2010

Development of a Remote Medical Image Browsing and Interaction System

Wei Ye
Wright State University

Follow this and additional works at: http://corescholar.libraries.wright.edu/etd_all
Part of the Computer Engineering Commons

Repository Citation

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact corescholar@www.libraries.wright.edu.
DEVELOPMENT OF A REMOTE MEDICAL IMAGE BROWSING AND INTERACTION SYSTEM

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Engineering

By

WEI YE
B.E., Beijing Institute of Technology, 2006

2010
Wright State University

_____________________
Yong Pei, Ph.D.
Thesis Advisor

_____________________
Thomas Sudkamp, Ph.D.
Chair, Computer Science and Engineering

Committee on
Final Examination

_____________________
Yong Pei, Ph.D.

_____________________
Bin Wang, Ph.D.

_____________________
Keke Chen, Ph.D.

_____________________
John A. Bantle, Ph.D.
Vice President for Research and Graduate Studies and Interim Dean of Graduate Studies
ABSTRACT

In this thesis a new capability - remote image browsing built upon our existing music telepresence platform is introduced. The implementation of this new image capability has two parts: one part is the local image viewing and the other part is distributed image-view interaction. Image viewing part is realized using the foundation of Linux GTK+ library. Most popular lossless and lossy image coding formats such as GIF, PNG, BMP and JPEG are supported currently. Image viewing part also provides several image operations such as: zoom in and zoom out, image information display and moreover users can select their own regions of interest to zoom in and view. Distributed image-view interaction uses TCP protocol to provide reliable image data packets delivery and browsing. The prototype system enables two users in the remote session to view exactly the same region of interest of the image. Furthermore, remote image browsing capability is enhanced with the other existing capabilities, such as text messaging, low latency audio and video interactions to construct a fully-fledged interactive environment for users to collaborate remotely.

Additionally, this thesis also closely evaluates the algorithms and techniques within the new image compression standard JPEG 2000 for their applicability in a distributed collaborative system. Specifically, compression ratio, encoding time and decoding details of two official JPEG 2000 testing images are analyzed by using an available
open source JPEG 2000 codec implementation – OPENJPEG project. It is found that compression ratio is not a critical factor to affect the encoding time if an image is encoded with only one tile option. Among the four decoding steps, tier-1 coding which includes three passes: significance pass, refinement pass, cleanup pass and entropy coding consumes the most decoding time. Inverse Discrete Wavelet Transformation needs second most time. Multi component transformation consumes third most decoding time and tier-2 coding requires the least time to finish. These results may set guide-lines for future adoption of the new JPEG 2000 image compression techniques in our prototype system.
# TABLE OF CONTENTS

Chapter 1 ................................................................................................................... 1
Introduction ............................................................................................................ 1
1.1 Data Compression .......................................................................................... 2
1.2 Image Compression ....................................................................................... 3
1.3 Lossless vs Lossy ......................................................................................... 4
1.4 Summary ........................................................................................................ 5
1.5 Organization of Thesis .................................................................................. 6

Chapter 2 ................................................................................................................... 7
Music Telepresence Project ................................................................................... 7
2.1 Telepresence ................................................................................................. 7
2.2 Project Introduction ...................................................................................... 7
2.3 System Capabilities ...................................................................................... 8
2.4 Project Test Result ......................................................................................... 9
2.5 Summary ....................................................................................................... 9

Chapter 3 ................................................................................................................. 11
Remote Image Browsing Capability ...................................................................... 11
3.1 Image Viewing Part ...................................................................................... 12
3.1.1 Image Operations .................................................................................... 13
3.1.2 Region of Interest Selection ................................................................... 16
3.2 Image-View Interaction Part ......................................................................... 18
3.2.1 Image Sending Procedure ....................................................................... 19
3.2.2 Image Receiving Procedure .................................................................... 21
3.3 Distributed Image Browsing .......................................................................... 23
3.4 Other Medical Image Browsing Applications .............................................. 25
3.5 Summary ....................................................................................................... 27

Chapter 4 ................................................................................................................. 29
JPEG 2000 ............................................................................................................. 29
4.1 Introduction .................................................................................................... 29
LIST OF FIGURES

Figure 3.1 Image Viewing Browser........................................................................ 13
Figure 3.2 Image Viewing Operations.................................................................... 16
Figure 3.3 Region of Interest Selection................................................................. 17
Figure 3.4 TextMessaging Notification.................................................................. 19
Figure 3.5 Sending and Receiving Procedures....................................................... 22
Figure 3.6 Region of Interest Transmission............................................................ 24
Figure 3.7 Interactive Image Browsing ................................................................. 25
Figure 4.1 JPEG and JPEG 2000 Encoder............................................................ 32
Figure 4.2 Tiling, DC Level Shifting, and DWT on Each Tile............................... 34
Figure 4.3 Dyadic Decomposition......................................................................... 36
Figure 4.4 D’s Stage DWT Producing 3D+1 Subbands......................................... 36
Figure 4.5 Stripe Oriented Scan Though Code Block Samples.............................. 39
Figure 4.6 Bit Planes ......................................................................................... 40
Figure 5.1 Original BMP image size: 2.02 MB (2,125,878 bytes) ......................... 43
Figure 5.2 JPEG2000 lossless image size: 1.24 MB (1,301,019 bytes)................. 43
Figure 5.3 JPEG2000 1:10 image size: 207 KB (212,026 bytes) ......................... 44
Figure 5.4 JPEG2000 1:50 image size: 41.4 KB (42,440 bytes) ......................... 44
Figure 5.5 JPEG2000 1:100 image size: 20.7 KB (21,255 bytes) ....................... 45
Figure 5.6 JPEG2000 1:200 image size: 10.1 KB (10,364 bytes) ....................... 45
Figure 5.7 JPEG2000 1:500 image size: 4.12 KB (4,224 bytes) ......................... 46
Figure 5.8 JPEG2000 1:1000 image size: 2.07 KB (2,121 bytes) ....................... 46
Figure 5.9 Original BMP image size: 26.7 MB (28,080,054 bytes) ..................... 49
Figure 5.10 JPEG2000 lossless image size: 8.81 MB (9,248,070 bytes).............. 49
Figure 5.11 JPEG2000 1:10 image size: 2.67 MB (2,807,643 bytes) ................. 50
Figure 5.12 JPEG2000 1:50 image size: 548 KB (561,596 bytes) ............... 50
Figure 5.13 JPEG2000 1:100 image size: 274 KB (280,658 bytes) ............... 51
Figure 5.14 JPEG2000 1:200 image size: 137 KB (140,397 bytes) ............... 51
Figure 5.15 JPEG2000 1:500 image size: 54.8 KB (56,141 bytes) ......................... 52
Figure 5.16 JPEG2000 1:1000 image size: 27.3 KB (28,030 bytes) ..................... 52
Figure 5.17 JPEG2000 1:2000 image size: 13.6 KB (13,953 bytes) ..................... 53
Figure 5.18 JPEG 2000 Lossless Image Waltham Each Step Decoding Time .......... 55
LIST OF TABLES

Table 1.1 Uncompressed Image Storage Space and Transmission Time...................... 1
Table 5.1 Image 7 Sisters Encoding Time ................................................................. 42
Table 5.2 Image Waltham Encoding Time ................................................................. 47
Table 5.3 JPEG 2000 Lossless Image Waltham Decoding Time ......................... 54
Table 5.4 Average Time of Each Decoding Step .................................................... 55
ACKNOWLEDGEMENTS

First of all, I would like to express my intense gratitude to my advisor Dr. Yong Pei who introduced me to the field of data compression and transmission networking. His knowledge, support, guidance and encouragement are really critical to me. Without him and his help this thesis would be impossible to complete.

