of bits, with is the maximum number of bits. Let this maximum number of bits be $q$, meaning the rage for all data in the class with $q$ number of bits is $2^q$. Following the procedures bellow, a sequence of optimal bit
allocations is produced with monotonically decreasing bit rates. Let \( B_i \) be the number of
bits allocated to class \( i \). Let \( M \) be the number of classes, and \( q \) be the maximum bits that
can be assigned to each class. The following procedures are from [6]

1. For \( i = 1,2,\ldots,M \), set \( B_i = q \). This is the initial bit allocation.
2. Calculate for \( i = 1,2,\ldots,M \) for \( j = 1,2,\ldots,q \), for each class I

\[
S_i(j, j-1) = -\frac{(\Delta D_{overall})}{(\Delta R_{overall})} \\
= -\frac{(d_i(j) - d_i(j-1))}{(j - (j-1))} = d_i(j-1) - d_i(j)
\]

3. Determine the class for which \( S_i(B_i, B_i-1) \) is the lowest. Assume it is class \( l \).
   (If the minimum \( S(.,.) \) is not unique, then select all classes with this value.)
   - Set \( B_l = B_l - 1 \)
4. Calculate the new overall average rate and distortion \( D \) and \( R \) as follows

\[
D = \sum_{i=1}^{M} p_i d_i(B_i)
\]

and

\[
R = \sum_{i=1}^{M} p_i B_i
\]

where, \( p_i \) is the ratio of size of class \( i \) to total image size.

Check if \( R = 0 \) or the target rate. If so, stop.

2.4 Quantization of the image data

The image data is quantized after the completion of optimal bit allocation. The
dictionary definition of quantization is the division of a quantity into discrete number of
small parts, often assumed to be integral multiples of a common quantity. The goal of
quantization is to encode a data from a source, characterized by its probability density function, into as few bits as possible in such as was that a reproduction my be recovered from the bits with as high quality as possible. The quality of a quantizer can be measured by the quality of the resulting reproduction in comparison to the original. The toolbox uses a scalar uniform quantizer.

There is scalar quantization and vector quantization. In scalar quantization, each input symbol is treated separately in producing the output, while in vector quantization the input symbols are grouped together and processed to give the output. There are uniform and non-uniform quantizers. A uniform quantizer is one where the levels are equally spaced, and non-uniform quantizer has levels with unequal spacing. A uniform quantizer can be easily specified by its lower bound and the step size. For example in the uniform quantizer shown below, if the input falls between n*r and (n+1)*r, the quantizer outputs the symbol n [7].

```
  n-2  n-1  n  n+1  n+2  <--- Output
---- ---- ---- ---- ---- ----
(n-2)r (n-1)r nr (n+1)r (n+2)r (n+3)r <--- Input
```

A quantizer partitions its outputs at discrete levels, a dequantizer is one which receives the output levels of a quantizer and converts them into normal data, by translating each level into a 'reproduction point' in the actual range of data. The quantization process can parallelized since each data of the input image is quantized independently.

### 2.5 Entropy Encoding of the quantized data

After the process of quantization, where data is represented using a finite set of values, it then is compressed using an entropy encoder. The process of data compression in general operates by taking “symbols” from an input “text”, processing them, and writing “codes” to a compressed file[1]. To be effective, a data compression scheme must be able to transform the compressed file back into an identical copy of the input text. Entropy refers to the amount of information represented in a data, usually expressed in bits/symbol. Two of the most popular entropy encoding schemes are the Huffman coding and Arithmetic coding.
Arithmetic Coding is claimed to perform better than the Huffman coding methods in many respects. Arithmetic coding takes a stream of input symbols and replaces it with a single number less than 1 and greater than 0. This single number can be decoded to reproduce the exact stream of symbols that went into its construction. The longer the input symbols get the more bits are needed in the output number. Arithmetic coding first assigns a set of probabilities to the input symbols being encoded. Then the individual symbols are assigned a range along the probability line, which is 0 to 1. The encoder and the decoder must use the same range table for symbols. The following example illustrates probability and range assignment for input symbols “K-STATE”.

K-STATE

<table>
<thead>
<tr>
<th>Character</th>
<th>Probability</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>1/7</td>
<td>0.00-0.14</td>
</tr>
<tr>
<td>A</td>
<td>1/7</td>
<td>0.14-0.28</td>
</tr>
<tr>
<td>E</td>
<td>1/7</td>
<td>0.28-0.43</td>
</tr>
<tr>
<td>K</td>
<td>1/7</td>
<td>0.43-0.57</td>
</tr>
<tr>
<td>S</td>
<td>1/7</td>
<td>0.57-0.71</td>
</tr>
<tr>
<td>T</td>
<td>2/7</td>
<td>0.71-1.00</td>
</tr>
</tbody>
</table>

Each character is assigned a range value that corresponds to the probability of its appearance. The first symbol to be encoded is “K”, which falls in the 0.43-0.57 ranges. After the first character is encoded, the range for the output number is bounded by the low number and the high number. During the rest of the encoding process, each new symbol to be encoded will further restrict the output number. The next symbol to be encoded “S” owns the range that corresponds to 0.57-0.71 in the new subrange of 0.43-0.57. That further restricts the output number to 0.511 – 0.531. The algorithm for encode a message of any length is as follows [1]:

Set low to 0.0
Set high to 1.0
While there are still input symbols do
  get an input symbol
  code_range = high - low.
  high = low + range*high_range(symbol)
  low = low + range*low_range(symbol)
End of While
output low

The algorithm to decode a message of any length is as follows[1]:
get encoded number
Do
  find symbol whose range straddles the encoded number
  output the symbol
  range = symbol low value - symbol high value
  subtract symbol low value from encoded number
  divide encoded number by range
until no more symbols

In summary, the encoding process narrows the range of possible numbers with every new symbol. The new range is proportional to the predefined probability attached to that symbol. Decoding is the inverse procedure, where the range is expanded in proportion to the probability of each symbol as it is extracted.

The arithmetic coding process requires each input symbol to be encoded sequentially. The input for a symbol to be encoded is dependent on the output of all the previous input symbols; therefore parallelizing this process is not possible. The entropy coding section of the code will remain sequential.

3.0 REFERENCES


http://www.debugmode.com/imagecmp/icomptbx.htm