Int. J. Advanced Networking and Applications 1481
Volume:04 Issue:01 Pages:1481-1485 (2012) ISSN : 0975-0290

UML Modeling for the Compression Of An Image File

Dr. Deepa Raj
Lecturer, Dept of Computer Science
B.B. Ambedkar University(Central University) Vidya Vihar, Rae Bareli RoadLucknow (U.P.) INDIA 226025
Email:deepa_raj200@yahoo.co.in

----------ABSTRACT----------
Unified Modeling language (UML) is one of the important modeling languages used for the visual representation of the research problem. In the present paper, UML model has been designed for the compression of any image using transform method because compressed image take a very less transmission time for sending image from one node to another node and at receiving end after decompression re find the original image. The class diagram and sequence diagram for the compression of an image are depicted in this paper. Some run length coding technique is also reported in this paper with activity diagram

KEYWORDS : transform, class, sequence, run length code

Date of submission: June 24, 2012 Date of Acceptance: July 02, 2012

1. Introduction

UML is a powerful modeling language used to represent the research problems visually. A lot of literature is available on modeling problems by the use of UML, but limited papers are reported in literature on applications of computer graphics. By the use of UML, Real time system based on UML is described by Selic and Rumbaugh [1]. The first represented of UML in the field of telecommunication sector is described by Holz [2]. The concept of UML was invented by the Greddy Booch et al. [3]. Sayeed [4] in his book on data compression explained about data compression technique. The computer architecture models which can be used for the further research work are available in [5]. The latest research in 2007 on distributed computing is reported by Martinez et al. [6]. UML based Vehicle control system is also reported in the literature by Walther et al. [7]. OMG is an important active group for inventing the different versions of the UML. The research papers on these are [8, 9] in which group describe the UML diagram based on XML meta data specification. Edge preserving compression technique using feed forward neural network is explained by singh et. al [10] [11] describe customizing the UML for modeling performance oriented applications. Recently Saxena et al. [13] proposed the UML model with Performance Evaluation for the Multiplex System for the process which are executing in the Distributed environment.

The present research work is based on the design of the UML model how the image is compressed at the source end in the network at the time of transforming from one node to another node. The class, sequence and activity diagram are given for the said problem. By the use of proposed model one can easily develop software for compress any image of any quality very easily. Now a day MATLAB tools is available for development of any software related of digital image processing.

2. Process

Let us first define the process which may be the group or block of instructions of program, macro, sub programs and subroutines. For defining the process, there is a need of the processing element. The processing element is defined as a stereotype and generally used to handle the concurrent process executing in the parallel and distributed environments. The following Fig. 1 shows the definition of processing unit

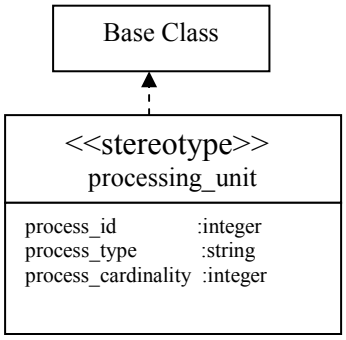
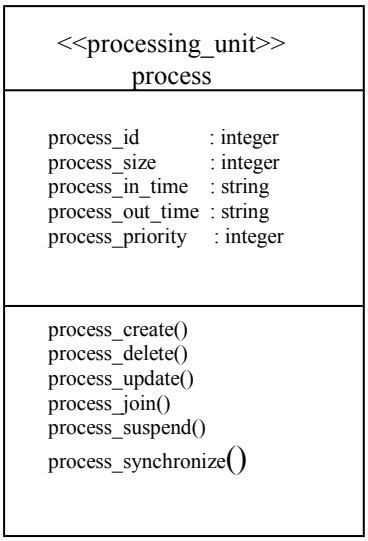


Figure 1. Definition of Processing Unit

Volume:04 Issue:01 Pages:1481-1485 (2012) ISSN : 0975-0290

The class diagram of process is defined in Fig 2



.

Figure 2. Class definition of process

The instance of the process is defined by the use of object xyz which is shown in Fig 3:

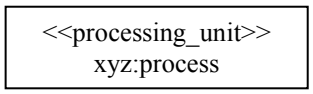


Figure 3. Instance of class

The set of the instances of the class process is modeled by the use of multiple objects which is shown below:

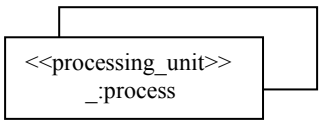


Figure 4. Multiple instances of object

In _:process shows the name of multiple objects.

