A NEW MULTI-FRAMES WATERMARKING TECHNIQUE FOR TAMPER DETECTION AND RECOVERY

KHOR HUI LIANG

Doctor of Philosophy
(Computer Science)

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : KHOR HUI LIANG

Date of Birth : 16th July 1971

Title : A New Multi-frames Watermarking Technique for Tamper Detection and Recovery

Academic Session :

I declare that this thesis is classified as:

- ☐ CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)*
- ☐ RESTRICTED (Contains restricted information as specified by the organization where research was done)*
- ☑ OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

(Student's Signature)	(Supervisor's Signature)
710716-10-5462	Dr.Liew Siau Chuin
New IC/Passport Number	Name of Supervisor
Date:	Date:

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Doctor of Philosophy in Computer Science

____

(Supervisor's Signature)

Full Name : Dr. Liew Siau Chuin

Position : Senior Lecturer

Date :

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

____

(Student's Signature)

Full Name : KHOR HUI LIANG

ID Number : PCC12005

Date : 6 May 2017

A NEW MULTI-FRAMES
WATERMARKING TECHNIQUE FOR
TAMPER DETECTION AND RECOVERY

KHOR HUI LIANG

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Doctor of Philosophy (Computer Science)

Faculty of Computer Systems and Software Engineering

UNIVERSITI MALAYSIA PAHANG

MAY 2017

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Liew Siau Chuin,for his invaluable guidance, continuous encouragement and constant support in making this research possible. I also will like to thank KPT MyBrain15-Myphd Units for three years of full-time scholarship. My greatest gratitude is extended to my mother and brothers for their constant support and encouragement. Lastly, special thanks should also go to IPS staff for their support for the entire duration of this study.

ABSTRAK

Pada masa kini kecaraan imej perubatan telah berdigital dan diseragamkan ke dalam DICOM format yang mempamerkan ciri-ciri berbilang-bingkai; oleh demikian imej perubatan yang berbilang-bingkai telah digunakan secara meluas dalam amalan klinikal tetapi kebanyakan kajian untuk penandaan air hanya dijalankan ke atas sebingkai imej perubatan kerana tujuan mereka adalah di atas intrinsik algoritma skim penandaan air. Berbilang-bingkai penandaan air boleh diproseskan secara berurutan tetapi ia memakan masa. Proses penandaan air adalah kerja latar belakang sebelum proses diagnosis bermula oleh itu masa pemprosesan perlu dipendekkan untuk mengurangkan masa menunggu dan masa rawatan pesakit dalam persekitaran hospital. Objektif kajian ini adalah untuk menyelesaikan kekangan masa proses berbilang-bingkai penandaan air. Proses penandaan air pelbagai bingkai berurutan boleh mempercepatkan dengan dua cara, cara pertama ialah mereka bentuk penandaan air cepat skim algoritma (ROI-DR) dalam rangka tunggal imej perubatan, dan cara kedua ialah membangunkan pelbagai bingkai baru penandaan air skim dengan menambah komponen keselarian ke berurutan pelbagai bingkai penandaan air proses. ROI-DR mempercepatkan masa pemprosesan penandaan air dengan mengeluarkan ciri-ciri penyetempatan contengan dan menggantikan seluruh imej ROI segera dengan imej ROI yang diekstrak pada pertemuan pertama dengan pixel yang berconteng. Keputusan eksperimen menunjukkan bahawa ROI-DR mempunyai nilai PSNR (~ 48 dB) yang lebih kurang sama dan masa pemprosesan yang lebih pendek berbanding dengan TALLOR dan TALLOR-RS skim penandaan air. ROI-DR mempunyai purata faktor penaikan kelajuan 22.55 dan 26.65 dalam proses benaman penandaan air; 21.89 dan 42,79 dalam proses pengesahan penandaan air berbanding dengan TALLOR dan TALLOR-RS masing-masing. Rangka kerja selari telah dibangunkan dengan menggunakan teknologi berbilang teras di mana imej perubatan telah dibahagikan antara teras dan melaksanakan proses penandaan air dengan selari. Dalam pengujian untuk kecekapan rangka kerja, dua skim penandaan air yang berbeza (TALLOR dan ROI-DR) telah dimasukkan ke dalam rangka kerja selari. Keseluruhan faktor penaikan kelajuan bagi ROI-DR dan TALLOR adalah 0.91 ~ 2.16 dan 5.21 ~ 6.60 masing-masing. Faktor penaikan kelajuan yang lebih tinggi dalam TALLOR menunjukkan bahawa prestasi skim penandaan air pelbagai bingkai cadangan dipengaruhi oleh kerumitan algoritma skim penandaan air yang terpakai dan bilangan bingkai dalam imej perubatan diproses.

ABSTRACT

Nowadays, medical image modalities have been digitalized and standardized into DICOM format which exhibit multi-frames characteristics; hence, multi-frames medical images have been widely used in clinical practice but it has been found that most of watermarking researches were conducted on single-frame medical image because their focus was on the intrinsic algorithm of watermarking scheme. Multi-frames watermarking could be processed sequentially but it is time-consuming. Watermarking process is the background work before diagnosis process starts; therefore, the processing time should be shortened in order to minimize patient's waiting and treatment time in a hospital environment. The objectives of the research are to resolve the time constraint of sequential multi-frames watermarking process. The sequential multi-frames watermarking process could be speed up by two methods, first method was designed a speedy watermarking scheme algorithm (ROI-DR) in a single frame of medical image, and the second method was developed a new multi-frames watermarking scheme by adding parallelism component into sequential multi-frames watermarking process. ROI-DR speed up the watermarking processing time by eliminating the tamper localization features and replacing the whole ROI image immediately with the extracted ROI image at the first encounter of tampered pixel. The experiment results show that ROI-DR has similar PSNR value (~ 48 dB) and shorter processing time as compared to TALLOR and TALLOR-RS watermarking scheme. ROI-DR has an average speed up factor of 22.55 and 26.65 in watermarking embedding process; 21.89 and 42.79 in watermarking authentication process relative to TALLOR and TALLOR-RS respectively. The proposed multi-frames watermarking scheme was developed by utilizing multicores technology where medical images have been divided among cores and perform watermarking process in parallel. Two different watermarking schemes (TALLOR and the ROI-DR) have been incorporated into the proposed multi-frames watermarking scheme. Overall speed up factor for ROI-DR and TALLOR range from 0.91 ~ 2.16 and 5.21 ~ 6.60 respectively. The higher speed up factor in TALLOR indicated that the performance of the proposed multi-frames watermarking scheme is affected by the algorithm complexity of the adopted watermarking scheme and the number of frames in medical images processed.

TABLE OF CONTENT

CHAPTER 4 RESULT

CHAPTER 5 CONCLUSION AND FUTURE WORK

APPENDICES

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CRC	Cyclic Redundancy Check
CT	Computerized Tomography
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DICOM	Digital Imaging and Communications in Medicine
DWT	Discrete Wavelet Transform
FH	High Frequency Band
FL	Low Frequency Band
FM	Medium Frequency Band
GPU	Graphics Processing Unit
HIS	Hospital Information System
HH	Diagonal Component
HL	Horizontal Component
IVUS	Intravascular Ultrasound
JPEG	Joint Photographic Experts Group
LH	Vertical Component
LL	Low Resolution Component
LSB	Least Significant Bit
MATLAB	Matrix Laboratory
MD5	Message-Digest Algorithm 5
MFD	Multi-Frames DICOM
MJS	MATLAB Job Scheduler
MRI	Magnetic Resonance Imaging
PACS	Picture Archiving and Communications System

PSNR	Peak Signal-to-Noise Ratio
RLE	Run Length Encoding
ROI	Region of Interest
ROI-DR	ROI Based Tampered Detection and Lossless Recovery
RONI	Region of Non-Interest
SHA	Secure Hash Algorithm
TALLOR	Tamper Localization and Lossless Recovery
TALLOR-RS	Tamper Localization and Lossless Recovery with ROI Segmentation
US	Ultrasound
XA	X-ray Angiography

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

This chapter introduces the background of the digital watermarking scheme in section 1.2. The problem statement of this research has been described in section 1.3. The research scope has been described in section 1.4. In section 1.6, a set of research outcomes are derived corresponding to the objectives in section 1.5. Section 1.7 presents the organization layout of the thesis.

1.2 BACKGROUND

With technological advancement in communications network and the Internet, the exchange of medical images between hospitals has become a common practice nowadays. It has facilitated teleconferences among clinicians, interdisciplinary exchanges between clinicians and radiologists for consultative purposes or discussion of diagnostic and therapeutic measures, and for distant learning of medical personnel (Memon, N.A. et al., 2009). Thus, medical images are exposed to an open network, where sensitive patient information is vulnerable to hackers' attack. Possible security breaches such as tampering of images include false data which may lead to wrong diagnosis and treatment. Consequently, medical image security has become an important issue that needs to be addressed.

All medical modalities such as X-rays, ultrasounds, and MRI have been digitalised in DICOM format in hospital and healthcare clinical practice where DICOM is a software integration standard that is widely used in Medical Imaging. All digital images are built from pixels including DICOM file, which could be easily tampered with using image processing tools and hence arise the need for digital watermarking as a layer of protection where its algorithm is developed by manipulating image pixels data elements and at the same time retains the medical image integrity. The integrity of medical image could be achieved in three levels (Anon, 2008):

1. tamper detection,
2. tamper localization, and
3. possible recovery by approximating the tampered region.

1.3 PROBLEM STATEMENT

Digital watermarking in medical images serves as a layer of protection against tampering. In medical image watermarking, a region of interest (ROI) is defined but untouched and the generated watermark is embedded in the region of non-interest (RONI). It is because ROI is the significant area for clinical diagnosis and modification was restricted to prevent any misdiagnosis from occurring. ROI bits was read and embedded into RONI as watermark. In watermarking authentication process, the embedded watermark will be extracted for tamper detection and recovery purposes.

Several studies (Guo, X. and Zhuang, T., 2008; Liew, S.C. et al., 2013; and Rayachoti et al. 2015) had divided a medical image into protection zone (ROI) and insertion zone (RONI) in their proposed watermarking schemes. All these watermarking schemes algorithm focused on a single frame medical image but most of the medical modalities are in multi-frames form, such as X-Rays, Ultrasounds, CT (Computed Tomography), and MRI (Magnetic Resonance Imaging). Multi-frames medical image is defined as an image whose pixel data consists of a sequential set of individual image pixel frames, and it is transmitted as a single contiguous stream of pixels (Mahmoud, I. et al, 2013).

Wenbo, D. et al. (2012) had researched on multi-frames watermarking scheme by exploited the 3-D property of volumetric (multi-frames) DICOM images and created an improved version of dual-layer watermarking scheme that developed by Tan, C.K. et al. (2011). They had utilized the advantage of 3-D property and manipulated them into their watermarking scheme algorithm. The limitation of theirs algorithm was frame-dependency, which means a specific extraction of watermarked medical images was not allowed. In order to achieve frame independency between watermarked medical images, the straight forward method will be sequential watermarking process in which watermark was embedded into medical images frame by frame sequentially. It is easy to migrate watermarking scheme from single frame to multi-frames environment by using a control loop, such as for loop to perform watermarking process on medical images sequentially but it may be time-consuming; for example the average processing time of TALLOR watermarking scheme (Liew, S.C. et al., 2013) was reported as 20.13 seconds per frame and it would be 20.13 minutes for 60 frames of medical images excluding the initial configuration and set up time.

Chen, B.L. et al (2010) and Najmuddin, A. et al (2010) have stated that waiting and treatment time are important determinants of patient satisfaction and service quality, and it would be beneficial for patients and hospital by reducing the waiting time. As watermarking process is the background work before diagnosis process starts, therefore, the processing time should be shortened in order to reduce waiting and treatment time. The processing time for sequential multi-frames watermarking process increased as the frame size of medical images increased. In order to reduce the processing time, two methods has been proposed.

The first proposed method is optimize or revamp the existing watermarking scheme algorithm to make it lighter and faster on a single frame so it will be faster in sequential multi-frames watermarking process as well. It has been found out that TALLOR watermarking scheme (Liew, S.C. et al., 2013) has bundled localization and recovery function into one process where it localized and recovered the tampered ROI pixel one by one consecutively until the end of ROI pixel, the checking process will be performed until the last of ROI pixel even if it is not tampered with which is time-consuming; therefore, in order to speed up the process, it is proposed to unbundled the tamper localization and recovery function by eliminating the tamper localization features and replaced the whole ROI image with the extracted ROI image at the first encounter of tampered pixel.

The second proposed method is to develop a new multi-frames watermarking scheme, which partitioning the sequential watermarking problem into smaller and manageable parts that can be performed in parallel. Wenbo, D. et al. (2012) had recommended using the power of parallel computing to further improve the volumetric watermarking processing time. Parallel computing is the simplest approach to leveraging multi-cores processor. The proposed parallel watermarking scheme requires little effort to separate the ultrasound medical images into a number of parallel tasks due to the low dependency (or communication) between those parallel tasks. By utilizing the power of parallel computing in multicores technology, a volumetric of medical images could be divided automatically according to the number of cores available in the processor and the divided set of medical images is then distributed into each core respectively to perform watermarking process in parallel.

Recent research demonstrates how parallel computing utilizing the power of multicores architecture has improved the performance in medical image processing such as image registration and image segmentation. Sanjay, S. et al. (2014) has provided a comprehensive review of the existing literature available on Image registration methods based on parallel computing in multi core architecture. Sanjay, S. et al. (2015) has proposed a segmentation approach by using multi core technologies. Parallel processing on multicores also has been researched on other areas besides medical image processing such as Amit, B. and Brijendra, K.J. (2015). Their research showed an improvement in lexical analysis phase by exploiting the inherent parallel processing capability of multi-core machines. All of these researches which utilized parallel computing on variety fields have achieved a significant speedup in their result, and this indicated that the proposed multi-frames watermarking scheme could improve performance and resolve the time constraint of sequential multi-frames watermarking process.

In summary, the first proposed method solve the time constraint of sequential watermarking process by having a speedy watermarking algorithm in a single frame, so it will speed up in sequential multi-frames watermarking process. The second proposed method solve the time constraint of sequential problem by incorporating parallelism into multi-frames watermarking scheme. Lastly, an integration of both methods expected to further improve the speedup factor.

1.4 SCOPE

As for the first proposed method, which is to develop a new algorithm of watermarking scheme, there was a limitation whereby the ROI region size is restricted by the embedding capacity of RONI. The segmentation of ROI and RONI will not able to cover the whole medical image and those areas that located outside the ROI and RONI will be untouchable by the watermarking scheme, in other words, any tampering occurred on those areas will go undetected by watermarking scheme. Therefore a careful selection of ROI is important.

As for the second proposed method, such as using the power of parallel computing to further improve multi-frames watermarking processing time. The watermarking processing time will depending on the number of cores available on the system. The maximum number of parallel threads cannot exceed the number of cores available on the system. The performance gain obtained by using multiple cores on a single system is also limited and varied, depending on the specific computation and the data size (Saddharth, S. et al., 2010). In this research, a multicores computer will be used to test the watermarking process on various lengths of ultrasound medical images, such as the three different sources of 15 and 30 frames of ultrasound medical images used to test whether or not different sources of same length will have impact on watermarking process performance. 15, 25, 30 and 79 frames of medical images are used to test whether the watermarking processing time is proportional to the size of medical images and whether the parallelism has significantly reduced the processing time.

In order to fulfil the integrity requirement of watermarking scheme in multi-frames environment, the watermarked image produced by parallel watermarking process should have visual quality similar to its original version, and it should be robust to tampering and have a significant improvement in elapsed time as compared to the sequential version.

1.5 OBJECTIVES

i. To develop an efficient tamper detection and loss-less recovery watermarking scheme.
ii. To implement multi-frames medical images watermarking scheme
iii. To evaluate efficiency of the proposed watermarking scheme

1.6 RESEARCH OUTCOME

i. The development of an efficient watermarking scheme on a single frame medical image and generation of comparative result of the proposed watermarking scheme and previous researched watermarking schemes.
ii. The implementation of a multi-frames medical images watermarking scheme
iii. An analytical result on performance of proposed watermarking scheme.

1.7 THESIS ORGANIZATION

The thesis has been divided into the following chapters:-

Chapter 1: Introduction - This chapter introduces the background of the research, justifies the problem statement, and describes the research scope, objectives and outcome.

Chapter 2: Literature Review - This chapter is a literature review on digital watermarking scheme on medical images, understand the watermarking requirement for medical images, investigate the multi-frames DICOM attributes, explore the parallel computing concept and review existing research on parallel computing.

Chapter 3: Methodology - This chapter presents research methodology, algorithm design, implementation details and experiment design of ROI-DR on a single frame and multi-frames medical images.

Chapter 4: Result - This chapter presents the experiment results analysis and discussion for ROI-DR and Multi-frames watermarking scheme.

Chapter 5: Conclusion – This chapter describes contributions and limitations of this research, and proposes future work for this research.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

Technological advancement in communications network has facilitated medical images exchange between hospitals for consultative and diagnostic purposes (Memon, N. A. and Gilani, S.A.M., 2008). Thus, medical images are exposed to cyber threats such as photo editing on medical image which may cause a misdiagnosis and wrong treatment. Consequently, the security of medical image has become an important issue that needs to be addressed.

All medical image modalities such as X-Rays, Ultrasounds, CT (Computed Tomography), and MRI (Magnetic Resonance Imaging) have been digitalised and standardized in DICOM format (which will be described further in sections 2.2) which exhibits the digital image characteristics such as pixel-oriented, multi-frames attributes and enables image file to be transfer via network or Internet and hence medical images are exposed to security risk such as image tampering which could not be tolerated for clinical diagnosis purposes. Watermarking scheme is developed to protect medical image from tampering. It manipulated pixel properties such as storing the compressed ROI bits into RONI's LSB in watermarking embedding process and it uses hash algorithm in watermarking authentication process (compression technique and hash algorithm will be discussed in sections 2.7 and 2.8 respectively).

A fundamental understanding of watermarking and its requirement for medical images are required to develop an ideal watermarking scheme (which will be described further in sections 2.3 and 2.4). A review on existing watermarking schemes has identified a few limitations (which will discuss in section 2.5) such as (1) most watermarking schemes were lacking of tamper recovery features; (2) Tamper localization function was time-consuming and (3) most researches were focused on watermarking intrinsic algorithm and hence only conducted on a single frame. All these limitations have led to research objectives 1 and 2, which are to develop a speedy watermarking scheme on a single frame medical image and incorporate parallelism components into the proposed multi-frame watermarking scheme. The parallelism is used to overcome the time consuming problem in sequential watermarking processing. An understanding of parallelism concept and its performance measure - speedup are required before exploration on the parallel computing with multicores technology (which will describe in section 2.9, 2.10 and 2.11). A review on the existing researches on parallel computing (which will be discuss in section 2.12) has shown that a significant speedup gain in their proposed parallelism system was theirs main contribution, and the speedup was measured by diving execution time of parallel mode with sequential mode.

2.2 DICOM IMAGES

Hospital Information System (HIS) and picture archiving and communication system (PACS) based on Digital Imaging and Communication in Medicine (DICOM) standard, which defines a set of security profiles that application entities should be complied with during the exchange of medical data. These profiles include secure use profiles, secure transport connection profiles, digital signature profiles, and media storage security profiles (Anon, 2006). PACS provides information or functions as following (Coatrieux, C. et al., 2000):

1. Picture viewing at remote workstations for consultation, diagnostic and reporting purposes.
2. Store information in magnetic or optical media.
3. Communications network, and
4. Serve as a modality interfaces which connected Healthcare Facility and Departmental Information Systems (HFDIS) and perceived as one integrated system.

All medical modalities have been digitalised in hospital and healthcare clinical practice and hence arise the need for DICOM images. DICOM is a software integration standard that is widely used in Medical Imaging such as X-Rays, Ultrasounds, CT

(Computed Tomography), and MRI (Magnetic Resonance Imaging). DICOM differs from other image formats in that it groups information into data sets (Dandu, R.V., 2012). DICOM has greatly improved communication among different medical equipment and reduced the cost and efforts in hardware and software integration problems.

The DICOM image file consists of a header that stores patient's information and image data. DICOM standard proposes to verify the authenticity of medical images by using digital signature and embedding the signature in the header. However, the authenticity of DICOM medical images can be compromised for the following reasons:-

1. The header of DICOM image file could be changed for identification for insurance claim.
2. A homonym could occur (two datasets have a same name but different content) since digital signature relies on the strength of cryptographic hash function (Tan, C.K. et al., 2011).
3. Digital signature does not provide tamper localization capability.
4. Image pixel data alone is not protected using the DICOM mechanisms if the pixel data are separated from the DICOM header.

Different medical images modalities such as MRI, ultrasounds and X-rays could be easily tampered or modified by image processing tools; therefore, a need for security protection and authentication in medical images have to be addressed (Fotopoulos, V., et al., 2008). Medical image integrity could be maintained in three stages (Anon, 2008):

1. Detect the tampered areas in medical images,
2. Localize the tampered areas, and
3. Recover the tampered areas.

2.2.1 The Heart of DICOM Images – Pixel Data Elements

All digital images are built from pixels including DICOM file; therefore, digital watermarking algorithm are manipulated image pixels data elements such as "Rows", "Columns," and "BitsStored" which will provide information that is required in programming. The commonly used pixels data elements are described as following (Jeff, M., 2002):-

- Rows and columns - define the dimension of the image size where rows are the height and columns are the width of the image.

- Sample per pixel - define the number of colour channels. The sample per pixel is set to 1 for grayscale images such as CT and MR whereas for colour images the sample per pixel is set to 3 for the RGB (red, green and blue colour channels); The sample per pixel for grayscale images (such as CT or US) is usually "MONOCHROME2" where 0 is represented as Black whereas “MONOCHROME1” is for some fluoroscopic images where 0 is represented as White;
- Bits Allocated –the total of buffer space allocated for every sample in bits.
- Bits stored –the total number of bits allocated are actually used.
- High Bit - defines the alignment of bits stored within the bits allocated.
- Number of Frames - computed the total number of frames in the image and this data elements are used to create multi-frame image objects in DICOM file.

A single frame DICOM image is defined as a single frame of medical image with a certain dimension (rows x columns), such as 640 x 476 means that image frame size is 640 pixels width and 476 pixels height. Multi-frames DICOM images is defined as multiple frames of medical images of the same size combine together and form a DICOM file. The combination of frames enable medical images to be viewed in a sequence like an animated gif, it usually called as cine loop and it facilitated doctor to spot abnormalities, such as tumours in the medical images. The details of multi-frames DICOM will be described in section 2.2.2.

2.2.2 Multi-frames DICOM

Nowadays, most medical images are stored as a set of single-frame composite DICOM data object that consists of a number of attributes, including items such as name, ID, image pixel data and etc. The advantage is that DICOM embedded patient ID within the header, and it could never accidentally detached from image. In the similar way, JPEG file also contains tags that describe the image properties. Multi-frames DICOM images could be formed by combining multiple DICOM images, but the same header data will be repeated for every DICOM image, which increased the data size and parsing overhead. This problem has been resolved by a creation of multi-frames DICOM (MFD) object, where it combined all DICOM images of the same series into a single DICOM object and only one header data was given. The same series means the DOCIM images were come from same machine and a same organs scan of a patient. For many medical imaging modalities, such as ultrasound and MRI medical image, a single image DICOM file can have only one attribute containing pixel data. However, the attribute may contain multiple "frames", allowing storage of cine loops or other multi-frame data. For example, ultrasound medical image is a multi-dimensional multi-frame image, this means that three- or four-dimensional data can be encapsulated in a single DICOM object. The MFD object concatenate the pixel data of all frames within the series and store it in a single-pixel data element. In other words, a MFD image is defined as an image whose pixel data consists of a sequential set of individual image pixel frames, and it is transmitted as a single contiguous stream of pixels. MFD have some limitation compare to a single-frame image series where all frames in the MFD must have the same size (Mahmoud, I. et al, 2013).

Table 2.1

Multi-frame Module Attributes

Attribute Name	**Tag**	**Type**	**Attribute Description**
Number of Frames	(0028, 0008)	1	Number of frames on a Multi-frame Image
Frame Increment Pointer	(0028, 0009)	1	Contain the data element tag of the attribute that is used as the frame increment in multi-frame pixel data

Source: DICOM Standards (2013)

Table 2.1 specifies the attributes of a multi-frame pixel data image. A Data Element Tag is represented as (xxxx,yyyy), where xxxx equates to the group number and yyyy equates to the element number within that group. Data Element Tags are represented in hexadecimal notation, as given in the DICOM specification. Image type identifies the image source, value 1 represented original image source, and value 2 represented derived image where an image pixel values have been derived from other images. Since only one data header is given for a MFD object, thus frame headers do not exist within the data stream. All frames were related to the first frame in the MFD where the data header was located. The total number of frames contained within a MFD is expressed as Number of Frames (0028, 0008). The frames within a MFD was treated as a logical sequence. The frame sequence number was determined by the Frame Increment Pointer (0028, 0009).

Frame Increment Pointer cannot be null and it is mandatory even if only a single frame is existed in MFD (DICOM Standards, 2013).

2.3 DIGITAL WATERMARKING

Digital watermarking insert information directly into medical image by modifying imperceptibly either the original data or some transformed version of them (Giakoumaki, A. et al., 2003). According to Lim, S.J. et al. (2009), a digital watermarking is basically formed by three components:

1. Watermark generator – watermark(s) are generated based on certain keys.
2. Watermark embedder - Watermark(s) are embedded into the medical image by using an embedder key
3. Watermark detector – Detecting the presence of embedded watermark(s) in the medical image, sometimes a message could be extracted from watermark(s).

Watermarking requirements for medical images are categorized as followed:-

1. Security and privacy - Coatrieux, C. et al. (2000) has outlined three watermarking requirements to protect medical image security and privacy such as:
   a. Data hiding - information are embedded into the image.
   b. Integrity control - to verify that the image has not been altered without authorization;
   c. Authenticity - to verify that the image is really what the user supposes it is.
2. Fidelity requirements – ensure that watermarked medical images are useable for clinical or diagnostic purposes.
3. Computational properties - obtain the feasibility analysis for practical implementation.

The stated watermarking requirements above have generated a set of watermarking design and evaluation parameters where design parameters will provide guidelines for watermarking scheme development and the evaluation parameters measure the developed scheme’s performance (as illustrated in Figure 2.1). A set up of watermarking scheme in medical image applications involved two phases such as development and validation phases. In the development phase, a set of design parameters for digital watermarking was derived based on the medical image requirements since all the design parameters of watermarking scheme often has impact on each other either directly or indirectly (Cox, I.J. et al., 2002). Whereas the validation phase is to check whether the existing/developing watermarking scheme has fulfilled the medical image requirement by benchmarking against a set of evaluation parameters, hence the evaluation parameters and its measurements need to be contemplated carefully in the validation phase.

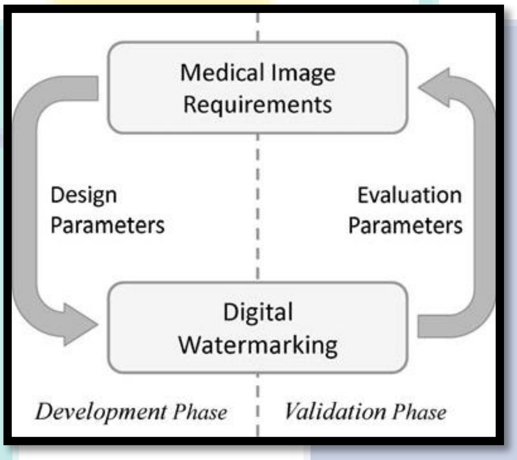


Figure 2.1. Interlinking of digital watermarking with medical image applications

Source: Nyeem, H. et al. (2013)

The watermark could be a logo, a text or an image, it is generated and embedded into medical image. The parameters for watermark generation and embedding include visibility, blindness, embedding capacity and imperceptibility/perceptual similarity. Whereas the parameter for detection and recovery include blindness, invertible, robustness, error probability, tamper localization and recovery (Nyeem, H. et al., 2013). The details of parameters are described as following:-

- **Visibility -** Digital watermark(s) should be invisible in medical image applications.
- **Robustness -** Robustness is defined as the degree of watermarking resistance towards modifications. It could be categorized as robust, fragile and semi-fragile watermarking schemes, which will be discussed in "Classification of watermarking scheme" section.
- **Blindness -** Blindness in watermarking is defined as a watermarking function (for example watermark generation, detection) which was able to be executed in the

absence of any input (for example watermark or original image). Non-blindness in watermark generation is vital in the watermark that depends on original image where it is useful in identifying forgery attacks (Cox, I.J. et al., 2008). Whereas non-blindness in watermark detection is useful in developing tamper recovery watermarking schemes where it needs original image to recover the tampered region.

- **Embedding capacity -** Embedding capacity is to measure the total number of embedding bits. A high embedding capacity is desirable in developing fragile or semi-fragile watermarking schemes (Al-Qershi, O.M. and Khoo, B.E., 2011). Research shows that Least Significant Bits (LSB) embedding techniques demonstrate a comparatively higher embedding capacity (Smitha, B. and Navas, K.A., 2007).
- **Invertible / reversible / lossless -** Invertible/reversible watermarking means the watermark can be removed and the image is restored to its original form.
- **Perceptual Similarity -** Perceptual similarity measures the degree of imperceptibility between the watermarked image and its original image (Cox, I.J. et al., 2008). Various metrics are used to measure this parameter such as correlation quality, mean square error, structural similarity (SSIM), mean SSIM, signal-to-noise ratio (SNR), peak SNR (PSNR), weighted PSNR and normalized cross-correlation (NCC). Perceptual similarity must be very high in medical image watermarking applications to avoid misdiagnosis occurrence.
- **Error Probability -** It is an important parameter to evaluate watermarking scheme performance. A zero value of error probability will be an ideal but difficult to achieve in practice as there is a high resistance to any distortions (Cox, I.J. et al., 2008). However a low value of error probabilities is always preferable in medical image application. The examples of general error probability metrics are bit error rate, false-positive rate and false-negative rate.
- **Tamper Localization -** Tamper localization function enables watermarking scheme to identify the location of the tampered pixel on the image prior image recovery process (Liew, S.C. et al., 2013). It is particularly useful in deducing the motive of image tampering.
- **Recovery –** The tampered pixel on the image could be recovered by extracting the original pixel values from the embedded watermark.

Ideally, the watermarking scheme should be an invisible, robust, non-blindness, high embedding capacity, reversible and able to recover the tampered pixels to its original form, but in reality, it is about striking a compromise between different parameters depending on an application framework. A robust watermark is desirable provided that it does not interfere with the image content interpretation

2.3.1 ADVANTAGE OF DIGITAL WATERMARKING

One of the important attributes of watermarking is data-hiding capability (Chao, H.M. et al., 2002 and Fallahpour, M. et al., 2009), which provides the following advantages:-

i. Patient sensitive information such as diagnostic results and the corresponding treatment methods can be protected by embedding the private data into the medical images.
ii. Enable database manipulation by embedding indices/keyword into image (Das, S. and Kundu, M.K., 2012).
iii. Watermarking system facilitated nonrepudiation in various multimedia applications (Cheung, S.C. et al., 2008 and Zhou, W. et al., 2002), where repudiation of not sending data would occurred among HISs. Therefore, a key-based (e.g., hospital logos or digital signature) watermarking application may facilitate nonrepudiation in teleradiology.
iv. Access authorization could be controlled by implementing keys in watermarking schemes.
v. By embedding the EPR (Electronic Patient Records) into image, it could save memory storage space, and also reduced bandwidth for the transmission in telemedicine applications (Das, S. and Kundu, M.K., 2012).

2.4 CLASSIFICATION OF WATERMARKING SCHEME

Figure 2.2 illustrates a hierarchy chart of watermarking scheme classification. Digital watermarking schemes can be broadly categorized into four classes such as robust, fragile, semi-fragile and reversible. All classes are required to be imperceptible, low embedding distortion and robust toward tampering; each different classes of watermarking scheme exhibited different characteristics which are suitable for different applications. Watermarking scheme has categorized into four types, such as Robust, Fragile, Semi-fragile and Reversible.

- **Robust watermarking scheme** – it is robustness to tampering such as common image processing operations. It usually applied in ownership proof and identification; Transaction tracking/fingerprinting; and Broadcast monitoring (Advertisement application).
- **Fragile watermarking scheme** – the embedded watermark is destroyed upon modification. It is usually applied in authentication (for example military intelligence and news broadcasting) and content-integrity verification (for example media recording of criminal events, medical image archiving, accident scene capturing for insurance and forensic purposes).
- **Semi-fragile watermarking scheme** – watermarks that can survive certain degree of legitimate manipulation such as compression and cropping. It is usually unsuitable for applications related to national security and legal issues.
- **Reversible watermarking scheme** – watermark is removed and image is restored to its original form. It usually used in teleradiology, where medical Image distortion is not be tolerated.

In summary, robust watermarks are mainly applied to ownership identification and copyright protection as they are designed to survive the attacks from common image processing operations. In contrast, fragile or semi-fragile watermarks are commonly applied to integrity verification and content authentication since they are vulnerable to attacks where it can detect and localize the tampered areas.

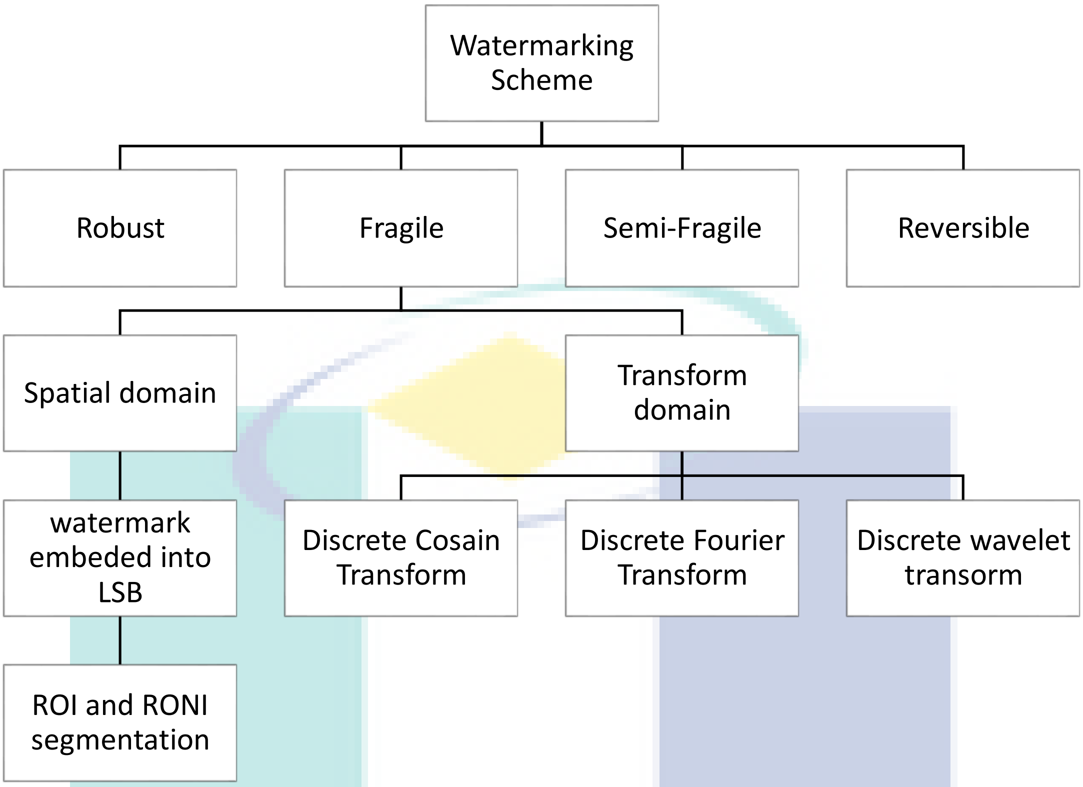


Figure 2.2. Hierarchy chart of watermarking scheme classification

Fragile watermarking scheme for image authentication can be categorized into two domains based on embedding and retrieval criteria (Liew, S.C. and Jasni, M.Z., 2010):

1) **Spatial domain watermarking** –the watermark code is embedded into the Least Significant Bits (LSBs) of the image. Since a change in LSB will only slightly shift image grey value scale thus the modification is unperceivable by human eyes.
2) **Transform domain watermarking** - based on discrete cosine transform, discrete Fourier transform, and discrete wavelet transforms.

Watermarking in spatial domain is simple and faster but vulnerable to compression, geometric distortion, and filtering as compared to transform domain relative of its computational time and complexity (Lim, S.J. et al., 2009 and Liew, S.C. and Jasni, M.Z., 2010). Transform domain can be further categorized into three domains, such as Discrete Cosine Domain (DCT), Discrete Fourier Domain (DFT) and Discrete Wavelet Domain (DWT). DCT watermarking uses the block based approach, it divided image into low (FL), medium (FM) and high (FH) frequency band in 8 x 8 block (as shown in Figure 2.3). The highlighted area in FM indicated that watermark will be embedded into a FM because modification in this region has lesser impact on image quality as compared to FL and FH.

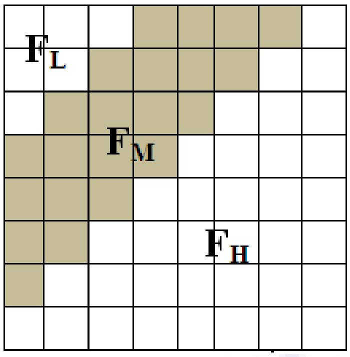


Figure 2.3. Discrete Cosine Transform Frequency 8X8 block

Source: Pooja, D. and Kavita, K. (2013)

The DFT is quite similar with DCT, where coefficients produced by a DCT operation on a block of pixels are similar to the frequency domain coefficients produced by a DFT operation. As an N point DCT is equivalent to a 2N-point DFT, it has the same frequency resolution. The differences are: (1) DCT is more focus into lower order coefficients than DFT; (2) The DCT is purely real whereas the DFT is complex (Roy, A. B, 2012).

DWT segregate image into Low resolution component (LL), horizontal component (HL), vertical component (LH) and diagonal component (HH). This segregation process could be repeated in component itself into a second level such as LL2, HL2, LH2 and HH2 (as shown in Figure 2.4). DWT retain the image quality with the cost of high computation, therefore the watermarking processing time is greater than spatial domains and others transform domains (Pooja, D. and Kavita, K., 2013)

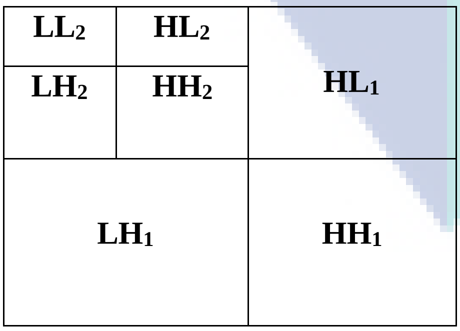


Figure 2.4. 2- Level Discrete Wavelet Transform

Source: Pooja, D. and Kavita, K. (2013)

2.5 REVIEW OF DIGITAL WATERMARKING ON MEDICAL IMAGES

Digital watermarking is used to ensure the integrity of medical images by providing tamper protection even when the images leave the network. It is because digital watermarking enables authentication information to be embedded into the medical images as watermark and maintain medical image's authenticity and integrity beyond the point of internal network. As digital medical images can be easily modified (Zhou, X.Q. et al., 2001), it is important to identify whether images have been altered and be able to localize regions that have been tampered. Watermarking techniques with tamper detection and recovery capability that allow the recipient to detect whether tampering of the medical image has occurred and to recover the tampered regions by extracting original image from the embedded watermark.

Guo, X. and Zhuang, T. G. (2009) suggested splitting medical image into region of interest (ROI) and region of non-interest (RONI) where ROI is the significant area for clinical diagnosis; thus, any modification on ROI will lead to misdiagnosis which is not tolerated. Therefore ROI was defined but left untouched and the generated watermark were embedded in the RONI, which has less or no significance in diagnosis. There is no clear method for segregation of ROI and RONI in medical images and ROI allocation may require the doctor/radiologist agreement and approval, which might be an obstacle in watermarking implementation. There is no standard procedure of making such segregation even if it has been applied in several watermarking schemes (Guo, X. and Zhuang, T. G., 2009; Liew, S.C. et al., 2013; and Rayachoti et al. 2015). The image distortion caused by watermarking could be reduced by embedding watermark into LSBs of the image. A reversible watermarking scheme is favourable where the medical image can be restored to its original form upon the removal of embedded watermark. However, this method often encountered limited storage capacity problem as compared to non-reversible method (Coatrieux, G. et al., 2006).

The practicality of watermarked medical images had been tested. Jasni, M.Z and Abdul, R. F. (2006) performed clinical evaluation of 225 medical images that were embedded with 256 bits watermark on RONI (Region of Non Interest) and 480K bits in both RON and ROI. The results show that watermarking did not alter the clinical diagnosis and is safe in terms of preserving image quality.

Fotopoulos, V. et al. (2008) had implemented a region based watermarking scheme on brain MRI image by inserting a ROI rectangle which contains the whole head shape into the RONI. ROI is compressed in a lossless way before embedding occur in order to avoid RONI storage capacity overflow problem.

Kim.K.S. et al. (2011) developed a region-based tamper detection and recovery watermarking scheme in which the image is divided into variable size block using quad tree decomposition. The average of each block is computed and used for recovery purpose. The limitations of this watermarking scheme are: (1) the image recovery process will fail if the block that stored recovery data is tampered with and (2) the tree decomposition algorithm is complex which leads to a high computational cost.

Tan, C.K. et al. (2011) proposed a dual layer reversible watermarking technique with tamper detection capability for medical images. The first layer contains source information and encrypted location signal and the second layer contains Cyclic Redundancy Check (CRC) values of image blocks for tampers detection. The limitations of these scheme are a lack of recovery methods after tamper detection and localization. The lack of recovery features may due to they are using transform domain watermarking technique which involved an intensive mathematical calculation, but it was unable to recover the image like spatial domain where the original images source was embedded as watermark.

Liew, S.C. et al. (2013) proposed a Tamper localization and Lossless Recovery (TALLOR) watermarking scheme by dividing image into one ROI (Region of Interest) and eight RONIs (Region of Non-Interest) on ultrasound medical image. ROI usually located at the centre of image which has a significant value for diagnosis usage and RONI is the area located outside the ROI. The ROI bits are compressed, read and embedded into the Least Significant Digits (LSBs) of the RONI areas during watermark embedding process. The embedded bits will be extracted to recovered tampered ROI areas during the watermarking authentication process. The strength of TALLOR was its tamper detection and localization features but also its limitation where all blocks in the image have to go through checking for tamper detection and localization even when the image has not been tampered with which is a wastage of time. Liew has further enhanced the TALLOR watermarking scheme to Tamper Localization and Lossless Recovery with ROI-Segmentation (TALLOR-RS) by further segmented ROI into non-overlapping blocks of 40 x 40 pixels and the RONI is divided into non-overlapping blocks of 2 x 2 pixels. Only the segments that were tampered with will be retrieved from the RONI for recovery purposes. Since the ROI is to be divided into segments, each segments needs to be authenticated individually and Cyclic Redundancy Check (CRC) was used to authenticate the segments of the ROI individually. CRC for each block is computed and embedded in its own block. The authentication can be performed in a multilevel manner where only suspected segments will be examined further for tampering. TALLOR-RS managed to reduce the processing time by approximately 53% relative to TALLOR but TALLOR-RS will take more time in implementing CRC during watermarking embedding process as compare to TALLOR.

Coatrieux, G.H. et al. (2013) has proposed another region based watermarking scheme where images are divided into ROI and RONI, and three watermarks are generated: first and second watermarks are used for tamper detection and localization and third watermark is used to identify the nature of alteration. Three signatures (H1, H2, and H3) are generated based on ROI and embedded into RONI where H1 is checked to verify whether the image is tampered, H2 is used to localize the tampered region and H3 is used to check whether the modification is global or local oriented and it uses multiclass support vector machines (SVM) classifier to further identified the types of modification. The drawback is that the method seems to be highly complex by using multiple watermarks, signatures and SVM, and each of them were pre-set in the algorithm, therefore any unanticipated alterations might cause the system failed.

Das, S. and Kundu, M.K. (2013) have proposed a region based watermarking technique to solve various issues in medical information management. In this method, two watermarks are embedded in zeroth and first Least Significant Bit (LSB) planes. First watermark consists of encrypted metadata and ROI information. Tamper detection is achieved by embedding second watermark consisting of the binary location map. This method achieves superior performance in terms of tamper localization capability, higher capacity, and imperceptibility. However, watermark is embedded in two bit planes which may result in image degradation. Image degradation may due to two bit planes was non-reversible.

Al-Haj, A. and Amer, A. (2014) have proposed a region based watermarking scheme where ROI is watermarked in the spatial domain and the RONI is watermarked in the frequency domain using a DWT-SVD hybrid transform. The algorithm uses robust watermark in the form of hospital logo for authentication purposes and uses fragile watermarks for tamper detection and localization. Al-Haj method has been further improved by the watermarking scheme developed by Aherrahrou, N. and Tairi, H. (2015). Their watermarking scheme is robust against various kinds of attacks by using a blind scheme in the Discrete Wavelet Transform (DWT)/Discrete Fourier Transform (DFT) transform domain. The limitation of both schemes are that their algorithms have not provided ROI recovery function after tamper detection. If a watermarking scheme without recovery function means that the medical image no longer can be used in clinical diagnosis once it was found tampered. The recovery function is to recover the medical image to its original form after tamper detection, which means the recovered medical image still can be used for clinical diagnosis.

Atta, R. et.al (2016) has presented a region-based watermarking scheme by using Residue Number System (RNS) and Chaos. Only ROI part is residue and residues that exceed bit size eight are converted to eight bits. Two watermarks are embedded in two stages: (1) robustness was achieved by using Spread Spectrum (SS) technique and the generated watermark is embedded in RONI pixels using the chaotic key, (2) fragility of image was achieved by calculating the digest of image and the hash values that calculated from the first stage is again embedded in RONI pixels based on the chaotic key. The drawback of this method is that it does not provide ROI recovery features.