My special thanks also go to the members of my thesis committee - Dr. Bin Wang and Dr. Keke Chen for their precious time in reviewing this paper and their valuable suggestions. I am grateful to them for their encouragement and support on my research and studies. I also would like to thank all the faculty and staff of the Department of Computer Science and Engineering at Wright State University for giving me lots of guidance and assistance.

I am very fortunate to work with a group of students in the Mobile Information and Communication Systems Lab at Wright State University, including Paul Bender, Jianing Ma and Isaac Keven Matthew. I had a great time working with them.

I would like to give my deepest gratitude and love to my parents. They have been always encouraging and assisting me throughout my life. Without their inspiration, unfailing love and faith in me it would be a tremendous struggle for me to make though this most important stage of my life.

Last but not least, I would like to thank my wife. Her sacrifice, support, encouragement and love made it possible for me to complete this study.
Dedicated to

My parents and my wife
Chapter 1

Introduction

Table 1.1 below shows the size transition from grayscale image to medical image and the storage space, transmission time needed to store and transmit such uncompressed image data.

<table>
<thead>
<tr>
<th>Media Data</th>
<th>Size</th>
<th>Bits/Pixel</th>
<th>Uncompressed Size (Bytes)</th>
<th>Transmission Time (using 1Mb/S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grayscale Image</td>
<td>1024 x 768</td>
<td>8 bpp</td>
<td>768 KB</td>
<td>6 sec</td>
</tr>
<tr>
<td>Color Image</td>
<td>1024 x 768</td>
<td>24 bpp</td>
<td>2.304 MB</td>
<td>18 sec</td>
</tr>
<tr>
<td>Medical Image</td>
<td>3072 x 2304</td>
<td>24 bpp</td>
<td>20.25 MB</td>
<td>2 min 42 sec</td>
</tr>
</tbody>
</table>

Table 1.1 clearly illustrates the need for sufficient storage space, large transmission bandwidth and long transmission time for images, especially for large images such as medical images and satellite images. Since medical images including more and more
color ultrasound and high definition CT images are becoming much larger than before, some of them even reach gigabytes, the demand to efficiently compress those data or images increases urgently.

1.1 Data Compression

In information technology we can use specific encoding schemes to encode information and make it have fewer bits than the source representation which is not encoded. This process is called data compression. After compressing the original data if we would like to transmit the compressed data only both the sender and the receiver understand the coding scheme (sender knows the encoding and receiver knows the decoding scheme) and then the communication between them can success.

Effective compression can help to reduce the consumption of expensive resources such as considerable storage capacity or network transmission bandwidth. However at the receiver side we must decode the data after we receive the compressed data from the sender otherwise we can not use it. The decoding process will also consume computational resources and processing time. For example, high definition video decoding may require expensive hardware such as powerful video card and central unit processor to decompress the video fast enough if we want to decode and watch the video at the same time. Although we may decompress the video before watching it extra waiting time and storage space are required. So there will always be a tradeoff between the transmission time and coding time. Here the coding time contains both the encoding time at the sender side and the decoding time at the receiver side.
1.2 Image Compression

A common characteristic of most images is that the neighboring pixels are correlated and therefore the image data contains redundant information. If we can find a less correlated representation of the image or remove the duplication from the image signal source data the image size can be reduced. In general, there are two types of redundancy in still image data can be identified:

(1) Spatial Redundancy exists between neighboring pixel values.

(2) Spectral Redundancy is among different color planes or spectral bands.

The objective of image compression is to reduce the number of bits needed to represent a digital image by removing the spatial and spectral redundancies as much as possible and after compression we are able to store more images or transmit the images much faster.

Image file size is relevant to two main factors: the number of pixels composing an image and the color depth of the pixels. The size of an image will increase if it has more rows, columns and higher resolution or the color depth of each pixel increases. An eight bits (or one byte) pixel stores 256 colors and 24 bits (or three bytes) pixels can store 16 million colors which is also called true color.

Ideal image compression scheme will minimize the size in bytes of a graphics file without degrading the quality of the image to an unacceptable level. The reduction in
file size allows more images to be stored in a given amount of disk or memory space. It also helps reduce the transmission time and bandwidth required for images to be sent over the Internet or downloaded from web pages.

1.3 Lossless vs Lossy

Lossless compression is often used to compress a file such as text or executable program without the introduction of any losses and errors, but only up to a certain extent. Beyond this point losses or errors will be introduced. In text and program files, lossless compression techniques are always utilized since a single error can seriously damage the meaning of a text file, or cause a program not to run properly.

In image compression, it is probably not noticeable to have a small loss. There is no “critical point” up to which compression works perfectly, but beyond which it becomes impossible. When there is some tolerance for loss, the compression factor can be larger than it can when there is no loss tolerance. For this reason, graphic images can be compressed more than text files or programs.

In lossless compression the decoded data from the compressed data should be exactly same as the original uncompressed data. But lossy compression may only reconstruct an approximation of the original data from the compressed data to achieve better compression ratio. Lossy compression techniques usually completely discard the redundant information.
Lossless compression is preferred for archival purposes and is often used for medical images or technical drawings which have more requirements about the high quality and details. Because lossy image compression methods may introduce compression artifacts, they are more suitable for natural images such as photos in which minor or imperceptible loss of fidelity (also called visually lossless) is acceptable to obtain much higher compression.

Currently two popular image compression techniques used on Internet are JPEG (often used with lossy option) and GIF (Graphics Interchange Format, only used with lossless option). JPEG is usually used for photographs, pictures and GIF is commonly used for line art and images whose geometric shapes are simple.

1.4 Summary

In this chapter the reasons and demands such as saving storage space, transmission time and bandwidth of the image data compression is illustrated first. An overview of data compression and some of its terminologies and theories are reviewed. The correlation and difference between lossy image compression and lossless image compression are discussed. Several detail definitions and explanations of the image coding are also described in this chapter. In this thesis we will compare two representation techniques of image compression: DCT (Discrete Cosine Transform) based JPEG, often used as lossy image coding and wavelet based JPEG 2000 which can do lossless image coding.
1.5 Organization of Thesis

In this paper I will introduce the remote medical image browsing capability based on our existing openh323 music telepresence project. The implementation of this new image capability is discussed as two parts: local image viewing part and image-view interaction part.

Some important algorithms and techniques at the heart of the JPEG 2000 image compression standard are studied. The software architecture and implementation of the image codec specified in the JPEG 2000 standard will also be briefly analyzed.