3. Load the Image

Process is use to load the image. In spatial coordinate system image is stored in the frame buffer of size NXN dimension. Each cell contain the coordinate of pixel (x,y). In the gray scale scheme each pixel use 8 bits to store the information but in the colored scheme, three plane are there for each R,G and B. The monochrome image f(x,y) is discretized both in spatial coordinates and gray level values to obtain the digital image. A digital image can be represented as matrix whose row and columns are used to locate a point in the image and corresponding element values give the gray level at that point.

In the MATLAB the function
image = imread('file name') is use to load the image and we can access the each pixel by using this method.
F(x,y) = image[(x,y)]

x vary from 0 to N and y vary from 0 to N.

4. Transformation of the Image

In image processing, spatial and frequency domain approaches are there but frequency domain approaches involve a very less computation time so this approach is widely used in image processing. There are different frequency transform approaches as *fourier transform* use for image enhancement and image reconstruction, *Discrete cosine transform* widely use for image compression and compression standard and *hotteling transform* use for recognition and aligning the object with principal eigen vector axis. The popular transform used for image compression is discrete cosine transform so it is discussed in this paper in detail. It is define as

$$C(u.v) = a(u)b(v) \sum f(x,y) \cos((2x+1)u\pi/2N) \cos((2y+1)v\pi/2N)$$

where x and y vary from 0 to N-1 and u and v varies from 0 to N-1.
where
a(u) = $\sqrt{1/N}$ for u= 0 and $\sqrt{2/N}$ for u = 1 to N-1
b(v) = $\sqrt{1/N}$ for v= 0 and $\sqrt{2/N}$ for v = 1 to N-1.
f(x,y) is image in spatial coordinate system.

Cosine transform is real and orthogonal i.e $C^{-1} = C^{T}$ and it has a excellent energy compaction for correlated data. In MATLAB the instruction used to transform the spatial coordinate into Discrete cosine transform is as

Image1 = Dct(image)

5. Compression Scheme

Compression is a process of reducing the amount of data required to represent a given quantity of information or it means to remove the redundant data in the information. Various redundancy methods are there for compression of an image as variable length coding and run length coding. For variable length coding, if variable length of bits is used to represent a pixel then one can eliminate redundancy compared to the fixed length bits of a pixel. In this paper Run length coding is explained in detail.

5.1 Run Length Coding.

To compress any data either in text or character run-length encoding is a data compression algorithm that is supported by most bitmap file formats, such as Tiff, BMP file. RLE is both easy to implement and quick to execute, making it a good alternative to either using a complex

Volume:04 Issue:01 Pages:1481-1485 (2012) ISSN : 0975-0290

compression algorithm or leaving image data uncompressed. By reducing the physical size of a repeating string of characters RLE works in this manner. Uncompressed, a character run of 15 A characters would normally require 15 bytes to store: AAAAAAAAAAAAAAA. The same string after RLE encoding would require only two bytes: 15A.The 15A code generated to represent the character string is called an *RLE packet*. Here, the first byte, 15 is the run count and contains the number of repetitions. The second byte, A is the run value and contains the actual repeated value in the run. A new packet is generated each time the run character changes, or each time the number of characters in the run exceeds the maximum count. Assume that our 15-character string now contains four different character runs: AAAAAAbbbXXXXXt.Using run-length encoding this could be compressed into four 2-byte packets:
6A3b5X1t

Thus, after run-length encoding, the 15-byte string would require only eight bytes of data to represent the string, as opposed to the original 15 bytes. In this case, run-length encoding yielded a compression ratio of almost 2 to 1. Let us take another example to encode this string

Xtmprsqzntwlfb

After RLE encoding, this string becomes:

1X1t1m1p1r1s1q1z1n1t1w1l1f1b

RLE schemes are simple and fast, but their compression efficiency depends on the type of image data being encoded. A black-and-white image that is mostly white, such as the page of a book, will encode very well, due to the large amount of contiguous data that is all the same color. An image with many colors that is very busy in appearance, however, such as a photograph, will not encode very well. This is because the complexity of the image is expressed as a large number of different colors. And because of this complexity there will be relatively few runs of the same color. One can easily decode the run length code at the time of decompression.

5.2 Variants on Run-Length Encoding

There are a number of variants of run-length encoding. Image data is normally run-length encoded in a sequential process that treats the image data as a 1D stream, rather than as a 2D map of data. In sequential processing, a bitmap is encoded starting at the upper left corner and proceeding from left to right across each scan line (the X axis) to the bottom right corner of the bitmap shown in fig 3a,but alternative RLE schemes can also be written to encode data down the length of a bitmap (the Y axis) along the columns as shown in fig 5b, to encode a bitmap into 2D tiles as shown in fig 3c), or even to encode pixels on a diagonal in a zig-zag fashion as shown in Fig 5d. This RLE variant works well only with real-world images that contain many subtle variations in pixel values.