Most of the watermarking scheme research as discussed above was focus on a single frame which is impractical in the real world where most of the medical image modalities are in volumetric/multi-frames form such as ultrasound, computed tomography (CT) and magnetic resonance imaging (MRI). Wenbo, D. et al. (2012) has enhanced Tan, C.K. et al (2011)'s watermarking scheme by developing a fully reversible digital watermarking scheme for the protection of volumetric/multi-frames DICOM images. Both of the watermarking scheme were based on transform domain. Wenbo, D. et al (2012) has utilized 3-D property of volumetric data to achieve shorter processing time in tamper detection and localization as compared to Tan, C.K. et al (2011)'s watermarking scheme. The shorter processing time is because they used average processing time of multi-frames for comparison, where some non-tampered frames may skip the watermarking process, indirectly it reduced the processing time in average. This is the first watermarking scheme that has been implemented in a volumetric medical image instead of a single frame. Some of the drawbacks identified with this method are as follows: (1) the concept of reversible watermarking scheme is to remove the watermark and restore image to its original form before diagnosis process; hence, the image is no longer protected and vulnerable to open network. In summary, reversible watermarking scheme only allowed one time authentication which is not favourable if the image will be used for multiple times; (2) a specific frame extraction from volumetric DICOM images for watermarking process is not permitted because of its 3-D property algorithm was frames-dependent and (3) it has not specified its recovery method after tamper detection and localization. They have suggested utilizing the power of parallel computing to further improve the volumetric watermarking processing time.

All the watermarking researches described above indicated that watermarked medical images must be imperceptible, robust to tampering and ROI able to recover from tampering after watermarking authentication process. The execution time of watermarking process might be negligible in a single frame but it will have an impact on multi-frames environment. It is easy to migrate watermarking scheme from single frame to multi-frames environment by using a control loop, such as for loop to perform watermarking process on medical images sequentially but it may be time-consuming, for example the average processing time of TALLOR watermarking scheme (Liew, S.C. et al, 2013) was reported as 20.13 seconds per frame and it would be 20.13 minutes for 60 frames of medical images regardless of the initial configuration and set up time. As watermarking process is the background work before diagnosis process starts, therefore the processing time should be shortened in order to minimize the waiting time. In order to speed up the watermarking processing time in multi-frames environment, two methods have been proposed: (1) analyse the existing watermarking algorithm in order to develop a light watermarking scheme, TALLOR watermarking scheme has been selected for performance comparison is due to its limitation as described above and its strength of having a complete package of watermarking process such as watermarking embedding, authentication and especially recovery process which was lacking in most existing watermarking schemes; and (2) develop a new watermarking scheme that able to perform parallel watermarking process on multi-frames/volumetric medical images. Parallel watermarking process could be achieved by utilizing multi-cores technology.

Table 2.2 has summarized existing watermarking schemes' characteristics in terms of spatial domain, transform domain, reversible, tamper localization, recovery, colour image, grey scale image, multi-frames environment and modalities applied. Reversible watermarking has been more widely implemented because of the clinical need which requires original medical images to be studied during diagnosis therefore watermark is removed before clinical diagnosis process which means the medical image is no longer protected and is exposed to security threat. This problem could be solved by retaining the watermarking in RONI LSB which allows for multiple times of authentication, provided that the digital watermarking has fulfilled the imperceptibility and fidelity requirements and could recovered exactly the ROI image, since ROI is the significant area for clinical diagnosis. Digital watermarking with recovery capability is crucial to maintaining the integrity of medical images since any distortion of medical image is not to be tolerated for diagnostic purposes but most of the reviewed watermarking schemes has not included recovery capability after tamper detection and localization, and tamper localization seems to be time consuming as the checking will not stop at the first encounter of tamper and will continue until the end of the authentication process.

Table 2.2

Review Table of Digital Watermarking on Medical Images

Researcher	SP	TD	Rev	TL	Rec	C	G	MF	Modalities
Giakoumaki, A. et al. (2003)		✓		✓			✓		Ultrasound
Jasni, M.Z and Abdul, R. F. (2006)	✓			✓	✓		✓		Ultrasound
Guo, X. and Zhuang, T.G. (2009)	✓		✓	✓			✓		
Fotopoulos, V. et.al (2008)	✓			✓	✓	✓			MRI
Kim, K.S. et.al (2011)		✓		✓	✓		✓		MRI, CT
Tan, C.K. et al. (2011)		✓	✓	✓		✓			DICOM
Wenbo, D. et.al (2012)		✓	✓	✓		✓		✓	DICOM
Das, S. and Kundu, M.K. (2013)	✓			✓	✓		✓		
Liew, S.C. et al. (2013)	✓		✓	✓	✓		✓		Ultrasound
Coatrieux, G.H. et al. (2013)	✓		✓	✓		✓	✓		DICOM
Al-Haj and Amer. A. (2014)		✓		✓	✓		✓		Ultrasound, MRI, CT, x-ray
Aherrahrou, N. and Tairi, H. (2015)		✓		✓	✓		✓		Ultrasound, MRI, CT, x-ray
Atta, R. et.al (2016)		✓		✓			✓		Ultrasound
Proposed method	✓			✓	✓		✓	✓	DICOM

SP: Spatial Domain
TD: Transform Domain
Rev: Reversible
TL: Tampering Localization
Rec: Recovery
C: Colored Image
G: Grey-scaled
MF: Multi-frames environment

It could be observed in Table 2.2 that most of the watermarking research was done in a single frame of medical image, it is because they were only concern on the watermarking scheme algorithm, and therefore a single frame will be sufficient. It could be noticed that only Wenbo, D. et al. (2012) has developed a watermarking scheme in volumetric/multi-frames medical images. But the proposed multi-frames watermarking scheme will not be compare with Wenbo’s approach, it is because Wenbo’s watermarking scheme was developed based on transform domain, which are using mathematical calculation for watermarking process. Whereas the proposed method will develop based on spatial domain, which are using least significant bit (LSB) for watermarking embedding process. The details of these two domains could be refer at section 2.4.

2.6 TYPE OF WATERMARKING ATTACK

There is no perfect security measures to protect watermarked image from attacks and it has been classified by Voloshynovskiy, S. et al. (2001) into four categories as following:-

1. **Removal Attack -** Removal attacks is a type of attack that eliminate watermark from watermarked image without breaching watermarking security. It does not required any effort to analysing the encryption algorithm or watermarking embedding method. The examples of removal attack are noising, sharpening, compression, and histogram equalization.
2. **Geometry Attack -** In geometry attack, the watermark signal is distorted rather than being removed from the image. It is possible to recover the original watermark if proper countermeasure is applied. The examples of geometry attack are image rotation, skewing and translation.
3. **Cryptographic Attack -** In geometry attack, the watermark signal is distorted rather than being removed from the image. It is possible to recover the original watermark if proper countermeasure is applied. The examples of cryptographic attack are image rotation, skewing and translation
4. **Protocol Attack -** Protocol attack is an attack on the whole watermarking application for example copyright protection, where a new watermark added into image to cause confusion in true ownership.

2.7 COMPRESSION IN IMAGE WATERMARKING

Compression is a technique that compress and pack data into a smaller size. Lossless compression is a technique that enable the original data to be recovered from the compressed data and it is widely applied in medical image due to its imperceptibility requirement in diagnosis process (Shih, F.Y. and Wu, Y.T., 2005). The examples of lossless compression are Huffman coding, arithmetic coding, RLE and lossless JPEG. Huffman coding (David A. Huffman, 1952) eliminate the repeated values between the successive pixels by encoding the remainder value between the actual and expected values. The low dynamic range of remainder values leads to fewer encoding bits and form a compression. Whereas Arithmetic coding uses probabilities of the source stream to subdivides the interval 0.0 and 1.0 continuously into subinterval which subdivides further by each incoming source symbols and it has a higher compression ratio than Huffman coding but consume more computer resources. RLE is a simple lossless compression algorithms that uses count number and a single value to representing a serial of the same data values, for example 11111100000011111 is encoded as (6,1),(6,0) and (5,1) and then it is converted into binary format, such as (110,1),(110,0),(101,1). As a result, the binary data of 17 bits has been compressed into 12 bits, it could be observe that RLE only efficient if the images have a large number of same successive bits such as bitmap files. JPEG is a popular compression technique for digital images. The degree of compression could be specified in JPEG compression where a high image quality will have a low degree of compression.

Compression technique has been widely applied in image watermarking scheme with the aim to reduce watermarking payload before watermarking embedding process. Especially in the case of region based watermarking scheme where ROI bits is embedded into RONI's LSB. Liew, S.C. (2011) has compressed ROI bits and embedded into RONI two LSB, where the ratio of ROI to RONI is 2:8, which means 2 bits ROI will required 8 bits of RONI for embedding purposes, therefore it is essential to reduce the ROI size by compression and lossless compression technique is preferable to ensure medical image is in its original form after decompression. Chang, C.C. et al. (2006) demonstrates the usage of RLE by encoding a bitmap file and embedding it into a gray-level image. Lin, R.S. and Hu, S.W. (2009) further improved this RLE based watermarking scheme to achieve higher embedding capacity by using an additional bit to represent the encoding. Osamah, M. and Khoo, B. E. (2011) had compressed ROI by using poor quality JPEG and compressed the average of block in the ROI by using Huffman coding in their watermarking scheme. Liew, S.C. (2011) has made a comparison analysis on JPEG and RLE compression techniques application on ROI in four samples of 8-bit monochrome grayscale ultrasound medical images and found out that JPEG performed better than RLE compression in term of compression ratio and reliability in successful compression.

2.8 HASH FUNCTION

SHA-2 (Secure Hash Algorithm 2) is a set of cryptographic hash functions designed by the NSA (U.S. National Security Agency). Cryptographic hash functions are mathematical operations run on digital data; Data's integrity could be determined by comparing the generated hash value from an algorithm to an expected hash value, if the compared result is not identical then it means the algorithm has been tampered or modified. Collision resistance is key feature of hash function where two different input values shouldn't produce the same hash output. The SHA-2 family consists of six hash functions such as SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and SHA-512/256. SHA-256 and SHA-512 are novel hash functions computed with 32-bit and 64-bit words respectively.

Hash function had been used widely in image watermarking for authentication and verification purposes. For example a block of pixels can be hashed and the hashed value is being embedded as part of the watermark. The hash value is retrieved and being compared with the hash value of the same block of pixels at the time of authentication. Das, S. and Kundu, M.K. (2010) applied SHA-256 hash function to the ROI of medical images for the usage of authentication. Tan, C.K. et al. (2011) applied the same hash function to verify the success of watermark removal in an image. MD5 is one in a series of hash function algorithms that produces a 128 bits hash value and it is not collision resistant. MD5 hash function had been demonstrated that it is possible to generate two inputs with different content but having the same hash value (Wang, X.Y. et al., 2004). Yang, C.W. and Shen, J.J. (2010) used MD5 hash function to hash image blocks and embeds the hash values into the LSBs of the corresponding blocks which is used for authentication purposes. Osamah, M. and Khoo, B. E. (2011) used the same hash function to authenticate the ROI. Fawad, A. et al. (2010) developed their own hash function to be used for image authentication in a robust watermarking scheme. The disadvantage of using self-developed hash function is that it may not be secure if proper testing is not performed.

2.9 PARALLELLISM CONCEPT

In recent years there has been a surge of interest in running application in parallel to take advantage of multiprocessor and multicore systems. Developments in microprocessor technologies have resulted in most processors having multiple computing cores in a single chip (Saddharth, S. et al., 2010). Parallel computing is a concept of performing tasks simultaneously by partitioning a large and complex problem into smaller tasks and solved each of them concurrently. Parallel watermarking process on multi-frames medical images has not been implemented before, Wenbo, D. et al (2012) have suggested utilizing the power of parallel computing to further improve the volumetric/multi-frames watermarking processing time. There are two forms of parallelism: Task parallelism and Data Parallelism, which are described as below.

- **Task Parallelism -** It is a form of parallelization of computer code across multiple processors in parallel computing environments (Rashmi, A.J. and Dinesh, V. P., 2012). Task parallelism focuses on distributing execution processes (threads) across different parallel computing nodes (Luo, G. and Zhang, D., 2010).
- **Data Parallelism** - It is a form of parallelization of computing across multiple processors in parallel computing environments (Luo, G. and Zhang, D., 2010). Data parallelism focuses on distributing the data across different parallel computing nodes.

Data parallelism emphasizes the distributed (parallelized) nature of the data, as opposed to the processing (task parallelism) (Wang, S.P and Ledley, R.S., 2012). Data parallelism is adopted in this experiment since each processor performs the same code (watermarking code) on different pieces of distributed ultrasound frames.

An application will partition into a few subtasks for parallel processing. Applications are often classified based on the frequency of synchronization and communication needs between their subtasks. Fine-grained parallelism is where an application has a high rate of communication among subtasks; Coarse-grained parallelism is where an application has none or low rate of communication among subtasks and it is term as “embarrassingly parallel” which means it is embarrassingly easy to parallelize (Vicat-Blanc, P. et al., 2013). In the embarrassingly parallel problems, speedup factors could be achieved approximately to the number of cores or even more if the problem is partition evenly to fit within each core's caches in which avoided using main memory which caused the system slow.

2.10 SPEEDUP

Speedup has been widely used as a performance metrics to measure the efficiency of a parallel algorithm. Sanjay, S. (2015) has used speedup to measure the performance of his proposed parallel algorithm by dividing image processing time of the sequential algorithm in relative to parallel algorithm in multi core architecture. Amit, B. and Brijendra, K.J. (2015) used speedup factor to measure his parallel lexical analysis performance. Speedup is used to measure the performance improvement between two systems processing the same problem. In other words, it is used to measure the improvement in execution speed of a processing task on two similar architectures. Speedup was established by Amdahl's' law in measuring the performance of parallel processing and it also could be used to demonstrate the effect after resource enhancement. With *n* processor, $Speeedup_n$ is:

$$Speeedup_n = \frac{T_1}{T_n} \tag{2.1}$$

Where T_1 is the execution time is for one core; T_n is the execution time for *n* cores; and the $Speeedup_n$ should be more than 1 (Mike, B., 2017). Parallelism will yield a linear speedup according to the number of processors. In other words, increase the number of processors will increase the speedup value. However, not many parallel algorithms yield an expected linear speed-up. Small numbers of processing elements tends to yield a near linear speedup but will into a constant value for large numbers of processing elements (Degroote, J. and Vierendeels, J., 2011).

According to Amdahl's law, the overall speed-up from parallelization would be restricted by some fraction of total operation that is inherently sequential and cannot be parallelized. This includes reading data, setting up calculations, control logic, storing results, etc (Mike, B., 2017).

$$Speeedup_n = \frac{T_1}{T_n} = \frac{1}{\frac{F_{parallel}}{n} + F_{sequential}} = \frac{1}{\frac{F_{parallel}}{n} + (1 - F_{parallel})} \tag{2.2}$$

Where $F_{parallel}$ is the fraction that can be reduced by deploying multiple processors; whereas $F_{sequential}$ is the fraction that cannot reduced by deploying multiple processors.

Gustafson's law is another law in computing, closely related to Amdahl's law (Saddharth, S. et al., 2010). Gustafson observed that when the same parallel program applied on larger datasets, the parallel fraction, F_p, increases. Let P be the amount of time spent on the parallel portion of an original task and S spent on the serial portion. Then

$$F_p = \frac{P}{P + S} \tag{2.3}$$

Amdahl's law assumes that the parallelism is independent with the number of processors, in contrast Gustafson's law assumes differently and stated that parallelism will yield a linear speedup according to the number of processors (Linczuk, M. et al., 2013).

2.11 HOW PARALLEL COMPUTING RUN A JOB

Scheduler coordinates the jobs execution and it can be executed on any computers in the network. The scheduler runs the submitted jobs in queue order, unless any jobs in its queue are promoted, demoted, cancelled, or deleted. Scheduler assigning task from the running job to each worker for execution, and fetch result from workers upon the task completion. The cycle is repeated with another task. Scheduler starts to submit another job to the next available worker when it has finish assigning all tasks of the running job to workers. Tasks were executed simultaneously by all workers in order to speedup execution of large jobs. Upon the completion of task execution, the results are return to the client by scheduler (MathWorks, 2015). The process of parallel computing described above is illustrated in Figure 2.5.

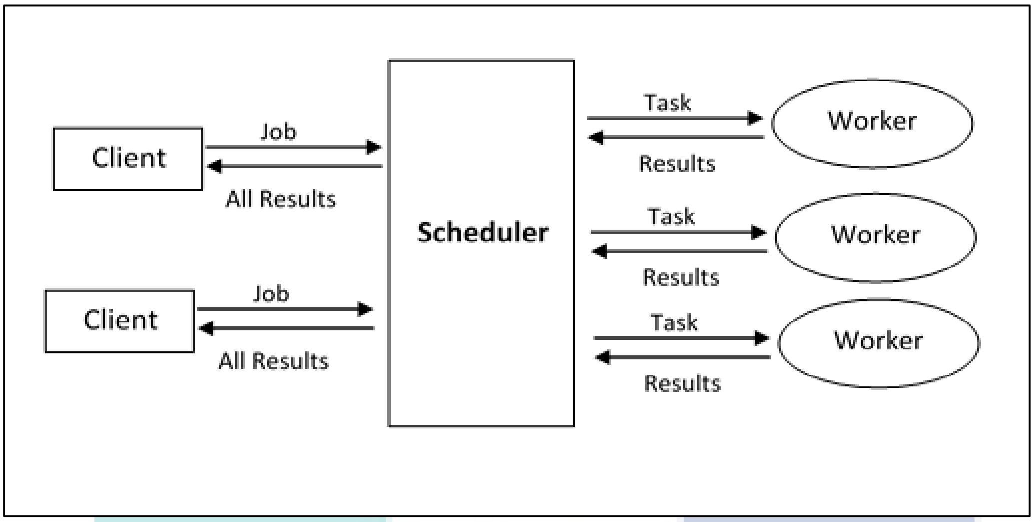


Figure 2.5. Interaction of parallel computing session

Source: MathWorks (2015)

2.11.1 Life Cycle of A Job

A job go through a number of stages upon its creation which is illustrated in Figure 2.6. In the scheduler, each stage of a job is categorized by its state, such as *pending*, *queued*, *running*, or *finished* (MathWorks, 2015). Functions used in job management are *createJob*, *submit*, and *fetchOutputs* (Vicat-Blanc et al., 2013). A job is in the pending stage when it is being created in client session, the pending jobs is then submitted into a queue for execution and the sequence of the queue can be changed with demote and promote function. When the queue is full, the scheduler will distribute the job's tasks to the worker session for execution and it turn the job status to "running". The scheduler start running the next job if there are worker available from the previous running job, so indirectly it makes jobs running in parallel. The job is in the "finished" stage when all tasks in a job have completed execution and generated a result which is being fetched by "fetchOutput" function. A job will be failed and deleted by scheduler if an error is encountered during commands execution.

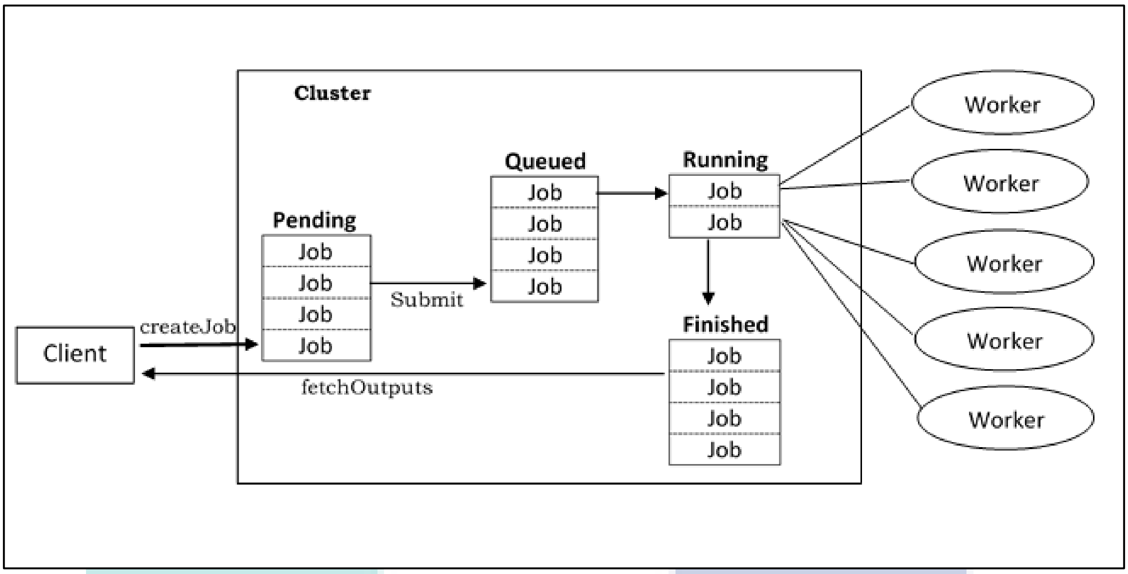


Figure 2.6 Stage of a job

Source: MathWorks (2015)

2.12 PARALLEL COMPUTING RESEARCHES

Recent research demonstrates parallel computing by utilizing the power of multicores architecture has improve the performance in medical image processing. Minye, W. et al. (2014) proposed a pipeline computation model which applies on cluster to make procedures more efficient by taking full advantages of cluster or Multi-core CPUs. Sanjay, S. et al. (2014) has provided a comprehensive review of the existing literature available on Image registration methods based on parallel computing in multi core architecture. Sanjay, S. et al. (2015) has proposed a segmentation approach using Otsu's method in multi core environment. Amit, B. and Brijendra, K.J. (2015) improve the lexical analysis phase by exploiting the inherent parallel processing capability of multi-core machines. Kadah, Y. M. et al. (2011) has reviewed and demonstrated the potential of parallel computation in medical imaging and visualization in a wide range of applications including image reconstruction, image denoising, motion estimation, deformable registration and modelling. Besides medical imaging fields, there are number of researchers of other fields (Qiang, L. et al. 2015; Jiang, H., and Ganesan, N., 2016) have utilized the power of parallel computing to increase the efficiency of the existing algorithms. Larsen, M. et al. (2015) has developed a parallel framework which provides a single code base that can run on many architectures. The parallel framework was constructed by utilizing multi-cores technologies where a job is divided into a few tasks which are executed in parallel. Li, Y. et al. (2014) have shown that the speedup gain of parallel framework is roughly proportional to the number of parallel cores. All of these research indicated that the proposed parallel multi-frames watermarking processing on multicores environment could improve performance and resolve the time constraint of sequential multi-frames watermarking process.

Speedup factor is a formula that has been used widely in time performance measurement. Speedup factors yield by comparing parallel execution time with sequential process execution time; therefore, an executable sequential process is pre-requisite and it serve as a benchmark against parallel process. Qiang, L. et al. (2015); Yan, C. et al. (2014) and Mahmood, Z. et al. (2015) have utilized multi-cores technology and developed a parallel framework in their research work, as a result they have proven that parallel execution has a high speedup compared to sequence execution, for example Yan, C. et al. (2014) has developed an efficient parallel framework for motion estimation in video technology on multi-cores processors, and has achieved 30 to 40 times speedup in parallel execution as compared to sequence execution. In this research, parallelism is incorporate into proposed multi-frames watermarking scheme so that a significant speedup could be achieved without compromising the functionality and accuracy of the existing watermarking scheme algorithm. The proposed multi-frames watermarking scheme is feasible if it exhibit data parallelism characteristic where less synchronization was required among the sub-tasks.

2.13 CONCLUSION

It has been found that most of the watermarking research has focused on watermarking intrinsic algorithm and only deal with single frame medical image but most of the medical modalities are in multi-frames form, such as MRI, X-rays and ultrasound medical images thus it is important to develop a watermarking scheme for multi-frames environment. Wenbo, D. et al. (2012) have demonstrated the possibility of implementing watermarking in multi-frame modalities. While it utilizes 3D property of volumetric DICOM images, the watermarking scheme is frame dependent, which means frame extraction from a volumetric medical images is not allowed. However, the doctor might want to extract and examine a specific frame for diagnostic purposes; therefore, watermarking frame-independency in multi-frame environment has become crucial. Process watermarking sequentially in multi-frame environment might be a solution but it is time consuming. Thus, it is recommended to utilize the power of parallel computing in order to speedup the watermarking process in multi-frames environment and therefore a multi-frames watermarking scheme is proposed. The strict requirement of medical images has hindered the acceptance of watermarking implementation in medical images in practice; therefore, in order to increase digital watermarking acceptance, it is crucial to develop a digital watermarking scheme which is imperceptible, robust to tampering and equipped with recovery capability, especially in multi-frames medical images.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

Watermarking research flow (as shown in Figure 3.1) serve as a research foundation for both region of interest based tamper detection and lossless recovery watermarking scheme (ROI-DR) and multi-frames watermarking scheme. A medical image was prepared, such as define the ROI and RONI, before it was input into watermarking embedding process. A watermarked medical image was generated after watermarking embedding process. Imperceptibility in term of PSNR was measured by comparing watermarked with raw medical image. A various type of tampering was applied on the watermarked medical image, such as applying “Pepper and Salt” noise, flipping a portion of images in ROI and etc. Watermarking authentication process was deployed on the tampered watermarked medical image. A tampering message and a recovered DICOM file were generated after the watermarking authentication process.

ROI-DR is a speedy watermarking scheme that able to perform recovery process immediately after the first detection of tamper pixel. ROI-DR is focus on watermarking scheme algorithm in a single frame of medical image, whereas multi-frames watermarking scheme execute ROI-DR watermarking scheme at multi-frame environment in parallel.

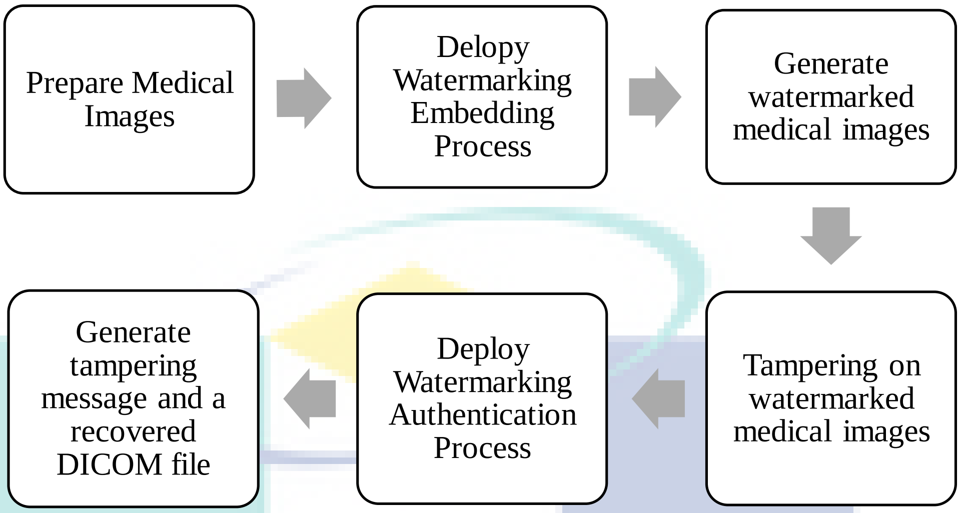


Figure 3.1 Watermarking research flow

Multi-frames watermarking scheme is implemented in both sequential and parallel modes. For sequential mode, the watermarking process is executed on frame by frame sequentially. For parallel mode, multi-frames watermarking scheme subdivided ultrasound medical images according to the number of cores available and perform sequential mode of watermarking process on each subdivided medical images in parallel. As illustrated in Figure 3.2, 15 frames of ultrasound medical images is subdivided into 4 cores and each core execute watermarking process frame by frame sequentially, and all four cores are run in parallel. A workable multi-frames watermarking scheme in sequential mode is a pre-requisite for parallel mode to be work successfully. Each rectangle in Figure 3.1 will be describe in the section 3.2 – 3.7.

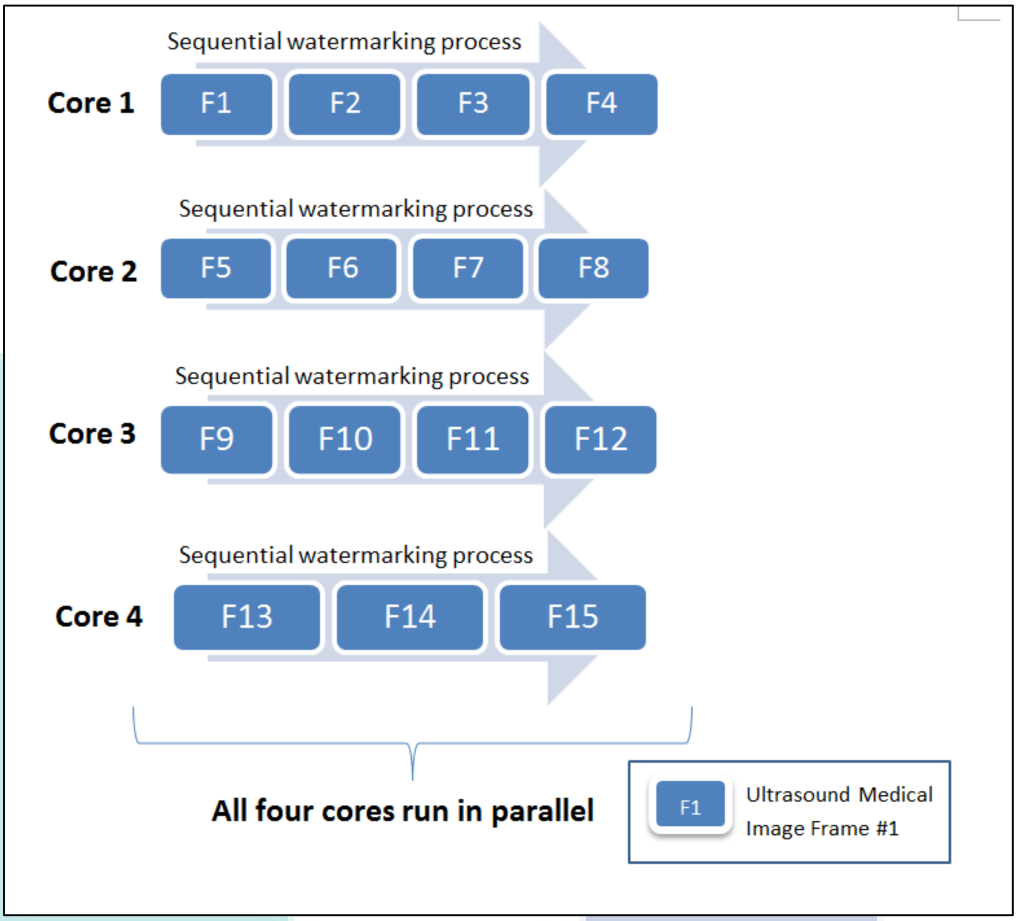


Figure 3.2 Multi-frames watermarking scheme in both parallel and sequential modes

3.2 PREPARE MEDICAL IMAGES

In Table 3.1(a), eight samples of ultrasound medical images in DICOM format were used as input files to the watermarking system. They have different image dimension and number of frames but same 8 bits per pixel. The different number of frames were used to test the function of multi-frames watermarking scheme, it is to test whether the number of frames will affect the speedup factor of multi-frames watermarking scheme. Each sample is 8 bits per pixel, and ultrasound_sample_1 consist of 640 x 480 x 8 = 2457600 bits per frame. The bits values is calculated to define the size of ROI and RONIs. The size of ROI [x, y, width, height] has been preset to [265, 110, 120, 320], where x and y are the coordinate value of the right corner of ROI rectangle, width is the width of ROI rectangle, and height is the height of ROI rectangle.

In Table 3.1 (b), other medical imaging modalities such as Magnetic Resonance Imaging (MRI), X-Ray Angiography (XA) and Computed Tomography (CT), were used to test the proposed system. The purpose is to ensure that the proposed watermarking schemes are executable on other medical imaging modalities besides ultrasound medical images. The ROI size of medical Image samples was redefined whereas RONIs size are remain the same

All samples have 8 bits per pixel, it is because the proposed watermarking scheme algorithm was restricted to 8 bits. One ROI pixel consists of 8 bits, which is 4 pairs of bits. Each pair of ROI bits are inserted into each RONI two LSB, which means one pixel of ROI will require four pixels of RONI for watermarking embedding process. Therefore it is essential to implement error handling which is to ensure sufficient RONI storage space for compressed ROI bits and its hashed values before watermarking embedding process.

Table 3.1

(a) *Ultrasound Medical Images Samples Properties*

Ultrasound Medical Images	**Image dimension in pixels**	**Bits per pixel**	**Number of frames**
Ultraound_Sample_1.dcm	640 x 480	8	30
Ultraound_Sample_2.dcm	640 x 480	8	15
Ultraound_Sample_3.dcm	640 x 476	8	15
Ultraound_Sample_4.dcm	642 x 460	8	79
Ultraound_Sample_5.dcm	642 x 460	8	31
Ultraound_Sample_6.dcm	670 x 480	8	25
Ultraound_Sample_7.dcm	640 x 476	8	15
Ultraound_Sample_8.dcm	640 x 459	8	33

(b) *Other Medical Images Samples Properties*

Medical Image Sample	**Image dimension in pixels**	**Bits per pixel**	**Number of frames**	**ROI size definition [x, y, width, height]**
MRI.dcm	640 x 476	8	16	[132, 135, 284, 149]
XA.dcm	512 x 512	8	12	[140, 127, 226, 149]
CT.dcm	512 X 512	8	1	[135, 124, 206, 168]

Each ultrasound medical image is divided into one rectangle for ROI and five rectangles for RONIs as shown in Figure 3.3. ROI and RONIs was predefined on ultrasound medical image before input into the watermarking embedding process. ROI is located at the center of ultrasound medical image which contains the significant diagnosis information and RONIs are located at the perimeter side of ultrasound medical image which contains no or less significant information. ROI bits and its related hash information will be compressed and embedded into designated RONIs areas during the watermarking embedding process, which will be discussed in section 3.3.

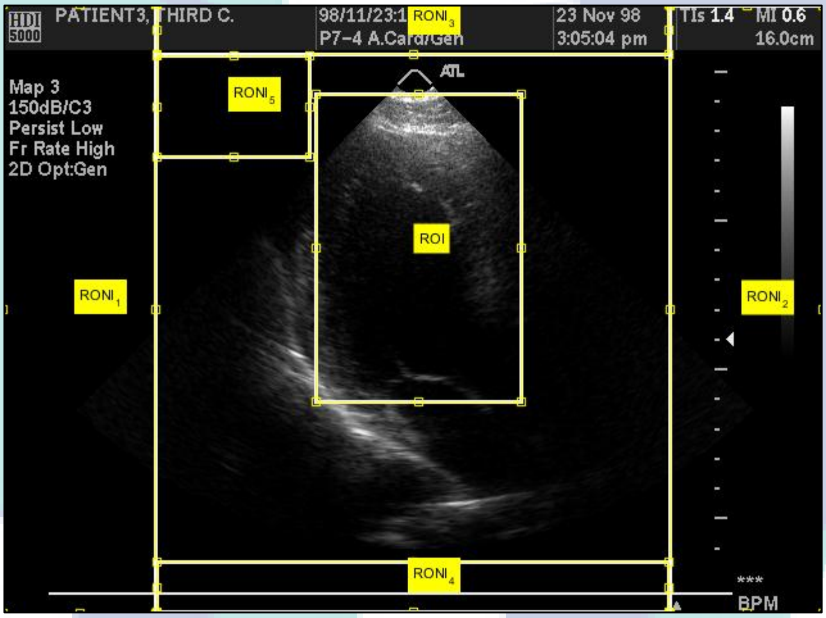


Figure 3.3. ROI and RONI layout for ROI-DR Watermarking Scheme

3.3 DEPLOY WATERMARKING EMBEDDING PROCESS

3.3.1 ROI-DR Watermarking Embedding Process on a Single Frame

Watermarking embedding process on a single frame of medical image is a process of inserting ROI bits into RONI areas of an ultrasound medical image to generate a watermarked ultrasound medical image, which later is used in watermarking authentication process. ROI of ultrasound medical image is identified, cropped, compressed with JPEG and hashed with SHA-256, which will be describe in details at the following paragraph. The generated compressed ROI and its hashed values are then convert into bits and stored into RONI areas as described in Table 3.2. The purpose of compression is to reduce the size of ROI embedding bits, and thus reduce the watermark payload and the elapsed time for watermarking embedding process. The compressed ROI bits will then split into two parts and embedded into RONI 1 and RONI 2 respectively.

Table 3.2

Bits information that embedded into RONIs area

RONI area	Embedded bits information	Objective
1 and 2	ROI bits	For ROI recovery process
3	Hashed ROI value	To verify tampering occurrence
4	Hashed compressed ROI values	To further verify tampering occurrence at ROI or RONI
5	Size of compressed ROI	Used in retrieving compressed ROI bits process.

There was 15 processes in the program flow chart of ROI-DR watermarking embedding process (as shown in Figure 3.4), and each of the process was described as following:-

1. **Read an Ultrasound Medical Image**: Input an ultrasound medical image file into ROI-DR watermarking system.
2. **Set ROI and RONI rectangle values**: Set the region of interest (ROI) and region of non-interest (RONI) rectangle values, which are [x-coordinate value, y-coordinate value, width, height].
3. **Crop ROI and RONI rectangle regions:** Crop out the image within the ROI and RONI rectangle regions.
4. **Convert all RONI regions to binary format:** Convert all the cropped RONI regions such as RONI_1 until RONI_5 to binary format and ready their last two least significant bits (LSB) for ROI bits and hashed values storage.
5. **Hash cropped ROI region with SHA-256:** Hash the cropped ROI region with SHA-256 to generate **ROI_hash_bin** and stored into RONI_3 region, in which later will be retrieved and used in watermarking authentication process.

6. **Compress ROI region to JPEG format**: Compress the ROI region into JPEG format in order to reduce the size and its payload into RONI storage.
7. **Convert the compressed ROI into binary format**: Compressed ROI will need to convert into binary format before it embedding bits into RONI last two LSB. The details will be illustrated in Figure 3.5.
8. **Split ROI_binary into two sections**: Split the ROI_binary (in which generated from (7)) into two sections and store into RONI_1 and RONI_2 respectively.
9. **Hash compressed ROI with SHA-256:** Compressed ROI are hashed with SHA-256 to generate comp_ROI_hash_bin and store into RONI_4.
10. **Store ROI bits into RONI LSB**: Every ROI pixel consist of 8 bits, in which is 4 x 2 bits, where each pixel is split into four pair of bits and store each pair of bits into the two LSB of each RONI pixel. In summary, one ROI pixel will require four RONI pixels for storage. The details are illustrates in Figure 3.6.
11. **Convert RONI_bin back to decimal format, c:** RONI_bin is converted back to decimal format as variable c after all ROI bits and hash values have stored into RONI_bin.
12. **Reshape c back to RONI original matrix size:** Reshape c variable from a single column matrix into RONI original matrix size.
13. **Convert RONI to uint8 data type:** Convert RONI from double to uint8 data type.
14. **Replace all RONI back to RONI region in the image:** Replaced all bits embedded RONIs (watermark) back to RONI region in the image, in which form a watermarked ultrasound medical image.
15. **Write into a DICOM file:** Lastly, write the watermarked ultrasound image into a DICOM file.

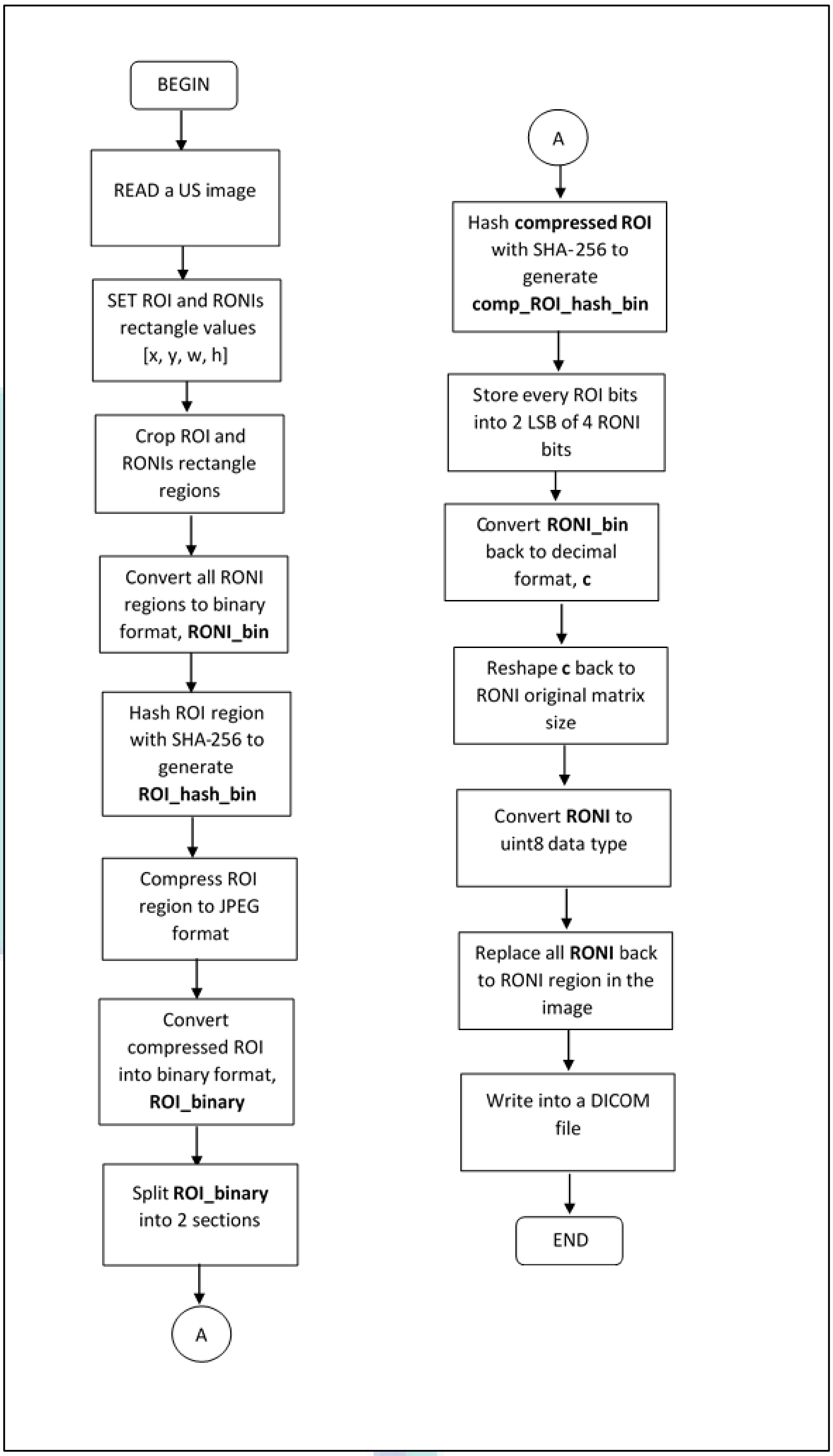


Figure 3.4. Program flow chart for ROI-DR watermarking embedding process

The implementation details of watermarking embedding process was illustrated in Figure 3.5, ROI region was hashed with SHA-256 and stored as ROI_hash in ROI 3. ROI region pixel values was read and compressed into JPEG format, the compressed ROI variable was used to perform three different tasks, which are:

1. Compressed ROI variable was converted into binary format before it embedded bits into RONI. The ROI pixel values in decimal format was read from top to bottom and left to right until the last pixel of ROI. The read decimal values was convert into an array of 8 bits. The array was then split into two sections and stored into RONI_1 and RONI_2.
2. Compressed ROI variable was hashed with SHA-256 and produced compressed_ROI_hash and stored into RONI 4.
3. The length of compressed ROI variable was calculated and stored as compressed_ROI_size into RONI 5.

The process of how watermark was being embedded into RONI_1 pixels was illustrated in Figure 3.6. The RONI_1 pixels values in decimal format was read and converted into an array of 8 bits. The last two least significant bits (LSB) of RONI pixels was replaced by two bits of ROI. As a result, 1 pixel (8 bits) of ROI required 4 pixels (4 x 2 bits) of RONI for watermark embedding capacity. After all ROI bits has been embedded into RONI's LSB, the array of RONI bits was converted into an array of decimal values, named as RONI_1 decimal matrix. RONI_1 decimal matrix was reshaped according to RONI_1 original matrix shape and size so that it could overwrite the original RONI_1.