In the last section of this thesis encoding and decoding tests of two official JPEG 2000 testing images are done by using an open source JPEG 2000 codec implementation – OPENJPEG project and the testing result is studied by compression ratio, encoding time and decoding detail steps.
Chapter 2

Music Telepresence Project

2.1 Telepresence

A set of technologies which allow people in different locations feel as if they were together, can give the appearance that they were present and can make an effect, at one location other than their true locations is referred as telepresence. Telepresence is used not only in business meeting area, but also can be used in some hazardous environments such as mining, rescue of victims from disasters and small diameter pipelines inspection.

Music telepresence project can help musicians or artists from different places to rehearsal or perform interactively. It also has the potential to help physicians to do remote medical diagnosis, treatment or even surgery.

2.2 Project Introduction

Internet based telepresence has the potential to make the social collaboration or cooperation much easier by eliminating the physical barriers between collaborators. The goal of this music telepresence project is to advance the state of internet based telepresence among a number of successively demanding applications. The model
applications to be addressed in this research range from high-quality multicast of audio transmission to low-latency duplex communication to enable real-time distributed interactive musical rehearsal, training and performance.

Music rehearsal, cooperation and performance are highly demanding of audio and video quality, and in interactive circumstances latency is a very important issue. The technical challenges of this project encompass issues ranging from the development of efficient and reliable network delivery protocol and multimedia compression to audio and video studies.

2.3 System Capabilities

Our music telepresence software is based on the OpenH323 project and running on the Linux operating system. Our software can help multiple musicians in different locations to collaborate and perform as they are together. It can provide high quality audio and video and can successfully satisfy the low latency requirement by musical performance. Instant text messaging is also supported by our music telepresence software as another capability.

The music telepresence system can support a wide range of audio sampling rates: 44K (standard CD quality stereo music), 22K, 11K and 8K and many video quality choices: 4CIF (704 x 576), CIF (352 x 288) and QCIF (176 x 144). The highest audio sampling rate 44100 Hz which we provide is not supported by most other existing voice over IP and video conferencing systems. If your network is limited you can
have many audio or video quality combinations. For instance, if perfect audio is needed without specific requirements of video then 44K Hz audio and QCIF video probably are the best selections.

### 2.4 Project Test Result

Our tests are carried out among the three universities participants which are The University of Rochester, including the Eastman School of Music, The University of Miami, including the Frost School of Music and Wright State University. We have 4 musicians rehearsing and performing together in the tests, one guitar at Rochester, New York, one piano at Miami, Florida and two guitars at Dayton, Ohio. These tests are stable, low latency and show highly acceptable audio and video effects.

Our software obtains a very low, only 35 milliseconds end to end latency in the two sites, Dayton and Rochester music session. If we use other network online communication tools such as Windows Live Messenger or Yahoo Messenger in the same test between Dayton and Rochester, the end to end latency is over 250 milliseconds. This long delay can not be accepted by musicians who will perform music together and tests show that 100 milliseconds is the higher bound for remote music performance.

### 2.5 Summary

This chapter gives a review of the goal, technical challenges, research and
development result of our Internet based critical low latency music telepresence project. Our H323 based model application achieves 35 milliseconds end to end latency between two campus sites which can fully and successfully support the music rehearsal and performance remotely. This project not only has the obvious benefits of distance music collaboration and education, but also enables individuals with physical limitations and people who are geographically isolated to participate in a wide range of essentially interactive activities from scientific and engineering cooperation to healthcare delivery. The development of effective telepresence technologies will have obvious economic benefits of reducing the need and cost to travel and the inestimable benefits of more and more human interaction and collaboration.
Chapter 3

Remote Image Browsing Capability

More and more people complain that hospitals or clinics are too far away from the medical imaging examination laboratories. And for some physicians who can not travel conveniently it is really difficult to discuss their patients’ medical images such as electrocardiograms (ECG), X-ray or CT images with cardiologists or internists in other special hospitals. Another aspect is that some patients may need cooperation or consultations between physicians and experts in different areas and it is also really difficult and expensive for the physicians or experts to get together.

To avoid the difficulties mentioned above our music telepresence system can also be applied to this medical cooperation area. Physicians and medicine experts in different cities, states or even countries can discuss the state of one patient’s illness, treatment method with both low latency audio and video. Besides they can also examine and study the medical images remotely but like sitting together at the same time.

For this purpose I added the remote interactive image browsing capability to our existing music telepresence system. The two parts of this capability: local image viewing part and distributed image-view interaction part also achieve the objective of basic collaboration and cooperation between physicians in different locations.
3.1 Image Viewing Part

Currently most dominant image formats including: JPEG, BMP (Windows Bitmap), PNG (Portable Network Graphics), GIF (Graphics Interchange Format), TIFF (Tagged Image File Format), XPM (X Window System Pixmap), PPM (Portable Pixmap), PGM (Portable Graymap) and PBM (Portable Bitmap) are supported in this image viewer.

Figure 3.1 is a screen shot taken when we open the image viewing part of the music telepresence software. The image browsing window will show up either we choose the “Image Browsing” option under “Tools” or we click the image browsing quick button on the left side of the software. The menu including two options: “Open local image” and “Open received image” will pop up if right button click of the mouse on the grey background of the image browser is taken.
3.1.1 Image Operations

This image viewer provides several basic image operations. We can use the “up” button on the keyboard to increase the size of the displayed image. Every time we press “up” button the image will be enlarged by twenty-five percent. We can press “down” button on the keyboard to decrease the size of the displayed image. The image is shrunk by twenty-five percent if “down” button is pressed. These two image resizing operations are done by firstly obtaining the current height and width of the image and then calculating the image new size through multiplying or dividing a fixed factor. The manipulated image is displayed with the new height and width.
Other useful buttons on the keyboard is described in the following: if “F” is pressed, an original “un-scaled” version of the image will be displayed. Since the original image and the manipulated image are stored in two separate pixel buffers so when “F” button is pressed the image in the original pixel buffer will be displayed. If “R” is pressed, the image will be restored back to its original scaled size. We can use “X” button to quit the image viewer.

If we press “I” button on the keyboard a new dialog box will be displayed to show the information about the image like in Figure 3.2. For instance, the dialog box will show the original and current size of the image: current height, width in pixels and original height, width in pixels. This image scaling information is also sent to the other user in the same session through text messages automatically to help the other user know some basic information about the image.

We can also use the scroll wheel of the mouse to resize the image: if the scroll wheel is rolled up the size of image is increased by twenty-five percent as we press the “up” button on the keyboard; scroll down has the same function as the keyboard “down” button. When the image is resized either larger or smaller the display window will fit the new image dimension automatically. This is particularly necessary while shrinking the image.

If we click the right button of the mouse in the display window a menu containing two options will pop up as shown in Figure 3.1. The first option is to open a local image residing on the internal hard drive or plugging and playing devices such as USB
drives, memory sticks. When we choose to open a local image a file chooser will display and we can select the image file we want to view or transmit. The image file we chose is put into an allocated GDK pixel buffer space and displayed. Its original measurements are recorded. If the image is bigger than the desktop screen size it will be resized to fit the screen. This part is discussed later. After we open one image file we can transfer the whole image or some part of it to the other user in the same session with us. The detail of this image transmission will be discussed in the next part.