RLE encoder always stops at the end of each scan line of bitmap data that is being encoded. There are several benefits to doing so. Encoding only a simple scan line at a time means that only a minimal buffer size is required. Encoding only a simple line at a time also prevents a problem known as *cross-coding*. Cross-coding is the merging of scan lines that occurs when the encoded process loses the distinction between the original scan lines. Cross-coding is sometimes done, When an encoder is encoding an image, an end-of-scan-line marker is placed in the encoded data to inform the decoding software that the end of the scan line has been reached. This marker is usually a unique packet, explicitly defined in the RLE specification, which cannot be confused with any other data packets. End-of-scan-line markers are usually only one byte in length, so they don't adversely contribute to the size of the encoded data.

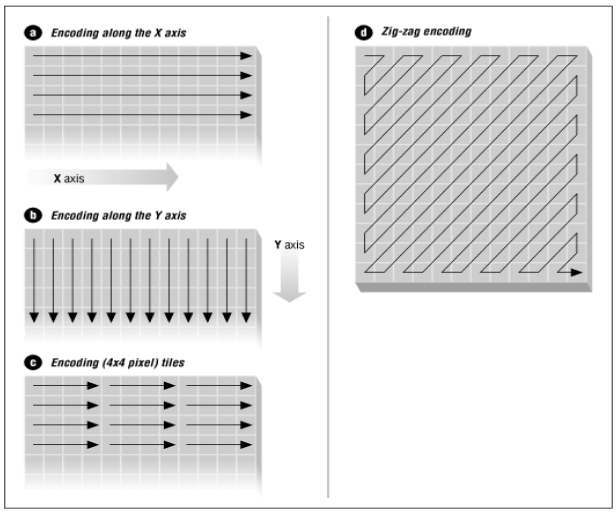


Figure 5: Run-length encoding variants

6. UML Model for Compression of Image

Step involved in compressing an image is as follows.

1. Load image
2 Take a sub image of size 8x8
3. Apply cosine transform for this sub image
4. Then these coefficients are normalized using the normalized matrix and apply run length coding technique to compress the image.
5. Repeat the same procedure until all image are scanned.
6. Exit

Volume:04 Issue:01 Pages:1481-1485 (2012) ISSN : 0975-0290

Unified Modeling Language is powerful Tool for modeling of object oriented software system. UML class Diagram is shown in Fig-6 which shows how the classes are associated to each other for compressing any image. Activity Diagram and Sequence Diagram are also shown in this paper. In the sequence diagram one can see that how message passing take place among the different classes like Load_image, Subimage, DCT_transform, quantization, Run_ length_code and compress_image. It also shows that how long time that all classes involve in Execution. Activity diagram shown in fig 7, which shows systematic steps involved in compressing any image of any type.