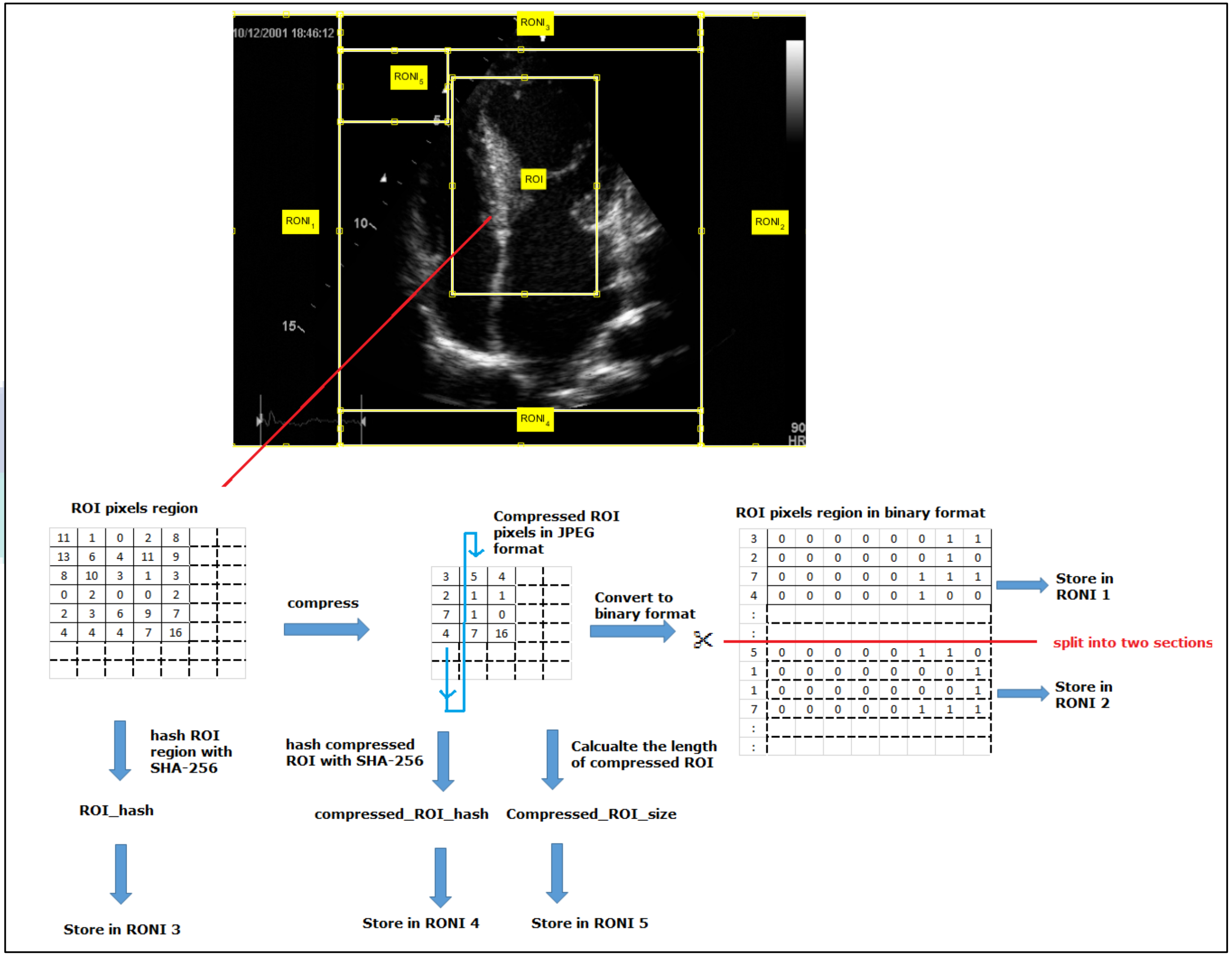


Figure 3.5 The implementation details of watermarking embedding process

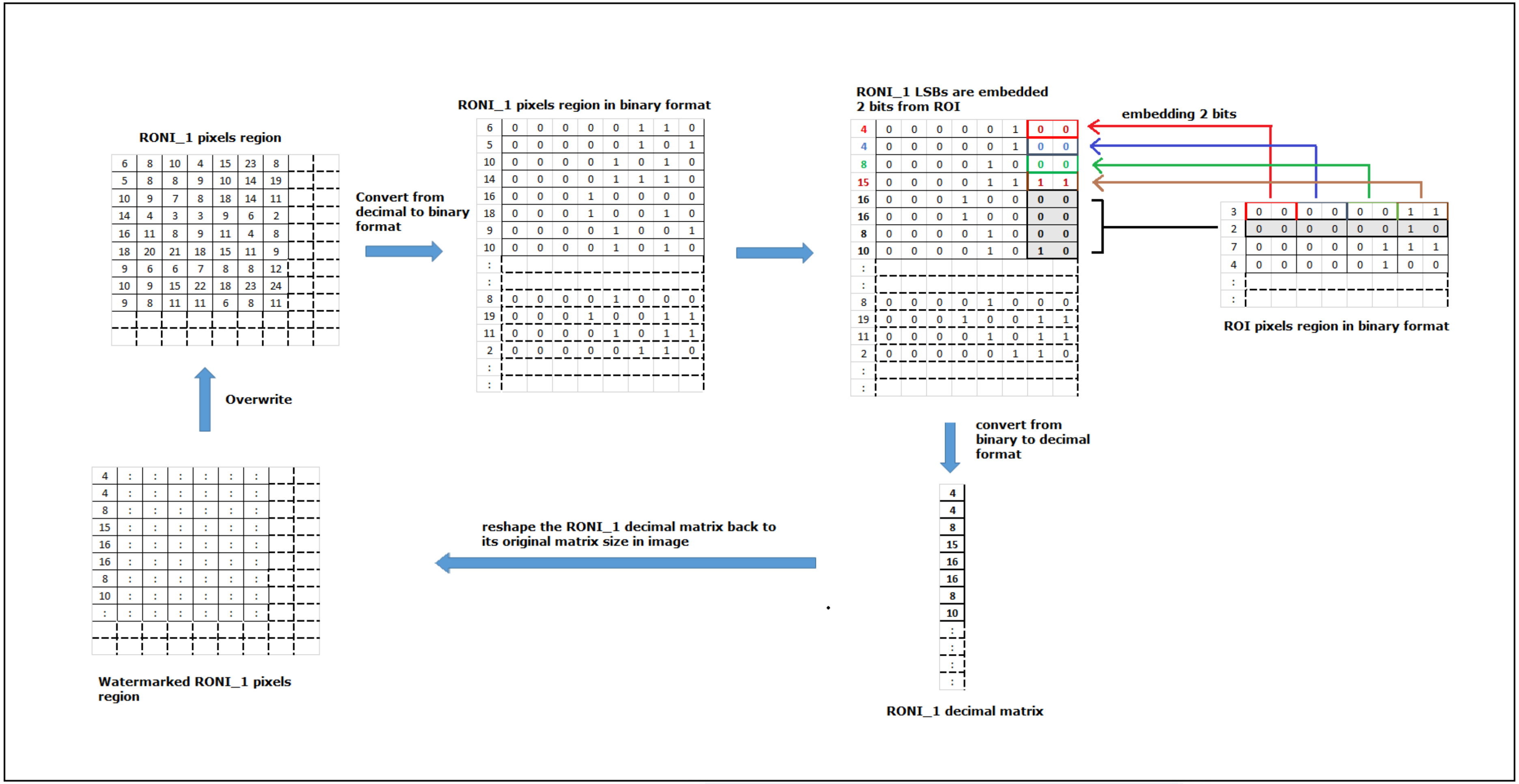


Figure 3.6 Watermark embedding into RONI_1 pixels

3.3.2 Multi-Frames Watermarking Embedding Process

Multi-frames watermarking embedding process executed in both parallel and sequential modes. Multi-frames ultrasound medical images were loaded into a quad core microprocessor/cluster and create a job on the scheduler; the job is then divided into tasks according to the number of cores in the microprocessor. The code implemented enables cluster to auto detect the number of cores available in the processor. If the processor used is a quad core then the job is divided into 4 tasks where ultrasound frames are equally divided by 4, so that the frames will be distributed to each core evenly. It is to ensure each core will handle the same amount of work load, so that they could finish the job at the same period of time in parallel. For example if the total number of ultrasound frames is 30 then it will be divided into 8,8,7,7 frames and if the total number of frames is 15 then it will divided into 4,4,4,3 frames. Those divided frames will then be distributed to 4 cores respectively. In each core, watermarking embedding process is carried out sequentially on the divided frames and at the same time it runs concurrently with other cores (as illustrated in Figure 3.7). Medical images was watermarked upon the completion of watermarking embedding process. The watermarked medical images output will be collected from each core and submitted as a job to microprocessor. The microprocessor will sort and output the watermarked medical image according to the frame number order. The output will be fetch, concatenate and write into a single DICOM file. A job is deleted on two circumstances: when the scheduler encounters an error, or when the job is finished.

Sequential watermarking process is important for two reasons: (1) it served as a benchmark against proposed multi-frames watermarking scheme, because it is a conventional method for multi-frames watermarking process. (2) Sequential watermarking process has playing an important part in the proposed multi-frames watermarking scheme construction. Sequential and parallel watermarking process will be described in details in the following section I and II respectively.

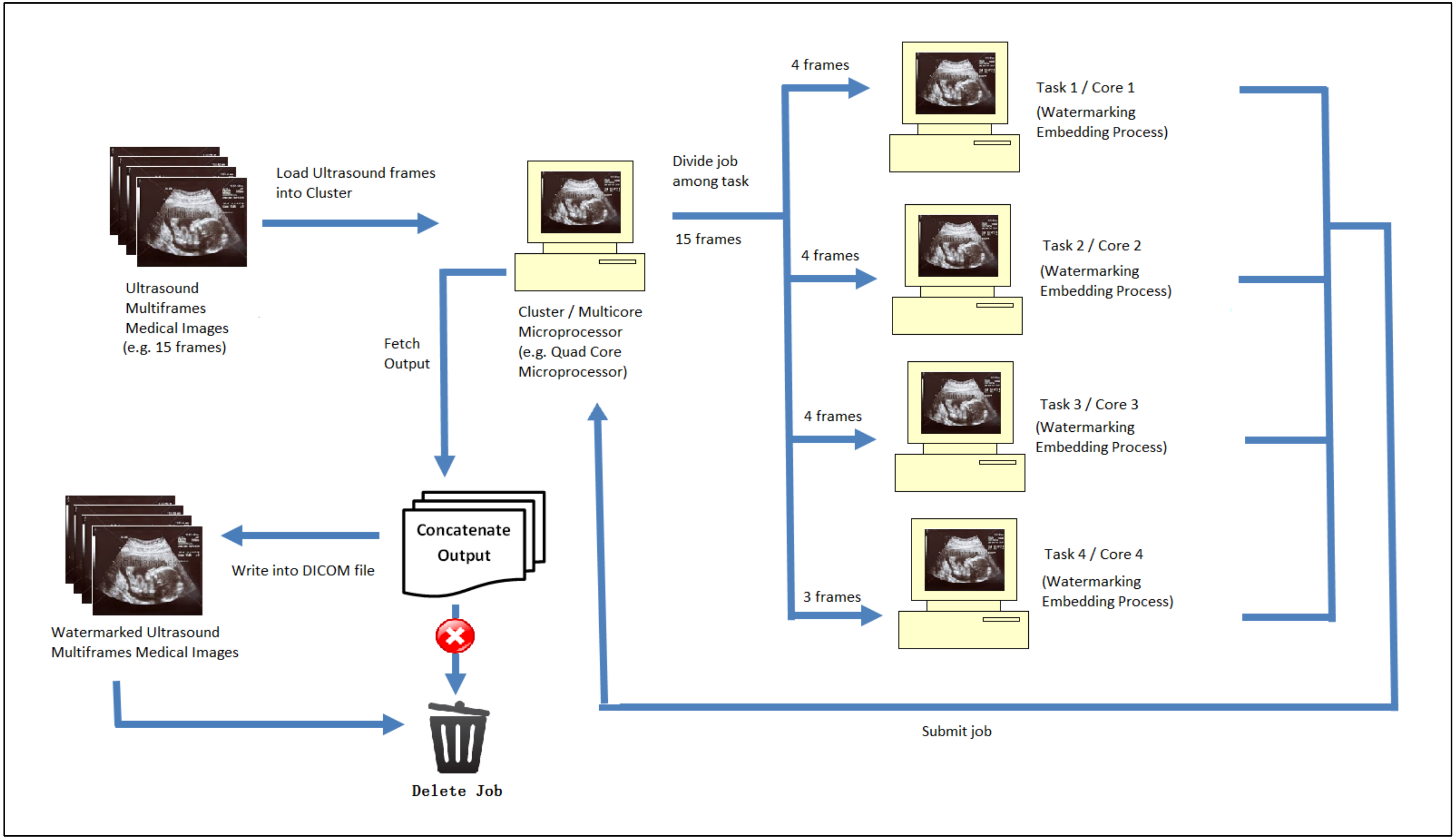
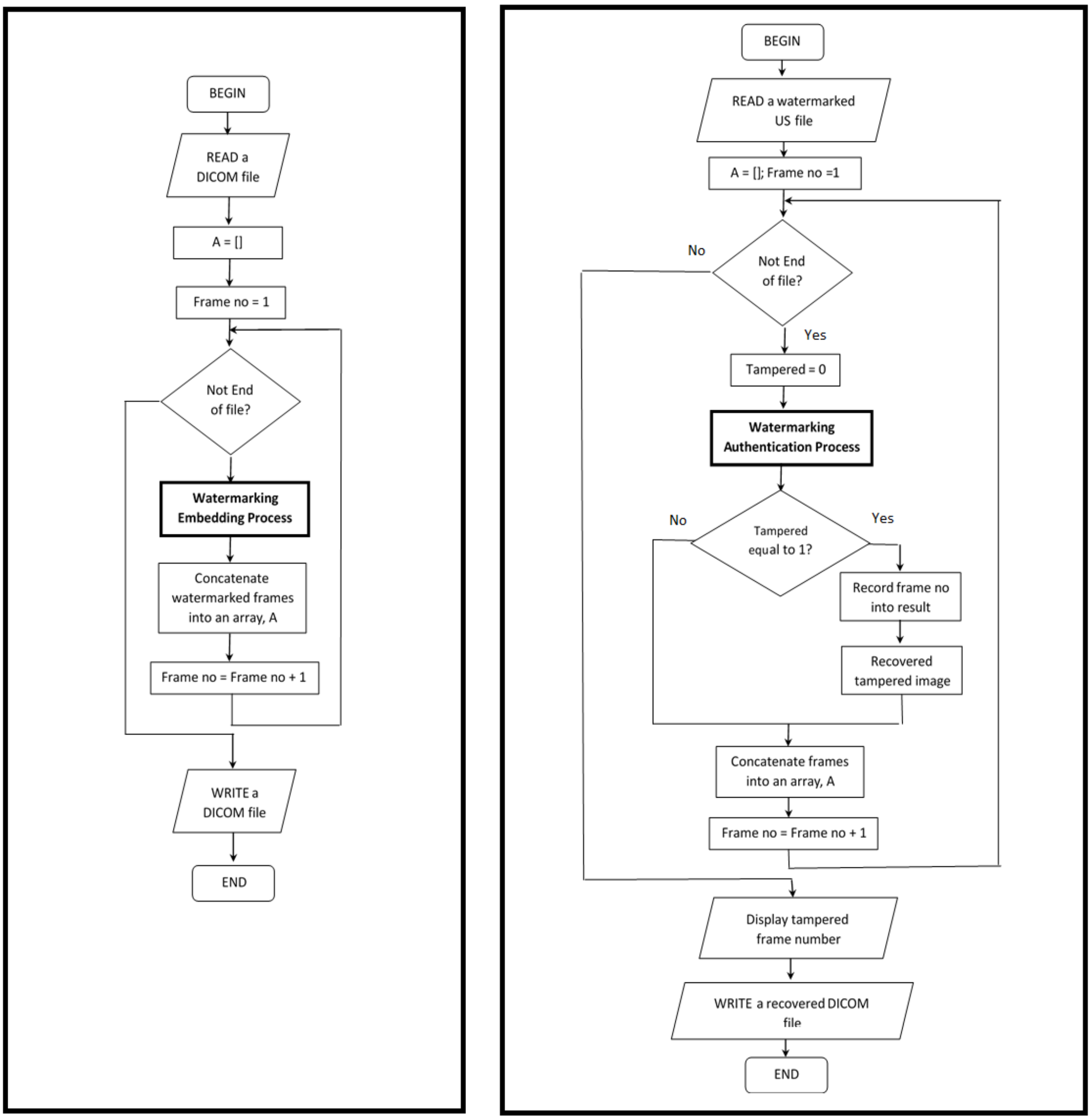


Figure 3.7. Multi-frames watermarking embedding process

I. Sequential Watermarking Process

Sequential watermarking process is a conventional method used in multi-frames medical images. Sequential watermarking embedding and authentication process are sharing a common process, where an ultrasound medical images in DICOM format is read and perform watermarking process frame by frame sequentially by using a for loop. Processed frames will then be concatenated into a file, A which later will be written into a DICOM format upon completion of watermarking process. The relationship between two processes is the output file of watermarking embedding process is the input file for watermarking authentication process. The difference between them is the sequential watermarking authentication process has added a flag as an indicator for tampered frame, if the flag is 0, means the frame is non tampered else the frame is tampered then image recovery is performed and tampered frame number is recorded and will be display upon the completion of watermarking process (as demonstrated in Figure 3.8).

The main algorithms of parallel watermarking process is dividing volumetric ultrasound medical images and distributed them into a number of cores and executing sequential watermarking processes on each core in parallel, therefore a successful sequential watermarking process is a prerequisite in parallel watermarking process.



(a) Sequential Watermarking Embedding Process

(b) Sequential Watermarking Authentication Process

Figure 3.8. Program flow chart for (a) Sequential watermarking embedding process and (b) Sequential watermarking authentication process

II. Parallel Watermarking Process

Figure 3.9 illustrated the main algorithm flow chart of parallel watermarking process, which applied for both parallel watermarking embedding and authentication process.

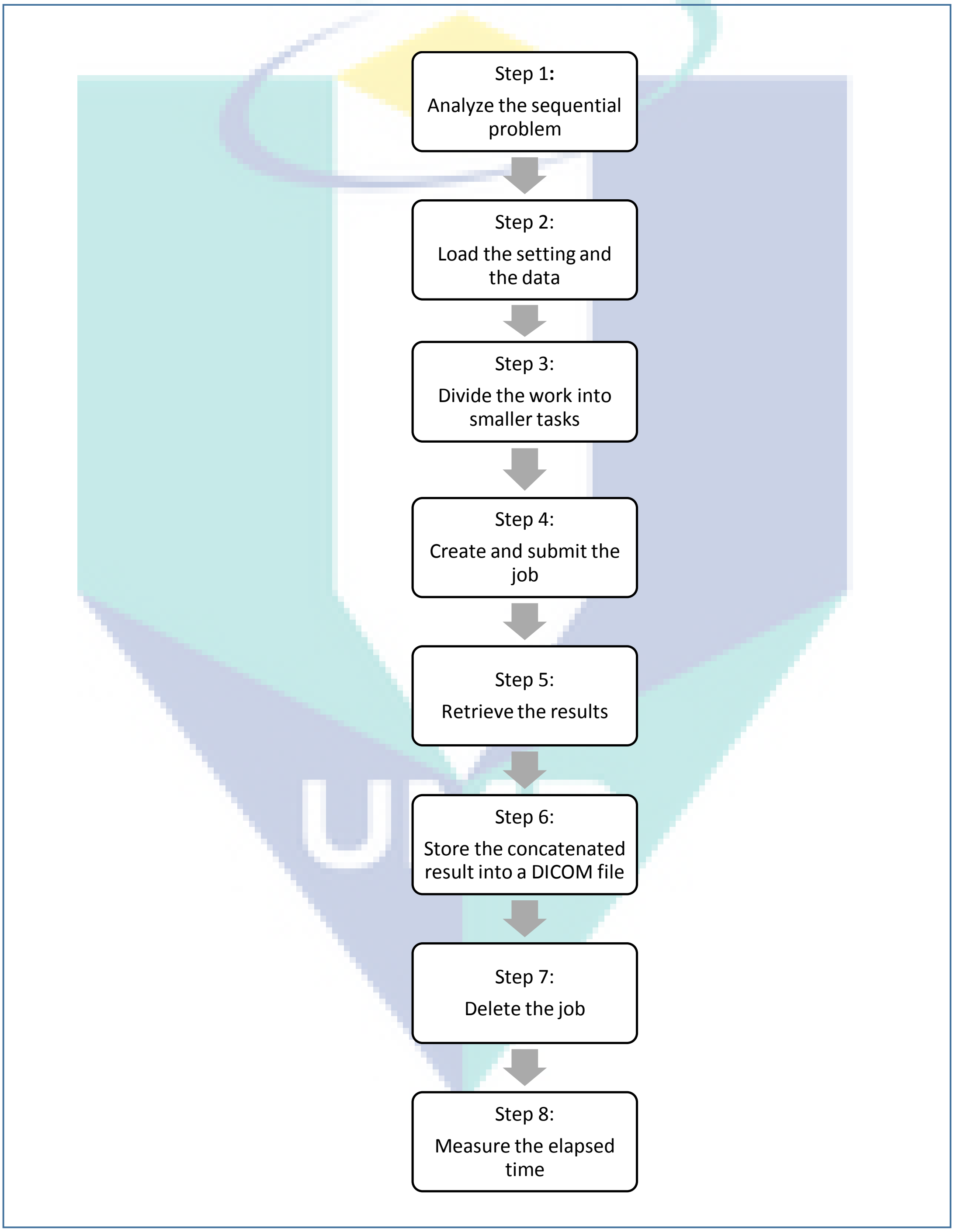


Figure 3.9. Main Algorithm Flow of Parallel Watermarking Process

The implementation details of parallel watermarking embedding process are as follow:

Step 1. Analyze the sequential problem

In multi-frames ultrasound medical images, watermarking is embedded in each frame sequentially and for each frame it will takes only seconds to complete the watermarking embedding process therefore it will need to takes minutes to completed the multi-frames processing. For example a frame takes 5 seconds then 15 frames will take approximately 1 minute to complete the task. Because the function `watermarking_embedding_fxn` can already perform multi-frames watermarking embedding process sequentially in a single function call, therefore it could be used directly as a task function in this program.

Step 2. Load the setting and the data

i. Retrieve a multi-frame ultrasound medical images and stored into `frames` variable.
ii. Identify the frame size and local cluster (microprocessor) used.
iii. Determine the number of cores in a cluster and assign it into a `numTasks` variable. The `numTasks` will return `4` since the cluster is using quad core microprocessor.

The related source code is illustrated as follow:

```
file_name = 'dcm3.dcm';
info = dicominfo(file_name);
frame_size = info.NumberOfFrames;
P=[];
clust = parcluster('local');
numTasks = clust.NumWorkers; % Tasks is split based on number of workers.
startClock = clock;
```

Step 3. Divide the work into smaller tasks

The function `pctdemo_helper_split_scalar` divides the multi-frames ultrasound medical images among the `numTasks` tasks. For example, 15 frames ultrasound images will be divided to 4,4,4,3 frames to four tasks respectively, which is display in `numPerTask`. This means four tasks will perform the watermarking embedding process simultaneously. The related source code is illustrated as follow:

```
[numPerTask, numTasks]= pctdemo_helper_split_scalar(frame_size, numTasks);
disp(numPerTask);
```

Step 4. Create and submit the job

Create the watermarking embedding job and the tasks in the job. Task `i` will perform `numPerTask(i)` watermarking embedding process. Notice that the task function `watermarking_embedding_fxn` is the same function that is used in the sequential version. This function has 3 inputs and 1 output. Keep track of the `frame_no` upon the completion of `createTask`. The related source code and its output are illustrated as below:

```
frame_no=1;
job = createJob(clust);
for i = 1:numTasks
       createTask(job, @watermarking_embedding_fxn, 1,
       {file_name,frame_no, numPerTask(i)});
       frame_no=frame_no + numPerTask(i);
end
```

Submit the job and wait for it to finish.

```
submit(job);
wait(job);
```

Step 5. Retrieve the results

Obtain the job results y, verify that all the tasks finished successfully, and then delete the job. `fetchOutputs` will throw an error if the tasks did not complete successfully, in which case the job needs to be deleted before throwing the error. The related source code is illustrated as below:

```
 try
       y = fetchOutputs(job);
catch err
       delete(job);
       rethrow(err);
  end
```

Format the results. Concatenate all the cells in y into one column vector, and store it into P.

```
disp(y(:,:,:));
P=cat(4,y{:});
```

Step 6. Store the concatenated result into a DICOM file

The related source code and its output are illustrated as below:

```
dicomwrite(P, Parallel_watermarked_multiframes_us.dcm');
display('Complete');
```

Step 7. Delete the job

Delete the job when all the verifications has finished. The related source code is illustrated as below:

```
delete(job);
```

Step 8. Measure the elapsed time

The time used for the distributed computations should be compared against the time it takes to perform in sequential version. The elapsed time varies with the underlying hardware and network infrastructure. The related source code is illustrated as below:

```
elapsedTime = etime(clock, startClock);
fprintf('Elapsed time is %2.1f seconds\n', elapsedTime);
```

3.4 GENERATE WATERMARKED MEDICAL IMAGES

A watermarked medical image was generated by embedding ROI bits into RONI's LSB during watermarking embedding process. The watermarked medical image is used as an input file of watermarking authentication process before clinical diagnosis. The digital watermark(s) should be invisible in medical image, and the watermarked medical image should have a high perceptual similarity to avoid misdiagnosis occurrence. Perceptual similarity measures the degree of imperceptibility between the watermarked image and its original image. The metrics used to measure perceptual similarity in this research are mean-squared-error (MSE) and peak signal-to-noise ratio (PSNR).

The perceptibility of a watermarked image can be judged according to its fidelity and quality. Fidelity measures the similarity between images before and after watermarking (Cox, I.J et al., 2002). A High fidelity means that watermarked image is very similar to the original image. The mean-squared-error (MSE) and peak signal-to-noise-ratio (PSNR) were calculated by comparing the watermarked image and original image. Watermarked images may bear visible or invisible distortion due to the embedding process. One way to quantify distortion is the mean-square error. This is defined as in equation 3.1:

$$MSE = \frac{1}{n}\sum_{i}^{n}(I_i' - I_i)^2 \tag{3.1}$$

which is the average term by term difference between the original image, I_i, and the watermarked image, I_i'. If I_i and I_i' are identical, then MSE (I_i',I_i) = 0. A related distortion measure is the peak signal-to-noise ratio (PSNR), measured in decibels (dB). The problem with mean-square error is that it depends strongly on the image intensity scaling and while PSNR rectifies this problem by scaling the mean-square error according to the image range (Smitha, B. and Navas, K.A., 2007). PSNR is defined as in equation 3.2:

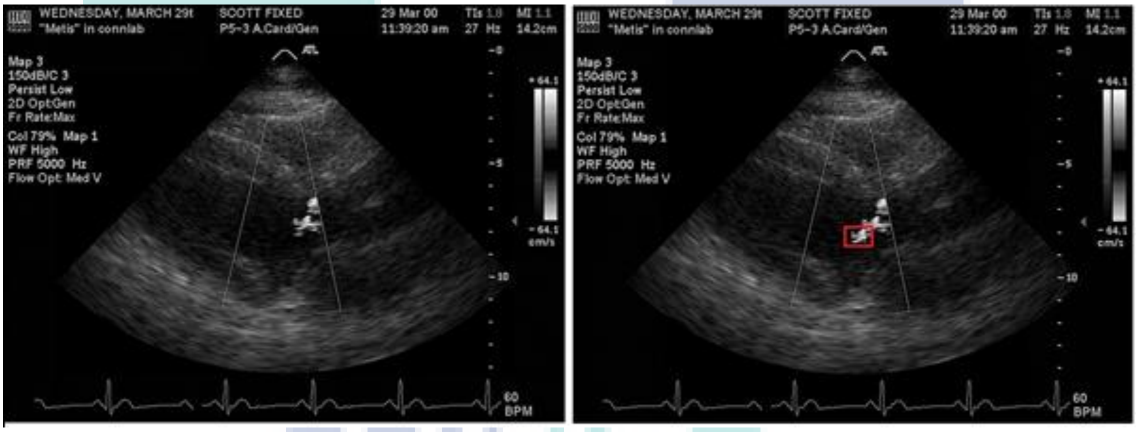
$$PSNR\ (dB) = 10\log_{10}\frac{\max I^2}{MSE'} \tag{3.2}$$

where $\max I$ is the peak value of the original image. If the signals are identical, then PSNR is equal to infinity. A high PSNR represents a high fidelity of a watermarked image.

In this thesis, PSNR is used as a measurement for image fidelity. A high fidelity watermarked image does not have any obvious noticeable distortion caused by the watermark embedding process. Typical values for the PSNR in 8 bits depth images are range from 30 to 50 dB. For 16-bit data typical values for the PSNR are range from 60 to 80 dB (Hamzaoui, R. and Saupe, D., 2006).

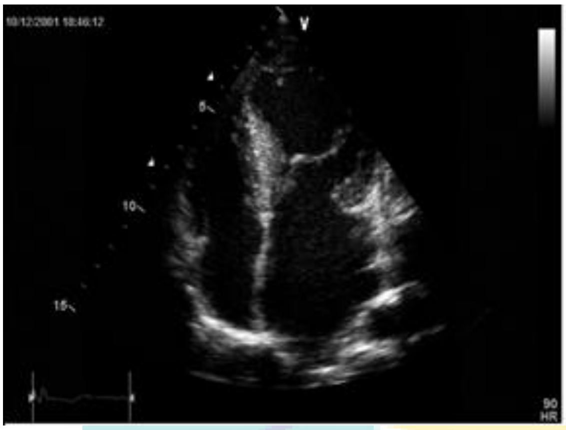
3.5 TAMPERING ON WATERMARKED MEDICAL IMAGES

In order to demonstrate the tamper localization function in detecting forgery, counterfeited images were created by manually modifying the pixel values in the watermarked images using image processing software—ImageJ 1.46r. Watermarked ultrasound medical images was tampered in different manners such as cloning a portion of image into ROI, adding "Pepper and Salt" noise into ROI and RONI, flipping a portion of image in ROI vertically and smoothening some areas in ROI (as shown in Figure 3.10).

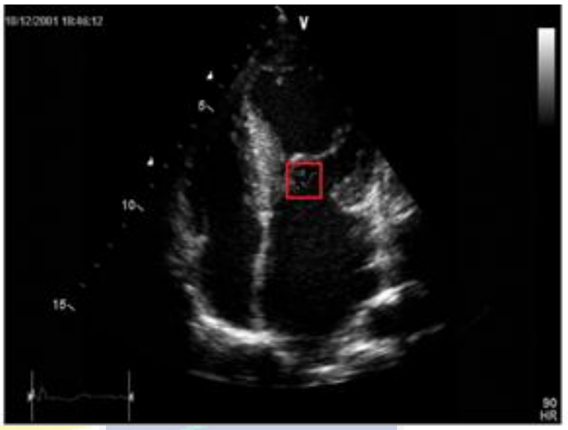


(a) Before tampered by cloning (b) After tampered by cloning

Figure 3.10. Tampering on watermarked ultrasound medical image in different manners.



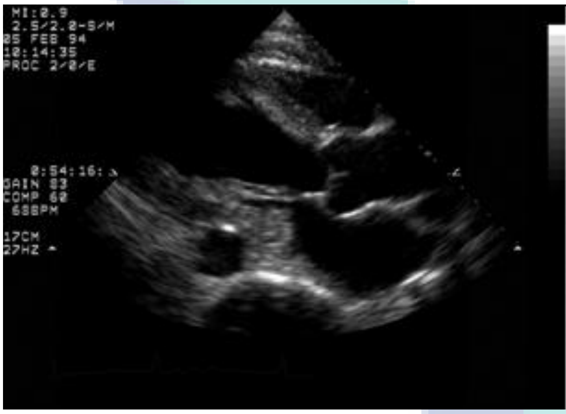
(c) Before tampered by “salt and pepper”



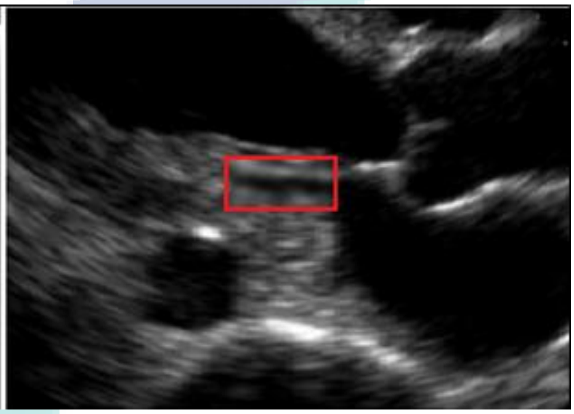
(d) After tampered by “salt and pepper”

(e) Before tampered by flipping

(f) After tampered by flipping



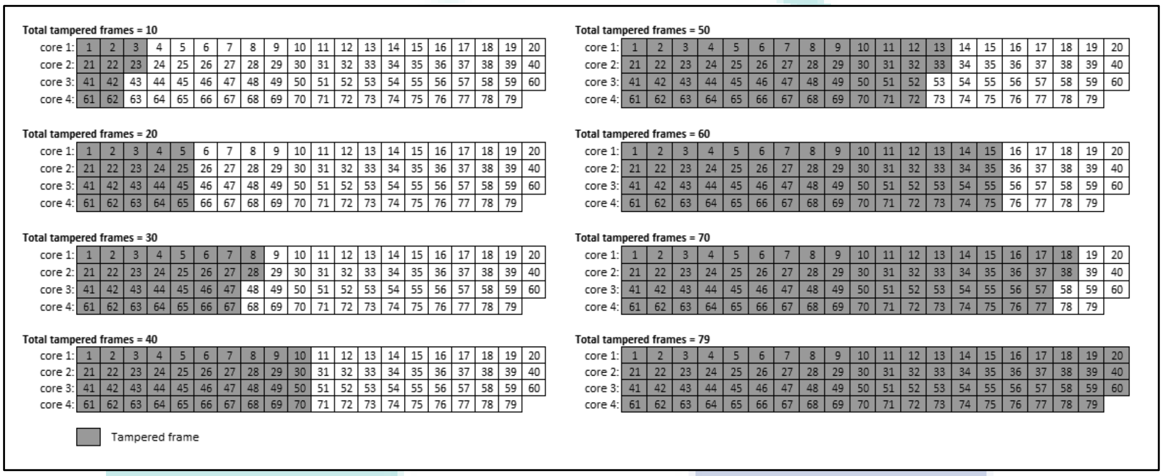
(g) Before tampered by smoothening



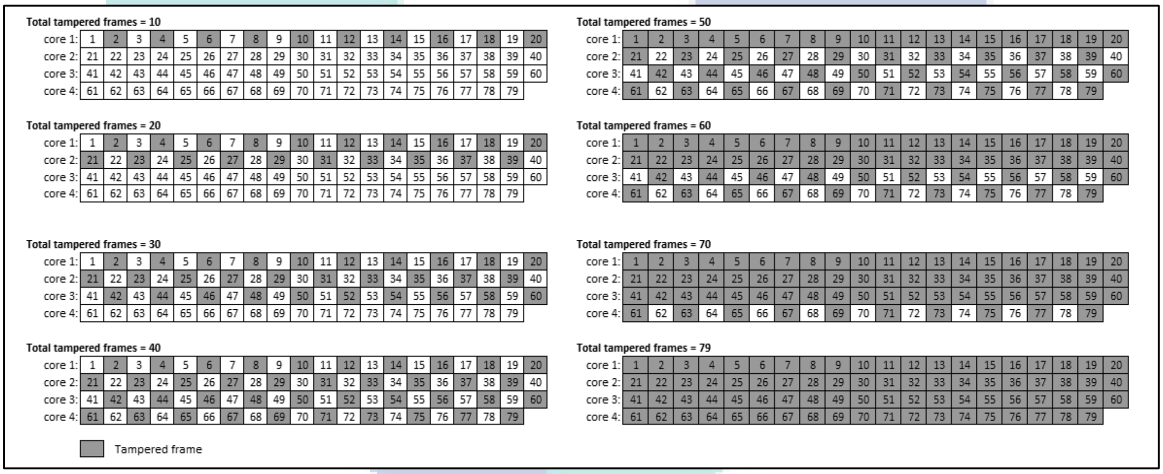
(h) After tampered by smoothening

Figure 3.10. Continued.

Besides tampering on watermarked medical image on different manners as shown in Figure 3.10 (a) to (h). Tampering locations within the medical image is a key factor that determine ROI-DR's robustness toward tampering. Any tampering occurred outside ROI and RONI will be ignored. As for multi-frames medical images, a number of frames will be tampered with a predefined order (as shown in Figure 3.11).



(a) Tampered frames organized evenly among four cores



(b) Tampered frames laid heavily on one sides of the cores

Figure 3.11. Tampered frames organization within the medical images

In theory, watermarking processing time for 20 tampered frames that organized evenly among four cores (as shown in Figure 3.11 (a)) will be faster than 20 tampered frames laid heavily on two cores (as shown in Figure 3.11 (b)). A experiment will be conducted to test on this theory, where 10 tampered frames that organized as (a) against 10 tampered frames that organized as (b), followed by 20 tampered frames organized as

(a) against 20 tampered frames organized as (b) and so on until full 79 tampered frames. The purpose is to check whether the tampered frames organization within multi-frames medical images has a significant impact on the processing time of multi-frames watermarking scheme.

3.6 DEPLOY WATERMARKING AUTHENTICATION PROCESS

3.6.1 ROI-DR Watermarking Authentication Process on Single Frame

Watermarking authentication process on a single frame is to verify whether any tampering has occurred at ROI areas and to perform recovery process once the tampering is detected. SHA-256 hashing method has been applied intensively in this process. The hashed values retrieved from RONI are compared with current hashed values of the examined watermarked medical image, if the hashed ROI bits comparison is positive, then ROI is not tampered else a further verification on tampering occurrence is required, where hashed compressed ROI values is retrieved and compared with the current hashed compressed ROI values of watermarked medical image, if the result is positive, it confirms that ROI has not been tampered with but RONI 3 has been tampered with. If ROI has been tampered with, then the recovery process will be performed. Before the recovery process is performed, it is necessary to ensure that RONI 1 or RONI 2 which stored the ROI bits is not tampered with. This could be accomplished by comparing the retrieved compressed ROI bits from RONI 1 and RONI 2 with current hashed compressed ROI bits. If both of them are equal, which means RONI 1 and RONI 2 have not been tampered with, and is safe to perform ROI recovery process or else tampering has occurred at RONI 1 or RONI 2, which means the stored ROI bits have been tampered with and no longer could be used to recover ROI as intended.

ROI-DR watermarking authentication process based on the program flow chart as shown in Figure 3.12 (a) to (d) are as follow:

(a) Watermarking authentication main process (as shown in Figure 3.12 (a))

1. **Read an Ultrasound Medical Image**: Input an ultrasound medical image file into ROI-DR watermarking system.
2. **Set ROI and RONI rectangle values**: Set the region of interest (ROI) and region of non-interest (RONI) rectangle values, which are [x-coordinate value, y-coordinate value, width, height].

3. **Crop ROI and RONI rectangle regions:** Crop out the image within the ROI and RONI rectangle regions.
4. **Convert all RONI regions to binary format:** Convert all the cropped RONI regions such as RONI_1 until RONI_5 to binary format and ready their last two least significant bits (LSB) for ROI bits and hashed values storage.
5. **Hash ROI region with SHA-256 to generate ROI_hash_bin:** Hash the cropped ROI region with SHA-256 to generate ROI_hash_bin and stored into RONI_3 region, in which later will be retrieved and used in watermarking authentication process.
6. **Retrieve ROI_hash_bin_retrieved from RONI 3:** Retrieve ROI_hash_bin that stored RONI 3 during the watermarking embedding process.
7. **ROI_hash_bin retrieved is equal to ROI_hash_bin?:** If ROI_hash_bin retrieved from process (6) is equal to ROI hash_bin generated from process (5), then it means no tampering has occurred on ROI region and the authentication process is terminated, or else it means tampering has occurred and a further verification is required to check whether it has occurred on ROI or RONI. Hence, proceed to process (b).

(b) Verification process on checking whether tampering is occurred on ROI or RONI (as shown in Figure 3.12(b))

1. **Compress ROI to jpeg format, Compressed_ROI:** ROI is compressed into JPEG and store as a variable named Compressed_ROI.
2. **Convert Compressed ROI into binary format, Compressed_ROI_bin:** Compressed_ROI that generated form process (1) above is convert into binary format and store as a variable named Compressed_ROI_bin
3. **Hash Compressed_ROI with SHA-256, hash_compressed_roi:** Compressed_ROI that generated from process (1) above is hashed with SHA-256 and store as a variable named hash_compressed_roi
4. **Retreive Compressed_ROI from RONI 4:** Retrieve compressed_ROI_hash_bin that stored in RONI 4 during the watermarking embedding process.
5. **Comp_ROI is equal to Comp_ROI_retrieved?**: If compressed_ROI_hash_bin retrieved is equal to hash_compressed_roi generated in step 3, then it means no tampering has occurred on ROI but has occurred on RONI 3 region instead, or else tampering has occurred on ROI region. Hence a further investigation is

required to ensure that tampering has not occurred on RONIs that stored ROI bits before proceeding to ROI recovery which leads to process (c).

(c) A process that ensures tampering has not occurred on RONIs that stored ROI bits before proceeding to ROI recovery (as shown in Figure 3.12 (c))

1. **Retrieve compressed_roi_size from RONI 5 region:** Retrieved the compressed_roi_size from RONI 5 in which it has been stored during the watermarking embedding process.
2. **m = compressed_roi_size/2:** Retrieve compressed ROI_size_bin from RONI 5 and divide it into half to generate m.
3. **Retrieve roi_1_bin and roi_2_bin from RONI 1 and RONI 2 respectively based on m values:** The length of m bits was retrieved from RONI 1 and RONI 2, the bits retrieved was stored as roi_1_bin and roi_2_bin respectively.
4. **Reassemble roi_1 bin and roi_2_bin into one binary form, compressed_ROI_2:** Merge roi_bin_1 and roi_bin_2 into one binary form, and stored as a variable named compressed_ROI_2.
5. **Hash compressed_ROI_2 with SHA-256 to generate comp_roi_hash_bin_2**
6. **Comp_roi_hash_bin_2 is equal to ROI_hash_bin_retrieved?:** Compare compressed_ROI_hash_bin_2 with values retrieved from RONI 4. If the result is equal then proceed to process (d), ROI recovery process or else it means tampering has occurred on RONIs which stored ROI bits, thus no recovery could be done.

(d) A process that recover the ROI region (as shown in Figure 3.12 (d))

This process has decompress the compressed_ROI_2 and reshape it according to ROI region size, and then replace it into ROI region. Lastly write the whole image into a DICOM file.