The second option is to open the received image. Just clicking this choice the received image can be displayed. Since currently only one last received image will be saved all previously received images were overwritten by the last one. Hence if users would like to keep one image they have to copy or move the received image to other file folders or drives manually.
3.1.2 Region of Interest Selection

The left button of the mouse is quite useful: it allows the user to select an area of the image and zoom in on it as shown in Figure 3.3. For instance, if we click the left button of the mouse in the display image then the image is zoomed in and the point pixel where we clicked will be the central point of the zoomed in image. The number of pixels by which to zoom is adjusted automatically in order to preserve the aspect ratio in regards to the already sized image display window. Also we have to ensure that the calculated starting point of the zoomed image portion does not exceed the boundaries of the existing image. The zoomed in image is a “sub image” of the
original display image and it is scaled to fit the display window. This zoom in feature allows the user to select his or her region of interest and if the user wants to share or discuss only this region of interest of the image the user can just transmit this region but not the whole image. This will be discussed more in the next section. After zoom in we can use either the scroll wheel of the mouse or the “up”, “down” buttons on the keyboard to restore back to the original display image.

![Figure 3.3 Region of Interest Selection](image)

This image viewer also has the function which ensures that the dimension of a displayed image can not exceed the capacity of the desktop. If one image is taller or wider than the desktop it will be resized to fit the screen and will keep its original height to width ratio. To accomplish this function we firstly obtain the desktop screen
height and width by using the GDK library procedures and leave some buffer room for things like taskbars on the screen around our image window. Then we compare the image size with the screen size if the image is bigger we will resize the image with keeping the same original height to width ratio and display the resized image.

### 3.2 Image-View Interaction Part

When two end users are in the same remote session and one of them uses the image viewer to open a local image then the user can transfer the whole opened image or only a portion of the image such as one region of interest of the image to the other user by using the “S” button on the keyboard. Since user can choose to either transfer the current whole display image or only some portion of it this enables some basic interaction or cooperation of the remote image browsing between users in different locations. When the image is transmitted to its destination place a text notification message is also sent to the image receiver automatically as shown in Figure 3.4.
We added a time stamp to the text message so it is convenient to be notified every time receiving a new image.

### 3.2.1 Image Sending Procedure

At the sender side, the system will firstly allocate enough memory to hold the entire image file which is opened locally. If the user would like to send the image to the other user within the same session by pressing the “S” button on the keyboard then the system creates a socket with a default port number and gets the IP address of the destination receiver. We use Transmission Control Protocol (TCP) as the transport
layer protocol since TCP can guarantee the error-free reliable transmission which is crucial and the first priority for medical image transmission.

After creating the sending socket and mapping TCP transport protocol the system will try to establish a connection with the specified host address (IP address of the receiver). If the TCP connection is established successfully system will send all data in the file buffer to the receiving socket. Since we already read the entire image file into memory we appropriately increment the pointer, so that each time we call send we only supply a size of at most size of the buffer. The operating system will cut up the buffer and send them as individual packets of appropriate size. Hence, we can not attempt to calculate how many packets we sent by counting how many times the socket send function we called.

In order to keep the other image operations alive the system spawns a child process that does the sending and parent process will reply for other user actions. After finishing sending all the image data held in the buffer the child sending process will clear the whole buffer and terminate gracefully. While sending the image the sending user will also send a text message to the receiving user to tell that one image has been sent to him or her.

If one user only wants to transfer a portion of the current entire image the user can select one area of the display image using the zoom in function. After selecting the interested region of the display image the sender can transfer only this part of the original image to the other user. This feature allows the two users in the same session
to focus on the same small region of interest of the original large image and enables their interactive discussion and cooperation of the same important portion of the image. It also saves the transmission bandwidth and time by transferring only part of the image instead of the delivery of the entire large image.

This image sending procedure is simply illustrated in Figure 3.5.

### 3.2.2 Image Receiving Procedure

At the receiver side, the system allocates a TCP socket with a default port number and a memory buffer to store the receiving TCP data packets from the sender and then repeatedly execute the following:

1. wait for the next connection from the sender
2. accept and handle the connection request
3. receive data packets from the sender
4. write the received data from the buffer to an output file
5. close the connection after receiving all the data packets

The same as the sending side, at the receiver side, the system will also spawn a child process which does the receiving and parent process takes care of all other user input actions.

Currently image capability can do both the local image viewing and transmitting the whole image or some portion of the image to another end user who is in the same
session. However the image-view interaction is only supported within two point to point connected users. In the future we would like to implement the ‘one server multiple clients’ model for the image transmission. In that situation one user can transfer an image to all other two or three users in the same session through the central server. Therefore multiple physicians can examine the same medical image and discuss or consult with others through both audio and video.

This image receiving procedure is also simply shown in Figure 3.5.

![Figure 3.5 Sending and Receiving Procedures](image-url)
3.3 Distributed Image Browsing

In most cases not the entire large medical image but only a small region of the original image is interested in by physicians or scientists. So in this remote image browsing capability we would like to stress this point and provide a simple way to help the users in their cooperative session to focus on only their common region of interest. For instance, one user is interested in the region which is shown in Figure 3.6 so this user can select his or her interest part of the image and zoom in. Only this portion of the image is sent to the other user if the first user wants to share and discuss based on this part. Since the entire image does not need to be transmitted first both the transmission time and bandwidth are saved to deliver audio or video at the same time. And the most important issue: showing exactly the same image region on the two user’s screen is guaranteed.
In the interactive image browsing part, we also consider that several correlated regions of one medical image are probably all interested or some of them need to be compared to obtain some results and treatment decisions. Two related zoomed in regions of interest can both be transmitted to the other user and displayed in two separate windows such as the left hand part and right hand part as shown in Figure 3.7. So users can easily compare the two parts visually together and get ideal results.

Figure 3.6 Region of Interest Transmission
In Figure 3.7 the original image is also transmitted and displayed in another window but this is not necessary. If two split parts of one image need to be compared together these two regions can transmitted separately without delivering the original entire image. And the receiving user can open and view the two split image regions in two windows for the goal of easy comparison.

3.4 Other Medical Image Browsing Applications

In medical image viewing and sharing area, several industrial software leading corporations are also focusing on developing interactive medical image browsing systems. Let us take a look at some most advanced remote medical image browsing
applications.

NetVue is a browser based medical image and document viewer developed by Accusoft Pegasus. It enables high speed image and document streaming display and supports many image formats such as BMP, GIF, JPEG and JPEG 2000. Several basic image operations like local save, zoom and rotate are supported by the viewer. NetVue delivers images and documents through a client and server architecture and the most impressive enhancement feature of it is the flexible annotation option. This annotation feature is very useful for collaborative medical image browsing system since physicians in different locations can share their perspectives and annotations on any image or document more powerful and clearer. Our future system should also support this feature.