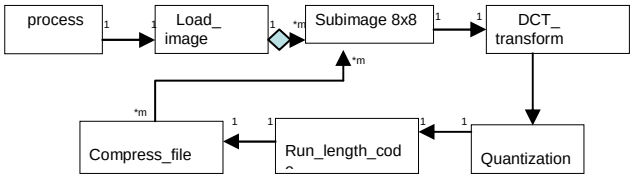


Figure 6. UML Class Diagram for compression of image

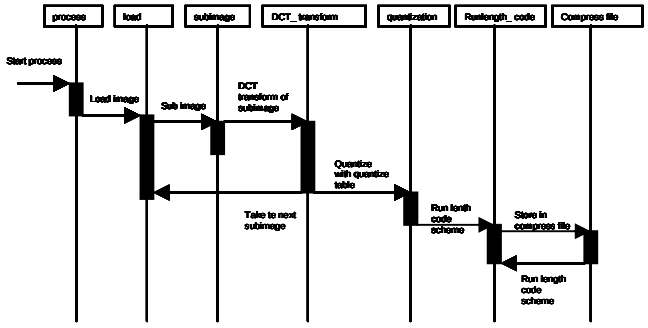


Figure 7. Sequence Diagram of Compression of image

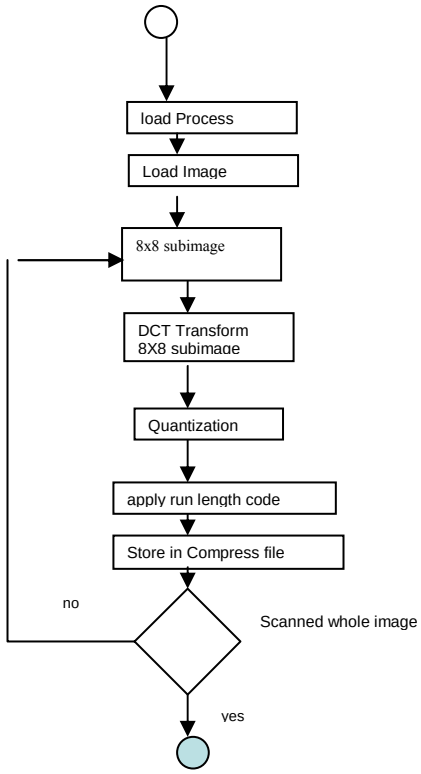


Figure 8. UML Activity diagram for compression of image

7. UML Modeling for Uncompressing an image

At receiving end uncompressing an image is required for finding an original image. Therefor in compressed file apply inverse run length coding for getting again 8x8 images after that apply quantization and inverse DCT transform and merging all the 8x8 sub images to get the decompress file as original image. UML diagram is depicted in the Fig-8 for decompressing a compressed file.

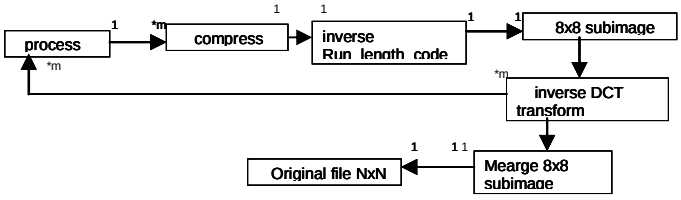


Figure 8. UML Class Diagram for uncompressing of image

Volume:04 Issue:01 Pages:1481-1485 (2012) ISSN : 0975-0290

8. Conclusion and Future scope of Work

All the application of image processing can be easily implemented with the help of UML modeling such as image segmentation, detection of line in the image, pattern recognition in the image as well as water marking in the image. This modeling technique will be suitable for software developer to develop a software regarding image processing.

Acknowledgement

Authors are very thankful to Prof. B. Hanumaiah,Vice Chancellor, Baba Saheb Bhimrao Ambedkar University(A Central University), Vidya Vihar, Rai Bareli Road, Lucknow, India, for providing the excellent computation facilities in the University Campus. Thanks also due to the University Grant Commission, India for providing financial assistance to the University for the Research.

References

[1]. B. Selic and J. Rum Baugh, "UML for Modeling Complex Real Time Systems", Available Online www.rational.com/Products/Whitepapers/100230.Jsp.

[2]. E. Holz, "Application of UML within the Scope of New Telecommunication Architectures" , *GROOM Workshop on UML*, Mannheim :Physicaverlag , 1997.

[3]. G. Booch , J. Rumbaugh, and I. Jacobson , "*The Unified Modeling Language User Guide*", Addision Wesley, Reading, MA ,1999.

[4]. K. Sayood, "Introduction to data compression", 2nd ed., 3rd ed., Academic Press, Morgan Kaufmann Publishers, 2000, 2006.

[5]. K. Hwang. "*Advanced Computer Architecture*", Mcgraw – Hill, Inc Publishing, 1993.

[6]. Martinez, Jesus, Merino, Pedro, Solmeron, Alberto, "Applying MDE Methodologies to Design Communication Protocols for Distributed Systems", *IEEE Transaction of Software Engineering, April* 2007.

[7]. M. Walther, J. Schirmer , P.T. Flores, A. Lapp, T. Bertram and J. Peterson . "Integration of the Ordering Concept for Vehicle Control System CARTRONIC into the Software Development Process using UML Modeling methods" , *SAI 2001 World Congress Detroit, Michigan, USA.*

[8]. OMG, "Unified Modeling Language Specification", Available Online Via www.omg.org, 2001

[9]. OMG, "OMG XML Metadata Interchange (XMI) Specification", Available Online Via. www.omg.org , 2002.

[10]. R. singh, J.singh, "Edge preserving compression technique using feed forward neural network", *IJCST vol.2*, issue I, march 2011

[11]. S.Pllana, and T. Fahringer, "UML based Modeling of Performance Oriented Applications", *Winter Simulation Conference*, 2002.

[12]. S. annadurai and R. Shanmugalakshmi, "*Fundamental of digital image processing*", Pearson education, ISBN 81-7758-479-0

[13]. V. Saxena, D. Arora and S. Ahmad, "Object oriented distributed architecture system through UML", *IEEE International Conference on Advanced in computer vision and information Technology,* November 28-30, 2007.