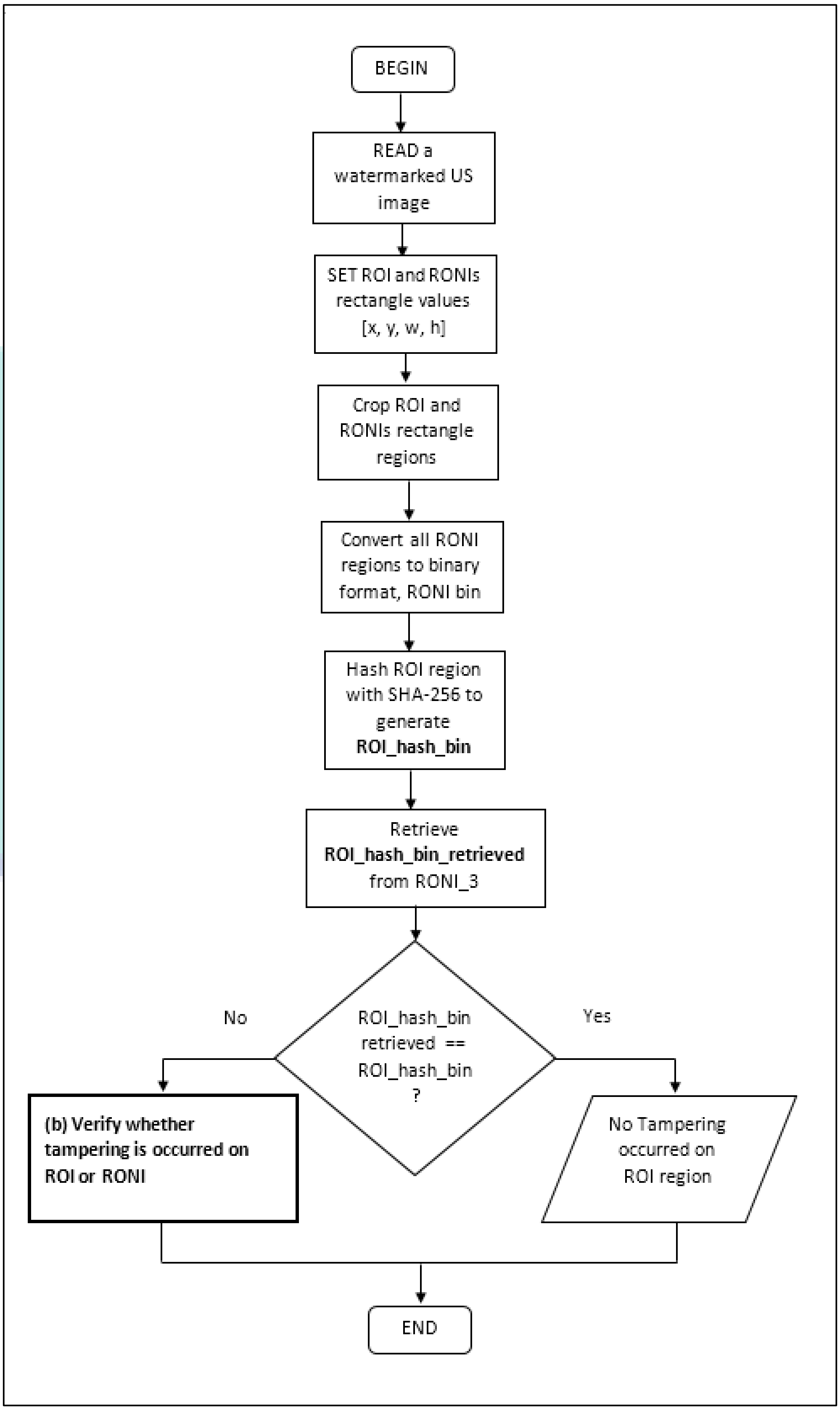


Figure 3.12(a) Watermarking Authentication Main Process

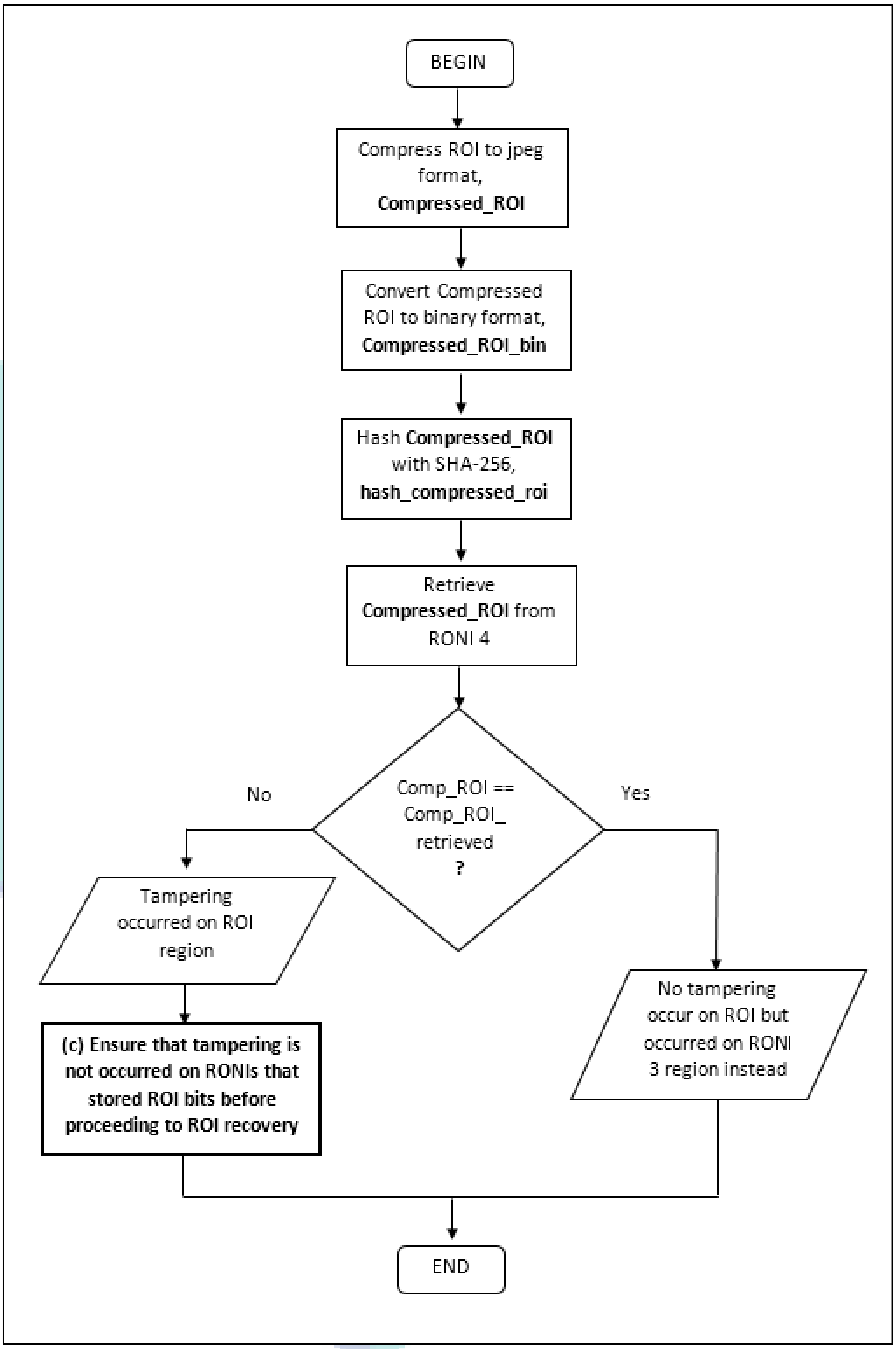


Figure 3.12(b) Process that verify whether tampering is occurred on ROI or RONI

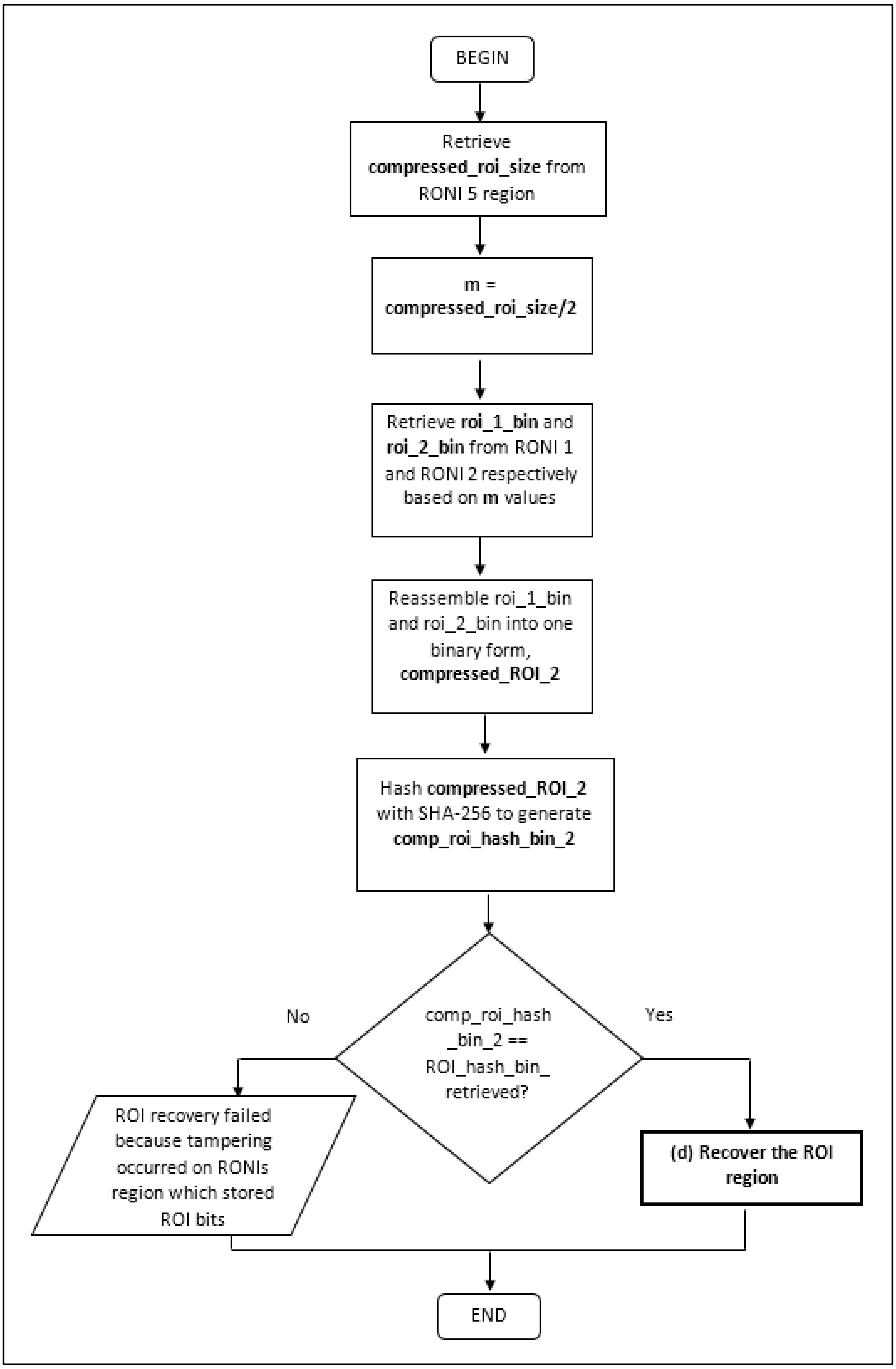


Figure 3.12(c) Process that ensures tampering is not occurred on RONIs that stored ROI bits before proceeding to ROI recovery

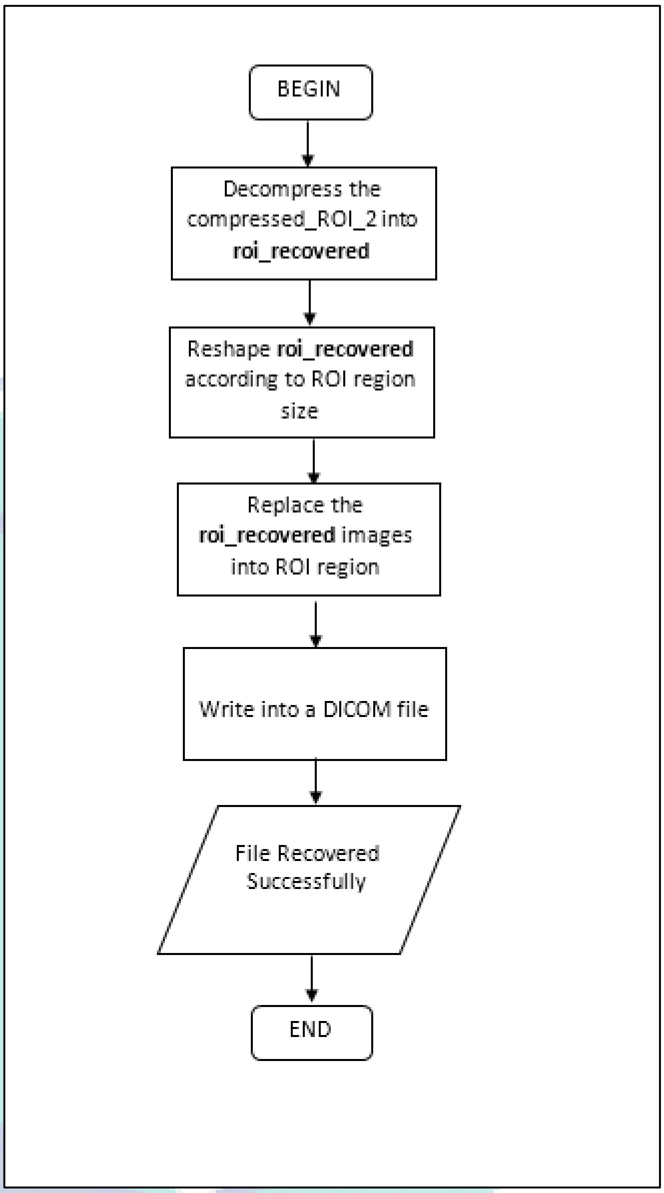


Figure 3.12(d) Process that recover the ROI region

3.6.2 Multi-Frames Watermarking Authentication Process

As for multi-frames watermarking authentication process (as shown in Figure 3.13), it has a similar process flow as multi-frames watermarking embedding process, except the input file of authentication process was the tampered output file of embedding process, and the output files was a message of tampered frame number and a recovered DICOM file.

The implementation details of parallel watermarking authentication process are as follow:

Step 1 and 8 of parallel watermarking authentication process are similar as parallel watermarking embedding process as described in section 3.3.2 part II, except in step 2, 4 and 5, which will be describe in details as follow.

Step 2. Load the setting and the data

i. Retrieve a tampered multi-frame ultrasound medical images and stored into `frames` variable.
ii. Identify the frame size and local cluster (microprocessor) used.
iii. Determine the number of workers/cores in a cluster and assign it into a `numTasks` variable. The `numTasks` will return `4` since the cluster is using quad core microprocessor. The related source code is illustrated as below:

```
file_name='tampered_WM_images_3_frames.tif';
frame_size=length(imfinfo(file_name));
P = [];
clust = parcluster('local');
numTasks = clust.NumWorkers; % split into this many tasks.
startClock = clock;
```

Step 4. Create and submit the job

Create the watermarking authentication job and the tasks in the job. Task `i` will perform `numPerTask(i)` watermarking embedding process. Notice that the task function `watermarking_authenticate_fxn` is the same function that is used in the sequential version. This function has 3 inputs and 2 outputs. Keep track of the `frame_no` upon the completion of `createTask`. The related source code is illustrated as below:

```
frame_no=1;
job = createJob(clust);
for i = 1:numTasks
       createTask(job,@watermarking_authenticate_fxn,2,{file_name,frame_no,
       numPerTask(i)});
       frame_no=frame_no + numPerTask(i);
end
```

Step 5. Retrieve the results

Obtain the job results y, verify that all the tasks finished successfully, and then delete the job. `fetchOutputs` will throw an error if the tasks did not complete successfully, in which case the job needs to be deleted before throwing the error. The related source code is illustrated as below:

```
       Try
              y = fetchOutputs(job);
       catch err;
              delete(job);
              rethrow(err);
       end
```

Format the results. Concatenate all the cells in y into two column vector, and store it into P and Result. P is the collection of frames output and Result is contain a string message of tampered frame. The related source code is illustrated as below:

```
       disp(y(:,:,:));
       P=cat(4,y{:,1});
       Result=[y{:,2}];
       fprintf('Tampered happen at: \n %s\n\n', Result);
```

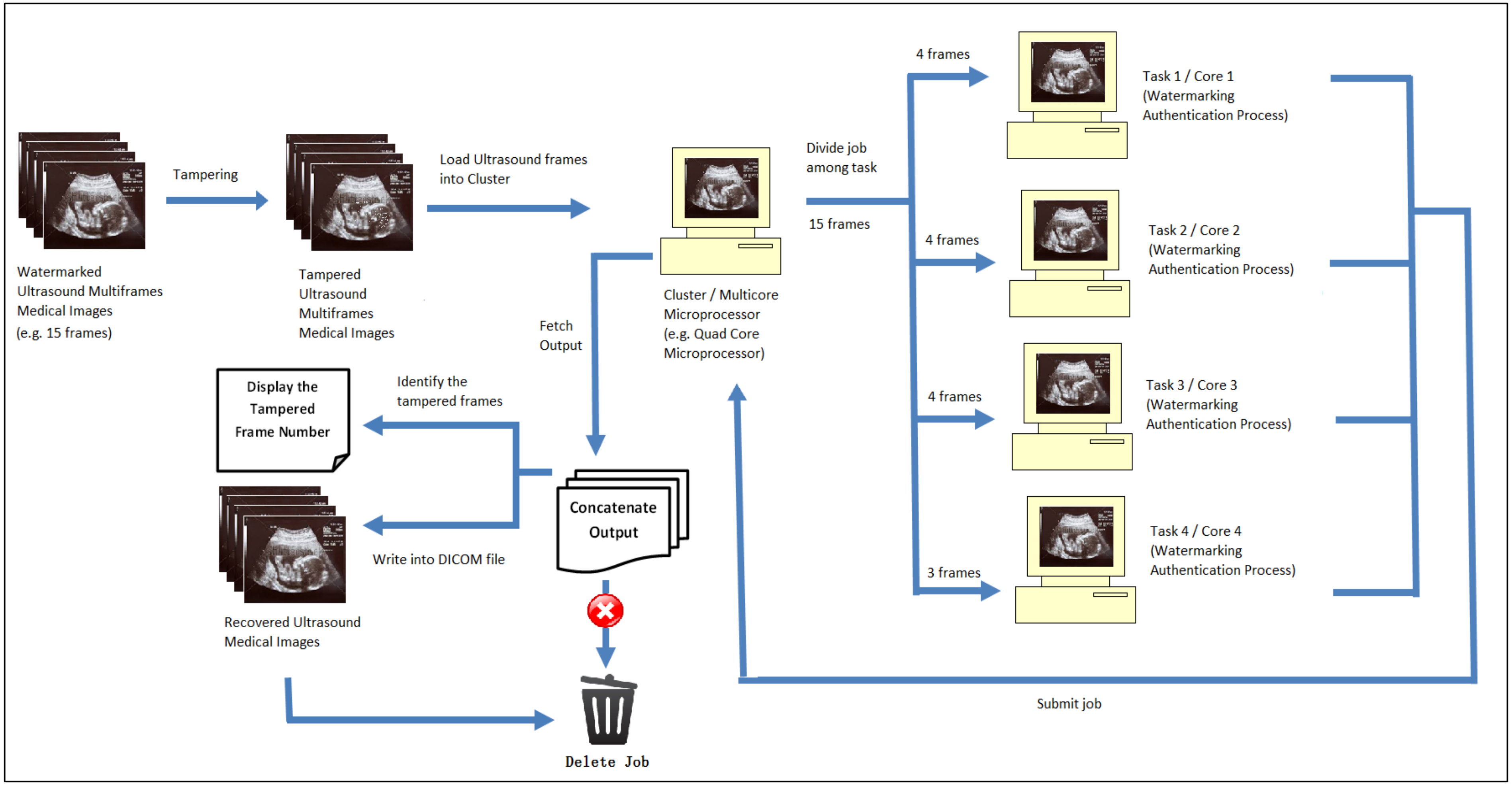


Figure 3.13. Multi-frames watermarking authentication process

3.7 GENERATE TAMPERING MESSAGE AND A RECOVERED DICOM FILE

Watermarking authentication process will generate a tampering message when there is a tampering occurred in ROI of watermarked medical image. The tampered ROI will be replaced by the ROI bits that extracted from RONIs. After the authentication process, the recovered file will save as a DICOM format. The DICOM file will fail to be recovered if the tampering occurred in the RONIs that stored ROI bits.

3.8 COMPARE ROI-DR WITH TALLOR AND TALLOR-RS ALGORITHM

TALLOR watermarking scheme experiment (Liew, S.C., 2011) revealed that significant amount of time was taken to embed and retrieve the JPEG file. This directly slows down the process of watermarking and authentication. It can be an issue when a user of an image has to spend a significant amount of time waiting for the image to be authenticated and recovered. TALLOR watermarking scheme has divided RONI into non-overlapping blocks of 2 x 2 pixels, and inserting one pixel (8 bits) of ROI into each block of RONI as shown in Figure 3.14. For example, one pixel (8 bits) of ROI consist of 1011 0001 has split into four pairs of bits, each pair of bits was replacing the last two LSB of 2 x 2 pixels block in RONI. The embedding process is conducted from left to right and then top to bottom in the RONI region.

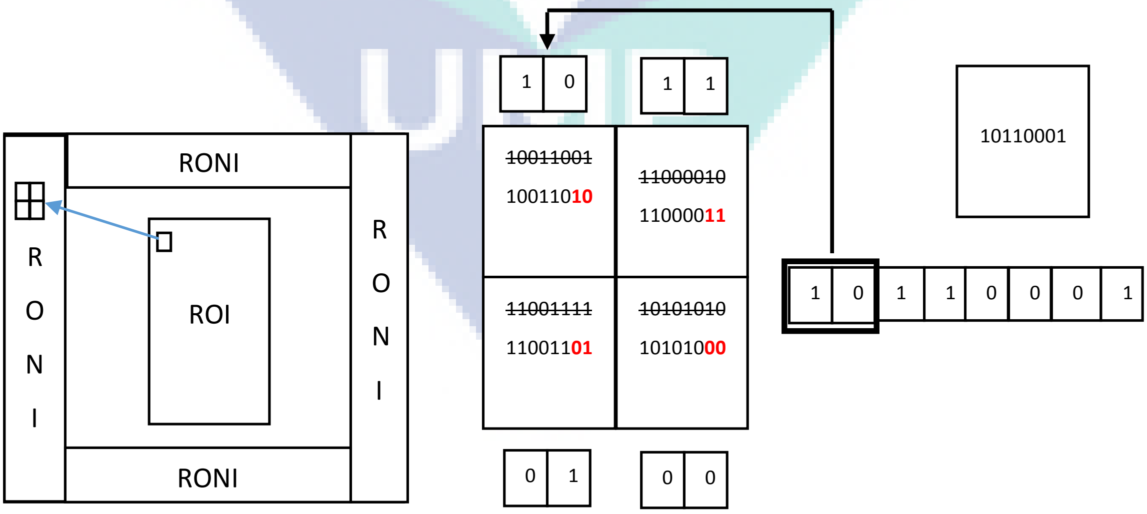


Figure 3.14. TALLOR watermarking embedding process

Whereas ROI-DR watermarking scheme reshapes ROI and RONI pixels into single column matrix and one pixel of ROI (8 bits, such as 1011 0001) will split into 4 pairs (such as 10,11,00,01) and replace four pairs of LSB in RONIs as shown in Figure 3.15. The embedding process is conducted from top down and then left to rigth in the RONi region. At the end of the process, the single column matrix of RONI will be reshape to its original form and placed back into medical image (as shown in Figure 3.16).

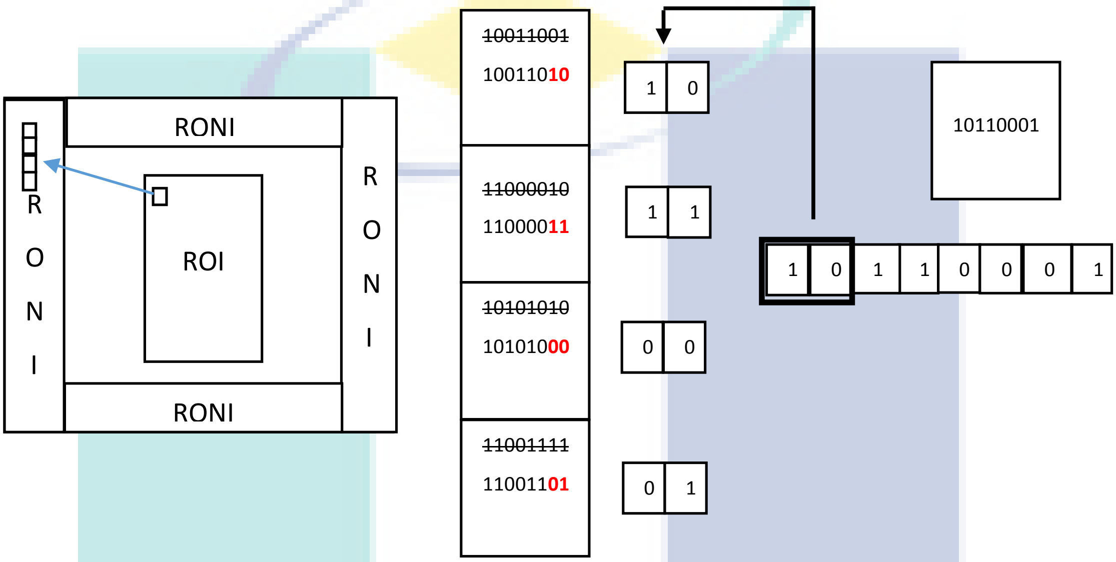


Figure 3.15. ROI-DR watermarking embedding process

The embedding process in TALLOR was rigid because RONI size has to be pre-calculated in order to fit 2 x 2 pixels blocks, for example 2 x 2 pixels blocks will required RONI size (width x height) to be an even number, such as 4 x 12 of RONI size will be able to fit 2 x 6 = 12 of 2 x 2 pixels blocks, whereas 3 x 12 of RONI size could only fit 1 x 6 = 6 of 2 x 2 pixels blocks and still remaining one column wasted in RONI. Whereas ROI-DR embedding process has no such issue due to the reshaping methods as shown in Figure 3.16 where it is not restricted by any block size and it could utilize the available space in maximum.

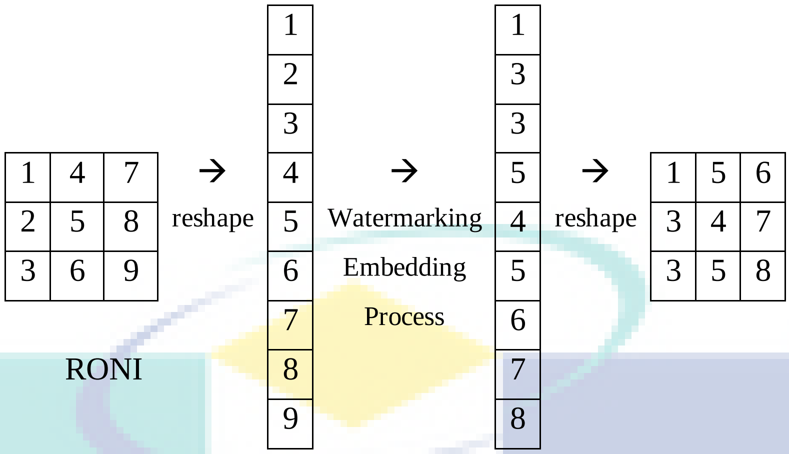


Figure 3.16. ROI-DR reshape M x N matric into single column matric and vice versa in watermarking embedding process

TALLOR watermarking scheme embeds the compressed ROI pixel by pixel into the RONI, since RONIs are located at different places in medical image, therefore RONI block location variable is required for tracking the current position of ROI bits storage. Block location variable is dynamic and will change during the embedding process therefore global memory is required for block location calculation which is high memory consumption and will affect the performance. Whereas ROI-DR watermarking scheme has eliminate the need of block location variable by splitting ROI into 2 segments and store each segment into different RONI blocks, therefore it has a better performance than TALLOR.

TALLOR authentication process go through pixel by pixel to check whether there is any difference between extracted ROI and existing ROI pixel values, if the result shows negative then the tampered pixel in existing ROI will be localized and replaced with extracted ROI pixel values. The process will repeat until the whole ROI pixels was checked. Whereas ROI-DR will stop the authentication process at the first encounter of tamper pixel and replacing the whole existing ROI image with the extracted ROI image. This has cut short the processing time significantly.

TALLOR-RS is the improved version of TALLOR by further segmented ROI into non-overlapping blocks of 40 x 40 pixels and the RONI is divided into non-overlapping blocks of 2 x 2 pixels. TALLOR-RS has used CRC to authenticate the segments of the ROI individually. Only the segments that were tampered with will be retrieved from the RONI for recovery purposes. Since the ROI is to be divided into segments, each segments needs to be authenticated individually. The authentication can be performed in a multilevel manner where only suspected segments will be examined further for tampering. As a result, it has reduced the authentication processing time as compare to TALLOR but it is not the case in watermarking embedding process because it needs additional processing time to implement CRC before watermarking embedding process. TALLOR-RS authentication process was faster than TALLOR if ROI is tampered but it will be slower if ROI is not tampered because TALLOR-RS will perform CRC checking from block to block in ROI until a tampering is detected. If there is no tampering occurred, then CRC will perform until the end of the blocks, therefore non-tampered ROI consume more time than tampered ROI. As for ROI-DR watermarking scheme hash algorithm has been used for authentication purposes and ROI recovery will perform immediately if the computed and extracted ROI hashed values is different. Therefore the time taken for ROI-DR is much shorter as compare to TALLOR and TALLOR-RS.

3.9 PERFORMANCE MEASUREMENT

As for performance comparison, ROI-DR watermarking scheme will compare with existing watermarking schemes (TALLOR and TALLOR-RS) in term of its processing/elapsed time. These three watermarking schemes has been conducted at the same experimental environment, such as same ultrasound samples and hardware and software testing environment. The elapsed time was obtained by recording the whole watermarking processing time. The result of total elapsed time of these three schemes will be collected and a speedup factor of TALLOR and TALLOR-RS relative to ROI-DR scheme will be calculated. The speedup factor of different watermarking schemes in a single frame is defined in equation 3.3:-

$$Speedup_{single\ frame} = \frac{\text{Elapsed time gain in exsiting watermarking process}}{\text{Elapsed time gain in proposed watermarking process}} \tag{3.3}$$

Different set of tampered watermarked ultrasound medical images has been used to test the effectiveness and efficiency of the tamper detection and recovery function in multi-frames environment. The function’s effectiveness was measured by checking whether it could detect and determine the tampering area, and able recovered ROI to its original form. The function’s efficiency was measured by calculating the speedup factor which should be above the threshold value, which means more than 1, where the elapsed time gain in ROI-DR watermarking scheme must lesser than TALLOR and TALLOR-RS in order achieved the efficiency testing requirement. Both effectiveness and efficiency testing were performed while testing the function's robustness to tampering.

As for multi-frames watermarking scheme, the speedup factor is defined in equation 3.4:-

$$Speedup_{Multiframes} = \frac{\text{Elapsed time gain in sequential watermarking process}}{\text{Elapsed time gain in parallel watermarking process}} \tag{3.4}$$

It is to measure the speedup factor of multi-frames watermarking scheme in parallel mode relative to sequential mode. For example, if the speedup factors obtained was 3, it means multi-frames watermarking scheme in parallel mode was 3 times faster than in sequential mode. This speedup factor in equation 3.4 could only applied for the same watermarking scheme, it is used to check whether parallelism has improvement in speed over the conventional method. If the speedup value is greater than 1 then parallel watermarking process is faster than sequential mode or vice versa. If the speedup value is one, it means than parallel and sequential watermarking processing have the same elapsed time. As for performance comparison with existing watermarking schemes in multi-frames environment, the speedup factor is measured similar as equation 3.3, which is used to measure speedup factor of different watermarking schemes.

3.10 EXPERIMENTAL DESIGN

Three important performance metrics that used for both ROI-DR watermarking scheme and multi-frames watermarking scheme are listed as follow:

1. Imperceptibility – this is to test the quality of medical images in term of invisibility of watermarking.
2. Elapsed Time - The time taken to perform watermarking embedding and authentication process on medical images. It is a variable used to measure speedup factor.
3. Robustness to tampering – This is to test the effectiveness and efficiency of the tamper detection and recovery function.

3.10.1 Experimental Design for ROI-DR on Single Frame

The research methodology flow as shown in Figure 3.1 has provided a perspective on digital watermarking experimental design.

Experiment will be focus on watermarking embedding and authentication process. Six samples of ultrasound medical images in DICOM format will be prepared as input files, and PSNR are used to measure the imperceptibility and integrity of watermarked ultrasound medical images generated, the higher PSNR values reflected the better image integrity, which mean less distortion in image. As for robustness to tampering, several tampering methods were applied on the watermarked ultrasound medical images generated, such as cloning a portion of image into ROI area, adding salt and pepper noise into ROI area, smoothening certain areas of ROI, flip a portion of image vertically and filled a portion of images with black color. Tampering will be applied at: (1) ROI areas, (2) RONIs areas, (3) both ROI and RONI areas, (4) outside ROI and RONI areas, for testing purposes. All of this tampering will change image pixel values and detected by watermarking authentication process, which will then display a message if there is any tampering occurred. As a result, ROI will be restored and generate a recovered DICOM file, in which ROI recovery will be tested by PSNR values on ROI areas. Elapsed time will be measured and used to make a comparison with previous research, TALLOR and TALLOR-RS (Liew, S.C., 2011), it is to prove that the current research, ROI-DR watermarking scheme has a better performance than TALLOR and TALLOR-RS.

A comparison on elapsed time of the current research (ROI-DR watermarking scheme) with previous research, TALLOR and TALLOR-RS (Liew, S.C., 2011) has been done under the same environment and conditions, such as same set of ultrasound medical image sources, same ROI size (160 X 240 pixels), same tampered areas in ROI and same testing environment. It is to ensure fairness in comparison. The main difference between ROI-DR, TALLOR and TALLOR-RS are: RONI layout and organization, the way of storing ROI bits into RONI LSB and its recovery method.

3.10.2 Experimental Design for Multi-Frames Watermarking Scheme

Two modes of watermarking process in multi-frames will be developed and compared; there are sequential (by using for loop and array manipulation) versus parallel watermarking process (watermarking on multicores). The purpose is to prove that the parallel watermarking will have a significant speedup on elapsed time.

Three same set of performance metrics were applied on multi-frames environment instead of single frame. These are:-

1. Imperceptibility – this is to test the quality of medical images in terms of invisibility of watermarking in multi-frames environment. PSNR and MSE values will be measured in each frame on both sequential and parallel watermarking embedding process. The result will be compared and it is expected that both modes will produce similar result on the same ultrasound medical image, since the watermark was embedded at the same location. The PSNR values are expected above 45 dB in order to achieve imperceptibility requirement and acceptable for diagnosis purposes.
2. Elapsed Time - The time taken to perform watermarking embedding and authentication process on medical images in multi-frames environment. The elapsed time collected from sequential and parallel watermarking process will be compared and used in measuring a speedup factor. The speedup factor determines the efficiency and hence suitability of parallelism adoption in watermarking scheme.
3. Robustness to tampering – This is to test the effectiveness and efficiency of the tamper detection and recovery function in multi-frames environment. "Salt and pepper" noises was applied on different frame number strategically, such as (1) distribute the tampered frame evenly on each core; (2) distribute tampered frame heavily on one core but none-tampered frames on others cores; (3) same tampering applied on different set of ultrasound medical images. All this tampering method is to see whether the way of tampering will have impact on parallelism efficiency while testing watermarking authentication functionality towards tampering.

3.11 EXPERIMENTAL SETUP

The evaluation was performed by running MATLAB program on a standalone computer with quad-core CPU of Intel) i7-4790 CPU @ 3.60 GHz, 3601 MHz, with RAM of 4 GB. Eight samples of ultrasound medical images in DICOM format were used to test the system (as shown in Table 3.1 and Figure 3.3). Existing watermarking schemes

(TALLOR and TALLOR-RS) were used to compare with the proposed watermarking scheme (ROI-DR) in term of PSNR and speedup factor.

3.12 CONCLUSION

ROI-DR and multi-frames watermarking schemes were undergoing the same pattern of research methodology. The difference between them was ROI-DR is emphasized on the watermarking algorithm in a single frame and multi-frames watermarking scheme provide a single code base for watermarking scheme to be execute in multi-frames environment. By executing multi-frames watermarking in parallel, it was foresee that it could reduce the processing time significantly as compared to sequential watermarking process.

The main purpose of ROI-DR and multi-frames watermarking scheme are to improve the watermarking processing time without compromising the integrity of medical image. Therefore, the algorithm design of ROI-DR is focus on code optimization. The algorithm design of multi-frames watermarking scheme is focus on parallelism on multi-frames environment. TALLOR and TALLOR-RS was selected for performance comparison is because ROI-DR has optimized the code based on TALLOR and TALLOR-RS. Imperceptibility and robustness to tampering is to ensure ROI-DR has fulfilled the basic requirements of watermarking scheme. Elapsed time is a variable used in measuring speedup factor, which is a key variable for performance comparison. Sequential watermarking process is important because it is a successful key for multi-frames watermarking scheme development and it also is a variable in measuring speedup factor for multi-frames watermarking scheme (which are in parallel relative to sequential watermarking process).

CHAPTER 4

RESULT

4.1 INTRODUCTION

Three performance metrics that used for both ROI-DR and multi-frames watermarking schemes are: (1) Imperceptibility, (2) Elapsed time, and (3) Robustness to tampering. The elapsed time data will be recorded while collecting data for (1) and (3).

4.2 RESULT ANALYSIS OF ROI-DR IN A SINGLE FRAME ULTRASOUND MEDICAL IMAGE

Imperceptibility (which measured in PSNR) and elapsed time data will be collected during the watermarking embedding process. Robustness to tampering and its elapsed time will be measured during the watermarking authentication process.

4.2.1 Imperceptibility and Elapsed Time in Watermarking Embedding Process

The proposed watermarking scheme (ROI-DR) was compared with existing watermarking scheme (TALLOR and TALLOR-RS) in term of PSNR and the elapsed time of watermarking embedding process. Speedup factor is derived by dividing elapsed time of existing watermarking scheme over elapsed time of proposed watermarking scheme. The purpose of the test is to check whether ROI-DR would achieved a significant speedup without compromised the medical image integrity.

Table 4.1

PSNR Values of Three Different Watermarking Schemes: ROI-DR, TALLOR and TALLOR-RS Watermarking Embedding Process

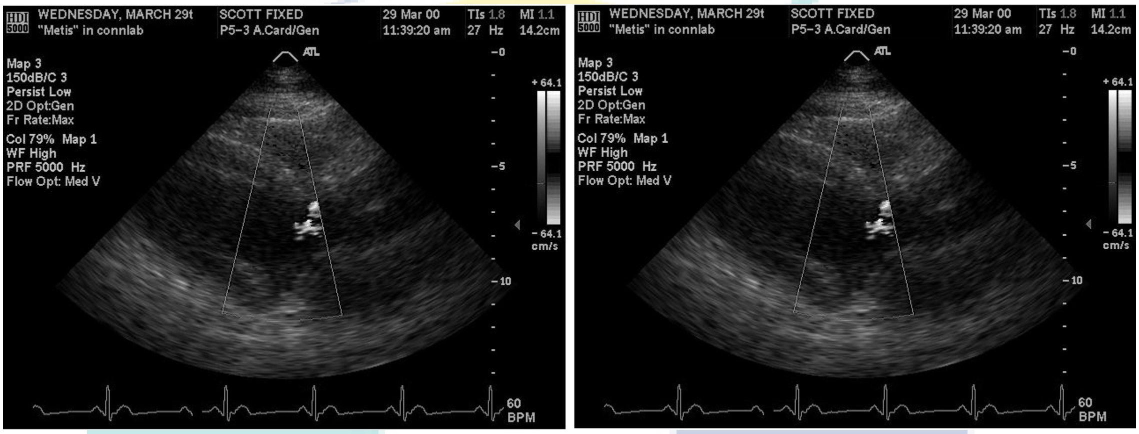
Watermarking Embedding Process						
	ROI-DR		**TALLOR**		**TALLOR-RS**	
Ultrasound samples	**PSNR (dB)**	**Elapse time (seconds)**	**PSNR (dB)**	**Elapse time (seconds)**	**PSNR (dB)**	**Elapse time (seconds)**
US_1	47.953	1.024	47.944	26.375	48.179	29.348
US_2	47.982	0.978	47.986	22.740	48.530	26.900
US_3	48.782	0.971	48.739	21.125	48.819	25.951
US_4	48.107	0.978	48.265	23.871	49.547	25.038
US_5	48.330	0.967	48.302	19.814	48.991	25.130
US_6	49.330	0.950	49.244	18.898	49.591	24.148
Average	**48.414**	**0.978**	**48.413**	**22.137**	**48.943**	**26.086**

Table 4.2

Speedup Factor of TALLOR and TALLOR-RS Relative to ROI-DR Watermarking Embedding Process

Ultrasound samples	**Speedup factor in relative to ROI-DR**	
	TALLOR	**TALLOR-RS**
US_1	25.769	28.674
US_2	23.257	27.512
US_3	21.612	26.733
US_4	24.407	25.601
US_5	20.387	25.989
US_6	19.889	25.415
Average	**22.554**	**26.654**

The PSNR values and elapsed time obtained from US_1 to US_6 are almost similar therefore the average values of PSNR and elapsed time were calculated and used for result analysis. This experiment has shown that the ROI-DR has a better performance in elapsed time relative to TALLOR and TALLOR-RS, where the average speedup factors are 22.554 and 26.654 for TALLOR and TALLOR-RS respectively in relative to ROI-DR Watermarking Embedding Process (as shown in Table 4.2). In other words, it means that ROI-DR was 22.554 times faster than TALLOR, and 26.654 times faster than TALLOR-RS in watermarking embedding process. The PSNR values is varied based on the RONI LSB which stored the ROI bits, the more coincidence of same value in ROI bits and RONI LSB values, the higher is the PSNR values. Table 4.1 shown that the average PSNR values of all three watermarking schemes are similar (48.41 ~ 48.94 dB) and have fulfilled the imperceptibility requirement, where ultrasound medical images before and after watermarking process are visually undisguisable (as shown in Figure 4.1). In other words, the quality of medical image was not degraded after the watermarking embedding process.



(a) Before watermarking embedding process (b) After watermarking embedding process

Figure 4.1. Ultrasound medical image before and after watermarking embedding process

4.2.2 Robustness To Tampering and Elapsed Time in Watermarking Authentication Process

A test plan for testing the robustness of watermarking towards tampering are shown in Table 4.3, in which derived six test cases as shown in Table 4.4 – 4.10.

Test case number 1 test on non-tampering ultrasound medical image, it is to check whether the ROI-DR watermarking scheme able to generate watermarked medical image as expected.

In test case number 2, ROI of US images are tampered in different ways, such as cloning a portion of image into ROI, adding salt and pepper noise into ROI, flipping a portion of image in ROI vertically and smoothening some areas in ROI. It is to test on ROI-DR robustness to tampering and to check whether it will display ROI tampered message and produced a recovered DICOM file.

In test case number 3, elapsed time of TALLOR, TALLOR-RS and ROI-DR watermarking authentication process were collected and derived a speedup factor for

ROI-DR relative to TALLOR and TALLOR-RS respectively. It is to test whether ROI-DR will achieved a significant speed up relative to TALLOR and TALLOR-RS.

In test case number 4, RONIs of US image was tampered. Any tampering occurred in RONIs will be ignored because of ROI is not tampered. Tampered on RONI 3 will generate a different message with other RONIs. It is because RONI 3 stored ROI hash key, which is used to determine whether ROI is tampered. A tampered RONI 3 will have change ROI hash key value, and is no longer valid for ROI tamper detection. Since ROI was not tampered, no recovery file was required.

In test case number 5, a tampering was applied outside of ROI and RONI rectangle area. It is an area that has not been covered by ROI-DR algorithm, therefore it is expected that any tampering occurred in this area will be ignored.

In test case number 6, a tampering was applied on both ROI and RONI. It is to check whether ROI-DR will perform watermarking recovery process if tampering occurred in ROI and RONIs that stored ROI recovery bits, such as RONI 1 and 2.

Table 4.3

Test Plan for Testing the Robustness of Watermarking to Tampering

Test No	**Description**
1	Test on non-tampering watermarked US images
2	Test on where ROI of US images are tampered in different ways
3	Comparison elapsed time between TALLOR, TALLOR-RS and ROI-DR watermarking authentication process
4	Test on where RONIs of US image is tampered
5	Test on where tampering is occurred outside of ROI and RONI rectangle area
6	Test on where both ROI and RONI were tampered

Table 4.4

Test Cases for Test No.1: Test on Non-tampering Watermarked US Images

Test Data	Elapsed time (seconds)	Expected Result	Actual Result
Watermarked US_1	0.0490	Display "***ROI Hash is Equal, Therefore NO Tampering occurred!!***" message	Same as expected result
Watermarked US_2	0.1085	Display "***ROI Hash is Equal, Therefore NO Tampering occurred!!***" message	Same as expected result
Watermarked US_3	0.1154	Display "***ROI Hash is Equal, Therefore NO Tampering occurred!!***" message	Same as expected result
Watermarked US_4	0.0585	Display "***ROI Hash is Equal, Therefore NO Tampering occurred!!***" message	Same as expected result
Watermarked US_5	0.1062	Display "***ROI Hash is Equal, Therefore NO Tampering occurred!!***" message	Same as expected result
Watermarked US_6	0.0421	Display "***ROI Hash is Equal, Therefore NO Tampering occurred!!***" message	Same as expected result

The message of "ROI Hash is Equal, Therefore NO Tampering occurred!!" indicated that the extracted ROI hash value (which stored in RONI 3) was equivalent to the generated ROI hash value, it means ROI has not been not tampered.

Table 4.5

Test Cases for Test No.2: Test on Where ROI of US Images Are Tampered in Different Ways

ROI of US images are tampered in different ways as indicated below	Elapsed time (seconds)	Expected Result	Actual Result	ROI recovery measured in PSNR
Cloning a portion of image into ROI (as shown in Figure 4.2)	0.4623	Display ROI tampered message and produced a recovered DICOM file	Same as expected result	infinity
Adding salt and pepper noise into ROI (as shown in Figure 4.3)	0.4713	Display ROI tampered message and produced a recovered DICOM file	Same as expected result	infinity
Flipping a portion of image in ROI vertically (as shown in Figure 4.4)	0.3915	Display ROI tampered message and produced a recovered DICOM file	Same as expected result	infinity
Smoothening some areas in ROI (as shown in Figure 4.5)	0.3343	Display ROI tampered message and produced a recovered DICOM file	Same as expected result	Infinity

The robustness of ROI-DR to tampering has been proven and the speed was satisfactory by having watermarking authentication processing time less than 0.5 seconds. Two identical images will yield an infinity result in PSNR. Therefore, an infinity result obtained in PSNR indicated 100 percent ROI recovery, where the recovered medical image was identical with the original medical image.

Figure 4.2 - 4.5 were the output of ultrasound medical images after being tampered in different ways.