Another advanced remote medical image browsing system is AccuRad ImageShare Platform which is developed by Aware. This platform is a client server based comprehensive solution for fast and efficient compression, streaming and viewing of medical images. It is comprised of two parts: AccuRad ImageShare Workstation is the client part and AccuRad ImageShare Server is the server side. The image sharing server utilizes JPEG 2000 for image compression and uses the Digital Imaging and Communications in Medicine (DICOM) approved JPEG 2000 Interactive Protocols (JPIP) for image data streaming to achieve reliable interoperability between the image archive and the client image viewer. The server prepares the streaming data by creating a study manifest and DICOM header files with JPIP links to the original
image data. When a JPIP request comes the server will locate the image or a region of the image and responds accordingly with JPIP image data. AccuRad ImageShare Workstation is a universal medical image client viewer. It sends JPIP requests to the image sharing server and retrieves the DICOM header files containing JPIP URL links to image pixel data and then receives the image data to view the entire image or a region of the image.

By studying some most state of art industrial remote medical image browsing applications we can see that our prototype system which binds the medical image browsing, low latency audio, video and instant text messaging together can be extended more such as incorporating the high performance image compression technique JPEG 2000 and its Interactive Protocols (JPIP) and adding the image annotation whiteboard functionality to our existing system. These more advanced techniques and functionalities with our current low latency audio, video, image browsing and text messaging capabilities can build a more interactive system to help physicians in distributed locations collaborate and cooperate more flexible and more efficiently.

3.5 Summary

In summary, we have developed a basic remote interactive image browser which can support most popular image formats and has some useful image operation functions. The remote image browser allows users to select their own regions of interest and
zoom in based on that part of the image. We also added the image-view interaction part based on TCP protocol to our existing music telepresence system. Users can choose either to transfer the entire image or only share a portion of the image with the other user in the same session. This helps the two users focus on the same interested region of the image and improves their interactive discussion on the most important parts of the image. It also saves the network transmission bandwidth and time through delivering only a portion of the image instead of the entire image. Currently the image-view interaction only works between point to point two users’ connections and in the future we would like to improve it to support three or four users in the same session. At last we analyzed some most advanced remote medical image browsing applications and from the analysis we obtain the future development and extension directions to our prototype system.
Chapter 4

JPEG 2000

This chapter describes some important algorithms and techniques at the heart of the JPEG 2000 image compression standard based on the JPEG 2000 international standard part 1: core coding system. A comparison of the encoders of the Discrete Cosine Transformation based JPEG and its successor wavelet based JPEG 2000 image coding techniques is proposed. The software architecture and implementation of the image codec specified in the JPEG 2000 standard will also be briefly analyzed.

4.1 Introduction

JPEG 2000 is a new standard for image coding published by the JPEG committee (Joint Photographic Experts Group, ISO/IEC JTC 1/SC 29/WG 1). The core, defined in Part 1 (core coding system) of the standard is built based on the new technologies of wavelet based compression and bit-plane coding.

JPEG 2000 supports both lossy and lossless compression of single component (for instance, grayscale) and multi-component (for instance, color) image. And scalable image representation is deeply emphasized in the standard. Portions of the compressed image data may be extracted and decoded independently, to reconstruct the image at a reduced resolution, at a reduced quality within any resolution, or within
a reduced spatial region, at the desired resolution and quality. Its excellent compression performance and many attractive features enable a very wide range of applications based on JPEG 2000: from portable digital cameras to web browsing, image archiving and medical imaging [1].

4.2 Comparison of JPEG and JPEG 2000 Encoders

JPEG 2000 is inherently more complicated than its predecessor, another important image compression standard published by the same ISO working group, JPEG. Discrete Wavelet Transform (DWT) which is used by JPEG 2000 and coding of wavelet coefficients in blocks together imply a significantly higher cost in memory consumption than the baseline JPEG algorithm [2].

I will simply compare the encoder architectures and encoding steps of the JPEG and JPEG 2000 algorithms.

4.2.1 JPEG Encoder

After reading a raw image the first thing to do for the JPEG encoder is the DC-level shifting which will make the sample data have a range centered about zero (that means, from zero ~ two hundred and fifty six to negative one hundred and twenty eight ~ positive one hundred and twenty seven). Then it will do the forward components transformation which maps data from RGB (red, green, blue color model) to YCrCb (Y: luma component, Cb: blue chroma component, Cr: red chroma component). After
the forward component transformation the forward Discrete Cosine Transformation (DCT) will be applied. Then the image data is quantized. Since JPEG coding system is always used in lossy image compression quantization is an important step for the JPEG encoding. After quantization the JPEG encoder chooses Huffman coder to do its entropy coding. At last the raw image is compressed to a JPEG image format file. These encoding steps are shown in Figure 4.1.

4.2.2 JPEG 2000 Encoder

At the JPEG 2000 side, before the DC-level shifting is used the raw image is firstly partitioned into Tiles which are rectangular non overlapping blocks. This step is called tiling and all the tiles are compressed independently as though they were entirely independent images. Both the tiling and DC-level shifting are together defined as pre-processing. After pre-processing the forward components transformation is applied just like in JPEG (from RGB to YCbCr). And then the forward Discrete Wavelet Transformation (DWT) is used. The next steps are quantization which is only used in lossy image compression and two tiers encoding. Each coefficient bit in the bit-planes is coded in only one of the three coding passes which will be discussed below by the tier-1 encoder. Then JPEG 2000 encoder will use the MQ-coder, a low complexity arithmetic coder, to do the entropy coding. The final step for the JPEG 2000 encoder is the packetization or bit stream organization. The compressed data from the bit-plane coding passes are separated into packets and then the packets are multiplexed together in an ordered manner to form one code-stream [4]. The JPEG
2000 encoder architecture is also showed in Figure 4.1.

Figure 4.1 JPEG and JPEG 2000 Encoder

While the DCT based JPEG image compression technique performs very well at moderate bit rates, at higher compression ratio, image quality degrades very much because of the artifacts introducing from the block based DCT scheme. Wavelet based image coding such as JPEG 2000 provides an improvement in image quality at low bit rates because of overlapping basis function and better energy compaction property of wavelet transforms [6]. Another inherent nature - multi resolution allows JPEG 2000 to fully support the image progressive transmission.
4.3 JPEG 2000 Codec Architecture

4.3.1 Pre-Processing

The JPEG 2000 standard works on image tiles. The source image is partitioned into rectangular non-overlapping blocks in a process called tiling. These tiles are compressed independently as though they were entirely independent images. All operations, including component transform, discrete wavelet transform, quantization, and entropy coding, are performed independently on each different tile. Tiling can reduce memory requirements, and since each tile is reconstructed independently, a specific part of the image can be decoded, rather than the entire image. Each tile can be thought of as an array of integers in sign-magnitude representation. This array is then described in a number of bit planes. These bit planes are a sequence of binary arrays with one bit from each coefficient of the integer array. The first bit plane contains the most significant bit (MSB) of all the magnitudes. The second array contains the next MSB of all the magnitudes, continuing like this until the final array, which contains the least significant bits of all the magnitudes.