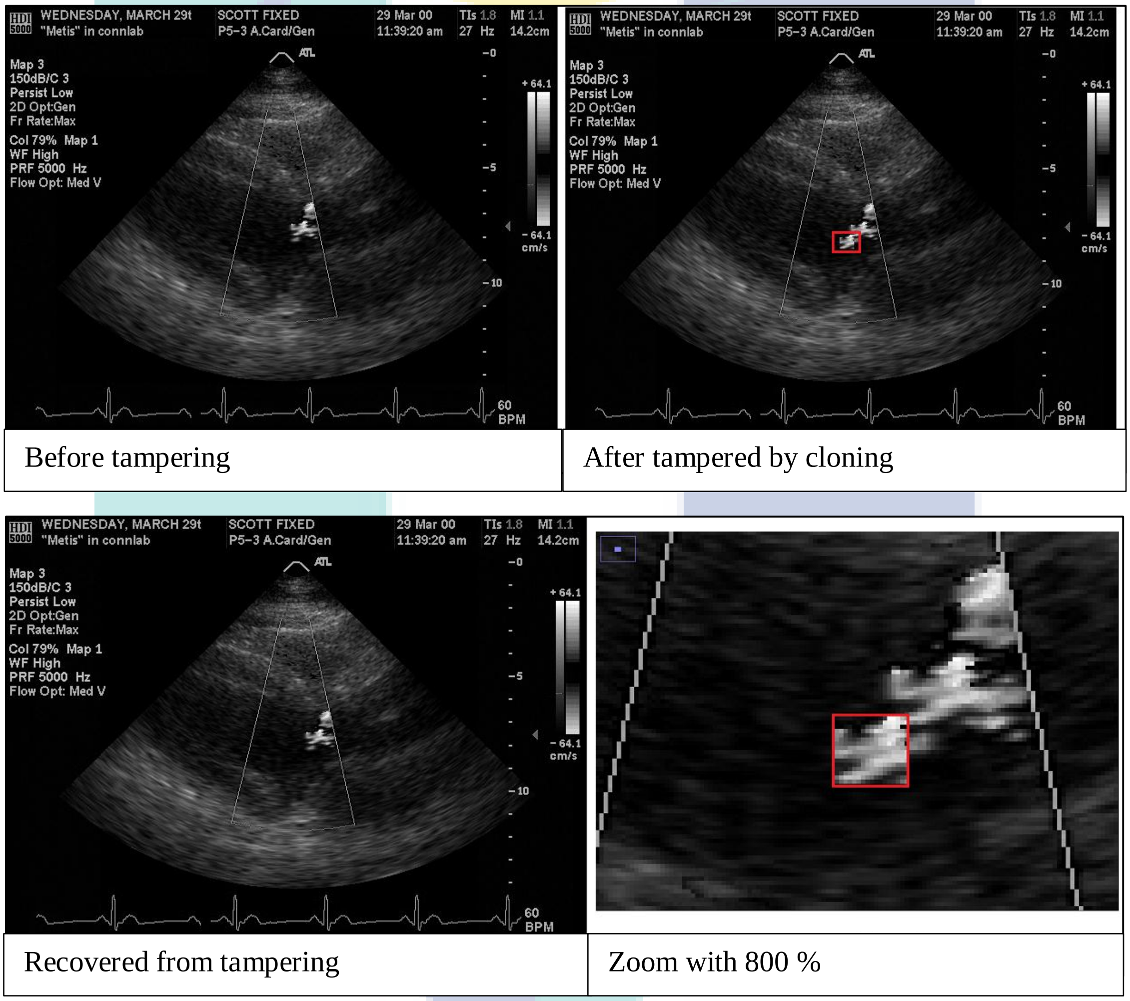


Figure 4.2 Ultrasound medical images before and after tampered by cloning a portion of image into ROI and the recovered ultrasound medical image

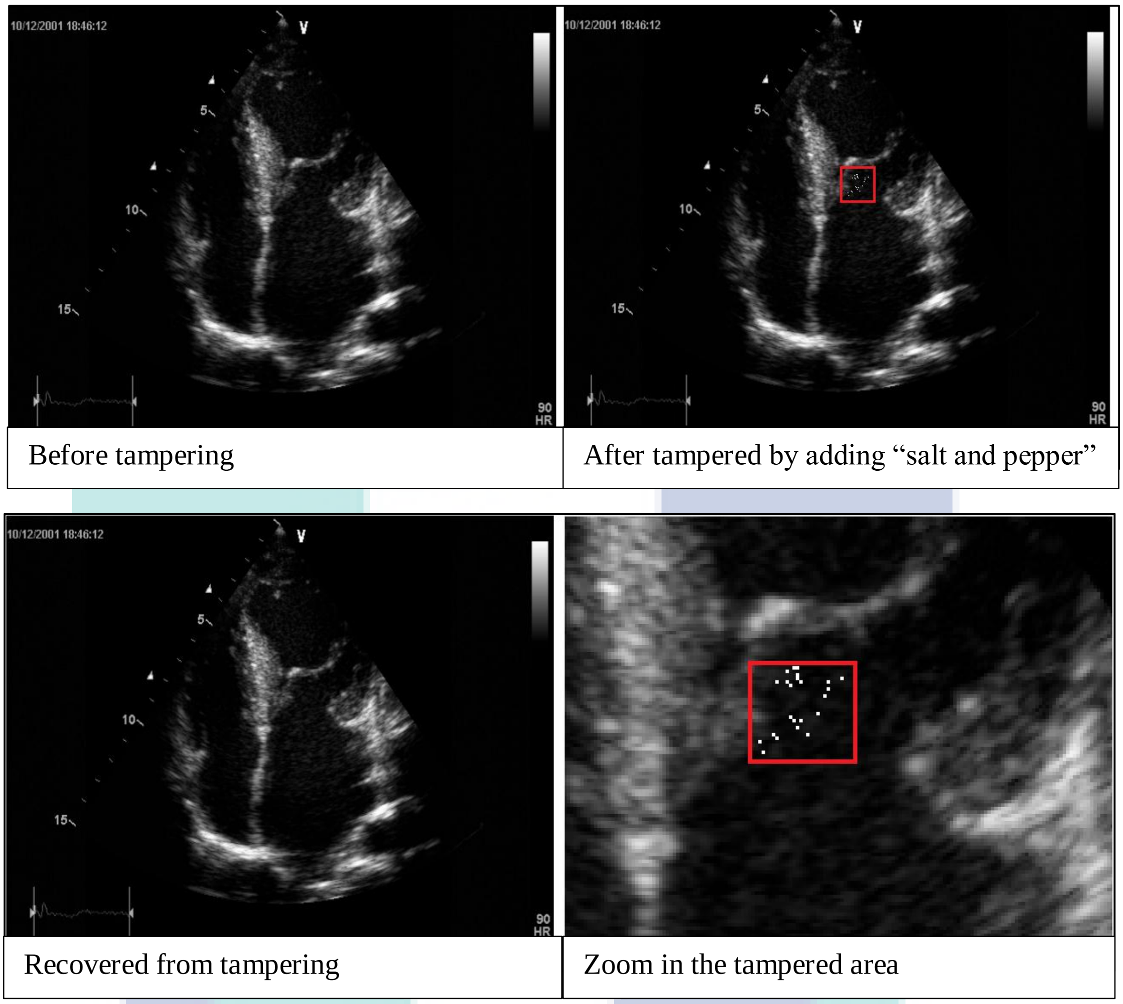


Figure 4.3 Ultrasound medical images before and after tampered by adding salt and pepper noise into ROI and the recovered ultrasound medical image

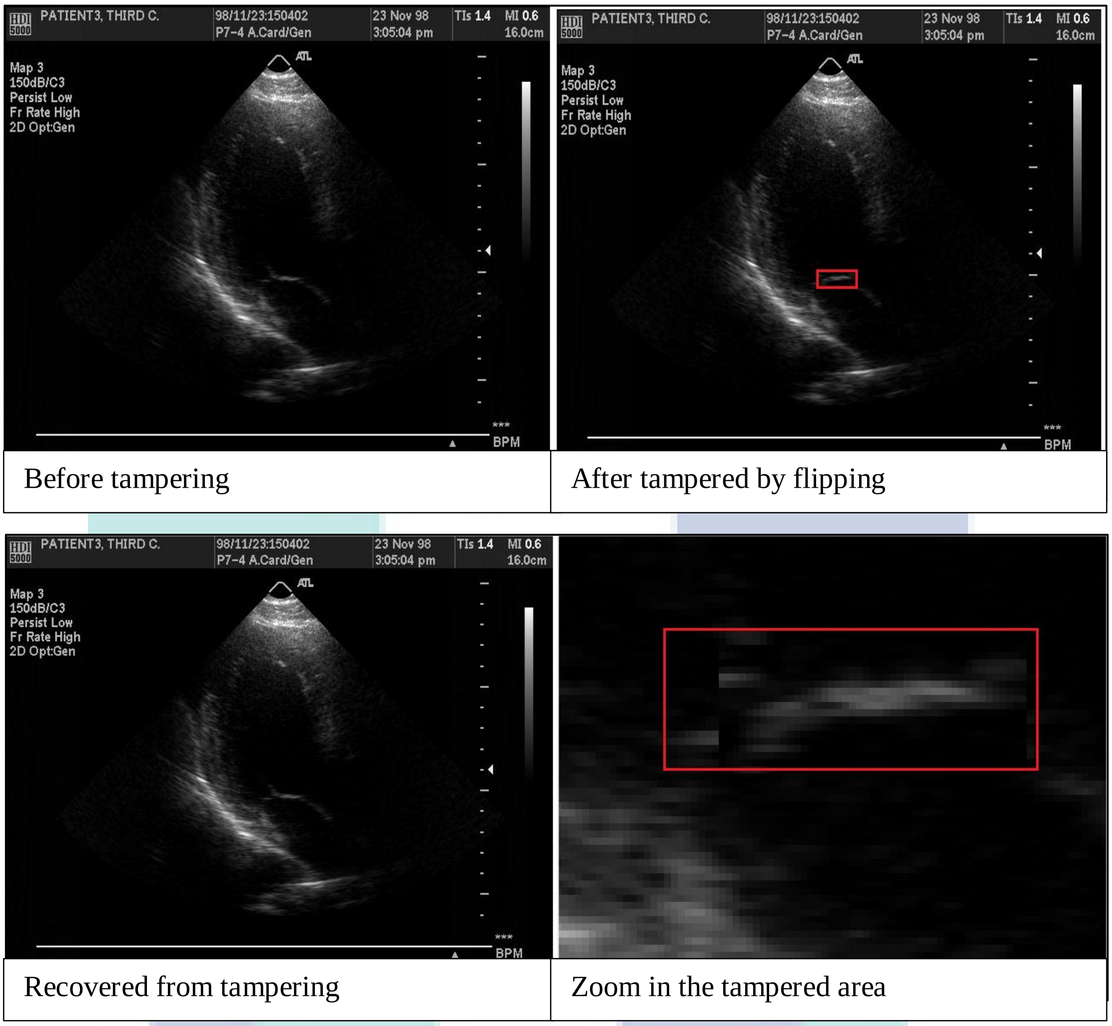
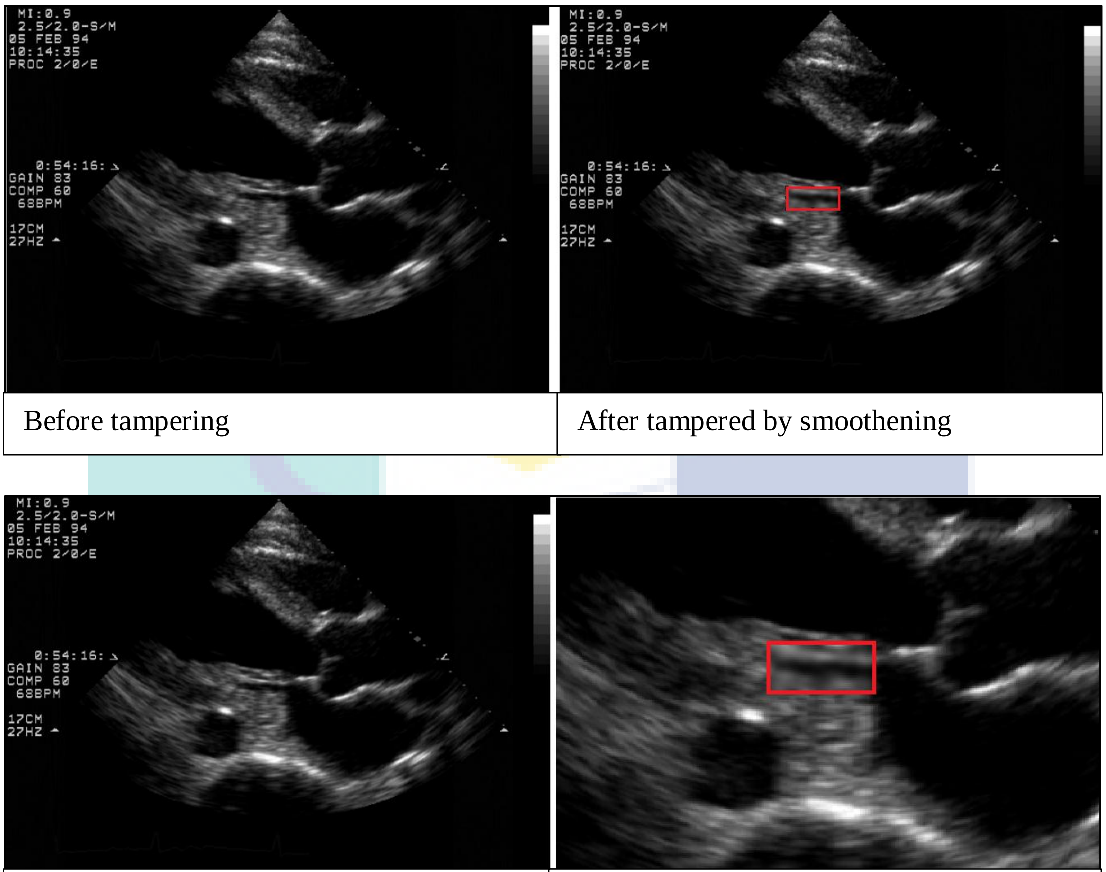


Figure 4.4. Ultrasound medical images before and after tampered by flipping a portion of image in ROI vertically and the recovered ultrasound medical image



Before tampering	After tampered by smoothening
Recovered from tampering	Zoom in the tampered area

Figure 4.5. Ultrasound medical images before and after tampered by smoothening some areas in ROI and the recovered ultrasound medical image

In Table 4.6, the elapsed time of TALLOR, TALLOR-RS and ROI-DR watermarking authentication process on different tampered medical images was recorded. It could be observed that ROI-DR was faster than TALLOR and TALLOR-RS. Three of them were perform faster in non-tampering ROI medical image than tampered ROI medical image. This is because when ROI is not tampered, the computed ROI hash value is equivalent with extracted ROI hash value, thus skip the whole authentication process and saved the processing time. Whereas, tampered ROI medical image will lead to inequality in ROI hash value, which will started the authentication process, therefore tampered ROI medical image was time consuming than non-tampered ROI medical image.

Table 4.6

Test Cases for Test No.3: Comparison of Elapsed Time between ROI-DR, TALLOR and TALLOR-RS of Watermarking Authentication Process

Test Data	**Elapsed time (seconds)**		
	ROI-DR	**TALLOR**	**TALLOR-RS**
Non-Tampering ROI	0.049	1.342	7.275
Cloning a portion of image into ROI	0.462	10.421	7.232
Adding salt and pepper noise into ROI	0.471	9.033	6.754
Flipping a portion of image in ROI vertically	0.392	8.268	6.597
Smoothening some areas in ROI	0.334	6.427	6.222
Average	**0.342**	**7.098**	**6.816**

Speedup factor in Table 4.7 was generated by dividing the elapsed time of TALLOR and TALLOR-RS with ROI-DR (refer Table 4.6) respectively.

Table 4.7

Speedup Factor of TALLOR and TALLOR-RS Relative to ROI-DR

Test Data	**Speedup factor in relative to ROI-DR**	
	TALLOR	**TALLOR-RS**
Non-Tampering ROI	27.390	148.527
Cloning a portion of image into ROI	22.543	15.645
Adding salt and pepper noise into ROI	19.166	14.331
Flipping a portion of image in ROI vertically	21.118	16.849
Smoothening some areas in ROI	19.227	18.614
Average	**21.889**	**42.793**

In Table 4.7, it could be observed that ROI-DR has a higher speedup factor in non-tampering ROI in TALLOR-RS than TALLOR. In other words, TALLOR-RS is slower than TALLOR. This is because TALLOR-RS will perform CRC checking until the tampered pixel was found, and it will proceed until the end of the ROI pixel in the case of non-tampered medical image, which is time consuming. The strength of CRC will only show in tampered medical images, where it will stop the checking process when tampered pixel was found. This results a higher speedup factor in TALLOR than TALLOR-RS relative to ROI-DR, where TALLOR-RS was faster than TALLOR. In authentication watermarking process on tampered image, ROI-DR has speedup factor of 19 to 22 and 14 to 18 relative to TALLOR and TALLOR-RS respectively. These speedup factor result indicated that ROI-DR has achieved its objective of reducing the watermarking processing time.

Table 4.8

Test Cases for Test No.4: Test on Where RONIs of US Image was Tampered

Test Data	Expected Result		Actual Result
	Message display	**Produced recovered file?**	
Tampered on RONI 1 (as shown in Figure 4.6)	ROI Hash is Equal, Therefore NO Tampering occurred!!	No	Same as expected result
Tampered on RONI 2 (as shown in Figure 4.6)	ROI Hash is Equal, Therefore NO Tampering occurred!!	No	Same as expected result
Tampered on right side of RONI 3(as shown in Figure 4.7)	ROI Hash is Equal, Therefore NO Tampering occurred!!	No	Same as expected result
Tampered on left side of RONI 3 (as shown in Figure 4.7)	No Tampering occurred on ROI BUT it occurred on RONI_3 instead, in which stored the ROI hash key!!!	No	Same as expected result
Tampered on RONI 4 (as shown in Figure 4.8)	ROI Hash is Equal, Therefore NO Tampering occurred!!	No	Same as expected result
Tampered on RONI 5 (as shown in Figure 4.8)	ROI Hash is Equal, Therefore NO Tampering occurred!!	No	Same as expected result

In Table 4.8, tampering has been applied solely on RONIs and ROI is left untouched. If ROI has not been tampered with, then no recovery is necessary, therefore no retrieval of stored ROI bits from RONI is required, thus tampering on RONIs would be ignored. When the ROI hash value that extracted from RONI 3 is equal to computed ROI hash value, it means ROI has not been tampered, and a message of "ROI hash is equal, therefore no tampering occurred!" will be displayed. A different message will be displayed if tampering is occurred in RONI 3 that stored ROI hash value. Tampered at left and right side of RONI 3 gave a different result because ROI Hash information is stored at the left side of RONI 3, therefore only the tampering on the left side will affect the stored ROI hash values.

The test data described in Table 4.8 are illustrated in Figure 4.6 – 4.8.

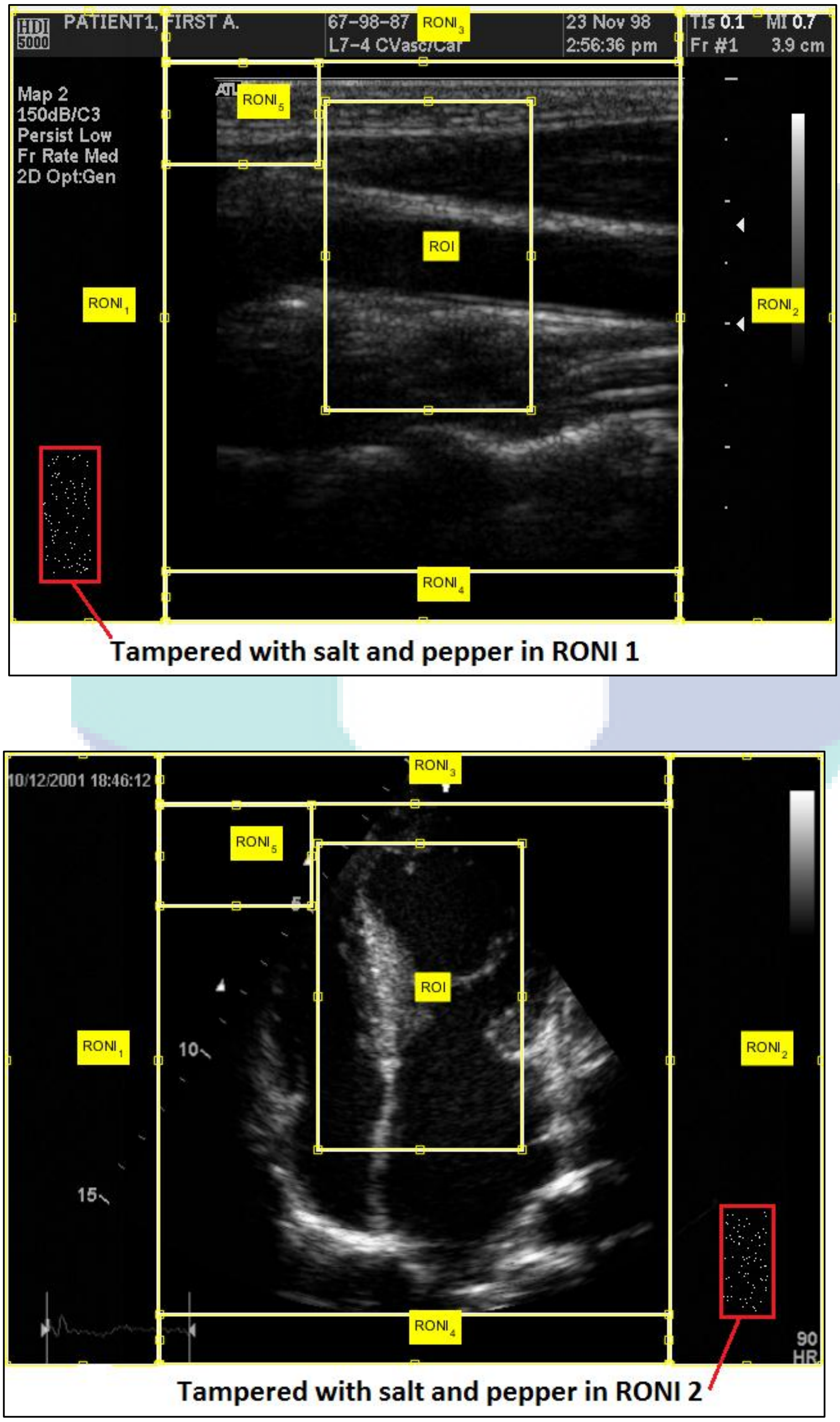
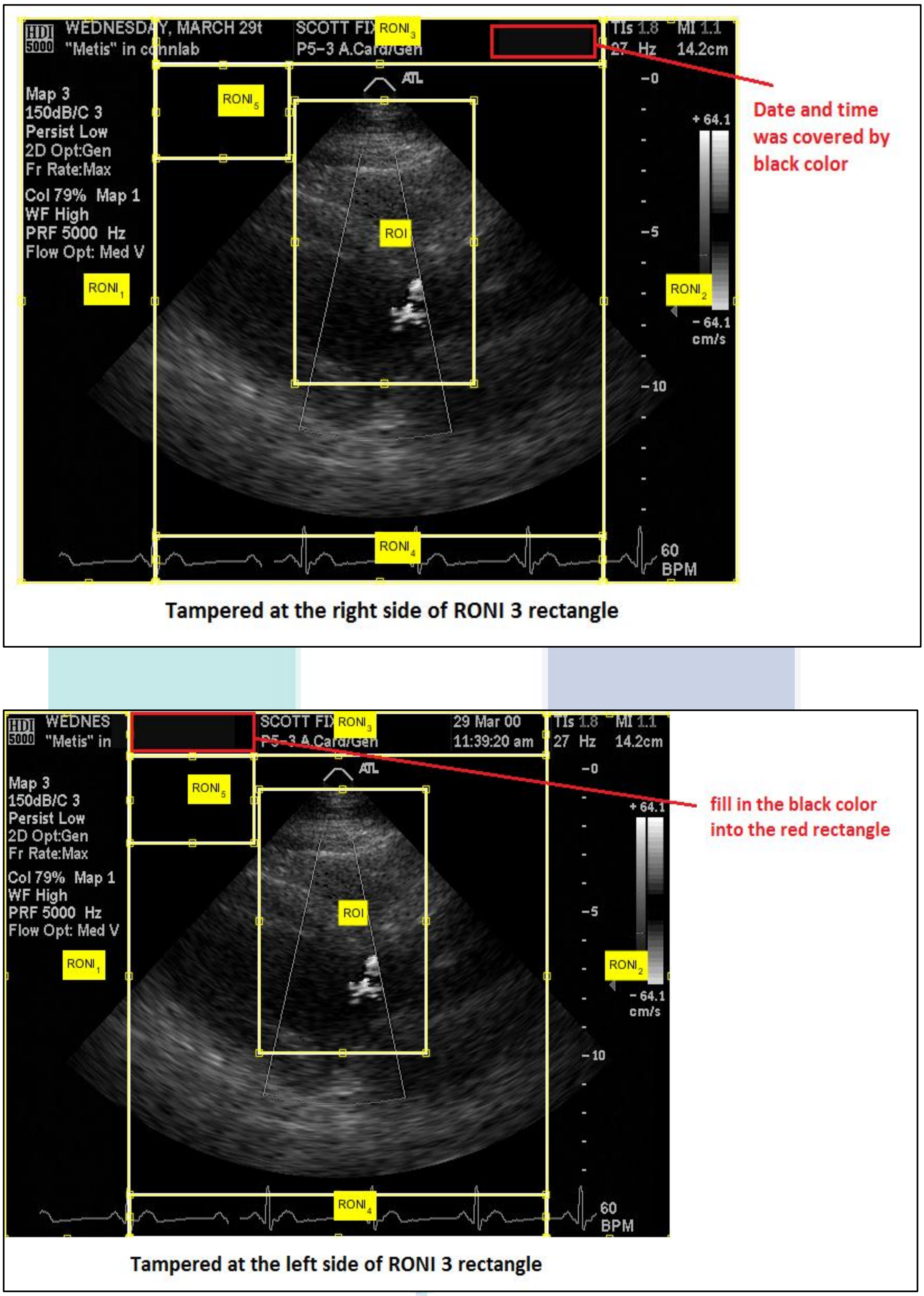


*Figure 4.6.*Ultrasound medical images tampered on RONI 1 and RONI 2



Tampered at the right side of RONI 3 rectangle

Tampered at the left side of RONI 3 rectangle

Figure 4.7. Ultrasound medical images tampered on RONI 3

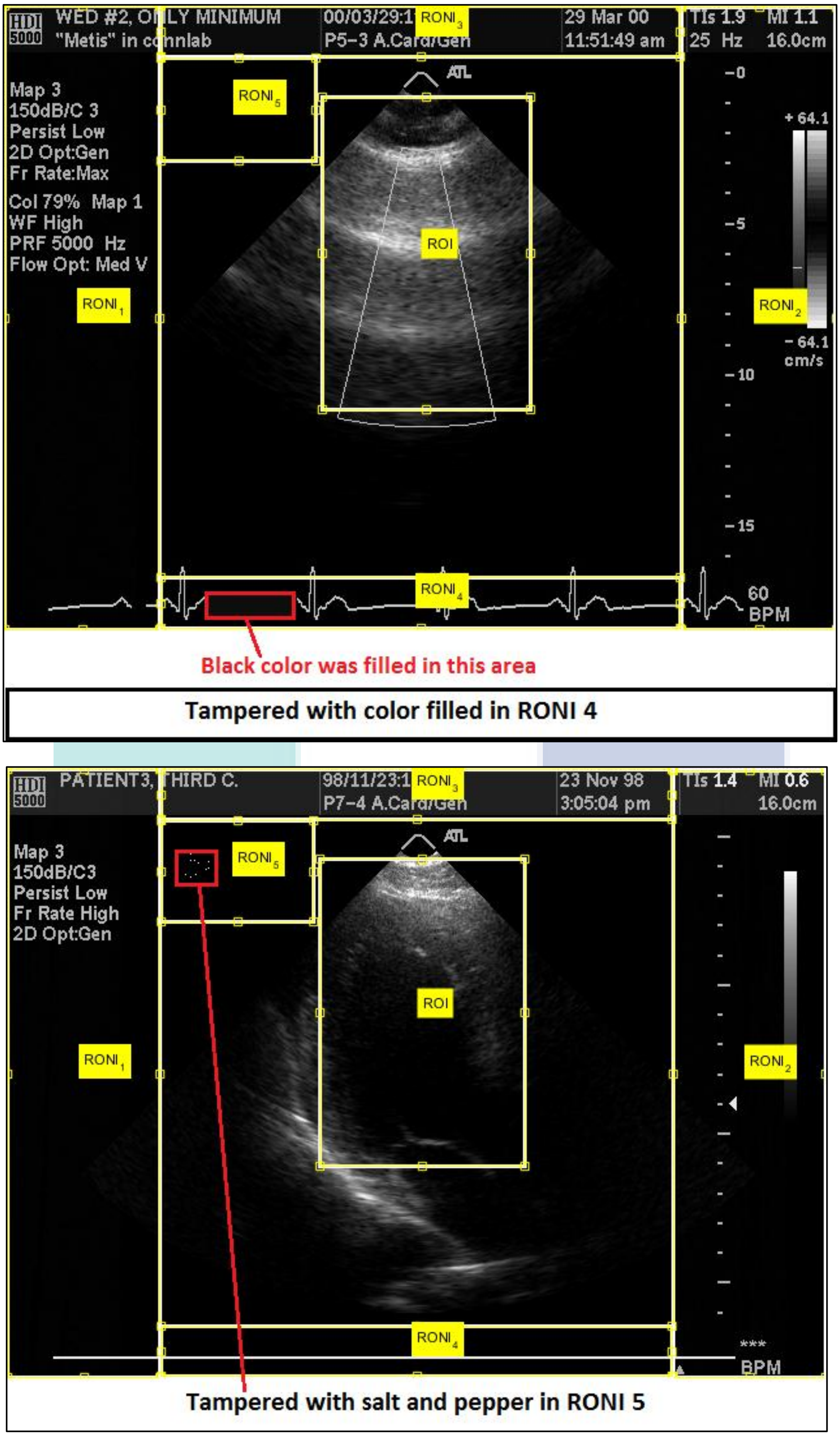


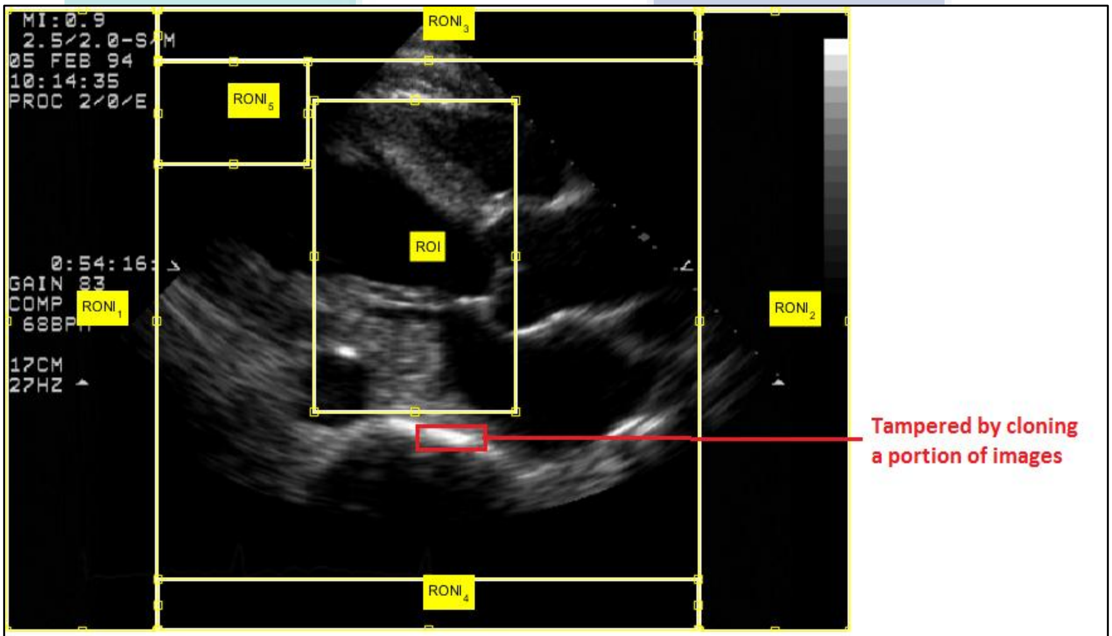
Figure 4.8. Ultrasound medical images tampered on RONI 4 and RONI 5

Table 4.9

Test Cases for Test No.5: Test on Where Tampering is Occurred Outside of ROI and RONI Rectangle Area

Test Data	Expected Result		Actual Result
	Message display	Produced recovered file?	
Tampered at outside the area of ROI and RONI	ROI Hash is Equal, Therefore NO Tampering occurred!!	No	Same as expected result

In Table 4.9, tampering was applied outside ROI and RONI area, which is an area that has not been covered by ROI-DR watermarking scheme algorithm. In other words, ROI-DR watermarking scheme algorithm is only able to detect the tampering occur within ROI area, thus any tampering occur beyond ROI area would be ignored. Since ROI has not been tampered, therefore no recovery file was required. The test data described in Table 4.9 is demonstrated in Figure 4.9.



Tampered at outside of ROI and RONI rectangle

Figure 4.9. Ultrasound medical images tampered with outside of ROI and RONI rectangles

Table 4.10

Test Cases for Test No.6: Test on Where Both ROI and RONI were Tampered

Test Data	**Expected Result**		**Actual Result**
	Message display	**Produced recovered file?**	
Tampered at ROI and RONI 1 (as shown in Figure 4.10)	ROI recovery failed because tampering occurred on RONI regions that contain ROI information!!	No	Same as expected result
Tampered at ROI and RONI 2 (as shown in Figure 4.11)	ROI recovery failed because tampering occurred on RONI regions that contain ROI information!!	No	Same as expected result
Tampered at ROI and the right side of RONI 3 (as shown in Figure 4.12)	Tampering occurred on ROI!!! Recovery Process starts... File Recovered Successfully!!	Yes	Same as expected result
Tampered at ROI and the left side of RONI 3 (as shown in Figure 4.12)	Tampering occurred on ROI!!! Recovery Process starts... File Recovered Successfully!!	Yes	Same as expected result
Tampered at ROI and RONI 4 (A) (as shown in Figure 4.13)	Tampering occurred on ROI!!! Recovery Process starts... File Recovered Successfully!!	Yes	Same as expected result
Tampered at ROI and RONI 4 (B) (as shown in Figure 4.13)	ROI recovery failed because tampering occurred on RONI regions that contain ROI information!!	No	Same as expected result
Tampered at ROI and RONI 5 (as shown in Figure 4.14)	Tampering occurred on ROI!!! Recovery Process starts... File Recovered Successfully!!	Yes	Same as expected result

The result in Table 4.10 indicated that RONIs will be a concern if ROI has been tampered with. RONI 1 and RONI 2 are storing recovery bits, if either one of them is tampered with, then no recovery could be done. In the test case of tampered at ROI and RONI 4(B), a tampering on RONI 4(B) has distorted the embedded bits value that required for ROI recovery process, and caused a failure in ROI recovery. But it has not happen in test cases of RONI 4 (A), this is because the embedded bits is not within the tampered area. The example of tampering on both ROI and RONI has been demonstrated in Figure 4.10 - 4.14.

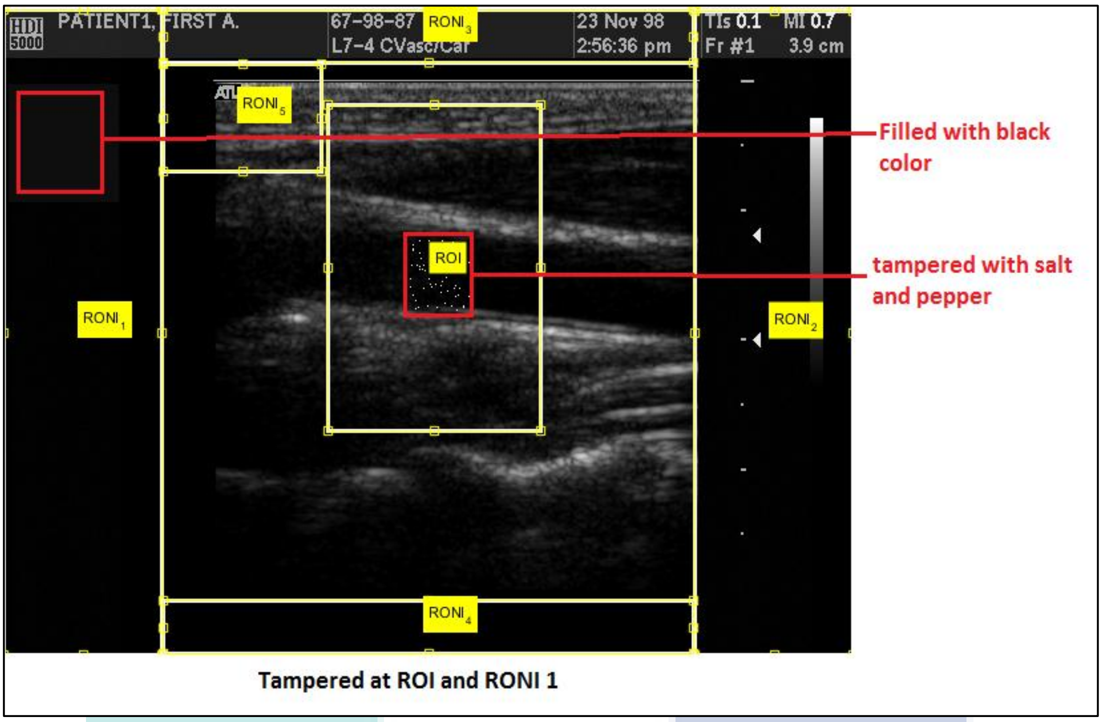


Figure 4.10. Ultrasound medical images tampered at ROI and RONI 1

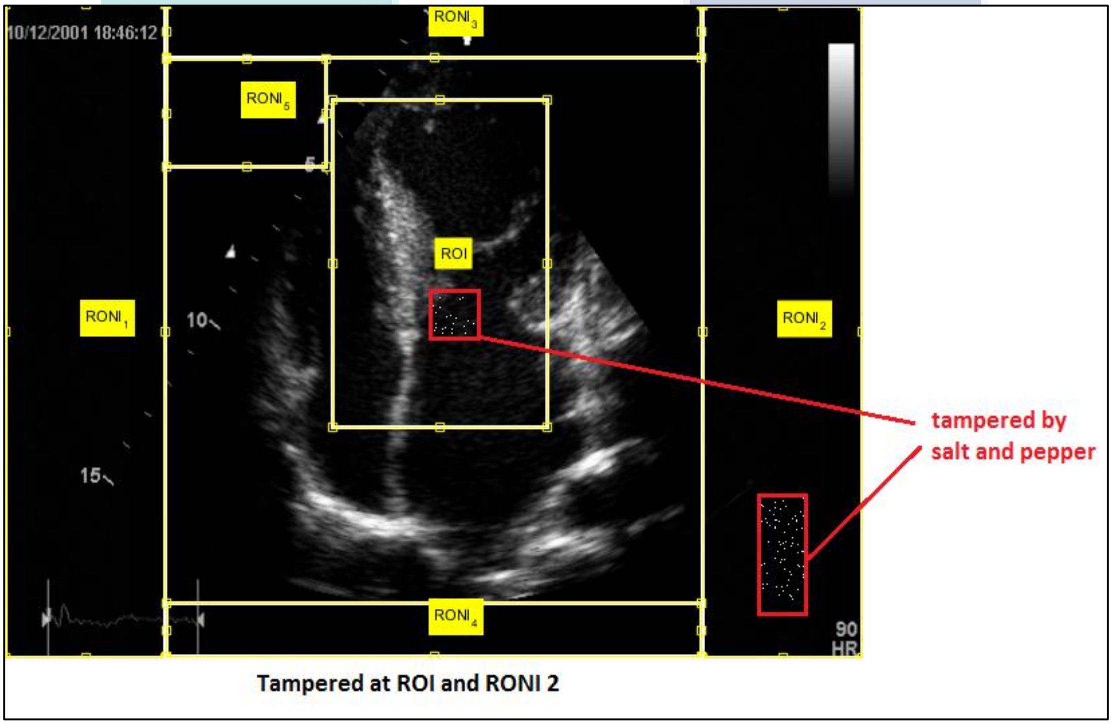
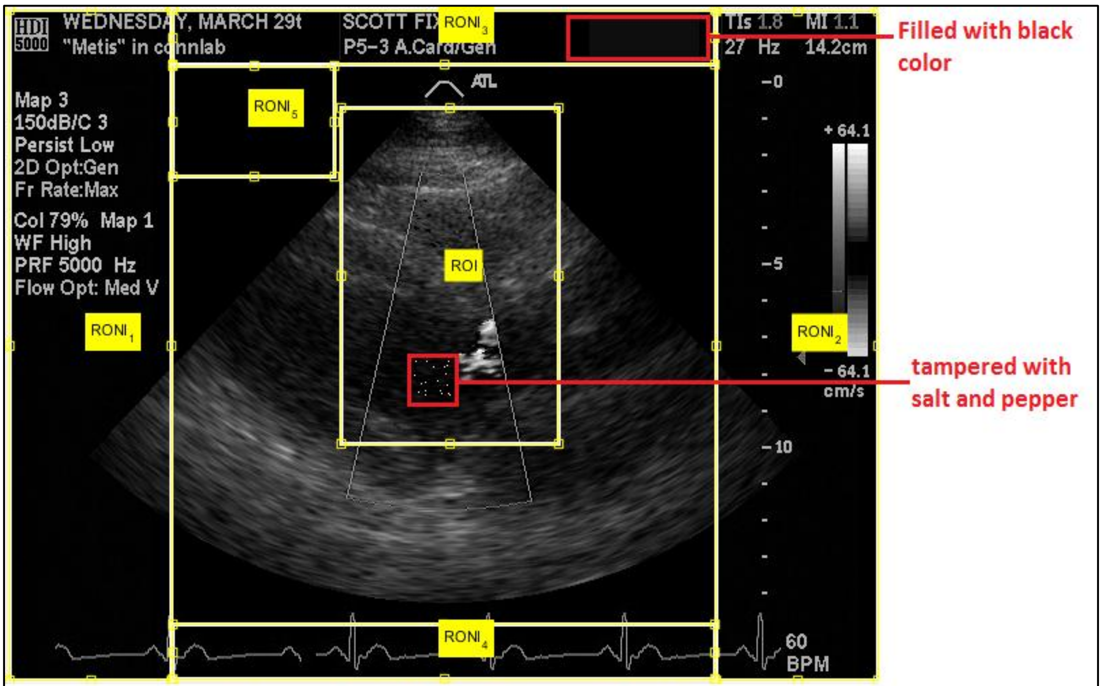
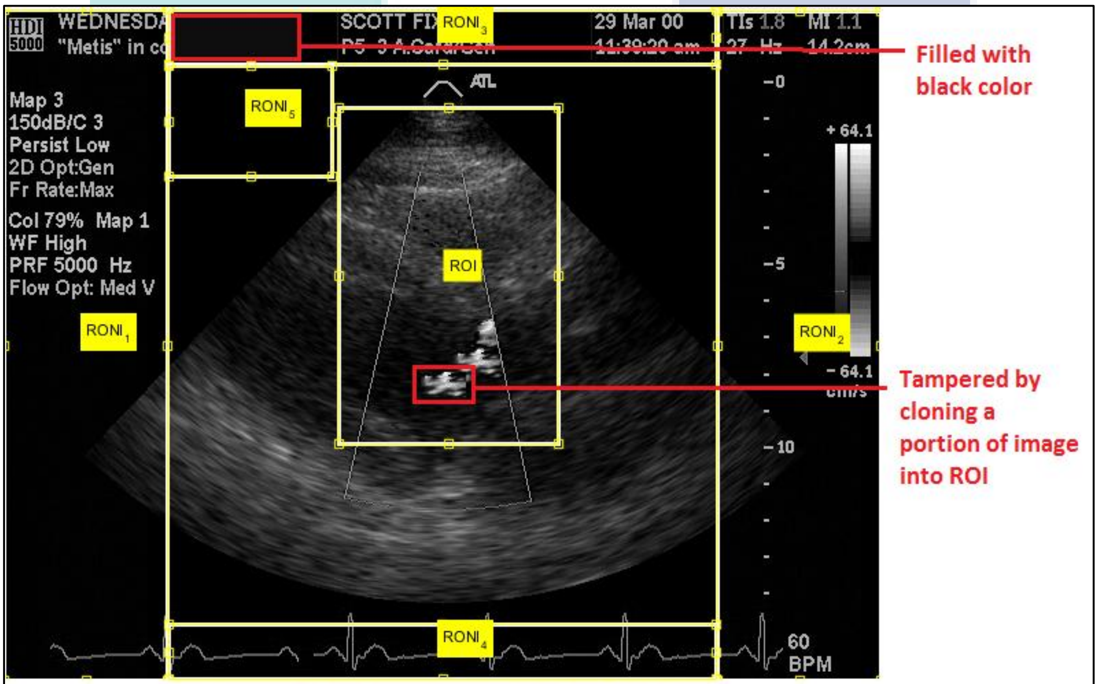