Before the forward DWT is applied to each tile, all image tiles are DC level shifted by subtracting the same quantity, such as the component depth, from each sample. The source sample data have a range centered about zero. DC level shifting will move the image tile to a desired bit plane, and is also used for region of interest coding, which is explained later.
4.3.2 Components Transformation

Component transformation maps the data from RGB to YCbCr. It will improve the compression performance and allow visually relevant quantization. There are two transformation methods because JPEG 2000 can do both lossy and lossless compression. On the lossy side, the codec uses the irreversible component transform (ICT) which is floating point real-to-real and used with irreversible (floating point 9/7) wavelet transform. On the lossless side, the codec applies the reversible component transform (RCT) that is integer approximation integer-to-integer and used with reversible (integer 5/3) wavelet transform for the lossless coding.

4.3.3 Discrete Wavelet Transformation (DWT)

![Figure 4.2 Tiling, DC Level Shifting, and DWT on Each Tile]

Each tile component is then decomposed using the DWT into a series of decomposition levels. Each decomposition level will contain a number of subbands as shown in Figure 4.2 [6]. These subbands contain coefficients that describe the
horizontal and vertical characteristics of the original tile component. The wavelet
transformation in JPEG 2000 compression method is one dimensional in nature.

Applying one dimensional transformation in the horizontal and vertical directions
forms two dimensional transformations. This creates four smaller image blocks; one
with both low resolutions, one with high horizontal resolution and low vertical
resolution, one with low horizontal resolution and high vertical resolution, and one
with all high resolutions. Applying the one-dimensional filters in both directions is
then done for a number of times on the low resolution image block. This procedure is
called dyadic decomposition which is shown in the Figure 4.3 [6].

If using lossy compression the DWT will be a floating point 9/7 wavelet filter which
has the best performance at low bit rate but high implementation complexity
especially for the hardware. On the other lossless side the DWT will be an integer 5/3
wavelet filter with low implementation complexity. The multi resolution nature of the
wavelet transform makes JPEG 2000 an ideal candidate for progressive transmission.

However, when wavelet filtering is applied to an input image, because the filter
coefficients are not necessarily integers, the filtered output is not integers but floating
point values. For lossless encoding, filtered coefficients should be integer values to
make the decoding process reversible.

Wavelet transformation allows lossless representation of images so progressive
transmission is easily achieved by first transmitting a lower resolution version of the
image followed by the transmission of successive details.
To perform the forward DWT, a one-dimensional subband is decomposed into a set of low pass samples and a set of high pass samples. The low pass samples are a smaller low resolution version of the original and the high pass samples are a smaller residual version of the original. The high pass samples are needed to reconstruct a perfect original set from the low pass set.

The $d$th stage DWT decomposes (analysis) its input image $LL_{d-1}$ to four subband images, which are $LL_d$, $LH_d$, $HL_d$ and $HH_d$. Each subband image has only half the height and width of the input image so $LL_1$ has the half height and width of $LL_0$ and $LL_2$ has half height and width of $LL_1$ and so on. $LL_0$ represents the original image and
LL₁ through LLₐ₋₁ are intermediate results, each will be the input to its subsequent stage [4].

Each stage may be synthesis, so the image is fully cut to 3d+1 subband images which can be shown in the Figure 4.4 [10]. If the code blocks corresponding to the r resolution detail subbands are discarded and omitting last r DWT synthesis stages, LLᵣ is rebuilt; this is a low resolution image whose dimensions are 2ʳ times smaller than those of the original image. This is the origin of resolution scalability in JPEG2000, since a reduced resolution version of the image can be reconstruct by a subset of the coded subbands.

After applying DWT the produced subband images are partitioned into smaller blocks, each of which is coded independently. Typical block sizes are 32 x 32 or 64 x 64 and each block bit stream has an embedded representation of the block’s samples.

### 4.3.4 Quantization

After transformation, a uniform quantizer with deadzone will be used to quantize the wavelet coefficients. This process will reduce the precision of coefficients and it is accomplished through dividing the magnitude of each coefficient by a quantization step size and rounding down. These step sizes can be chosen in a way to obtain a given level of quality. This operation is lossy, unless the coefficients are integers which are produced by the reversible integer 5/3 wavelet. In this lossless procedure the quantization step size is set to one so actually no quantization is done and all of the
coefficients will not be changed.

4.3.5 Tier-1 and Tier-2 Coding

Following quantization, each subband is subjected to a packet partition. Each packet includes a successively improved resolution level on one tile. The image is divided into first a low quality representation of the original, and then is kept improving until it gets to its maximum quality level. Finally dividing each packet partition location into regular non overlapping rectangles will obtain the code blocks. These code-blocks are used for employing entropy coding [4].

Entropy coding is done by the MQ coder which is an adaptive, binary arithmetic coder, characterized by multiplier-free approximation, renormalization-driven probability estimation and bit-stuffing. The entropy coding is also applied independently on each code block. This coding is carried out as context dependant binary arithmetic coding of bit planes [7]. Arithmetic coding is done through a process which scans each bit plane in a series of three coding passes. In which pass a given bit is coded in is decided based on the significance of that bit’s location and the significance of the neighboring locations. A location will be significant if a one has been coded for that location in current or previous bit plane.

The first pass in a new bit plane is called the significance propagation pass. If a bit’s location is not significant, but at least one of its eight connected neighbors is significant this bit will be coded in this pass. The second pass is the magnitude refinement pass. In
this pass, all bits from locations which became significant in a previous bit plane are coded. The third and final pass is the clean-up pass, which takes care of any bits not coded in the first two passes. All the three passes use a stripe oriented scan through code block samples as shown in Figure 4.5 [10]. Each coefficient bit in the bit-plane is coded in only one of the three coding passes. The image is ready to be stored as a compressed version of the original image after entropy coding [4].

Figure 4.5 Stripe Oriented Scan Though Code Block Samples

Let us take a look at the bit-plane: a bit plane of a digital medium such as image or sound is a set of bits having the same position in the respective binary numbers as shown in Figure 4.6 [11]. For instance, for an eight bit data there will be eight bit planes: the first bit plane includes the set of the most significant bit and the eighth contains the least significant bit. Bit plane coding is a technique for embedded coding.
The embedded block coding algorithm utilizes the same low complexity binary arithmetic coding and has a careful design of the bit-plane coding primitives. The coder provides additional advantages: memory locality, spatial random access and ease of geometric manipulation such as rotation. The blocks can be coded or decoded in parallel and in any order.

### 4.4 Summary

The current chapter compares the difference between the encoding structures of two image coding systems: Discrete Cosine Transformation based JPEG and its successor wavelet based JPEG 2000. After the comparison we described some of the most important features of the JPEG 2000 standard and analyzed the JPEG 2000 encoder step by step.

This chapter helps us to gain a basic understanding and some knowledge of the JPEG 2000 image compression technique. In next chapter we will use an available open source JPEG 2000 image compression codec implementation to have a test on its encoding and decoding time.
Chapter 5

Test Result

In this chapter two official JPEG 2000 testing images are compressed and decompressed using OpenJPEG --- an open source implementation of JPEG 2000 codec with different options.

5.1 JPEG 2000 Encoder Test

The two official testing images are firstly encoded with the following options: 1 tile and 1 precinct. Size of code block is 64 x 64 and number of resolution is 6 which is the default Discrete Wavelet Transformation decomposition level. The compression ratio of the two original images is: lossless, 10, 50, 100, 200, 500, 1000 and 2000. Encoding times are recorded below.
Table 5.1 Image 7 Sisters Encoding Time

<table>
<thead>
<tr>
<th>Compression Ratio</th>
<th>Encoding Time</th>
<th>Image Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lossless</td>
<td>5.291226 seconds</td>
<td>1.24 Megabytes</td>
</tr>
<tr>
<td>1:10</td>
<td>5.301194 seconds</td>
<td>207 Kilobytes</td>
</tr>
<tr>
<td>1:50</td>
<td>5.338188 seconds</td>
<td>41.4 Kilobytes</td>
</tr>
<tr>
<td>1:100</td>
<td>5.382181 seconds</td>
<td>20.7 Kilobytes</td>
</tr>
<tr>
<td>1:200</td>
<td>5.294196 seconds</td>
<td>10.1 Kilobytes</td>
</tr>
<tr>
<td>1:500</td>
<td>5.377183 seconds</td>
<td>4.12 Kilobytes</td>
</tr>
<tr>
<td>1:1000</td>
<td>5.312208 seconds</td>
<td>2.07 Kilobytes</td>
</tr>
</tbody>
</table>

The original size of the first testing image 7sisters is 2.02 megabytes. The lossless encoded image is 1.24 megabytes, about half of the original size. For medical images, only lossless compression or with compression ratio 10 could be accepted. For other uses such as web applications compressed images with compression ratio 10, 50, 100 and even 200 can be accepted. Compressed images with compression ratio 500 and 1000 have too much losses.
Figure 5.1 Original BMP image size: 2.02 MB (2,125,878 bytes)

Figure 5.2 JPEG2000 lossless image size: 1.24 MB (1,301,019 bytes)
Figure 5.3 JPEG2000 1:10 image size: 207 KB (212,026 bytes)

Figure 5.4 JPEG2000 1:50 image size: 41.4 KB (42,440 bytes)
Figure 5.5 JPEG2000 1:100 image size: 20.7 KB (21,255 bytes)

Figure 5.6 JPEG2000 1:200 image size: 10.1 KB (10,364 bytes)
Figure 5.7 JPEG2000 1:500 image size: 4.12 KB (4,224 bytes)

Figure 5.8 JPEG2000 1:1000 image size: 2.07 KB (2,121 bytes)
The original size of the second testing image Waltham is 26.7 megabytes. The lossless encoded image is 8.81 megabytes, about one third of the original size. For medical images, only lossless compression or with compression ratio 10, 50 could be accepted. For other uses such as web applications compressed images with compression ratio 10, 50, 100, 200 and even 500 can be accepted. Compressed images with compression ratio 1000 and 2000 have too much losses.

From table 5.1 and 5.2 we can see that compression ratio is not a critical factor for the encoding time. For actual web delivery, if we want to transmit the original first testing image 7sisters whose size is 2.02 megabytes and we assume that network speed is 0.2
megabytes per second. So the transmission time would be 2.02/0.2, which is 10.1 seconds. If we compress the image with lossless option first and then we deliver it with the same network speed the amount of time would be lossless compression time 5.30 seconds plus 1.24/0.2 seconds, which is 11.5 seconds.

If we want to transmit the original second testing image Waltham whose size is 26.7 megabytes and we assume that network speed is also 0.2 megabytes per second. So the transmission time would be 26.7/0.2, which is 133.5 seconds. If we compress the image with lossless option first and then we deliver it with the same network speed the amount of time would be lossless compression time 35.4 seconds plus 8.81/0.2 seconds, which is only 79.45 seconds.

From this calculation we can discover that the larger image we want to deliver the more time we can save if we compress the image using JPEG 2000 first (Although decoding also requires time compressing the large images first still can save the transmission time and the time amount). Since medical images are often 20 megabytes or more using JPEG 2000 to compress the medical images first can save lots of delivery time of medical images for hospitals.
Figure 5.9 Original BMP image size: 26.7 MB (28,080,054 bytes)

Figure 5.10 JPEG2000 lossless image size: 8.81 MB (9,248,070 bytes)
Figure 5.11 JPEG2000 1:10 image size: 2.67 MB (2,807,643 bytes)

Figure 5.12 JPEG2000 1:50 image size: 548 KB (561,596 bytes)
Figure 5.13 JPEG2000 1:100 image size: 274 KB (280,658 bytes)

Figure 5.14 JPEG2000 1:200 image size: 137 KB (140,397 bytes)
Figure 5.15 JPEG2000 1:500 image size: 54.8 KB (56,141 bytes)

Figure 5.16 JPEG2000 1:1000 image size: 27.3 KB (28,030 bytes)
5.2 JPEG 2000 Decoder Test

At the decoder side, JPEG 2000 official testing image Waltham, originally a bitmap is firstly encoded with the following options: 1 tile lossless, so the Discrete Wavelet Transformation (DWT) uses the integer 5/3 wavelet filter and the compressed file extension is J2K. Size of code block is 64 x 64 and number of resolution is 6 which is the default DWT decomposition level. The amount time of decoding and the time used by each decoding step are recorded.

Based on the open source implantation of JPEG 2000 image compression codec - OPENJPEG, I made some modifications to the project. From the source code we can cut the decoder to several parts and each part will do a specific function to decode the
compressed image data. For instance, the first part of the decoder will do the tier-2 coding and the next part does the tier-1 three passes and entropy coding and then the third part will utilize the inverse Discrete Wavelet Transformation and so on. Before each decoding step begins I let the software return a time from the operating system and after the current decompression step finishes the software will return another system time. Through calculating the difference between the two times we can obtain the time which is consumed by each decoding step. And they are recorded in Table 5.3 (Time unit is second):

<table>
<thead>
<tr>
<th>Operation</th>
<th>Time(seconds)</th>
<th>Time(seconds)</th>
<th>Time(seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tier-2 coding</td>
<td>0.110983</td>
<td>0.110983</td>
<td>0.102985</td>
</tr>
<tr>
<td>Tier-1 coding</td>
<td>27.861765</td>
<td>30.964293</td>
<td>29.298611</td>
</tr>
<tr>
<td>Inverse DWT</td>
<td>1.950702</td>
<td>1.943705</td>
<td>1.960699</td>
</tr>
<tr>
<td>Multi-component T</td>
<td>0.205969</td>
<td>0.189971</td>
<td>0.200969</td>
</tr>
<tr>
<td>Time amount</td>
<td>30.129419</td>
<td>33.208952</td>
<td>31.563264</td>
</tr>
</tbody>
</table>
Table 5.4 Average Time of Each Decoding Step

<table>
<thead>
<tr>
<th>Operation</th>
<th>Average Time(seconds)</th>
<th>Time Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tier-2 coding</td>
<td>0.108317</td>
<td>0.34%</td>
</tr>
<tr>
<td>Tier-1 coding</td>
<td>29.37489</td>
<td>92.86%</td>
</tr>
<tr>
<td>Inverse DWT</td>
<td>1.951702</td>
<td>6.06%</td>
</tr>
<tr>
<td>Multi-component T</td>
<td>0.1989697</td>
<td>0.63%</td>
</tr>
<tr>
<td>Time amount</td>
<td>31.633878</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.18 JPEG 2000 Lossless Image Waltham Each Step Decoding Time

From table 5.3, we can find out that tier-2 coding only consumes about 0.1 seconds and the multi-component transformation uses approximately 0.2 seconds. The inverse
Discrete Wavelet Transformation takes about 2 seconds. All these three decoding steps only take about less than 2.3 seconds in all. Tier-1 coding will consume from 28 to 30 seconds in the three tests.