Figure 4.11. Ultrasound medical images tampered at ROI and RONI 2



Tampered at ROI and right side of RONI 3



Tampered at ROI and the left side of RONI 3

Figure 4.12. Ultrasound medical images tampered at ROI and RONI 3

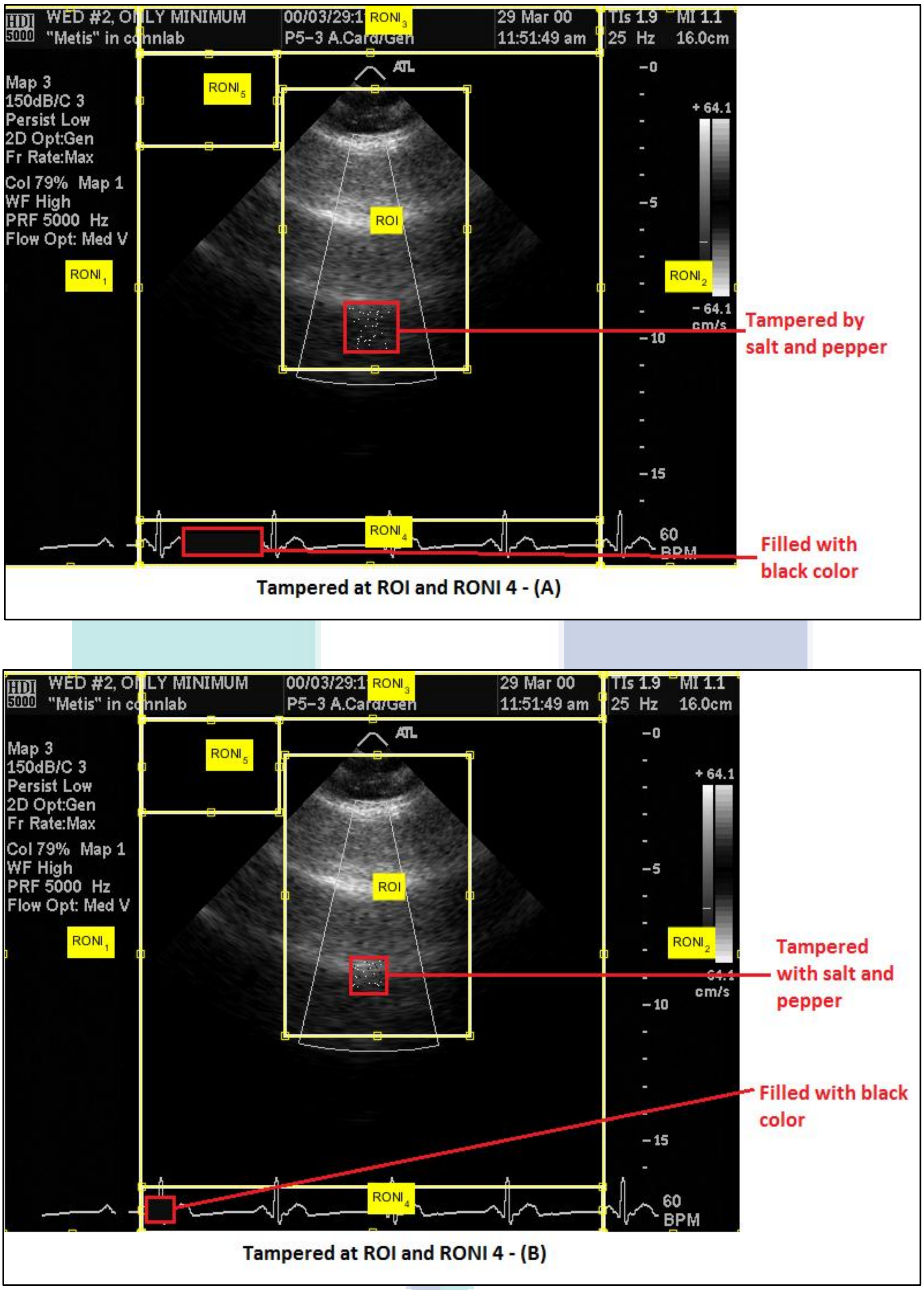


Figure 4.13. Ultrasound medical images tampered at ROI and RONI 4

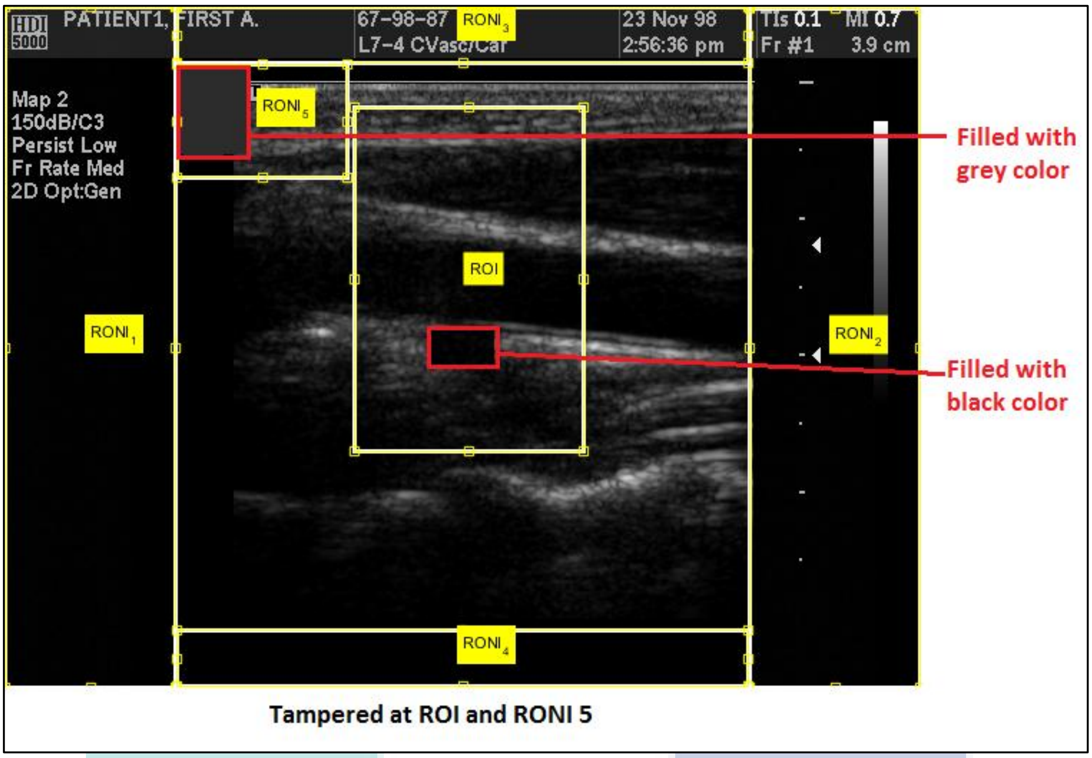


Figure 4.14. Ultrasound medical images tampered at ROI and RONI 5

4.2.3 Performance Comparison of ROI-DR with TALLOR and TALLOR-RS

In Table 4.11, a table of comparison in algorithms, experiment and result has been made among ROI-DR, TALLOR and TALLOR-RS. It has been shown that three of them have some similarity in algorithm, such as use of jpeg compression and hash-256 and embedding two bits of each ROI pixel into two LSB of each RONI pixel. They also have a same ROI size but with the different RONI layout and organization. Both ROI-DR and TALLOR-RS are using segmentation technique, where ROI-DR watermarking scheme split ROI into two sections and stored them into RONI 1 and RONI 2 respectively, whereas TALLOR-RS divided ROI into 40 x 40 blocks and implement each with CRC. Both TALLOR and TALLOR-RS are using block location and matrix concatenation method to store ROI bits before transfer to RONI LSB, which required high memory consumption, thus slow down the process. Whereas ROI-DR watermarking scheme crop out the ROI and RONI image and pairing up 1 pixel (8 bits) of ROI to 4 pixels (4 x 2 bits) of RONI, and then overwrite 2 bits of each ROI pixel into 2 LSB of each RONI pixel iteratively. This process is more direct and less memory consumption. Fairness of comparison of these three watermarking schemes was ensured by having them tested under the same environment and using the same set of ultrasound medical images as an input file for watermarking embedding process, and the watermarked ultrasound medical image are generated as a DICOM file. Same tampering technique was applied on watermarked ultrasound medical image for testing these three scheme's authentication process. The result indicated that ROI-DR Watermarking Scheme has similar PSNR value (~ 48 dB) and shorter elapsed time as compared to TALLOR and TALLOR-RS watermarking scheme. ROI-DR has an average speedup factor of 22.55 and 26.65 in relative to TALLOR and TALLOR-RS respectively in watermarking embedding process. Whereas, in watermarking authentication process, ROI-DR has an average speedup factor of 21.89 and 42.79 relative to TALLOR and TALLOR-RS respectively. The high speedup factor in relative to TALLOR-RS is due to the time taken in authenticating non-tampered ultrasound medical images was too high, which compromised the average of speedup factor in TALLOR-RS watermarking authentication process. This is because TALLOR-RS needs to go through the whole process of CRC checking even though there is no tampering has occurred. The actual average speedup factor of ROI-DR in relative to TALLOR-RS could be counted as 16.36 if excluding the non-tampered ultrasound medical image in calculating the average of speedup factor for watermarking authentication process. CRC checking in TALLOR-RS has sped up the authentication process in relative to TALLOR, but more time is required in implementing CRC into watermarking embedding process as compared to TALLOR. In conclusion, the current research, ROI-DR watermarking scheme has maintained integrity of ultrasound medical images and improved the time performance significantly as compared to the previous research, TALLOR and TALLOR-RS watermarking scheme.

The three important performance metrics as described above have been successfully tested in ROI-DR watermarking scheme. Firstly, the generated watermarking is invisible in watermarked ultrasound medical image, and has produced a satisfactory PSNR result. Secondly, the elapse time taken to perform watermarking embedding and authentication process has shown a high speedup factor in relative to TALLOR and TALLOR-RS watermarking scheme. Thirdly, the robustness to tampering has been thoroughly tested by applying a various kind of tampering on ROI and RONI areas, and the result shown that the detection and recovery function work as expected.

Table 4.11

A Table of Comparison between ROI-DR, TALLOR and TALLOR-RS Watermarking Scheme

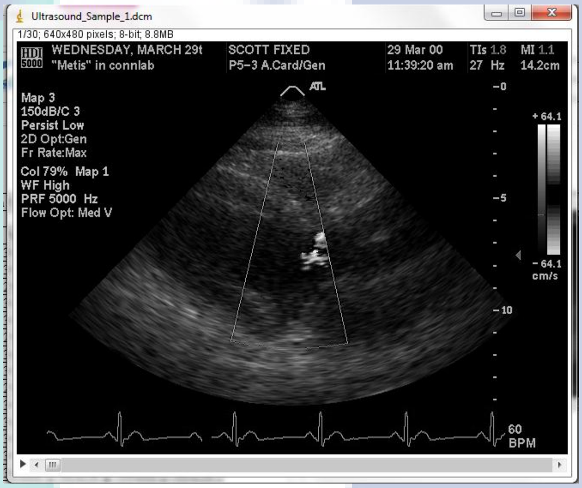
	Proposed Method	**Existing Works (Liew S.C., 2011)**	
Watermarking Scheme	**ROI-DR**	**TALLOR**	**TALLOR-RS**
Algorithm:			
ROI Size	160 x 240 pixels	160 x 240 pixels	160 x 240 pixels
Using Hash-256	Yes	Yes	Yes
Using jpeg compression	Yes	Yes	Yes
Store ROI bits into 2 LSB of RONI	Yes	Yes	Yes
Use segmentation	Split into 2 sections	No	40 x 40 pixels
Use CRC	No	No	Yes
Use matrix concatenation method in storing ROI bits	No	Yes	Yes
Way of storing ROI bits into RONIs	Split compressed ROI bits into two and stored into RONI1 and 2	Store ROI bits into RONI circularly by block location.	Same as TALLOR plus CRC
Way of image recovery	Recovered by replacing the whole tampered ROI with stored ROI image.	Recovered by replacing the tampered pixels that has been localized.	Recovered by replacing the tampered block.
Experiment:			
Use same set of ultrasound medical images samples as input file for watermarking embedding process	Yes	Yes	Yes
Use same set of tampered images for authentication process	Yes	Yes	Yes
Conduct under the same hardware and software environment.	Yes	Yes	Yes
Result:			
Average of PSNR (dB)	48.4141	48.413	48.943
Elapsed time in watermarking embedding process	0.98 seconds	22.14 seconds	26.09 seconds
Elapsed time in watermarking authentication process	0.34 seconds	7.10 seconds	6.82 seconds
Able to detect tampering	Yes	Yes	Yes
Able to produce exact ROI recovery	Yes	Yes	Yes
Able to generate a recovered DICOM file	Yes	Yes	Yes

4.3 RESULT ANALYSIS OF ROI-DR IN MULTI-FRAMES ULTRASOUND MEDICAL IMAGES

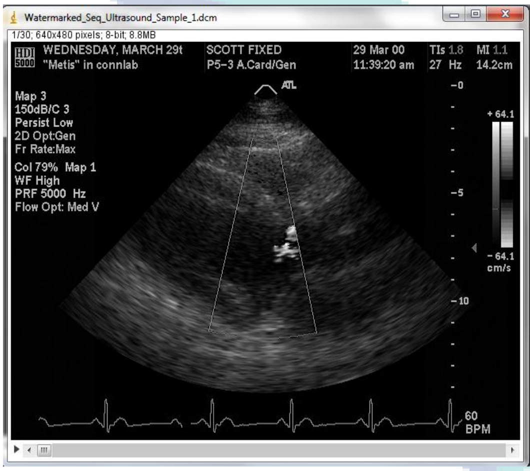
Parallel watermarking process is a main attribute of the proposed multi-frames watermarking scheme. Sequential watermarking process was developed and used as a benchmark against parallel watermarking process. The experiment result in Table 4.2 and 4.7 shown that ROI-DR has achieved a high speedup relative to TALLOR, which means ROI-DR will also achieve a high speedup relative to TALLOR in sequential multi-frames watermarking process. The following experiment is to investigate whether the parallel ROI-DR watermarking process will further improve the processing time relative to sequential ROI-DR watermarking process. The proposed multi-frames watermarking scheme also tested on TALLOR to check whether it could solve the time constraint of sequential problem which has been stated in the problem statement. In summary, experiment objective of the proposed multi-frames watermarking scheme is to investigate whether it could improve the multi-frames watermarking processing time without compromising its functionality (its robustness to tampering) and integrity of medical images (the degree of imperceptibility). Lastly, a performance comparison will be conducted between ROI-DR and TALLOR multi-frames watermarking schemes.

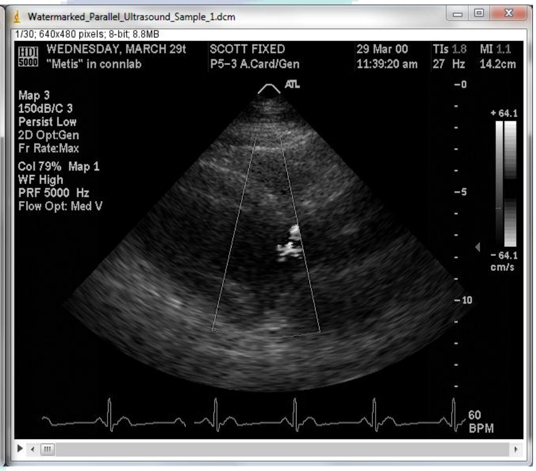
4.3.1 Imperceptibility

Eight different set of ultrasound medical images samples (as indicated in Table 3.1) has been watermarked in two different ways: (1) sequentially and (2) parallel. It is important to ensure that the quality and fidelity of images were not affected by the way watermarking embedding process performed. Both sequential and parallel watermarking embedding processes have produced same MSE and PSNR result as indicated in Table 4.12 (a) to (h), except some negligible difference in the highlight area, which may due to the instability of software operating environment. The same result means that the sequential and parallel watermarking embedding process have produced the same effect on medical images. It is because both processes were originated from the same watermarking scheme, they will embedded bits at the same pixel location in medical image, therefore produced the same result. The MSE and PSNR formula has been defined in chapter 3, where MSE is a part of PSNR formula. There are only a minor variance (~ 1 dB) across the frames in the same set of ultrasound sample, therefore an average was derived for result analysis purposes. The average PSNR values calculated for all images ranging between 47.966~49.316 dB. Hamzaoui, R. and Saupe, D. (2006) stated that image in 8 bits depth should have PSNR value between 30 to 50 dB in order to maintain the image quality. This means that the operation either in sequential or parallel mode does not affect the image quality and its fidelity, irrespective on the number of frames processed. As a conclusion, multi-frames watermarking scheme has met the imperceptibility requirement, where the medical images after watermarking embedding process are visually indistinguishable as the original images (as shown in Figure 4.15).



(a) Ultrasound Sample 1 Raw Image – Before Watermarking Process





(b) Watermarked ultrasound medical images generated by Sequential ROI-DR Watermarking Process

(c) Watermarked ultrasound medical images generated by Parallel ROI-DR Watermarking Process

Figure 4.15. Ultrasound medical images before and after watermarking process

Table 4.12 (a)

PSNR and MSE Values for Watermarked US_Sample_1 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_1				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	32.657	47.953	32.657	47.953
2	33.360	48.048	33.360	48.048
3	33.045	48.072	33.045	48.072
4	32.690	48.052	32.690	48.052
5	33.432	48.160	33.432	48.160
6	33.046	48.120	33.046	48.120
7	32.751	48.034	32.751	48.034
8	33.437	48.110	33.437	48.110
9	33.137	48.168	33.171	48.202
10	32.695	47.847	32.695	47.847
11	32.294	47.999	32.294	47.999
12	33.023	48.101	33.023	48.101
13	32.695	47.847	32.695	47.847
14	33.294	47.999	33.294	47.999
15	33.023	48.101	33.023	48.101
16	32.829	47.797	32.829	47.797
17	33.388	47.984	33.388	47.984
18	33.150	48.153	33.150	48.153
19	32.857	48.073	32.857	48.073
20	33.575	48.192	33.575	48.192
21	33.248	48.215	33.248	48.215
22	32.887	48.104	32.887	48.104
23	33.596	48.197	33.596	48.197
24	33.247	48.197	33.247	48.197
25	32.912	48.142	32.912	48.142
26	33.621	48.229	33.621	48.229
27	33.286	48.252	33.286	48.252
28	32.878	48.140	32.878	48.140
29	33.562	48.210	33.562	48.210
30	33.225	48.218	33.225	48.218
Average	**33.095**	**48.090**	**33.096**	**48.092**

Table 4.12 (b)

PSNR and MSE Values for Watermarked US_Sample_2 and Watermarked US_Sample_3 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_2				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	30.678	47.982	30.678	47.982
2	30.622	47.990	30.622	47.990
3	30.754	47.910	30.754	47.910
4	30.498	47.914	30.498	47.914
5	30.429	47.911	30.429	47.911
6	30.597	47.858	30.597	47.858
7	30.387	47.902	30.387	47.902
8	30.324	47.905	30.324	47.905
9	30.490	47.851	30.490	47.851
10	30.243	47.849	30.243	47.849
11	30.183	47.853	30.183	47.853
12	30.942	48.397	30.942	48.397
13	30.327	47.866	30.308	47.848
14	30.274	47.878	30.274	47.878
15	31.040	48.423	31.040	48.423
Average	**30.519**	**47.966**	**30.518**	**47.965**
Watermarking Embedding Process on US_Sample_3				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	32.096	48.782	32.096	48.782
2	32.015	48.753	32.015	48.753
3	32.015	48.717	32.015	48.717
4	32.929	48.652	32.929	48.652
5	31.857	48.619	32.817	49.579
6	31.717	48.559	31.717	48.559
7	31.672	48.580	31.672	48.580
8	31.607	48.571	31.607	48.571
9	31.590	48.576	31.384	48.37
10	31.553	48.495	31.553	48.495
11	31.505	48.387	31.505	48.387
12	31.580	48.371	31.580	48.371
13	31.658	38.377	31.658	38.377
14	31.794	48.466	31.794	48.466
15	31.867	48.503	31.867	48.503
Average	**31.830**	**47.894**	**31.881**	**47.944**

Table 4.12 (c)

PSNR and MSE Values for Watermarked US_Sample_4 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_4				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	30.394	48.982	30.394	48.982
2	30.764	48.978	30.764	48.978
3	30.498	48.980	30.498	48.980
4	30.381	48.990	30.381	48.990
5	30.752	48.984	30.752	48.984
6	30.486	48.986	30.486	48.986
7	30.332	49.008	30.332	49.008
8	30.700	48.995	30.700	48.995
9	30.343	49.002	30.343	49.002
10	30.320	49.050	30.320	49.050
11	30.703	49.047	30.703	49.047
12	30.440	49.060	30.440	49.060
13	30.406	49.011	30.406	49.011
14	30.762	48.000	30.762	48.992
15	30.536	49.034	30.536	49.034
16	30.387	48.919	30.387	48.919
17	30.753	48.916	30.753	48.916
18	30.483	48.910	30.483	48.910
19	30.392	48.932	30.392	48.932
20	30.759	48.928	30.759	48.928
21	30.504	48.936	30.545	48.977
22	30.457	48.919	30.457	48.919
23	30.831	48.929	30.831	48.929
24	30.572	48.930	30.572	48.930
25	30.574	49.042	30.574	49.042
26	30.915	49.019	30.915	49.019
27	30.677	49.041	30.677	49.041
28	30.418	48.959	30.418	48.959
29	30.781	48.952	30.781	48.952
30	30.517	48.952	30.517	48.952
31	30.361	48.909	30.361	48.909
32	30.731	48.909	30.731	48.909
33	30.458	48.901	30.458	48.901
34	30.339	48.903	30.339	48.903
35	30.705	48.896	30.705	48.896
36	30.446	48.902	30.446	48.902
37	30.033	48.682	30.033	48.682
38	30.404	48.674	30.404	48.674
39	30.142	48.683	30.142	48.683
40	30.235	49.101	30.235	49.101

Table 4.12 (c) Continued

Watermarking Embedding Process on US_Sample_4				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
41	30.621	49.088	30.621	49.088
42	30.329	49.080	30.329	49.080
43	29.908	49.040	29.908	49.040
44	30.320	49.031	30.320	49.031
45	30.032	49.044	30.032	49.044
46	29.913	49.189	29.913	49.189
47	30.339	49.179	30.339	49.179
48	30.039	49.188	30.039	49.188
49	29.923	49.142	29.923	49.142
50	30.349	49.139	30.349	49.139
51	30.061	49.157	30.061	49.157
52	30.122	49.297	30.122	49.297
53	30.527	49.277	30.527	49.277
54	30.222	49.276	30.222	49.276
55	30.237	49.380	30.237	49.380
56	30.660	49.381	30.660	49.381
57	30.363	49.386	30.363	49.386
58	30.122	49.275	30.122	49.275
59	30.538	49.267	30.538	49.267
60	30.253	49.284	30.253	49.284
61	30.159	49.326	30.159	49.326
62	30.599	49.339	30.599	49.339
63	30.280	49.322	30.280	49.322
64	30.161	49.333	30.161	49.333
65	30.568	49.315	30.568	49.315
66	30.267	49.318	30.267	49.318
67	30.179	49.343	30.179	49.343
68	30.590	49.330	30.590	49.330
69	30.304	49.347	30.304	49.347
70	30.110	49.297	30.110	49.297
71	30.545	49.306	30.545	49.306
72	30.242	49.308	30.242	49.308
73	30.044	49.281	30.044	49.281
74	30.500	49.276	30.500	49.276
75	30.192	49.274	30.192	49.274
76	30.101	49.323	30.101	49.323
77	30.510	49.332	30.510	49.332
78	30.235	49.333	30.235	49.333
79	30.103	49.301	30.103	49.301
Average	**30.396**	**49.089**	**30.396**	**49.102**

Table 4.12 (d)

PSNR and MSE Values for Watermarked US_Sample_5 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_5				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	31.200	48.513	31.175	48.488
2	31.428	48.527	31.428	48.527
3	31.286	48.514	31.286	48.514
4	31.172	48.481	31.172	48.481
5	31.391	48.486	31.391	48.486
6	31.250	48.474	31.250	48.474
7	31.168	48.502	31.168	48.502
8	31.360	48.478	31.360	48.478
9	31.252	48.500	31.238	48.486
10	31.230	48.568	31.230	48.568
11	31.420	48.544	31.420	48.544
12	31.295	48.547	31.295	48.547
13	31.100	48.459	31.100	48.459
14	31.300	48.441	31.300	48.441
15	31.181	48.453	31.181	48.453
16	31.089	48.466	31.089	48.466
17	31.289	48.345	31.323	48.482
18	31.163	48.453	31.163	48.453
19	31.134	48.518	31.134	48.518
20	31.370	48.535	31.370	48.535
21	31.207	48.504	31.207	48.504
22	30.990	48.422	30.990	48.422
23	31.208	48.419	31.208	48.419
24	31.076	48.420	31.076	48.420
25	31.082	48.525	31.044	48.487
26	31.312	48.532	31.312	48.532
27	31.175	48.530	31.175	48.530
28	30.986	48.452	30.986	48.452
29	31.213	48.457	31.213	48.457
30	31.078	48.456	31.078	48.456
31	31.008	48.469	31.008	48.469
Average	**31.207**	**48.484**	**31.205**	**48.485**

Table 4.12 (e)

PSNR and MSE Values for Watermarked US_Sample_6 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_6				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	32.583	49.330	32.583	49.330
2	32.776	49.331	32.776	49.331
3	32.660	49.327	32.660	49.327
4	32.600	49.362	32.600	49.362
5	32.929	49.386	32.929	49.386
6	32.678	49.358	32.678	49.358
7	32.610	49.364	32.610	49.364
8	32.803	49.364	32.803	49.364
9	32.691	49.364	32.691	49.364
10	32.597	49.421	32.597	49.421
11	32.787	49.416	32.787	49.416
12	32.673	49.416	32.673	49.416
13	32.697	49.592	32.697	49.592
14	32.882	49.578	32.882	49.578
15	32.784	49.595	32.784	49.595
16	32.813	49.663	32.813	49.663
17	33.006	49.659	33.006	49.659
18	32.895	49.662	32.895	49.662
19	32.784	49.689	32.784	49.689
20	32.968	49.674	32.968	49.674
21	32.860	49.681	32.860	49.681
22	32.841	49.708	32.841	49.708
23	33.036	49.706	33.036	49.706
24	32.918	49.702	32.918	49.702
25	32.722	49.650	32.722	49.650
Average	**32.784**	**49.520**	**32.184**	**49.520**

Table 4.12 (f)

PSNR and MSE Values for Watermarked US_Sample_7 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_7				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	31.878	48.33	31.878	48.33
2	31.889	48.317	31.889	48.317
3	31.936	48.351	31.936	48.351
4	31.949	48.35	31.949	48.35
5	31.966	48.36	31.966	48.36
6	31.991	48.378	31.991	48.378
7	32.007	48.386	32.007	48.386
8	31.981	48.32	31.981	48.32
9	31.993	48.287	31.993	48.287
10	32.114	48.389	32.114	48.389
11	32.118	48.373	32.118	48.373
12	32.093	48.337	32.093	48.337
13	32.103	48.362	32.103	48.362
14	32.083	48.355	32.083	48.355
15	32.141	48.425	32.141	48.425
Average	**31.950**	**48.349**	**31.950**	**48.349**

Table 4.12 (g)

PSNR and MSE Values for Watermarked US_Sample_8 after Sequential and Parallel Watermarking Embedding Process

Watermarking Embedding Process on US_Sample_8				
Frame Number	**Sequential**		**Parallel**	
	MSE	**PSNR**	**MSE**	**PSNR**
1	33.637	49.358	33.637	49.358
2	33.662	49.358	33.662	49.358
3	33.680	49.358	33.680	49.358
4	33.770	49.466	33.770	49.466
5	33.795	49.466	33.795	49.466
6	33.813	49.466	33.813	49.466
7	33.691	49.386	33.691	49.386
8	33.716	49.386	33.716	49.386
9	33.783	49.386	33.783	49.386
10	33.783	49.438	33.783	49.438
11	33.807	49.438	33.807	49.438
12	33.824	49.438	33.824	49.438
13	33.533	49.363	33.533	49.363
14	33.558	49.363	33.558	49.363
15	33.577	49.364	33.577	49.364
16	33.569	49.292	33.569	49.292
17	33.595	49.292	33.622	49.318
18	33.612	49.291	33.612	49.291
19	33.444	49.158	33.444	49.158
20	33.469	49.158	33.469	49.158
21	33.487	49.158	33.487	49.158
22	33.554	49.197	33.554	49.197
23	33.579	49.197	33.579	49.197
24	33.596	49.197	33.596	49.197
25	33.482	49.227	33.482	49.227
26	33.517	49.227	33.517	49.227
27	33.524	49.227	33.524	49.227
28	33.497	49.270	33.497	49.270
29	33.522	49.270	33.522	49.270
30	33.540	49.270	33.540	49.270
31	33.457	49.314	33.457	49.314
32	33.483	49.315	33.483	49.315
33	33.501	49.314	33.501	49.314
Average	**33.608**	**49.315**	**33.609**	**49.316**

4.3.2 Elapsed Time

Elapsed time is the time taken to perform watermarking embedding and authentication process on medical images in multi-frames environment. This section is to test the speedup factor in parallel mode as compared to sequential mode in watermarking process.

(a) Watermarking Embedding Process

In Table 4.13, eight ultrasound samples of different frame size served as input files for both sequential and parallel watermarking embedding process, the elapsed time was recorded at the end of both processes. The speedup value was obtained by dividing the elapsed time recorded in sequential mode with parallel mode. It could be observed that the elapsed time of 15 frames ultrasound medical images (such as ultrasound sample 2, 3 and 7) have a similar result and double up at 30 frames (such as ultrasound sample 1, 5 and 8). This means that the watermarking embedding scheme are pixel oriented regardless of the sources of ultrasound medical images input, and elapsed time is proportional to frame size. In other words, the amount of bits embedded into RONI pixels was determined by the ROI size. The same ROI size have been preset in ROI-DR algorithm, which means the amount of embedded bits will be same for all samples. Therefore, the medical images of same frame size produced similar result of elapsed time, and it was double up as the frame size was double up.

The watermarking embedding process in parallel has achieved a minor speedup (1.14 ~ 2.35) relative to sequential process, speedup increased as the number of frames increased. It could be observed that the speedup for ultrasound_sample_4 (frame size of 79) is almost double of ultrasound_sample_1 (sample size of 30). But the speed up ultrasound_sample_1 is almost similar with ultrasound_sample_2 (frame size of 15). This means that parallel watermarking embedding process is more efficient in high volume of ultrasound medical images relative to sequential mode.

Table 4.13

Speedup Gain in Watermarking Embedding Process in Elapsed Time

Input File (Ultrasound Medical Images)	No. of frames	Elapsed Time in Watermarking Embedding Process (seconds)		Speedup	Output File (Watermarked ultrasound medical images)
		Sequential	Parallel		
Ultraound_Sample_1	30	29.5	19.3	1.53	Watermarked_US1
Ultraound_Sample_2	15	14.3	12.6	1.14	Watermarked_US2
Ultraound_Sample_3	15	14.4	11.7	1.23	Watermarked_US3
Ultraound_Sample_4	79	71.0	30.2	2.35	Watermarked_US4
Ultraound_Sample_5	31	28.0	19	1.47	Watermarked_US5
Ultraound_Sample_6	25	23.0	14.8	1.55	Watermarked_US6
Ultraound_Sample_7	15	14.2	12.7	1.12	Watermarked_US7
Ultraound_Sample_8	33	29.2	22.2	1.32	Watermarked_US8

(b) Watermarking Authentication Process

Different from watermarking embedding process, watermarking authentication process is to verify whether there is any tampering has occurred in the watermarked ultrasound medical images and then recovered the tampered frame to its original state.

In Table 4.14, eight fully tampered watermarked ultrasound samples of different frame size served as input files for both sequential and parallel watermarking authentication process. Fully tampered means all frames has been tampered in the sample. The elapsed time was recorded at the end of both sequential and parallel watermarking authentication processes. The speedup value was obtained by dividing the elapsed time recorded in sequential mode with parallel mode. The speedup factor gain are range from 0.61 ~ 1.76.

Table 4.14

Speedup Gain in Watermarking Authentication Process

Input File	Tampered frame / Total frame	Elapsed Time (seconds)		Speedup	Output File
		Sequential	Parallel		
Tampered_Watermarked_US1	30/30	12.680	10.500	1.210	Recovered_US1.dcm
Tampered_Watermarked_US2	15/15	5.700	9.300	0.610	Recovered_US2.dcm
Tampered_Watermarked_US3	15/15	5.510	8.400	0.660	Recovered_US3.dcm
Tampered_Watermarked_US4	79/79	25.580	14.500	1.760	Recovered_US4.dcm
Tampered_Watermarked_US5	31/31	10.220	9.300	1.100	Recovered_US5.dcm
Tampered_Watermarked_US6	25/25	7.440	9.300	0.800	Recovered_US6.dcm
Tampered_Watermarked_US7	15/15	5.310	8.200	0.650	Recovered_US7.dcm
Tampered_Watermarked_US8	33/33	12.550	10.500	1.200	Recovered_US8.dcm

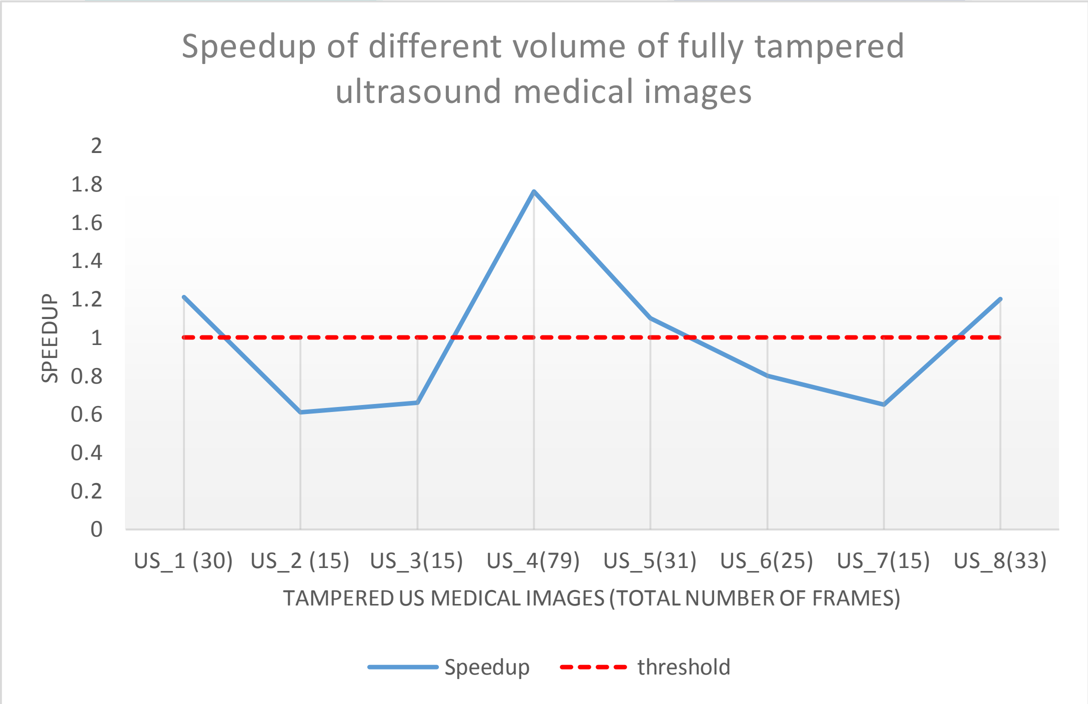


Figure 4.16. Speedup of different volume of fully tampered ultrasound medical images

If the speedup factor is 1, then it means both sequential and parallel process have same elapsed time, if the speedup factor is less than 1, then it means the watermarking authentication process in sequential mode is faster than parallel mode or vice versa.

In Figure 4.16, it could be observed that watermarking authentication process in parallel mode is faster if the volume of tampered file is more than 30 frames, this might due to the initial set up time required for parallel mode is equivalent to the time taken to process 30 frames in sequential watermarking authentication process. The different ultrasound medical images with the same frame size, such as US_2, US_3 and US_7 will give a similar speedup values, this is because watermarking process is pixel oriented, which means only the number of pixel values changes will have impact on the processing time.

(c) Overall Performance of Watermarking Process

In Table 4.15, the elapsed time for sequential watermarking embedding process was obtained from Table 4.13, and the elapsed time for parallel watermarking embedding process was obtained from Table 4.14. Speedup values was derived by dividing elapsed time of sequential mode with parallel mode. Since the whole package of watermarking process involves two steps: (1) watermarking embedding process (2) watermarking authentication process, therefore it is necessary to test the overall elapsed time involved in both process. The high speedup in watermarking embedding process is compromised by the low speedup in watermarking authentication process.

Table 4.15 has shown the overall speedup factor for eight ultrasound medical images samples are range from 0.91 ~ 2.16. If the speedup value is 1, it means that overall watermarking processing time for parallel mode is equivalent as sequential mode. If the speedup value is lesser than 1, it means that the overall watermarking processing time for parallel mode is greater/slower than sequential mode, such as ultrasound sample 2, 3 and 7, which all of them consist of 15 frames. If the speedup is greater than 1, it means that overall processing time for parallel mode is lesser/faster than sequential mode, such as ultrasound sample 1,4,5,6 and 8. Ultrasound sample 4 with a highest frame size yield the highest speedup values, it means that parallel mode have a better effect in high volume of medical images.

Table 4.15

The Speedup Gain in Overall Watermarking Process (Embedding plus Authentication)

Ultrasound Sample	**Watermarking Process**	**Sequential (seconds)**	**Parallel (seconds)**	**Speedup**
1 (frame size=30)	Embedding	29.540	19.300	1.530
	Authentication	12.680	10.500	1.210
	Overall time taken:	42.220	29.800	1.420
2 (frame size=15)	Embedding	14.320	12.600	1.140
	Authentication	5.700	9.300	0.610
	Overall time taken:	20.020	21.900	0.910
3 (frame size=15)	Embedding	14.420	11.700	1.230
	Authentication	5.510	8.400	0.660
	Overall time taken:	19.930	20.100	0.990
4 (frame size=79)	Embedding	71.040	30.200	2.350
	Authentication	25.580	14.500	1.760
	Overall time taken:	96.620	44.700	2.160
5 (frame size=31)	Embedding	28.010	19.000	1.470
	Authentication	10.220	9.300	1.100
	Overall time taken:	38.230	28.300	1.350
6 (frame size=25)	Embedding	22.990	14.800	1.550
	Authentication	7.440	9.300	0.800
	Overall time taken:	30.430	24.100	1.260
7 (frame size=15)	Embedding	14.210	12.700	1.120
	Authentication	5.310	8.200	0.650
	Overall time taken:	19.520	20.900	0.930
8 (frame size=33)	Embedding	29.190	22.200	1.320
	Authentication	12.550	10.500	1.200
	Overall time taken:	41.740	32.700	1.280

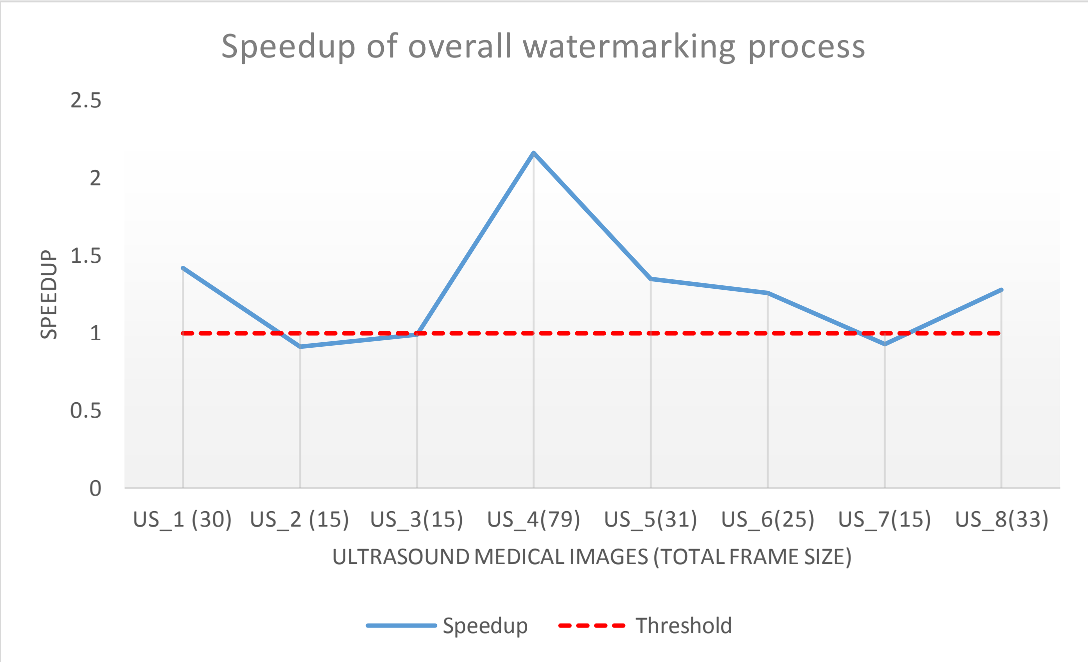


Figure 4.17. Speedup of overall watermarking process

If the speedup factor is 1, then it means both sequential and parallel process have same elapsed time, it has been established as threshold value. If the speedup factor is less than 1, then it means the overall watermarking process in sequential mode is faster than parallel mode or vice versa.

In Figure 4.17, it could be observed that the total frame size must more than 15 frames in order to achieve a positive speedup in parallel watermarking process, or else the sequential process will be faster than parallel mode. The speedup for parallel mode has increased substantially with a larger frame size, which means parallel ROI-DR watermarking processing is efficient in high volume of ultrasound medical images.

4.3.3 Robustness to Tampering

This section is to test on the robustness of multi-frames watermarking scheme in terms of its ability to: (1) detect and display tampered frames number, (2) recover to ultrasound medical images original form after the authentication process.

Table 4.16

Elapsed Time for Watermarking Authentication Process on Different Set of Tampered Frames in Watermarked Ultrasound Sample 1 Medical Images

Total of tampered frames	**Tampered frame number**	**Elapsed Time for Watermarking Authentication Process on Watermarked_US_1 Medical Images (seconds)**							**Speedup**
		Seq	**D**	**R**	**Par**	**Diff**	**D**	**R**	
0	0	1.6	Yes	Yes	9.5	-	Yes	Yes	0.17
2	1,9	1.9	Yes	Yes	10.4	0.9	Yes	Yes	0.18
4	.., 17, 24	2.4	Yes	Yes	9.5	-0.9	Yes	Yes	0.25
6	.., 2,10	3.1	Yes	Yes	10.5	1	Yes	Yes	0.30
8	.., 18, 25	3.9	Yes	Yes	10.5	0	Yes	Yes	0.37
10	.., 3, 11	4.8	Yes	Yes	10.4	-0.1	Yes	Yes	0.46
12	.., 19, 26	5.5	Yes	Yes	11.4	1	Yes	Yes	0.48
14	.., 4,12	6.2	Yes	Yes	12.6	1.2	Yes	Yes	0.49
16	.., 20, 27	6.9	Yes	Yes	11.7	-0.9	Yes	Yes	0.59
18	.., 5, 13	7.8	Yes	Yes	11.6	-0.1	Yes	Yes	0.68
20	.., 21, 28	8.6	Yes	Yes	12.7	1.1	Yes	Yes	0.68
22	.., 6, 14	9.5	Yes	Yes	12.5	-0.2	Yes	Yes	0.76
24	.., 22, 29	10.1	Yes	Yes	12.6	0.1	Yes	Yes	0.80
26	.., 7, 15	10.7	Yes	Yes	12.5	-0.1	Yes	Yes	0.86
28	.., 23, 30	11.8	Yes	Yes	12.7	0.2	Yes	Yes	0.93
30	.., 8, 16	12.4	Yes	Yes	13.8	1.1	Yes	Yes	0.90

.. – Means values same as above, for example .., 2,10 means 1,9,17,24,2,10
Seq – Sequential;
Par - Parallel
D – Able to detect and display the tampered frame number?
R – Able to recover to its original form after authentication process?
Diff = current parallel elapsed time value – previous parallel elapsed time value
Speedup = Sequential / Parallel

In Table 4.16, different set of tampered frames in same set of watermarked ultrasound medical images result has been collected. The watermarked ultrasound medical images are tampered two frames gradually in watermarked_US_1 medical images (which contains 30 frames). The elapsed time for sequential and parallel watermarking authentication process on will be recorded. A “Yes” result in D field indicated that ROI-DR multi-frames watermarking scheme was able to detect and display the tampered frame number, and a “Yes” in R field indicated that it was able perform recovery process after the tamper detection. Figure 4.18 has demonstrate the watermarked ultrasound medical images which has been tampered in frame number 1, 3 and 5 by “Pepper and Salt” painting tools. A message of tampered frame number will be display and recovered ultrasound medical images will be saved as a DICOM file after the watermarking authentication process (as illustrated in Figure 4.19).

Figure 4.18 Ultrasound Medical Images Tampered on Frame number 1, 3 and 5

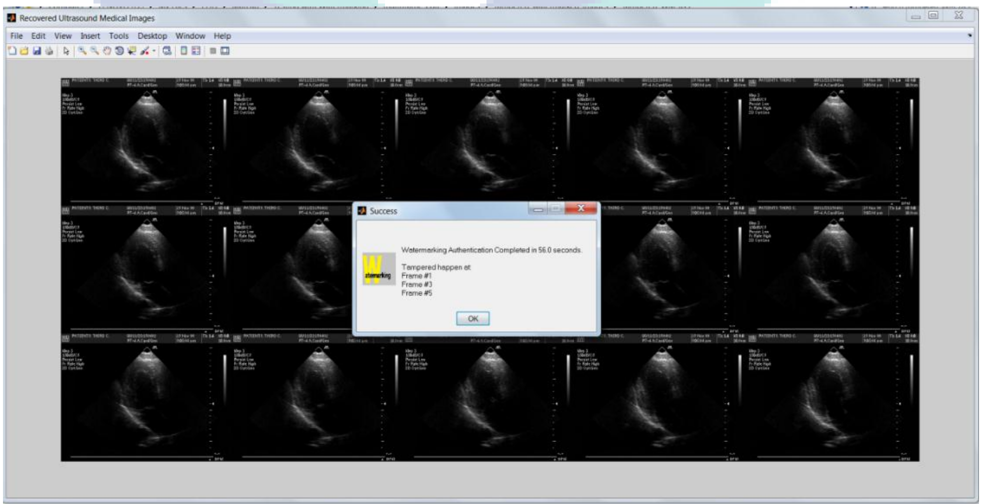
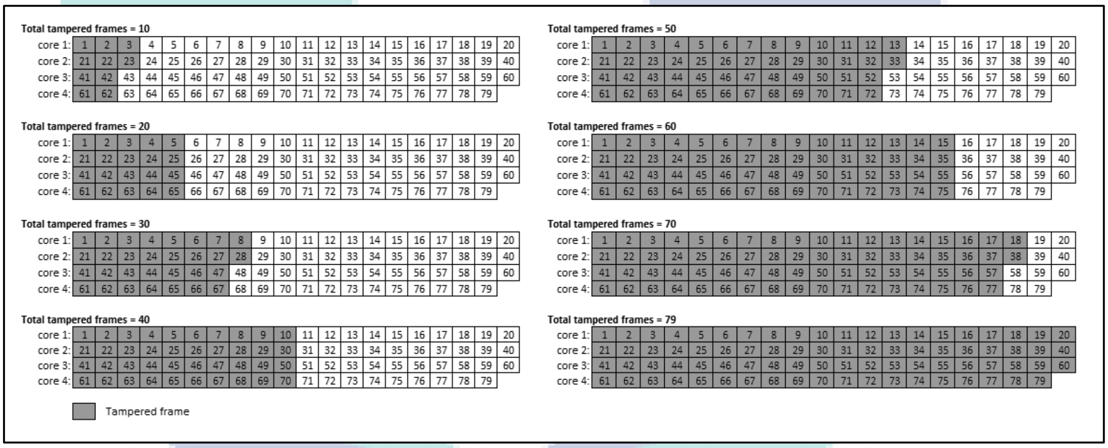


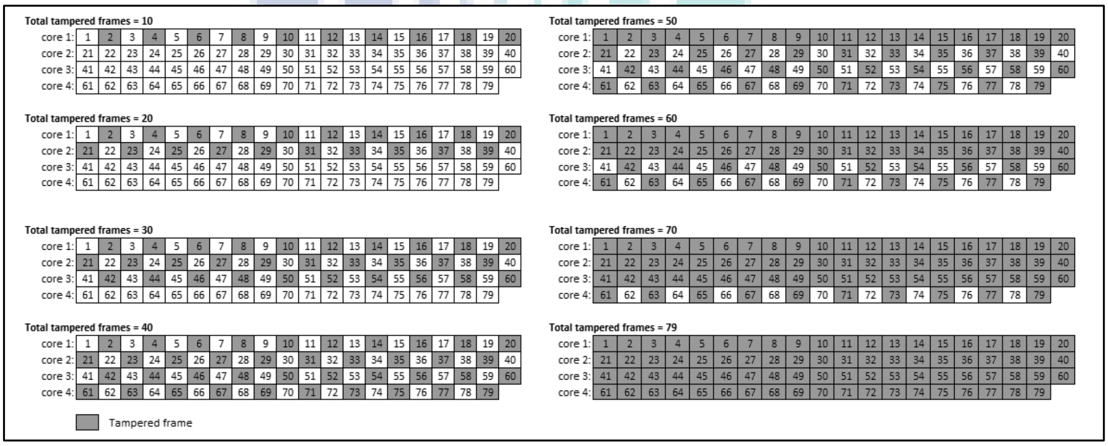
Figure 4.19 A recovered ultrasound medical images was generated and an alert message was displayed after Watermarking Authentication Process

In Table 4.16, all “Yes” results in the field of “D” and “R” indicated the robustness to tampering has been tested successfully. It could be observed that all the speedup values are less than 1, below the threshold value which means the sequential mode is faster than parallel mode, in other words, the frame size of 30 is insufficient for parallel mode to work effectively. Therefore a higher volume of ultrasound medical images (such as ultrasound sample 4 which consist of 79 frames) has been chosen for robustness testing towards tampering in parallel watermarking authentication process.

In Figure 4.20, Watermarked_US_4 has duplicated into two copies, which are watermarked_US_4A and watermarked_US_4B, and ten frames were tampered with accumulatively but in different frame numbers. This is to test whether the different layout of tampered frame organization of the same ultrasound medical image source will have impact on the elapsed time.



(a) The tampered frames layout of watermarked_US_4A



(b) The tampered frames layout for watermarked_US_4B

Figure 4.20. The tampered frames layout for watermarked_US_4A and 4B

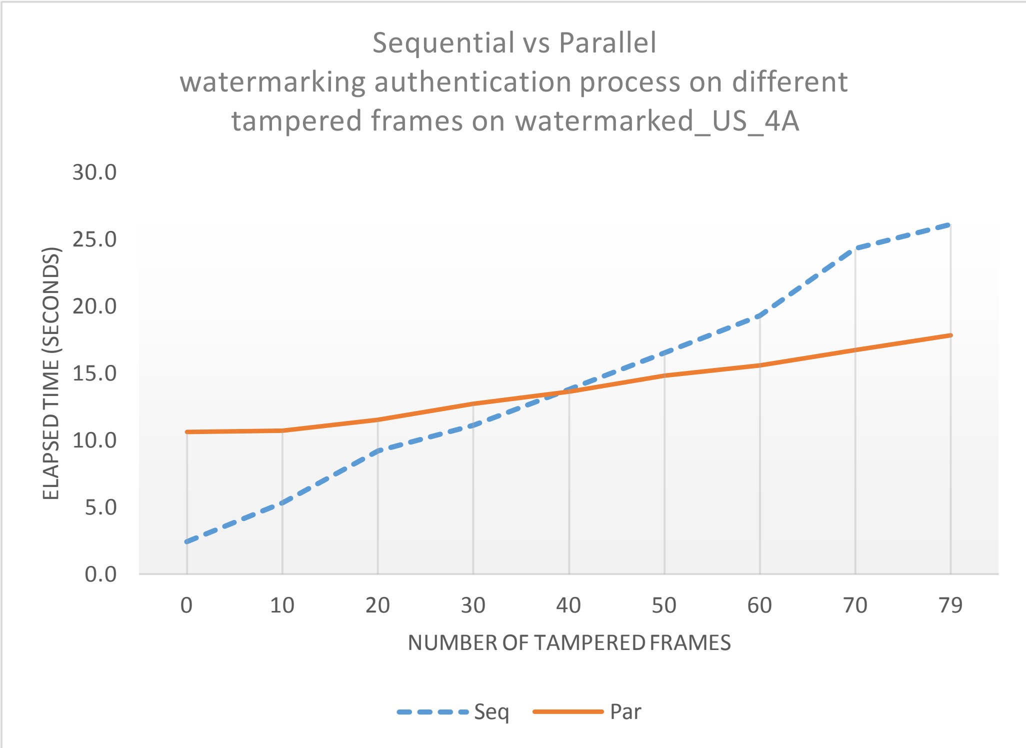
Experiment has been conducted according to the tampered layout shown in Figure 4.20 and generated a result as shown in Table 4.17. The “Yes” result obtained in D and R indicated a successful of tamper detection and recovery function in both sequential and parallel multi-frames watermarking authentication process.

Table 4.17

Watermarking Authentication Process on Watermarked_US_4A and 4B

Sample	**Total of tampered frames**	**Tampered frame number**	**Seq (sec)**	**D**	**R**	**Par (sec)**	**D**	**R**	**Speed up**
Watermarked_US_4A	0	0	2.4	Yes	Yes	10.6	Yes	Yes	0.23
	10	1-3,21-23,41,42,61,62	5.3	Yes	Yes	10.7	Yes	Yes	0.50
	20	1-5, 21-25, 41-45, 61-65	9.2	Yes	Yes	11.5	Yes	Yes	0.80
	30	1-8, 21-28, 41-47, 61-67	11.1	Yes	Yes	12.7	Yes	Yes	0.87
	40	1-10, 21-30, 41-50, 61-70	13.8	Yes	Yes	13.6	Yes	Yes	1.01
	50	1-13, 21-33, 41-52, 61-72	16.5	Yes	Yes	14.8	Yes	Yes	1.12
	60	1-15, 21-35,41-55, 65-75	19.3	Yes	Yes	15.6	Yes	Yes	1.24
	70	1-18, 21-38, 41-57, 61-77	24.3	Yes	Yes	16.7	Yes	Yes	1.45
	79	all frames	26.1	Yes	Yes	17.8	Yes	Yes	1.47
Watermarked_US_4B	0	0	2.3	Yes	Yes	10.5	Yes	Yes	0.22
	10	even number from 2-20	5.5	Yes	Yes	13.8	Yes	Yes	0.40
	20	…, odd number from 21-39	8.8	Yes	Yes	13.5	Yes	Yes	0.65
	30	…, even number from 42-60	11.7	Yes	Yes	13.5	Yes	Yes	0.87
	40	…, odd number from 61-79	14.7	Yes	Yes	15.6	Yes	Yes	0.94
	50	…, 1-20	17.2	Yes	Yes	16.8	Yes	Yes	1.02
	60	…, 21-40	20.1	Yes	Yes	19.8	Yes	Yes	1.01
	70	…, 41-60	22.7	Yes	Yes	17.8	Yes	Yes	1.27
	79	all frames	25.1	Yes	Yes	17.8	Yes	Yes	1.41

.. – Means values same as above, for example .., 2,10 means 1,9,17,24,2,10
Seq – Sequential; **Par** – Parallel; **D** – Able to detect and display the tampered frame number?
R – Able to recover to its original form after authentication process?
Speedup = Sequential / Parallel



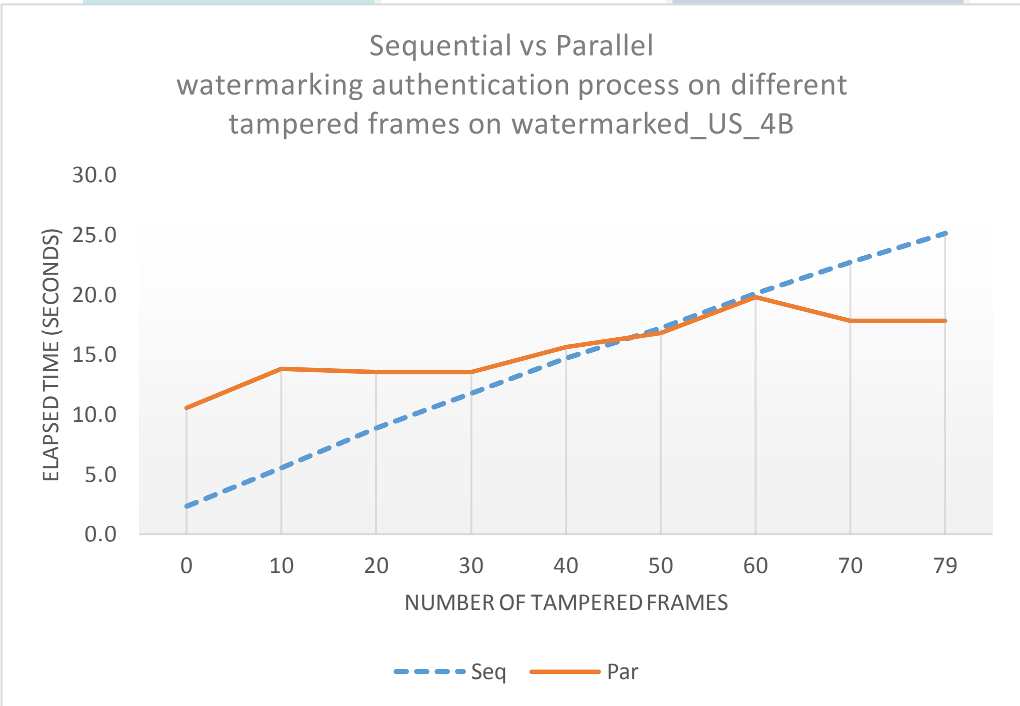


Figure 4.21. Elapsed time of sequential versus parallel watermarking authentication process on watermarked_US_4A and watermarked_US4_B respectively

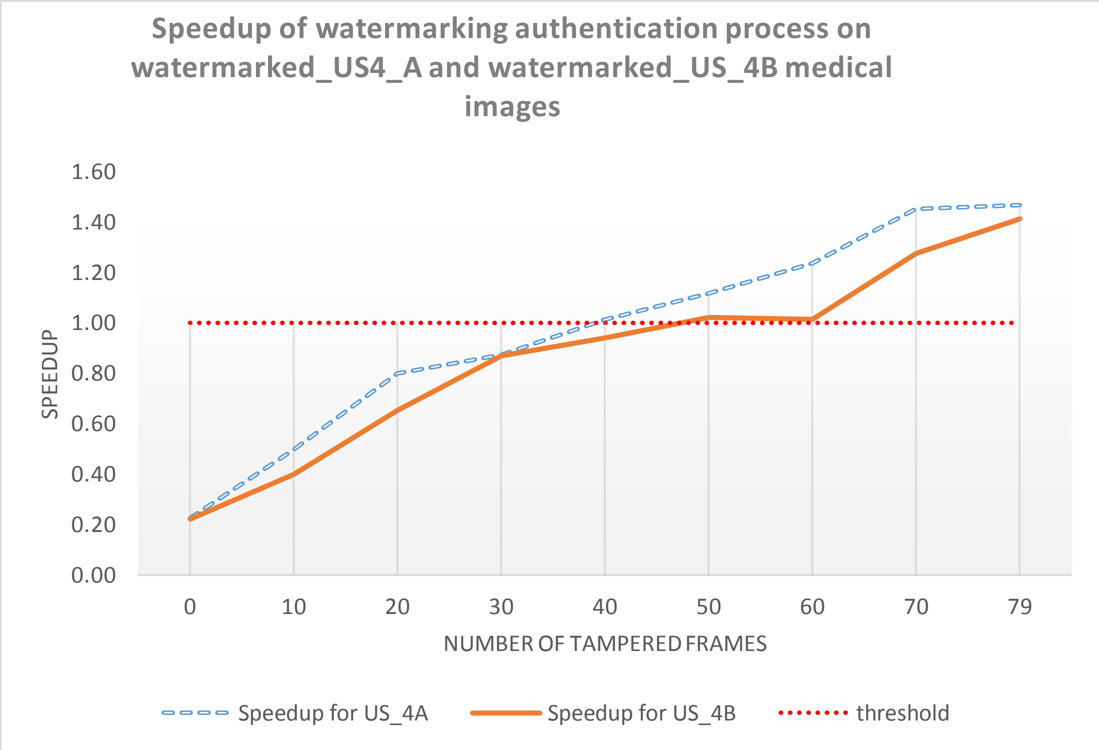


Figure 4.22. Speedup of watermarking authentication process on watermarked_US_4A and watermarked_US4_B medical images

Speedup of watermarking authentication process on watermarked_US_4A is higher than watermarked_US_4B medical images (as shown in Figure 4.22), this indicated the layout of tampered frame organization would affect the speedup performance, which means the way of tampered frame order arrangement has more impact on parallel mode than sequential mode. The line plotted for sequential mode in both watermarked_US4_A and watermarked_US_4B are almost identical (as shown in Figure 4.21), which indicated that the arrangement of tampered frame order has no impact on sequential mode. Whereas in parallel mode, the line plotted in watermarked_US_4B tend to be more fluctuated than watermarked_US_4A, it is because tampered frames are distributed into four cores evenly and fairer in watermarked_US_4A, whereas watermarked_US_4B has distributed tampered frames unevenly and heavily on one core than the others (as shown in Figure 4.20), in which it has under-utilized the power of parallel computing on multi-cores. In summary, parallel watermarking authentication processing on multi-core would work best if the tampered frames are distributed evenly on each core, in which every core would share the same work load and finish within the same time frames. In Table 4.17, it could be observed that parallel watermarking authentication process required more initial set up time than sequential mode, and they only break even at 40~50 tampered frames. Therefore it could be concluded that small tampered frame size of ultrasound medical images, below 40 frames, might under-utilize the parallel framework.

4.3.4 Summary

Both sequential and parallel watermarking embedding processes have produced same MSE and PSNR which indicate operation either in sequential or parallel mode does not affect the image quality and its fidelity. The imperceptibility was achieved where images after watermarking embedding process are visually indistinguishable as the original images. The average PSNR values calculated for all images ranging between 47.966~49.316 dB, which it is within acceptable range for clinical diagnosis purposes. The speedup for parallel mode has increased substantially with a larger frame size, which means parallel ROI-DR watermarking processing is efficient in high volume of ultrasound medical images. In other words, sequential watermarking process will be a better choice for a small volume of ultrasound medical images, which less than 15 frames. The organization of tampered frames will have impact on processing time, where evenly distributed tampered frame across the cores will have shorter elapsed time. As for robustness to tampering, the effectiveness and efficiency of the tamper detection and recovery function in multi-frames environment has been tested successfully.

4.3.5 Performance Comparison between ROI-DR and TALLOR Multi-frames Watermarking Schemes

ROI-DR and TALLOR multi-frames watermarking scheme were tested under the same condition, such as same set of raw or tampered ultrasound medical image samples, same hardware and software testing environment. It is to ensure fairness and accuracy in performance comparison. Three ultrasound medical image samples has been used for performance comparison between ROI-DR and TALLOR multi-frames watermarking scheme (as shown in Table 4.18). Three performance metrics has been used for both schemes in proposed multi-frames watermarking scheme: (1) Imperceptibility, (2) Robustness to tampering, and (3) Elapsed time of sequential and parallel watermarking process, a division between them yield speedup factor.

Table 4.18

A Table of Performance Comparison between ROI-DR and TALLOR Multi-frames Watermarking Scheme

Ultrasound medical image sample	**Sample 1 (60 frames)**		**Sample 2 (15 frames)**		**Sample 3 (15 frames)**	
Performance metrics	ROI-DR	TALLOR	ROI-DR	TALLOR	ROI-DR	TALLOR
(1) Imperceptibility in PSNR values (dB)	48.09	48.08	47.97	47.97	47.94	48.51
(2) Robustness to tampering						
- Tamper detection	Yes	Yes	Yes	Yes	Yes	Yes
- Tamper recovery	Yes	Yes	Yes	Yes	Yes	Yes
(3) Elapsed Time for watermarking Embedding process						
- in sequential mode	29.5	738.7	14.3	336.40	14.4	332.60
- in parallel mode	19.3	38.3	12.6	23.80	11.7	22.80
- Speedup	1.53	19.29	1.14	14.13	1.23	14.59
Elapsed Time for watermarking Authentication process						
- in sequential mode	12.68	597.1	5.70	270	5.51	241.2
- in parallel mode	10.50	164.2	9.30	92.6	8.40	73.3
- Speedup	1.21	3.64	0.61	2.92	0.66	3.29

Multi-frames watermarking scheme has tested successfully in both ROI-DR and TALLOR watermarking schemes in multi-frames environment, both schemes have achieved a good result in imperceptibility with average PSNR values around 48 dB, where image after watermarking embedding process are visually undisguisable as original image. Both of them function effectively in the multi-frames watermarking scheme, where it was robust to tampering, able to detect tampering, display the message of tampered frame number and recovered to its original form. The efficiency of multi-frames watermarking scheme is largely depend on the speedup factor where the amount of time that required to process multi-frames watermarking process sequentially relative to parallel mode, and the speedup factor is affected by the number of frames processed and the watermarking scheme algorithm. The results have shown that proposed multi-frames watermarking scheme work efficiently in high volume of ultrasound medical images. In other words, sequential watermarking processing would be a better choice for low volume of medical images, where the speedup factor is below than 1. The higher speedup factor in TALLOR indicated that the proposed multi-frames watermarking scheme was much suitable in

TALLOR as compare to ROI-DR. This is due to TALLOR algorithm more complex and it required more processing time than ROI-DR, in which indirectly affected the speedup factor. Whereas, sequential watermarking process may be suitable approach for ROI-DR.

4.4 RESULT ANALYSIS OF ROI-DR IN OTHER MEDICAL IMAGES

4.4.1 Imperceptibility and Elapsed Time in Watermarking Embedding Process

ROI-DR multi-frames watermarking embedding process has been applied on MRI and XA medical imaging modalities.

Table 4.19

PSNR and MSE Values after ROI-DR multi-frames watermarking embedding process on MRI medical images

ROI-DR Multi-Frames Watermarking Embedding Process				
MRI.dcm	**Sequential mode**		**Parallel Mode**	
Frame_No	**PSNR**	**MSE**	**PSNR**	**MSE**
1	49.34	35.64	49.00	37.45
2	49.00	37.45	49.00	37.45
3	49.11	35.11	49.11	35.11
4	49.12	35.51	49.12	35.51
5	49.06	34.51	49.06	34.51
6	49.02	34.79	49.02	34.79
7	49.01	35.85	49.01	35.85
8	49.07	36.39	49.07	36.39
9	49.02	35.86	49.02	35.86
10	48.96	35.65	48.99	35.69
11	49.10	36.09	49.10	36.09
12	49.15	36.21	49.15	36.21
13	49.15	36.31	49.16	36.33
14	49.10	36.88	49.10	36.88
15	49.10	37.50	49.10	37.50
16	49.11	37.42	49.11	37.42
Average:	**49.09**	**36.07**	**49.07**	**36.19**

Table 4.20

PSNR Values after ROI-DR multi-frames watermarking embedding process on XA medical images.

ROI-DR Multi-frames Watermarking Embedding Process				
XA.dcm	**Sequential mode**		**Parallel Mode**	
Frame_No	**PSNR**	**MSE**	**PSNR**	**MSE**
1	52.70	38.96	52.70	38.96
2	52.69	39.54	52.69	39.54
3	52.65	39.69	52.65	39.69
4	52.65	39.80	52.65	39.80
5	52.65	39.74	52.65	39.74
6	52.72	39.93	52.72	39.93
7	52.68	39.90	52.68	39.90
8	52.69	39.87	52.69	39.87
9	52.69	39.87	52.69	39.87
10	52.71	39.90	52.71	39.90
11	52.68	39.89	52.68	39.89
12	52.67	39.86	52.67	39.86
Average:	52.68	39.75	52.68	39.75

In Table 4.19 and 4.20, the average PSNR values for MRI is 49 dB and XA is 52.68 dB, both of them are above minimum acceptable PSNR value (30 dB). The PSNR value in XA is better than MRI, it may due to the size of ROI in XA is smaller than MRI, therefore less embedded bits in RONI, which means less distortion, and hence achieved high PSNR value. The similar result of average PSNR values in both sequential and parallel mode indicated that multi-frames watermarking embedding process will have the same performance regardless it is on sequential or parallel mode.

In Table 4.21, ROI-DR watermarking scheme was tested on a single frame of MRI, XA, CT and US. The result shown that XA has a highest PSNR values followed by MRI, US and CT. The elapsed time in watermarking embedding is lesser than 1 second, it proved that ROI-DR indeed was a speedy watermarking scheme.

Table 4.21

Result of ROI-DR watermarking scheme on different medical image modalities.

Measurement	**RESULT**			
	MRI	**XA**	**CT**	**US**
PSNR (dB)	49.3372	52.6952	47.5813	48.4141
Elapsed time in watermarking embedding process (seconds)	1.02	0.91	1.03	0.98

In Table 4.22, the speedup factor was generated by diving elapsed time of watermarking embedding process in sequential mode with parallel mode. The speedup factor for MRI and XA are less than 1, it means that sequential mode is faster than parallel. This might due to the frames size of MRI and XA is not big enough for proposed multi-frames watermarking scheme (with parallelism component) to take effect.

Table 4.22

Speedup factor for ROI-DR multi-frames watermarking embedding process on different medical image modalities.

Medical Image Modalities	**Number of frames**	**Elapsed time (seconds)**		**Speedup factor = Sequential/Parallel**
		Sequential	**Parallel**	
MRI	16	16.57	18.50	0.90
XA	12	10.94	16.10	0.68
US	15	14.30	12.60	1.14

4.4.2 Robustness to Tampering and Elapsed Time in Watermarking Authentication Process

MRI, XA and CT has been tampered by adding "Pepper and Salt" noise into ROI, cloning a portion of image into ROI and flipping a portion of ROI vertically (as shown in figure 4.23– 4.25). The odd number of frames has been selected for tampering, such as frame number 1, 3, 5 until the last odd number of the frame.

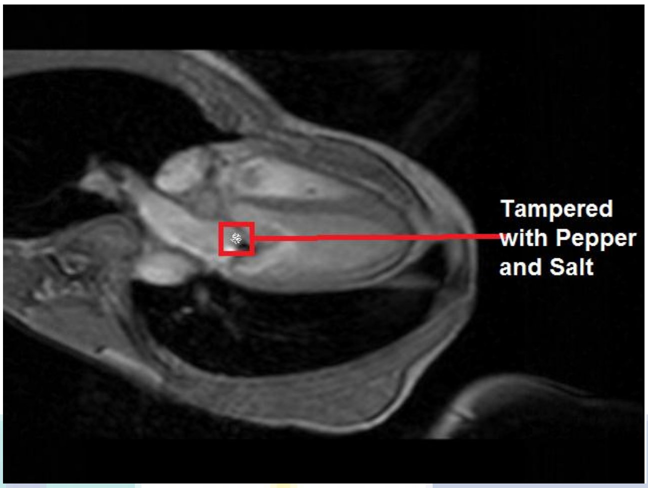


Figure 4.23. Watermarked MRI tampered by adding pepper and salt noise into ROI

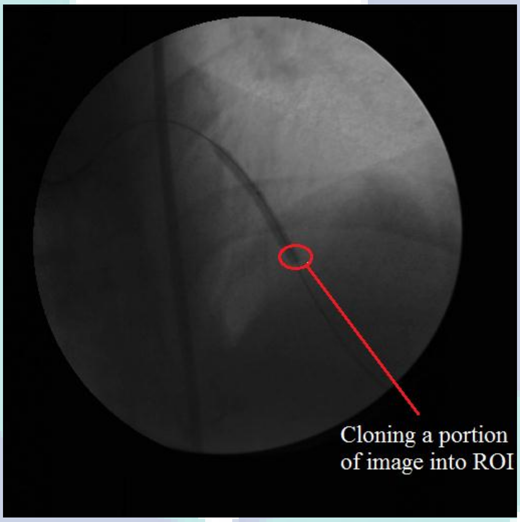


Figure 4.24. Watermarked XA tampered by cloning a portion of image into ROI

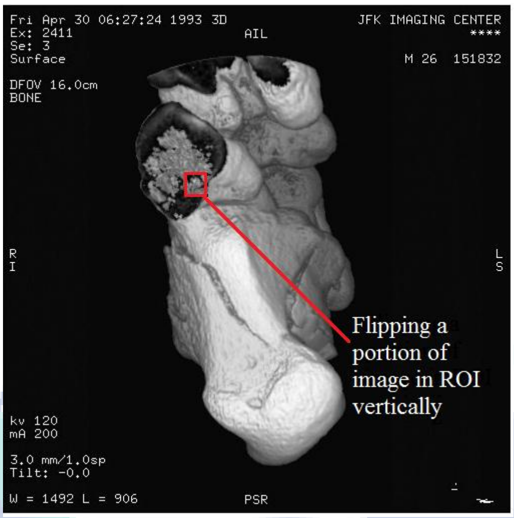


Figure 4.25. Watermarked CT tampered by flipping a portion of image in ROI vertically

In Table 4.23, the elapsed time for ROI-DR watermarking authentication process was lesser than 0.5 second, it means that ROI-DR was efficient. The “Yes” result indicated that ROI-DR was robust to tampering and able to perform tamper detection and recovery function regardless the type of medical imaging modalities.

Table 4.23

Speedup factor for ROI-DR multi-frames watermarking authentication process on different medical images

Measurement	**RESULT**			
	MRI	**XA**	**CT**	**US**
Elapsed time in watermarking authentication process (seconds)	0.46	0.29	0.43	0.34
Able to detect tampering	Yes	Yes	Yes	Yes
Able to produce exact ROI recovery	Yes	Yes	Yes	Yes
Able to generate a recovered DICOM file	Yes	Yes	Yes	Yes

In Table 4.24, the high elapsed time in parallel watermarking authentication process indicated that the proposed multi-frames watermarking scheme was not a suitable choice to be adopted. This might due to the frame size was too small to utilize the power of parallelism in multi-frames watermarking scheme.

Table 4.24

Speedup factor for ROI-DR multi-frames watermarking authentication process on different medical image modalities.

Medical Image Modalities	**Number of frames**	**Elapsed time (seconds)**		**Speedup factor = Sequential/Parallel**
		Sequential	**Parallel**	
MRI	16	3.28	16.90	0.19
XA	12	1.69	12.70	0.13
US	15	5.70	9.30	0.61

4.5 CONCLUSION

In multi-frames watermarking scheme, the imperceptibility test with PSNR values (~ 48 dB) which more than 30 dB indicated medical images was not degraded irrespective to the number of frames processed. The robustness to tampering has been successfully tested, where it could perform recovery function and display tampered frames message. As for elapsed time measurement, ROI-DR has proven that a light and optimized code in watermarking scheme algorithm could reduce the processing time significantly and achieved a high speedup factor relative to TALLOR in sequential multi-frames watermarking process. If a watermarking scheme could produce a high speed in a single frame (such as ROI-DR), then watermarking process in sequential mode might be faster than parallel mode. Parallelism in proposed multi-frames watermarking scheme help to reduce processing time significantly especially in those watermarking scheme with a complex algorithm (such as TALLOR) and high volume of medical images that required high processing time. ROI-DR multi-frames watermarking scheme work successfully on others medical imaging modalities besides ultrasound medical images provided it is 8 bits depth.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 INTRODUCTION

This chapter have listed contributions and limitations of the thesis in sections 6.2 and 6.3 respectively. Section 6.4 described the future work based on the outcome of this thesis. Lastly section 6.5 summarised the chapter.

5.2 CONTRIBUTIONS

The contributions of this thesis are listed as below:-

- Successfully develop a new watermarking scheme, which is ROI-DR that has achieved a high speedup factor relative to TALLOR and TALLOR-RS watermarking schemes. ROI-DR watermarking scheme has achieved average speedup factor of 22.55 and 26.65 in watermarking embedding process, 21.89 and 42.79 in watermarking authentication process in relative to TALLOR and TALLOR-RS respectively.
- ROI-DR watermarking scheme is able to detect whether the tampering region has occurred on ROI or RONI.
- Successfully develop and implement multi-frames watermarking scheme. It has been found out that processing time could be reduced by utilizing parallel computing on multicores, the proposed multi-frames watermarking scheme has been developed and tested on two watermarking schemes: ROI-DR and TALLOR. It has shown that the proposed scheme is reusable and could work on different watermarking schemes. It

has been proven that a significant speedup in parallel watermarking processing in relative to sequential watermarking process.

- In the proposed multi-frames watermarking scheme, all frames are independent, which means a specific frame extraction from volumetric watermarked medical images for authentication process is allowed, but it is not the case in the watermarking schemes that proposed by Wenbo, D. et al. (2012), who exploits the 3-D property of volumetric DICOM images where the watermarking process is dependent on the leading image (first image in a stack volume of medical images), thus frames extraction is not allowed. Furthermore it does not provide ROI recovery function.
- ROI-DR multi-frames watermarking scheme are able to work on other medical imaging modalities besides ultrasound medical images.

5.3 LIMITATIONS

The limitations of this thesis are listed as below:-

- ROI-DR watermarking scheme is not reversible, in which the watermarks embedded into RONI is not removed and RONI bits is not restored after the watermarking authentication process. Non-reversible has provided an advantage of allowing multiple times authentication as long as the watermarks is still embedded in RONI. The reason of retaining the watermark is because RONI is insignificant for clinical diagnosis and the watermarked ultrasound medical image are visually undisguisable as compared to original ultrasound medical image, therefore doctor may not even realised the watermark existences.
- ROI-DR watermarking scheme has not included tamper localization features because it is time consuming as checking process will not stop until the whole ROI has been checked, which conflicts with the objective of improving the time performance.
- ROI need to be pre-defined before watermarking process.
- The need of parallelism in multi-frames watermarking scheme is lesser if the watermarking processing time in a single frame is less than one second, such as ROI-DR multi-frames watermarking scheme. This is because the initial set up time of parallel computing will be higher than the watermarking processing time itself. Therefore parallel watermarking process on multicores work best in high volume of

ultrasound medical images, whereas sequential watermarking process will be a best choice for low volume of ultrasound medical images.

5.4 FUTURE WORK

Some possible future works are listed as following:

- Develop a watermarking scheme that is able to auto detect the ROI and allow doctor to select a specific ROI regions manually.
- Implement a Graphical Processing Unit (GPU) computing on medical images for large dataset of medical images in order to achieve a better performance in processing time.

5.5 SUMMARY

This chapter presented the research summary as well as the contributions and limitations of the research. The outcome from this research has opened up some possibilities for future work. Based on the results and evaluations, the objectives of this research outlined in chapter 1 have been achieved.

Research Process	Contribution
Theory	• Review on existing watermarking scheme and watermarking concept. • Analyse TALLOR and TALLOR-RS watermarking schemes and find ways to improve processing time. • Review parallel computing on multicores and its concept. • Explore the MATLAB parallel computing toolbox.
Practice	• Develop an ROI-DR watermarking scheme that improves the processing time as compared to TALLOR and TALLOR-RS. • Test these three schemes under the same conditions and environments. • Develop a sequential watermarking scheme on multi-frames ultrasound medical images. • Develop a multi-frames watermarking scheme.

	• Integrated ROI-DR and TALLOR watermarking schemes into the multi-frames watermarking scheme. • Test and compare sequential and parallel watermarking process, and measure the speedup factors.
Outcome	• ROI-DR watermarking scheme on single frame medical image. • A comparative Table of three watermarking schemes: ROI-DR, TALLOR and TALLOR-RS. • Sequential watermarking scheme on multi-frames medical images. • Parallelism in multi-frames watermarking scheme enable digital watermarking process to perform on multicores in parallel.

REFERENCES

Aherrahrou, N. and Tairi, H. (2015). PDE based scheme for multi-modal medical image watermarking. *Journal of BioMedical Engineering OnLine, 14*(1), pp.108

Al-Haj, A. and Amer, A. (2014). Secured telemedicine using region-based watermarking with tamper localization. *Journal of Digital Imaging, 27*(6), pp. 737–50.

Al-Qershi, O.M. and Khoo, B.E. (2011). Authentication and data hiding using a hybrid ROI-based watermarking scheme for DICOM images. *Journal of Digital Imaging 24*, pp. 114-125.

Amit, B. and Brijendra, K.J. (2015). Improved parallel lexical analysis using OpenMp on multi-core machines. *Procedia Computer Science, 49*(1), pp. 211-219.

Anon. (2008). Medical image integrity control seeking into the detail of the tampering. *Engineering in Medicine and Biology Society*, pp. 414–417.

Anon.(2006). Digital imaging and communications in medicine (DICOM) standard. (Online)
Retrieved from http://medical.nema.org/ dicom/2006/2006 on 20 May 2016

Atta, R., Naseem, M.T., Qureshi, I.M. and Muzaffar, M.Z. (2016). Spread spectrum based invertible watermarking for medical images using RNS & Chaos. *The International Arab Journal of Information Technology.13*(2), pp. 223 - 231

Brooks, F. P. (1996). *The mythical man month essays on software engineering*. Anniversary ed. USA: Addison-Wesley.

Chang, C.C., Lin, C.Y. and Wang, Y.Z. (2006). New image steganographic methods using run-length approach. *Information Sciences, 176* (22), pp. 3393-3408.

Chao, H.M., Hsu, C.M. and Miaou, S.G. (2002). A data-hiding technique with authentication, integration, and confidentiality for electronic patient records. *IEEE Transactions on Information Technology in Biomedicine, 6*(1), pp. 46–53.

Chen BL, Li ED, Yamawuchi K, Kato K, Naganawa S. and Miao WJ. (2010). Impact of adjustment measures on reducing outpatient waiting time in a community hospital: application of a computer simulation. *Chinese Medical Journal (English Edition). 123*(5), pp. 574.

Cheung, S.C, Chiu, D.K.W. and Ho, C. (2008). The use of digital watermarking for intelligence multimedia document distribution. *Journal of Theoretical and Applied Electronic Commerce Research, 3*(3), pp. 103-118.

Coatrieux, G., Maitre, H., Sankur, B., Rolland, Y. and Collorec, R. (2000). Relevance of watermarking in medical imaging. *Proceedings of the Information Technology Applications in Biomedicine* (pp. 250-255). IEEE.

Coatrieux, G., Puentes, J., Roux, C., Lamard, M., & Daccache, W. (2006). A low distortion and reversible watermark: application to angiographic images of the retina. *Proceedings of the Engineering in Medicine and Biology Society* (pp. 2224-2227). IEEE.

Coatrieux, G.H., Huang, H., Shu, L., Luo, and Roux, C. (2013). A watermarking-based medical image integrity control system and an image moment signature for tampering characterization. *IEEE Journal of Biomedical and Health Informatics, 17*(6), 1057–1067.

Cox, I.J., Miller, M.L. and Bloom. (2002). *Digital watermarking*. San Francisco: Morgan Kaufmann.

Cox, I.J., Miller, M.L., Bloom, J.A., Fridrich, J. and Kalker, T. (2008). *Models of watermarking. in: digital watermarking and steganography*. Second ed. San Francisco: Morgan Kaufmann, pp. 61–103.

Dandu, R.V. (2012). Managing DICOM images: tips and tricks for the radiologist. *Indian Journal of Radiology Imaging*, *22*(1), pp. 4–13.

Das, S. and Kundu, M.K. (2010). Lossless ROI medical image watermarking technique with enhance security and high payload embedding. Istanbul, Turkey, *Proceeding of the 20th International Conference on Pattern Recognition, ICPR 2010* (pp. 1457-1460). IEEE

Das, S. and Kundu, M.K. (2012). Effective management of medical information through a novel blind watermarking technique. *Journal of Medical Systems*, *36*(5), pp. 3339-3351.

Das, S. and Kundu, M.K. (2013). Effective management of medical information through ROI-lossless fragile image watermarking technique. *Computer Methods and Programs in Biomedicine*, *111*(3), pp. 662–675.

David A. Huffman. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE(Institute of Radio Engineers)*, *40*(9) (pp.1098-1101). IEEE

Degroote, J. and Vierendeels, J. (2011). Multi-solver algorithms for the partitioned simulation of fluid-structure interaction. *Computer Methods in Applied Mechanics and Engineering, 200*(25), pp. 2195-2210.

DICOM Standards (2013). Digital imaging and communications in medicine ps 3 2013. Rosslyn, Virginia, USA: *National Electrical Manufactures Association* (online). Retrieved from http://dicom.nema.org/dicom/2013/output/chtml/part03/sect_C.7.html #sect_C.7.6.6 on 3 April 2017.

Fallahpour, M., Megias, D. and Ghanbari, M. (2009). High capacity, reversible data hiding in medical images. *Proceedings of the 16th IEEE International Conference on Image Processing (ICIP)* (pp. 4241–4244). IEEE

Fawad, A., Siyal, M.Y. and Vali, U.A. (2010). A secure and robust hash-based scheme for image authentication. *Signal Processing*, *90*(2010), pp. 1456-1470.

Fotopoulos, V., Stavrinou, M.L. and Skodras, A.N. (2008). Medical image authetication and self correction through an adaptive reversible watermarking technique. *Proceedings of the 8th IEEE International Conference on BioInformatics and BioEngineering* (pp. 1-5). IEEE

Giakoumaki, A., Pavlopoulos, S. and Koutsouris, D. (2003). A medical image watermarking scheme based on wavelet transform. *Proceedings of the 25th Annual International Conference of the IEEE-EMBS* (pp. 856–859). IEEE

Guo, X. and Zhuang, T. G. (2009). Lossless watermarking for verifying the integrity of medical images with tamper localization. *Journal of Digital Imaging*, *22*(6), pp. 620–628.

Hamzaoui, R. and Saupe, D. (2006). *Fractal image compression: document and image compression (signal processing and communications)*. M. Barni (Ed.). USA: CRC Press

Jiang, H., and Ganesan, N. (2016). CUDAMPF: a multi-tiered parallel framework for accelerating protein sequence search in HMMER on CUDA-enabled GPU. *BMC Bioinformatics, 17*(1), pp.106

Jasni, M.Z and Abdul, R. F. (2006). Medical image watermarking with tamper detection and recovery. *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (pp. 3270-3273). IEEE

Jeff, M. (2002). Accessing data in DICOM files (Technical Report). USA: MathWorks.

Kadah, Y. M., Abd-Elmoniem, K. Z., and Farag, A. A. (2011). Parallel computation in medical imaging applications. *International Journal of Biomedical Imaging 2011*, vol. 2011, Article ID 840181, 2 pages. doi:10.1155/2011/840181

Kim.K.S., Lee, M.J., Lee, J.W., Oh, T.W. and Lee, H.Y. (2011). Region-based tampering detection and recovery using homogeneity analysis in quality-sensitive imaging. *Computer Vision and Image Understanding*, *115*(9), pp.1308–1323.

Larsen, M., Meredith,J.S., Navratil, P.A.and Childs, H. (2015). Ray tracing within a data parallel framework. *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis)* (pp. 279 – 286). IEEE

Li, Y., Chow, P., Jiang, J., Zhang, M. and Wei, S. (2014). Software/hardware parallel long-period random number generation framework based on the WELL method. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *22*(5), pp. 1054-1059

Liew, S.C. and Jasni, M.Z. (2010). Reversible medical image watermarking for tamper detection and recovery. *Proceedings of the Computer Science and Information Technology (ICCSIT)*, (Vol.5, pp. 417-420). IEEE

Liew, S.C. (2011). *Tamper localization and recovery watermarking schemes for medical images in PACS*. Ph.D. Thesis. Universiti Malayisa Pahang, Malayisa

Liew, S.C., Liew, S.W. and Jasni, M.Z. (2013). Tamper localization and lossless recovery watermarking scheme with ROI segmentation and multilevel authentication. *Journal of Digital Imaging*, *26*(2), pp. 316-325.

Lim, S.J., Moon, H.M., Chae, S.H. and Chung, Y. and Pan, S.B. (2009). JPEG 2000 and digital watermarking technique using in medical image. *Proceedings of the 3rd IEEE International Conference on Secure Software Integration Reliability Improvement* (pp. 413–416). IEEE

Lin, R.S. and Hu, S.W. (2009). A Modified Run-length Image Data Hiding for High Embedding Capacity. *Proceedings of the 5th International Conference on Information Assurance and Security* (pp. 673-676). IEEE

Linczuk, M., Pozniak, K., & Romaniuk, R. (2013). Review of parallel computing methods and tools for FPGA technology. *Proceedings of the Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2013* (pp. 890321-890321). IEEE

Luo, G. and Zhang, D. (2010). Efficiency improvement for data-processing of partial discharge signals using parallel computing. *Proceedings of the 10th IEEE International Conference on Solid Dielectrics* (pp. 1-4). IEEE

Mahmood, Z., Khan, M. U. S., Jawad, M., Khan, S. U. and Yang. L. T. (2015). A parallel framework for object detection and recognition for secure vehicle parking. *Proceedings of the 2015 IEEE 12th International Conference on Embedded Software and Systems (ICESS)* (pp. 892-895). IEEE

Mahmoud, I. and Jalme, P. (2013). Multi-series DICOM: an extension of DICOM that stores a whole study in a single object. *Journal of Digital Imaging*, *26*(4), pp.691–697

MathWorks. (2015). How parallel computing products run a job. (online) Retrieved from http://www.mathworks.com/help/distcomp/how-parallel-computing-products-run-a-job.html on 7 April 2015.

Memon, N. A. and Gilani, S.A.M. (2008). NROI watermarking of medical images for content authentication. *Proceedings of 12th IEEE International Mutitopic Conference (INMIC'08)* (pp. 106-110). IEEE

Memon, N.A., Gilani, S.A.M. and Ali, A. (2009). Watermarking of chest CT scan medical images for content authentication. *Proceedings of the International Conference on Information and Communication Technologies* (pp. 175–180). IEEE

Mike, B. (2017). *Parallel programming: speedups and Amdahl's law. Slide*. Oregon, USA: Oregon State University

Minye, W., Yang, Z., Zhikang, D. and Xing, W. (2014). A parallel processing model for big medical image data. *Proceedings of the Information Technology and Artificial Intelligence Conference (ITAIC)* (pp.266-269). IEEE

Najmuddin, A., Ibrahim, I. and Ismail, S. (2010). A simulation approach: improving patient waiting time for multiphase patient flow of obstetrics and gynaecology department (O and G Department) in local specialist centre. *WSEAS Transactions on Mathematics*. *10*(5), pp. 778–90.

Nyeem, H., Boles, W., & Boyd, C. (2013). A review of medical image watermarking requirements for teleradiology. *Journal of Digital Imaging*, *26*(2), pp. 326-343.

Osamah, M. and Khoo, B. E. (2011). Authentication and data hiding using a hybrid ROI based watermarking scheme for DICOM images. *Journal of Digital Imaging*, *24*(1), pp. 114-125.

Pooja, D. and Kavita, K. (2013). A study on spatial and transform domain watermarking techniques. *International Journal of Computer Applications (0975 – 8887)*, *7*(14). pp. 38 - 41

Qiang, L., Xi, C., Yi, L., ZhongHua, L. and YanGang, W. (2015). A new parallel framework of distributed SWAT calibration. *Journal of Arid Land*, *7* (1), pp. 122-131.

Rashmi, A.J. and Dinesh, V. P. (2012). Scalable and flexible heterogeneous multi-core system. *International Journal of Advanced Computer Science and Applications*, *3*(12), pp. 186-191

Rayachoti, E.and Edara, S.R. (2015). Robust medical image watermarking technique for accurate detection of tampers inside region of interest and recovering original region of interest. *IET Image Processing*, *9*(8), pp. 615-625

Roy, A. B., Dey, D., Mohanty, B., & Banerjee, D. (2012). Comparison of FFT, DCT, DWT, WHT Compression Techniques on Electrocardiogram & Photoplethysmography Signals. In *IJCA Special Issue on International Conference on Computing, Communication and Sensor Network CCSN* (pp. 6-11). IEEE

Saddharth, S., Vijay, G. and Ashok, K. (2010). MATLAB for signal processing on multiprocessors and multicores. *IEEE Signal Processing Magazine*, *27*(2), pp. 40-49

Sanjay, S., Shiru, S. and Neeraj, S. (2014), Image registration techniques using parallel computing in multicore environment and its applications in medical imaging: an overview. *Proceedings of the 5th International Conference on Computer and Communication Technology (ICCCT)* (pp.97-104). IEEE

Sanjay, S., Shiru, S. and Neeraj, S. (2015). A significant approach for implementing image processing using MATLAB with java threads and its implementation in segmentation using Otsu's method in multi core environment, *International Journal of Applied Engineering Research*, *10*(17), pp.37651-37657

Shih, F.Y. and Wu, Y.T. (2005). Robust watermarking and compression for medical images based on genetic algorithms. *Journal of Information Sciences*,*175*(3), pp. 200-216.

Smitha, B. and Navas, K.A. (2007). Spatial domain—high capacity data hiding in ROI images. *Proceedings of the Signal Processing, Communications and Networking* (pp. 528-533). IEEE

Tan, C.K., Ng, J.C., Xu, X., Poh, C.L., Guan, Y.L. and Sheah, K. (2011). Security protection of dicom medical images using dual-layer reversible watermarking with tamper detection capability. *Journal of Digital Imaging*, *24*(3), pp. 528–540.

Vicat-Blanc, P., Soudan, S., Guillier, R. and Goglin, B. (2013). *Utilization of on network computing technologies, in computing networks: from cluster to cloud computing*. USA: John Wiley & Sons, Inc.