Table 5.4 calculates the average time of each decoding step and the percentage of their times. On average tier-2 coding will use 0.108317 seconds which is 0.34 percent of the decoding time amount. Multi-component Transformation takes 0.1989697 seconds which is 0.63 percent and the inverse DWT needs 1.951702 seconds to finish that is 6.06 percent of the time amount. Tier-1 coding consumes the most time: 29.37489 seconds on average and it occupies almost 93 percent time needed to decode a JPEG 2000 image.

This test result also gives us a clue for the reason why compression ratio is not a critical factor to affect the coding time. Since Tier-1 coder which does the three passes bit plane coding: significance pass, refinement pass, cleanup pass and the entropy coding is applied to all the input image source data and it consumes the most time of the image coding process. While we code an image with different compression ratios the Tier-1 coding procedure has no significant difference so the coding time of different compression ratios also does not change significantly.

The average amount time of the whole decoding process which includes reading the input JPEG 2000 image file in memory is about 32 seconds. Since both the compression ratio and the coding time are critical we need to consider both if we want to apply JPEG 2000 compression technique to some time demanding or real time
projects. For instance, a JPEG 2000 image may have better quality, larger resolution and smaller size than a JPEG file or other image format files but probably decoding a JPEG 2000 image file will also consume much more time than others. This tradeoff between the compression latency and the transmission latency will need more comparisons and tests between JPEG 2000 and JPEG or other image compression algorithms to be found out in specific circumstances.

Figure 5.18 shows the direct visual consuming time comparison between the four main steps in decoding a JPEG 2000 image. From this chart we can know that tier-1 coding which includes three passes: significance pass, refinement pass, cleanup pass and the entropy coding uses more than 90 percent of the decoding time. It also gives us a clue that if we want to decrease the time to decode a JPEG 2000 image file tier-1 coding algorithms have the first priority to be optimized.

5.3 Adoption of JPEG 2000

If we would like to adopt the new image compression technique JPEG 2000 to a distributed interactive system several aspects including both merits and drawbacks need to be fully considered and evaluated:

1. JPEG 2000 encoder can do lossless image compression and performs very well at low bit rate. From our test we can also see that compression ratio does not affect the encoding time much.

2. JPEG 2000 lossless compression can help a lot on saving the image storage space
and image transmission time and bandwidth by compressing large images to a quite reduced size.

3. The encoding and decoding time of a JPEG 2000 image are also critical to decrease the image transmission delay since the whole delay time contains both the network delay and the image coding delay. JPEG 2000 image coding system can probably reduce the network delay time but also may require more time to decode a received image than other image coding methods.

5.4 Summary

From the experimental tests in this chapter we can know that lossless or nearly lossless image compression by JPEG 2000 can help to save lots of transmission time for large images’ transmission, especially for several types of extra large images such as medical images or satellite images whose sizes has increased to hundreds of megabytes or even more than a gigabyte. Another discovery of the JPEG 2000 standard is the tier-2 coding step including three passes: significance pass, refinement pass, cleanup pass and the entropy coding consumes the most time in the decoding process. If we want to improve the efficiency of JPEG 2000 and its progressive and interactive further uses the tier-2 coding has the first priority to optimize.
Chapter 6

Conclusions and Future Works

6.1 Conclusions

In this thesis remote interactive image browsing capability is developed to be added to our low latency music telepresence system. The new image capability does not have any impact on the existing music (audio), video and text messaging features and their transmission channels. The image capability has two main parts: image viewing part and distributed image-view interaction part. Currently the image viewing capability supports most dominant image formats such as JPEG, PNG, BMP and GIF and has some basic image operation functions. The most useful functionality of image viewing capability is to allow users to select their own regions of interest and then create a sub image which zooms in based on that region.

The image-view interaction capability is based on the TCP protocol so the transmission reliability is guaranteed and it works between two point to point users in one session right now. Image-view interaction capability enables a user to transfer an entire image or only to transmit a portion of the image to the other user in the same session. Transferring only some certain part of the image helps the two users to focus on the same interested region of the image and improves their interactive discussion or cooperation on the most crucial parts of the image. Delivering only a portion of the
image but not the entire image also saves the network transmission time and bandwidth.

We also compared the major differences and encoding features of the DCT based JPEG and Wavelet based JPEG 2000 image compression techniques. Experimental tests of the JPEG 2000 encoder and decoder are proposed using the open source JPEG 2000 codec implementation – OPENJPEG project. From the test result we discover that at the encoder side compression ratio does not have a distinct effect on the encoding time and at the decoder side tier-1 coding which includes three passes: significance pass, refinement pass, cleanup pass and entropy coding consumes the most about 93 percent of the decoding time. So if we want to improve the efficiency of JPEG 2000 image compression codec the first priority of optimizing the implementation is probably given to tier-1 coding step.

6.2 Contributions

1. We introduced a new capability into our music telepresence software: remote interactive image browsing capability which has the local image viewing part and the distributed image-view interaction part.

2. We implemented the local image viewing part with the foundation of Linux GTK+ library to support most popular image formats and several basic image operations such as zoom in, zoom out and region of interest selection.

3. We implemented the distributed image-view interaction part by using TCP
protocol to provide reliable image transmission and interactive image browsing between two users in one remote session.

4. The new image capability with the existing capabilities: instant text messaging, low latency audio and video constructs an interactive environment for users to do the distributed collaborative image browsing.

5. We evaluated and analyzed the new image compression standard JPEG 2000 such as its compression ratio, encoding and decoding time by using an available open source JPEG 2000 codec implementation – OPENJPEG project. Our testing results will help applying JPEG 2000 to a distributed collaborative system.

### 6.3 Future Works

Our future objectives include incorporating the JEPG 2000 image compression codec, especially the decoder to our image viewer and optimizing the JPEG 2000 image compression implementation. We probably will improve the efficiency and extend the functionality of both encoding and decoding processes to make the future image transmission more flexible and progressive.

Future development work also includes more image operation functions for the image viewing capability and improves the image-view interaction capability to support among three or four users in the same session through one central server.


[5] OpenJPEG free open source project, Communications and Remote Sensing Laboratory, University Catholique de Louvain (UCL), Belgium