Voloshynovskiy, S., Pereira, S., Pun, T., Eggers, J.J. and Su, J.K. (2001). Attacks on digital watermarks: classification, estimation based attacks, and benchmarks. *IEEE Communications Magazine*. *39*(8), pp.118-126.

Wang, S.P and Ledley, R.S. (2012). *Computer architecture and security: fundamentals of designing secure computer systems*. USA: John Wiley and Sons.

Wang, X.Y, Feng, D.G., Lai X.J. and Yu, H.B. (2004). Collisions for hash functions MD4, MD5 HAVAL-128 and RIPEMD, *IACR Cryptology ePrint Archive Report*, *2004*, pp. 199.

Wenbo, D., Chueh, L.P. and Yong, L.G. (2012). An improved tamper detection and localization scheme for volumetric DICOM Images. *Journal of Digital Imaging*, *25*(6), pp. 751–763.

Yan, C., Zhang, Y., Dai, F. and Li, L. (2014). Efficient parallel framework for HEVC motion estimation on many-core processors. *IEEE Transactions on Circuits and Systems for Video Technology*, *24*(12), pp. 2077-2089.

Yang, C.W. and Shen, J.J. (2010). Recover the tampered image based on VQ indexing. *Signal Processing*, *90*(1), pp. 331-343.

Zhou, W., Rockwood, T. and Sagetong, P. (2002). Non-repudiation oblivious watermarking schema for secure digital video distribution. *Proceedings of the 2002 IEEE Workshop* (pp. 343-346). IEEE

Zhou, X.Q., Huang, H.K. and Lou, S.L. (2001). Authenticity and integrity of digital mammography images. *IEEE Transactions on Medical Imaging*, *20*(8), pp. 784–791.

APPENDICES A

PUBLICATIONS

The following publications has been made out of this thesis.

Journals

Hui Liang Khor, Siau-Chuin Liew, and Jasni Mohd. Zain (2017). Region of Interest-Based Tamper Detection and Lossless Recovery Watermarking Scheme (ROI-DR) on Ultrasound Medical Images. *Journal of Digital Imaging (2017), pp. 1-22,* DOI: 10.1007/s10278-016-9930-9

Hui Liang Khor, Siau-Chuin Liew, and Jasni Mohd. Zain (2016). Parallel Digital Watermarking Process on Ultrasound Medical Images in Multicores Environment. *International Journal of Biomedical Imaging. vol. 2016, Article ID 9583727*, 14 pages, 2016. doi:10.1155/2016/9583727

Conferences

Hui Liang Khor, Siau-Chuin Liew, and Jasni Mohd. Zain (2015). Algorithm Design of Digital Watermarking Scheme for Multi-frames Medical Images. *Proceeding of 2nd International Conference on Computational Science and Information Management (ICoCSIM 2015).*

Hui Liang Khor, Siau-Chuin Liew, and Jasni Mohd. Zain (2015). A Review on Parallel Medical Image Processing on GPU, *Proceeding of 4th International Conference on Software Engineering and Computer Science (ICSECS 2015)*, pp.45-48. DOI:10.1109/ICSECS.2015.7333121

Hui Liang Khor, Siau-Chuin Liew, and Jasni Mohd. Zain (2014). A Review of Reversible Medical Image Watermarking Scheme with Tamper Localization and Recovery Capability. *Proceeding for International Conference on Computer, Communication and Control Technology,* pp.188-192. DOI: 10.1109/I4CT.2014.6914172

APPENDICES B1

CODE LISTING FOR

ROI-DR WATERMARKING EMBEDDING PROCESS

```
%*****************************************************************
        %   Title       :       ROI-DR Watermarking Embedding Process
        %   Author      :       Khor Hui Liang
        %   Date        :        9 Sept 2015
%*****************************************************************
tic
        %% Read a DICOM file
        im = dicomread ('US_6.dcm');

        %% ROI & RONI property setting for [x,y,w,h]
        L=120;H=40; % set RONI Length and Height
        roi_x=245;roi_y=71;w=160; h=240; % set ROI x,y,w and h
        roni1_x =1; roni1_y=1;roni1_w=L; roni1_h=size(im,1); % set RONI1
        roni2_x=size(im,2)-L+1; roni2_y=1+1; roni2_w=L; roni2_h=size(im,1)-1; % set
RONI2
        roni3_x=L+1; roni3_y=1;roni3_w=size(im,2)-2*L; roni3_h=H; % set RONI 3
        roni4_x=L+1; roni4_y=size(im,1)-H;roni4_w=size(im,2)-2*L; roni4_h=H; % set
RONI 4
        roni5_x=L+1; roni5_y=H+1; roni5_w=L; roni5_h=2*H; %set for RONI 5

        % set ROI and 4 RONI rectangle properties
        roi_rect=[roi_x, roi_y, w-1, h-1];
        roni1_rect=[roni1_x, roni1_y, roni1_w-1, roni1_h-1];
        roni2_rect=[roni2_x, roni2_y, roni2_w-1, roni2_h-1];
        roni3_rect=[roni3_x, roni3_y, roni3_w-1, roni3_h-1];
        roni4_rect=[roni4_x, roni4_y, roni4_w-1, roni4_h-1];
        roni5_rect=[roni5_x, roni5_y, roni5_w-1, roni5_h-1];

        %% crop ROI and RONI images
        roi_region = imcrop(im(:,:,:,1),roi_rect);
        roni1_region = imcrop(im(:,:,:,1),roni1_rect);
        roni2_region = imcrop(im(:,:,:,1),roni2_rect);
        roni3_region = imcrop(im(:,:,:,1),roni3_rect);
        roni4_region = imcrop(im(:,:,:,1),roni4_rect);
        roni5_region = imcrop(im(:,:,:,1),roni5_rect);

        %% convert all RONI regions to binary format
        roni1_region_bin = dec2bin(roni1_region,8);
        roni2_region_bin = dec2bin(roni2_region,8);
        roni3_region_bin = dec2bin(roni3_region,8);
        roni4_region_bin = dec2bin(roni4_region,8);
        roni5_region_bin = dec2bin(roni5_region,8);

        %% hash ROI_region with SHA-256
```

```
roi_hash_hex = hash(roi_region, 'SHA-256');
roi_hash_bin = hex2bin(roi_hash_hex);

%% compress ROI region to jpg format
imwrite(roi_region,'roi.jpg','jpg','Mode','lossless');
% retrieve from file
fid=fopen('roi.jpg');
compressed_roi=fread(fid);
% convert compressed ROI to binary format
roi_bin = dec2bin(compressed_roi,8);

%% split ROI_binary into two section
compressed_roi_size = length(compressed_roi);
compressed_roi_bin = dec2bin(compressed_roi,8);
m=floor(compressed_roi_size/2);
roi_bin_a = roi_bin(1:m,:);
roi_bin_b = roi_bin(m+1:end,:);
compressed_roi_size_bin = dec2bin(compressed_roi_size,16);

%% hash compressed_ROI with SHA-256
compressed_roi_hash_hex = hash(compressed_roi, 'SHA-256');
compressed_roi_hash_bin = hex2bin(compressed_roi_hash_hex);

%% store 1 ROI pixel and ROI hash values into 4 LSB of RONI pixels in binary
[roni_bin_1, flag1]=insert_one2fourpixels(roni1_region_bin, roi_bin_a);
[roni_bin_2, flag2]=insert_one2fourpixels(roni2_region_bin, roi_bin_b);
[roni_bin_3, flag3]=insert_hash2RONI(roni3_region_bin,roi_hash_bin);
[roni_bin_4,
flag4]=insert_hash2RONI(roni4_region_bin,compressed_roi_hash_bin);
[roni_bin_5,                    flag5]=insert_hash2RONI(roni5_region_bin,
compressed_roi_size_bin);

%% Replace RONIs back to image and write to DICOM file
% check for the sufficiency of RONIs storage space
if (flag1 == 1 || flag2 ==1 || flag3 == 1 || flag4 == 1|| flag5 == 1)
   disp('**** Insufficient RONI pixels space!!! ***');
else
   %convert RONI_binary back to decimal in [size x 1] matrix
   c1=bin2dec(roni_bin_1);
   c2=bin2dec(roni_bin_2);
   c3=bin2dec(roni_bin_3);
   c4=bin2dec(roni_bin_4);
   c5=bin2dec(roni_bin_5);

   %reshape c back to original matrix size
   roni1_double = reshape(c1,size(roni1_region,1), size(roni1_region,2));
   roni2_double = reshape(c2,size(roni2_region,1), size(roni2_region,2));
   roni3_double = reshape(c3,size(roni3_region,1), size(roni3_region,2));
   roni4_double = reshape(c4,size(roni4_region,1), size(roni4_region,2));
   roni5_double = reshape(c5,size(roni5_region,1), size(roni5_region,2));
```

```
    % convert double to uint8 data type
    roni1_uint = uint8(roni1_double);
    roni2_uint = uint8(roni2_double);
    roni3_uint = uint8(roni3_double);
    roni4_uint = uint8(roni4_double);
    roni5_uint = uint8(roni5_double);

    %replace all roni back to RONI regions respectvely in image
    im(roni1_y:roni1_y+roni1_h-1,roni1_x:roni1_x+roni1_w-1) = roni1_uint;
    im(roni2_y:roni2_y+roni2_h-1,roni2_x:roni2_x+roni2_w-1) = roni2_uint;
    im(roni3_y:roni3_y+roni3_h-1,roni3_x:roni3_x+roni3_w-1) = roni3_uint;
    im(roni4_y:roni4_y+roni4_h-1,roni4_x:roni4_x+roni4_w-1) = roni4_uint;
    im(roni5_y:roni5_y+roni5_h-1,roni5_x:roni5_x+roni5_w-1) = roni5_uint;

    %write into dicom file
    dicomwrite(im(:,:,:,:),'watermarked_US_6.dcm');
end

%% Measure Elapsed time and PSNR values
toc;
im1=dicomread('US_6.dcm');
im2=dicomread('watermarked_US_6.dcm');
disp(psnr(im1,im2));
disp(psnr(im1(:,:,:,1),im2(:,:,:,1)));


%*****************************************************************
% Function Name   : hash
% Description     : Convert an input variable into a message digest using
%                   any of several common hash algorithms
% Author          : Michael Kleder
% Source          : http://www.mathworks.com/matlabcentral/fileexchange
%                   /8944-compute-hash-usingmd2-% md5-sha-1-sha-256-
%                   sha-384-or-sha-512
%*****************************************************************

function h = hash(inp,meth)
inp=inp(:);
% convert strings and logicals into uint8 format
if ischar(inp) || islogical(inp)
    inp=uint8(inp);
else % convert everything else into uint8 format without loss of data
    inp=typecast(inp,'uint8');
end

% verify hash method, with some syntactical forgiveness:
meth=upper(meth);
switch meth
```

```
    case 'SHA1'
        meth='SHA-1';
    case 'SHA256'
        meth='SHA-256';
    case 'SHA384'
        meth='SHA-384';
    case 'SHA512'
        meth='SHA-512';
    otherwise
end
algs={'MD2','MD5','SHA-1','SHA-256','SHA-384','SHA-512'};
if isempty(strmatch(meth,algs,'exact'))
    error(['Hash algorithm must be ' ...
        'MD2, MD5, SHA-1, SHA-256, SHA-384, or SHA-512']);
end

% create hash
x=java.security.MessageDigest.getInstance(meth);
x.update(inp);
h=typecast(x.digest,'uint8');
h=dec2hex(h)';
if(size(h,1))==1 % remote possibility: all hash bytes < 128, so pad:
    h=[repmat('0',[1 size(h,2)]);h];
end
h=lower(h(:)');
clear x
return


%******************************************************************
% Function Name     : hex2bin
% Description       : Convert Hexadecimal to binary format
% Author            : Khor Hui Liang
% Date              : 9 Sept 2015
%
******************************************************************
function [binary] = hex2bin(x)
    decimal = hex2dec(x);
    binary = dec2bin(decimal,length(x)*4);
 end


%*********************************************************************
% Function Name     : insert_one2fourpixels
% Description       : Insert binary values (which one pixel <= 256) into RONI
% Author            : Khor Hui Liang
% Date              : 9 Sept 2015

% *******************************************************************
function [roni_bin, flag] = insert_one2fourpixels(roni_bin, roi_bin)
```

```
flag=0;
if (size(roni_bin,1) >= 4*size(roi_bin,1))
   n=1;
   for i=1:size(roi_bin,1)
      for b=1:+2:8
         roni_bin(n,7:8)=roi_bin(i, b:b+1);
         n=n+1;
      end
   end
else
   flag=1;
end
end


% **********************************************************************
   % Function Name    : insert_hash2RONI
   % Description      : insert binary values (which one pixel > 256) into RONI
   % Author           : Khor Hui Liang
   % Date             : 9 Sept 2015

%***********************************************************************
function [roni_bin, flag] = insert_hash2RONI(roni_bin, h_bin)
flag = 0;
if (length(roni_bin) >= size(h_bin,2)/2)
    n=1;
    for b=1:+2:size(h_bin,2)
         roni_bin(n,7:8)=h_bin(1, b:b+1);
         n=n+1;
    end
else
   flag = 1;
end
end
```

APPENDICES B2

CODE LISTING FOR

ROI-DR WATERMARKING AUTHENTICATION PROCESS

```
%*****************************************************************
    %  Title     :    ROI-DR Watermarking Authentication Process
    %  Author    :    Khor Hui Liang
    %  Date      :    9 Sept 2015
%*****************************************************************
    tic
    %% Read a DICOM file
    im=imread('tampered_RONI_5_and_ROI.bmp');

    %% ROI & RONI property setting for [x,y,w,h]
    L=120;H=40; % set RONI Length and Height
    roi_x=245;roi_y=71;w=160; h=240; % set ROI x,y,w and h
    roni1_x =1; roni1_y=1;roni1_w=L; roni1_h=size(im,1); % set RONI1
    roni2_x=size(im,2)-L+1; roni2_y=1+1; roni2_w=L; roni2_h=size(im,1)-1; % set
RONI2
    roni3_x=L+1; roni3_y=1;roni3_w=size(im,2)-2*L; roni3_h=H; % set RONI 3
    roni4_x=L+1; roni4_y=size(im,1)-H;roni4_w=size(im,2)-2*L; roni4_h=H; % set
RONI 4
    roni5_x=L+1; roni5_y=H+1; roni5_w=L; roni5_h=2*H; %set for RONI

    % set ROI and 4 RONI rectangle properties
    roi_rect=[roi_x, roi_y, w-1, h-1];
    roni1_rect=[roni1_x, roni1_y, roni1_w-1, roni1_h-1];
    roni2_rect=[roni2_x, roni2_y, roni2_w-1, roni2_h-1];
    roni3_rect=[roni3_x, roni3_y, roni3_w-1, roni3_h-1];
    roni4_rect=[roni4_x, roni4_y, roni4_w-1, roni4_h-1];
    roni5_rect=[roni5_x, roni5_y, roni5_w-1, roni5_h-1];

    %% Crop ROI and RONI images
    roi_region = imcrop(im(:,:,:,1),roi_rect);
    roni1_region = imcrop(im(:,:,:,1),roni1_rect);
    roni2_region = imcrop(im(:,:,:,1),roni2_rect);
    roni3_region = imcrop(im(:,:,:,1),roni3_rect);
    roni4_region = imcrop(im(:,:,:,1),roni4_rect);
    roni5_region = imcrop(im(:,:,:,1),roni5_rect);

    %% Convert all RONI regions to binary format
    roni1_region_bin = dec2bin(roni1_region,8);
    roni2_region_bin = dec2bin(roni2_region,8);
    roni3_region_bin = dec2bin(roni3_region,8);
    roni4_region_bin = dec2bin(roni4_region,8);
    roni5_region_bin = dec2bin(roni5_region,8);

    %% Hash ROI_region with SHA-256
    roi_hash_hex = hash(roi_region, 'SHA-256');
```

```
roi_hash_bin = hex2bin(roi_hash_hex);
 %% Retrieve ROI_hash_bin from RONI_3 region
roi_hash_bin_retrieved = retrieve_bits_from_RONI(roni3_region_bin, 256, 'h');
 %% Compare ROI hash bin values from both calcualted and retrieved
if isequal(roi_hash_bin,roi_hash_bin_retrieved) == 1
    disp('*** ROI Hash is Equal, Therefore NO Tampering occured!! ***');
else
    disp('*** ROI hash is NOT Equal!! Now check on whether the tamper is occur on RONI or ROI... ***');

    % compress ROI region to jpg format
    imwrite(roi_region,'roi.jpg','jpg','Mode','lossless');
    % retrieve from file
    fid=fopen('roi.jpg');
    compressed_roi=fread(fid);
    % convert compressed ROI to binary format
    roi_bin = dec2bin(compressed_roi,8);
     % hash compressed_ROI with SHA-256
    compressed_roi_hash_hex = hash(compressed_roi, 'SHA-256');
    compressed_roi_hash_bin = hex2bin(compressed_roi_hash_hex);
       % Retrieve compressed_roi_hash_bin from RONI_4 region
    compressed_roi_hash_bin_retrieved = retrieve_bits_from_RONI(roni4_region_bin, 256, 'h');

    %% compare compressed_roi_hash_bin to check RONI 3 is tampered?
    if isequal(compressed_roi_hash_bin,compressed_roi_hash_bin_retrieved) == 1
       disp('*** No Tampering occurred on ROI BUT it occured on RONI_3 instead, in which stored the ROI_hash key!!! ***');
    else
       disp('*** Tampering occurred on ROI !!! Recovery Process start...***');

       % Retrieve compressed_roi_size_bin from RONI 5 region
       compressed_roi_size_bin_retrieved = retrieve_bits_from_RONI(roni5_region_bin, 16, 'h');

compressed_roi_size_retrieved=bin2dec(compressed_roi_size_bin_retrieved);

       % Reassemble compressed_ROI_binary sections into one
       m=floor(compressed_roi_size_retrieved/2);
       roi_bin_1 = retrieve_bits_from_RONI(roni1_region_bin, m, 'r');
       roi_bin_2 = retrieve_bits_from_RONI(roni2_region_bin, compressed_roi_size_retrieved-m, 'r');
       compressed_roi_bin_retrieved = vertcat(roi_bin_1, roi_bin_2);
       compressed_roi_2 = bin2dec(compressed_roi_bin_retrieved);
       % Convert compressed_roi_hash from hexa to binary format
       compressed_roi_hash_hex_2 = hash(compressed_roi_2, 'SHA-256');
       compressed_roi_hash_bin_2 = hex2bin(compressed_roi_hash_hex_2);

       %% to check whether RONIs that stored ROI is tampered?
```

```
                if isequal(compressed_roi_hash_bin_2,compressed_roi_hash_bin_retrieved)
== 1 %roni not tampered
                    % decompress ROI_jpg and restore to its original state
                     fid=fopen('temp2.jpg','w');
                     fwrite(fid, compressed_roi_2);
                     restored_double=imread('temp2.jpg','jpg');
                     restored_uint = uint8(restored_double);
                     roi_recovered=reshape(restored_uint, size(roi_region));
                     roi_recovered_uint=uint8(roi_recovered);

                     % replace the restored image to ROI
                     im(roi_y:roi_y+h-1,roi_x:roi_x+w-1,1,1) = roi_recovered_uint;
                      % write into a dicom file
                     dicomwrite(im, 'recovered_tampered_RONI_5_and_ROI.dcm');
                     disp('*** File Recovered Successfully!! ***');
                else
                     disp('ROI recovery failed because tampering occured on RONI regions
that contains ROI information!!');
                end
            end
        end
        toc;

%*************************************************************************
        %   Function Name    :      Retrieve bits from RONI region
        %   Author           :      Khor Hui Liang
        %   Date             :      9 Sept 2015
%*************************************************************************
        function [roi_bin] = retrieve_bits_from_RONI(roni_region_bin, roi_bin_size, tag)
          n=1;
        if isequal(tag,'r') % 'r' is process pixel value <= 256, which is picture bits such as
ROI
            roi=zeros(roi_bin_size,1);
            roi_bin = dec2bin(roi,8);
              for i=1:roi_bin_size
                for b=1:+2:8
                    roi_bin(i, b:b+1) = roni_region_bin(n,7:8);
                    n=n+1;
                end
              end
        else % process pixel value is > 256, which is calculated values such as hash values
              for b=1:+2:roi_bin_size
                  roi_bin(1, b:b+1) = roni_region_bin(n,7:8);
                  n=n+1;
             end
        end
        end
```

APPENDICES B3

CODE LISTING FOR

SEQUENTIAL ROI-DR WATERMARKING EMBEDDING PROCESS

```
%*******************************************************************
    % Title    :    Sequential ROI-DR Watermarking Embedding Process
    % Author   :    Khor Hui Liang
    % Date     :    9 Sept 2015
%*******************************************************************
    %% Read a DICOM file and stored into |frames|
    % file_name='dcm3.dcm';
    %file_name='3EAF6500';
    file_name ='D:\MatLab\ Raw US Images\Ultrasound_Sample_8.dcm';
    file=dicomread(file_name);
    info=dicominfo(file_name);
    frame_size = info.NumberOfFrames;
    A=[];
    psnr_array=[];
    frame_no_array=[];
     tic;
    %% Sequnetial TALLOR Watermarking Embedding Process on Multiframes
Medical Images
    for n=1:frame_size
        im=file(:,:,:,n);
        %% ROI & RONI property setting for [x,y,w,h]
        L=120;H=40; % set RONI Length and Height
        roi_x=245;roi_y=71;w=160; h=240; % set ROI x,y,w and h
        roni1_x =1; roni1_y=1;roni1_w=L; roni1_h=size(im,1); % set RONI1
        roni2_x=size(im,2)-L+1; roni2_y=1+1; roni2_w=L; roni2_h=size(im,1)-
    1; % set RONI2
        roni3_x=L+1;  roni3_y=1;roni3_w=size(im,2)-2*L;  roni3_h=H;  %  set
    RONI 3
        roni4_x=L+1;        roni4_y=size(im,1)-H;roni4_w=size(im,2)-2*L;
    roni4_h=H; % set RONI 4
        roni5_x=L+1; roni5_y=H+1; roni5_w=L; roni5_h=2*H; %set for RONI 5

        % set ROI and 4 RONI rectangle properties
        roi_rect=[roi_x, roi_y, w-1, h-1];
        roni1_rect=[roni1_x, roni1_y, roni1_w-1, roni1_h-1];
        roni2_rect=[roni2_x, roni2_y, roni2_w-1, roni2_h-1];
        roni3_rect=[roni3_x, roni3_y, roni3_w-1, roni3_h-1];
        roni4_rect=[roni4_x, roni4_y, roni4_w-1, roni4_h-1];
        roni5_rect=[roni5_x, roni5_y, roni5_w-1, roni5_h-1];

        %% crop ROI and RONI images
        roi_region = imcrop(im(:,:,:,1),roi_rect);
        roni1_region = imcrop(im(:,:,:,1),roni1_rect);
        roni2_region = imcrop(im(:,:,:,1),roni2_rect);
```

```
roni3_region = imcrop(im(:,:,:,1),roni3_rect);
roni4_region = imcrop(im(:,:,:,1),roni4_rect);
roni5_region = imcrop(im(:,:,:,1),roni5_rect);

%% convert all RONI regions to binary format
roni1_region_bin = dec2bin(roni1_region,8);
roni2_region_bin = dec2bin(roni2_region,8);
roni3_region_bin = dec2bin(roni3_region,8);
roni4_region_bin = dec2bin(roni4_region,8);
roni5_region_bin = dec2bin(roni5_region,8);

%% hash ROI_region with SHA-256
roi_hash_hex = hash(roi_region, 'SHA-256');
roi_hash_bin = hex2bin(roi_hash_hex);

%% compress ROI region to jpg format
imwrite(roi_region,'roi.jpg','jpg','Mode','lossless');
% retrieve from file
fid=fopen('roi.jpg');
compressed_roi=fread(fid);
% convert compressed ROI to binary format
roi_bin = dec2bin(compressed_roi,8);

%% split ROI_binary into two section
compressed_roi_size = length(compressed_roi);
compressed_roi_bin = dec2bin(compressed_roi,8);
m=floor(compressed_roi_size/2);
roi_bin_a = roi_bin(1:m,:);
roi_bin_b = roi_bin(m+1:end,:);
compressed_roi_size_bin = dec2bin(compressed_roi_size,16);

%% hash compressed_ROI with SHA-256
compressed_roi_hash_hex = hash(compressed_roi, 'SHA-256');
compressed_roi_hash_bin = hex2bin(compressed_roi_hash_hex);

%% store 1 ROI pixel and ROI hash values into 4 LSB of RONI pixels in
binary
[roni_bin_1, flag1]=insert_one2fourpixels(roni1_region_bin, roi_bin_a);
[roni_bin_2, flag2]=insert_one2fourpixels(roni2_region_bin, roi_bin_b);
[roni_bin_3, flag3]=insert_hash2RONI(roni3_region_bin,roi_hash_bin);
[roni_bin_4,
flag4]=insert_hash2RONI(roni4_region_bin,compressed_roi_hash_bin);
[roni_bin_5,                    flag5]=insert_hash2RONI(roni5_region_bin,
compressed_roi_size_bin);

%% Replace RONIs back to image and write to DICOM file
% check for the sufficiency of RONIs storage space
if (flag1 == 1 || flag2 ==1 || flag3 == 1 || flag4 == 1|| flag5 == 1)
   disp('**** Insufficient RONI pixels space!!! ***');
else
```

```
            %convert RONI_binary back to decimal in [size x 1] matrix
            c1=bin2dec(roni_bin_1);
            c2=bin2dec(roni_bin_2);
            c3=bin2dec(roni_bin_3);
            c4=bin2dec(roni_bin_4);
            c5=bin2dec(roni_bin_5);

            %reshape c back to original matrix size
            roni1_double = reshape(c1,size(roni1_region,1), size(roni1_region,2));
            roni2_double = reshape(c2,size(roni2_region,1), size(roni2_region,2));
            roni3_double = reshape(c3,size(roni3_region,1), size(roni3_region,2));
            roni4_double = reshape(c4,size(roni4_region,1), size(roni4_region,2));
            roni5_double = reshape(c5,size(roni5_region,1), size(roni5_region,2));

            % convert double to uint8 data type
            roni1_uint = uint8(roni1_double);
            roni2_uint = uint8(roni2_double);
            roni3_uint = uint8(roni3_double);
            roni4_uint = uint8(roni4_double);
            roni5_uint = uint8(roni5_double);

            %replace all roni back to RONI regions respectvely in image
            im(roni1_y:roni1_y+roni1_h-1,roni1_x:roni1_x+roni1_w-1)          =
roni1_uint;
            im(roni2_y:roni2_y+roni2_h-1,roni2_x:roni2_x+roni2_w-1)          =
roni2_uint;
            im(roni3_y:roni3_y+roni3_h-1,roni3_x:roni3_x+roni3_w-1)          =
roni3_uint;
            im(roni4_y:roni4_y+roni4_h-1,roni4_x:roni4_x+roni4_w-1)          =
roni4_uint;
            im(roni5_y:roni5_y+roni5_h-1,roni5_x:roni5_x+roni5_w-1)          =
roni5_uint;

            A=cat(4,A, im);
         end

         %% Measure MSE and PSNR values
         [psnr_value, mse_value]=psnr(im,file(:,:,:,n));
          psnr_array=[psnr_array [psnr_value; mse_value]];
          frame_no_array=[frame_no_array n];

end
%% Measure and display the Elapsed time
toc;

%% Display Table of frames PSNR values
c=cat(1, frame_no_array, psnr_array);
d=transpose(c);
psnr_Table = array2Table(d,...
    'VariableNames',{'Frame_No' 'PSNR' 'MSE'});
```

```
display(psnr_Table);
 %% Stored A into a DICOM format
dicomwrite(A,'D:\MatLab\my watermarking\US used for testing multiframe\Watermarked US Images\Sequential Watermarked US Images\Watermarked_Seq_Ultrasound_Sample_8.dcm');
display('Complete');
```

APPENDICES B4

CODE LISTING FOR

SEQUENTIAL ROI-DR WATERMARKING AUTHENTICATION PROCESS

```
%***************************************************************
	% Title		:	Sequential ROI-DR Watermarking Authentication Process
	% Author	:	Khor Hui Liang
	% Date		:	 9 Sept 2015
%***************************************************************
	A=[];
	result='Tampered happen at: \n';
	file_name
='D:\MatLab\Watermarked_Seq_Ultrasound_Sample_4_B_tampered_79.tif';
	frame_size=length(imfinfo(file_name));
	count=0;
	tic;
	for frm =1:frame_size

fprintf('===========================================================\n');
	 fprintf('		Scan for Frame %d \n',frm);

fprintf('===========================================================\n\n'
);


	% Read a DICOM file
	im=imread(file_name,frm);

	% ROI & RONI property setting for [x,y,w,h]
	L=120;H=40; % set RONI Length and Height
	roi_x=245;roi_y=71;w=160; h=240; % set ROI x,y,w and h
	roni1_x =1; roni1_y=1;roni1_w=L; roni1_h=size(im,1); % set RONI1
	roni2_x=size(im,2)-L+1; roni2_y=1+1; roni2_w=L; roni2_h=size(im,1)-1; % set
RONI2
	roni3_x=L+1; roni3_y=1;roni3_w=size(im,2)-2*L; roni3_h=H; % set RONI 3
	roni4_x=L+1; roni4_y=size(im,1)-H;roni4_w=size(im,2)-2*L; roni4_h=H; % set
RONI 4
	roni5_x=L+1; roni5_y=H+1; roni5_w=L; roni5_h=2*H; %set for RONI

	% set ROI and 4 RONI rectangle properties
	roi_rect=[roi_x, roi_y, w-1, h-1];
	roni1_rect=[roni1_x, roni1_y, roni1_w-1, roni1_h-1];
	roni2_rect=[roni2_x, roni2_y, roni2_w-1, roni2_h-1];
	roni3_rect=[roni3_x, roni3_y, roni3_w-1, roni3_h-1];
	roni4_rect=[roni4_x, roni4_y, roni4_w-1, roni4_h-1];
	roni5_rect=[roni5_x, roni5_y, roni5_w-1, roni5_h-1];
```

```
% Crop ROI and RONI images
roi_region = imcrop(im(:,:,:,1),roi_rect);
roni1_region = imcrop(im(:,:,:,1),roni1_rect);
roni2_region = imcrop(im(:,:,:,1),roni2_rect);
roni3_region = imcrop(im(:,:,:,1),roni3_rect);
roni4_region = imcrop(im(:,:,:,1),roni4_rect);
roni5_region = imcrop(im(:,:,:,1),roni5_rect);

% Convert all RONI regions to binary format
roni1_region_bin = dec2bin(roni1_region,8);
roni2_region_bin = dec2bin(roni2_region,8);
roni3_region_bin = dec2bin(roni3_region,8);
roni4_region_bin = dec2bin(roni4_region,8);
roni5_region_bin = dec2bin(roni5_region,8);

% Hash ROI_region with SHA-256
roi_hash_hex = hash(roi_region, 'SHA-256');
roi_hash_bin = hex2bin(roi_hash_hex);

% Retrieve ROI_hash_bin from RONI_3 region
roi_hash_bin_retrieved = retrieve_bits_from_RONI(roni3_region_bin, 256, 'h');

% Compare ROI hash bin values from both calcualted and retrieved
if isequal(roi_hash_bin,roi_hash_bin_retrieved) == 1
    disp('*** ROI Hash is Equal, Therefore NO Tampering occured!! ***');
else
    disp('*** ROI hash is NOT Equal!! Now check on whether the tamper is occur on RONI or ROI... ***');

    % compress ROI region to jpg format
    imwrite(roi_region,'roi.jpg','jpg','Mode','lossless');
    % retrieve from file
    fid=fopen('roi.jpg');
    compressed_roi=fread(fid);
    % convert compressed ROI to binary format
    roi_bin = dec2bin(compressed_roi,8);

    % hash compressed_ROI with SHA-256
    compressed_roi_hash_hex = hash(compressed_roi, 'SHA-256');
    compressed_roi_hash_bin = hex2bin(compressed_roi_hash_hex);

    % Retrieve compressed_roi_hash_bin from RONI_4 region
    compressed_roi_hash_bin_retrieved = retrieve_bits_from_RONI(roni4_region_bin, 256, 'h');

    % compare compressed_roi_hash_bin to check RONI 3 is tampered?
    if isequal(compressed_roi_hash_bin,compressed_roi_hash_bin_retrieved) == 1
```

```
                disp('*** No Tampering occurred on ROI BUT it occured on RONI_3 instead, in which stored the ROI_hash key!!! ***');
            else
                disp('*** Tampering occurred on ROI !!! Recovery Process start...***');

                % Retrieve compressed_roi_size_bin from RONI 5 region
                compressed_roi_size_bin_retrieved = retrieve_bits_from_RONI(roni5_region_bin, 16, 'h');
compressed_roi_size_retrieved=bin2dec(compressed_roi_size_bin_retrieved);

                % Reassemble compressed_ROI_binary sections into one
                m=floor(compressed_roi_size_retrieved/2);
                roi_bin_1 = retrieve_bits_from_RONI(roni1_region_bin, m, 'r');
                roi_bin_2 = retrieve_bits_from_RONI(roni2_region_bin, compressed_roi_size_retrieved-m, 'r');
                compressed_roi_bin_retrieved = vertcat(roi_bin_1, roi_bin_2);
                compressed_roi_2 = bin2dec(compressed_roi_bin_retrieved);

                % Convert compressed_roi_hash from hexa to binary format
                compressed_roi_hash_hex_2 = hash(compressed_roi_2, 'SHA-256');
                compressed_roi_hash_bin_2 = hex2bin(compressed_roi_hash_hex_2);

               % to check whether RONIs that stored ROI is tampered?
                if isequal(compressed_roi_hash_bin_2,compressed_roi_hash_bin_retrieved) == 1 %roni not tampered
                   % decompress ROI_jpg and restore to its original state
                    fid=fopen('temp2.jpg','w');
                    fwrite(fid, compressed_roi_2);
                    restored_double=imread('temp2.jpg','jpg');
                    restored_uint = uint8(restored_double);
                    roi_recovered=reshape(restored_uint, size(roi_region));
                    roi_recovered_uint=uint8(roi_recovered);

                    % replace the restored image to ROI
                    im(roi_y:roi_y+h-1,roi_x:roi_x+w-1,1,1) = roi_recovered_uint;
                 else
                    disp('ROI recovery failed because tampering occured on RONI regions that contains ROI information!!');
                end
            end
           count=count+1;
           str_frm = num2str(frm);
           result=strcat(result,'\t\tFrame #',str_frm,'\n');
        end
        fprintf('*************<< - End of Frame %d  - >>*************\n\n',frm);
        A=cat(4,A, im);
        end
        toc;
        % Stored A into a DICOM format
```

```
dicomwrite(A,'D:\ Tampered US 4\Recovered_Seq_US.dcm');
str_count = num2str(count);
fprintf('Total tampered frames is %s \n',str_count);
fprintf(result);
display('Complete');
```

APPENDICES B5

CODE LISTING FOR

PARALLEL ROI-DR WATERMARKING EMBEDDING PROCESS

```
%*****************************************************************
% Title     :    Parallel Sequential ROI-DR Watermarking Embedding Process
% Author    :    Khor Hui Liang
% Date      :     9 Sept 2015
%*****************************************************************

    %% Load the Settings and the Data
    % Dicomread ultrasound file into frames variable.
    % Identify the frame size and local cluster (microprocessor) used.
    % Determine the number of workers/cores in a cluster and assign it into a
numTasks variable.
    % The numTasks will return 4 since the cluster is using quad core microprocessor.

    file_name = 'dcm3.dcm';
    file=dicomread(file_name);
    info = dicominfo(file_name);
    frame_size = info.NumberOfFrames;
    P=[];
    clust = parcluster('local');
    numTasks = clust.NumWorkers; % We want to split into this many tasks.
    startClock = clock;

    %% Divide the Work into Smaller Tasks
    % The function pctdemo_helper_split_scalar to divide the ultrasound multiframes
among the numTasks tasks.
     [numPerTask, numTasks] = pctdemo_helper_split_scalar(frame_size,
numTasks);
    disp(numPerTask);

    %% Create and Submit the Job
    % Create the watermarking embedding job and the tasks in the job.
    % Task |i| will perform |numPerTask(i)| watermarking embedding process.
    % Notice that the task function |tallor_par_3_2_1| is the same function
    % that used in the sequential version.
    frame_no=1;
    job = createJob(clust);
    for i = 1:numTasks
        createTask(job, @optimized_watermarking_embedding_fxn, 1,
{file,frame_no, numPerTask(i)});
        frame_no=frame_no + numPerTask(i);
    end

    %%
    % We can now submit the job and wait for it to finish.
    submit(job);
```

```
wait(job);

%% Retrieve the Results
% Let us obtain the job results |y|, verify that all the tasks finished successfully,
% and then delete the job.
% fetchOutputs will throw an error if the tasks did not complete successfully,
% in which case we need to delete the job before throwing the error.
try
    y = fetchOutputs(job);
catch err
    delete(job);
    rethrow(err);
end

%%
% Let us format the results.  Concatenate all the cells in y into one column vector,
and store it into P.
disp(y(:,:,:));
P=cat(4,y{:});

%% Store the concatenated result into Dicom file
dicomwrite(P,'parallel_optimized_watermarked_multiframes_images.dcm');
display('Complete');

%% Delete the Job
% We have now finished all the verifications, so we can delete the job.
delete(job);

%% Measure the Elapsed Time
% The time used for the distributed computations should be compared against the
time it takes to perform in sequential version.
% The elapsed time varies with the underlying hardware and network
infrastructure.
elapsedTime = etime(clock, startClock);
fprintf('Elapsed time is %2.1f seconds\n', elapsedTime);
```

```
%*******************************************************************
% Function Name      : ROI-DR Watermarking Embedding Process Function
% Description        : This function is to embed watermarking into ultrasound images
%                      The input argument are ultrasound image frame, its frame
%                      number and the input frame size. It will return watermarked
%                      frames as an
%                      output
%
********************************************************************

     function        [A]=optimized_watermarking_embedding_fxn(file,frame_no,
frame_size)
        A=[];
        for n=frame_no:frame_no+frame_size-1
          im=file(:,:,:,n);
          % ROI & RONI property setting for [x,y,w,h]
          L=120;H=40; % set RONI Length and Height
          roi_x=245;roi_y=71;w=160; h=240; % set ROI x,y,w and h
          roni1_x =1; roni1_y=1;roni1_w=L; roni1_h=size(im,1); % set RONI1
          roni2_x=size(im,2)-L+1; roni2_y=1+1; roni2_w=L; roni2_h=size(im,1)-1; %
set RONI2
          roni3_x=L+1; roni3_y=1;roni3_w=size(im,2)-2*L; roni3_h=H; % set RONI
3
          roni4_x=L+1; roni4_y=size(im,1)-H;roni4_w=size(im,2)-2*L; roni4_h=H; %
set RONI 4
          roni5_x=L+1; roni5_y=H+1; roni5_w=L; roni5_h=2*H; %set for RONI 5

          % set ROI and 4 RONI rectangle properties
          roi_rect=[roi_x, roi_y, w-1, h-1];
          roni1_rect=[roni1_x, roni1_y, roni1_w-1, roni1_h-1];
          roni2_rect=[roni2_x, roni2_y, roni2_w-1, roni2_h-1];
          roni3_rect=[roni3_x, roni3_y, roni3_w-1, roni3_h-1];
          roni4_rect=[roni4_x, roni4_y, roni4_w-1, roni4_h-1];
          roni5_rect=[roni5_x, roni5_y, roni5_w-1, roni5_h-1];

          % crop ROI and RONI images
          roi_region = imcrop(im(:,:,:,1),roi_rect);
          roni1_region = imcrop(im(:,:,:,1),roni1_rect);
          roni2_region = imcrop(im(:,:,:,1),roni2_rect);
          roni3_region = imcrop(im(:,:,:,1),roni3_rect);
          roni4_region = imcrop(im(:,:,:,1),roni4_rect);
          roni5_region = imcrop(im(:,:,:,1),roni5_rect);

          % convert all RONI regions to binary format
          roni1_region_bin = dec2bin(roni1_region,8);
          roni2_region_bin = dec2bin(roni2_region,8);
          roni3_region_bin = dec2bin(roni3_region,8);
          roni4_region_bin = dec2bin(roni4_region,8);
          roni5_region_bin = dec2bin(roni5_region,8);
```

```
% hash ROI_region with SHA-256
roi_hash_hex = hash(roi_region, 'SHA-256');
roi_hash_bin = hex2bin(roi_hash_hex);

% compress ROI region to jpg format
imwrite(roi_region,'roi.jpg','jpg','Mode','lossless');
% retrieve from file
fid=fopen('roi.jpg');
compressed_roi=fread(fid);
% convert compressed ROI to binary format
roi_bin = dec2bin(compressed_roi,8);

%% split ROI_binary into two section
compressed_roi_size = length(compressed_roi);
%compressed_roi_bin = dec2bin(compressed_roi,8);
m=floor(compressed_roi_size/2);
roi_bin_a = roi_bin(1:m,:);
roi_bin_b = roi_bin(m+1:end,:);
compressed_roi_size_bin = dec2bin(compressed_roi_size,16);

%% hash compressed_ROI with SHA-256
compressed_roi_hash_hex = hash(compressed_roi, 'SHA-256');
compressed_roi_hash_bin = hex2bin(compressed_roi_hash_hex);

%% store 1 ROI pixel and ROI hash values into 4 LSB of RONI pixels in
binary
[roni_bin_1, flag1]=insert_one2fourpixels(roni1_region_bin, roi_bin_a);
[roni_bin_2, flag2]=insert_one2fourpixels(roni2_region_bin, roi_bin_b);
[roni_bin_3, flag3]=insert_hash2RONI(roni3_region_bin,roi_hash_bin);
[roni_bin_4,
flag4]=insert_hash2RONI(roni4_region_bin,compressed_roi_hash_bin);
[roni_bin_5,                    flag5]=insert_hash2RONI(roni5_region_bin,
compressed_roi_size_bin);

%% Replace RONIs back to image and write to DICOM file
% check for the sufficiency of RONIs storage space
if (flag1 == 1 || flag2 ==1 || flag3 == 1 || flag4 == 1|| flag5 == 1)
   disp('**** Insufficient RONI pixels space!!! ***');
else
   %convert RONI_binary back to decimal in [size x 1] matrix
   c1=bin2dec(roni_bin_1);
   c2=bin2dec(roni_bin_2);
   c3=bin2dec(roni_bin_3);
   c4=bin2dec(roni_bin_4);
   c5=bin2dec(roni_bin_5);

   %reshape c back to original matrix size
   roni1_double = reshape(c1,size(roni1_region,1), size(roni1_region,2));
   roni2_double = reshape(c2,size(roni2_region,1), size(roni2_region,2));
   roni3_double = reshape(c3,size(roni3_region,1), size(roni3_region,2));
```

```
        roni4_double = reshape(c4,size(roni4_region,1), size(roni4_region,2));
        roni5_double = reshape(c5,size(roni5_region,1), size(roni5_region,2));

        % convert double to uint8 data type
        roni1_uint = uint8(roni1_double);
        roni2_uint = uint8(roni2_double);
        roni3_uint = uint8(roni3_double);
        roni4_uint = uint8(roni4_double);
        roni5_uint = uint8(roni5_double);

        %replace all roni back to RONI regions respectvely in image
        im(roni1_y:roni1_y+roni1_h-1,roni1_x:roni1_x+roni1_w-1) = roni1_uint;
        im(roni2_y:roni2_y+roni2_h-1,roni2_x:roni2_x+roni2_w-1) = roni2_uint;
        im(roni3_y:roni3_y+roni3_h-1,roni3_x:roni3_x+roni3_w-1) = roni3_uint;
        im(roni4_y:roni4_y+roni4_h-1,roni4_x:roni4_x+roni4_w-1) = roni4_uint;
        im(roni5_y:roni5_y+roni5_h-1,roni5_x:roni5_x+roni5_w-1) = roni5_uint;

        A=cat(4,A, im);
      end

    end
end
```

APPENDICES B6

CODE LISTING FOR

PARALLEL ROI-DR WATERMARKING AUTHENTICATION PROCESS

```
%% Load the Settings and the Data
% Dicomread ultrasound file into frames variable.
% Identify the frame size and local cluster (microprocessor) used.
% Determine the number of workers/cores in a cluster and assign it into a numTasks variable.
% The numTasks will return 4 since the cluster is using quad core microprocessor.

file_name ='D:\MatLab\my watermarking\US used for testing multiframe\Watermarked US Images\Sequential Watermarked US Images\Tampered Watermarked Seq US Images\Tampered US 4\Watermarked_Seq_Ultrasound_Sample_4_B_tampered_79.tif';
frame_size=length(imfinfo(file_name));
P = [];
clust = parcluster('local');
numTasks = clust.NumWorkers; % We want to split into this many tasks.
startClock = clock;

%% Divide the Work into Smaller Tasks
% The function pctdemo_helper_split_scalar to divide the ultrasound multiframes among the numTasks tasks.
% For example, 15 frames ultrasound images will be divided to 4,4,4,3 frames to four tasks respectively,
% which is display in numPerTask. This means four tasks will perform the watermarking authentication process simultaneously.
[numPerTask, numTasks] = pctdemo_helper_split_scalar(frame_size, numTasks);
disp(numPerTask);

%% Create and Submit the Job
% Create the watermarking embedding job and the tasks in the job.
% Task |i| will perform |numPerTask(i)| watermarking embedding process.
% Notice that the task function |tallor_par_3_2_1| is the same function
% that used in the sequential version.
frame_no=1;
job = createJob(clust);
for i = 1:numTasks
    createTask(job, @optimized_watermarking_authentication_fxn, 2, {file_name,frame_no, numPerTask(i)});
    frame_no=frame_no + numPerTask(i);
end

%%
% We can now submit the job and wait for it to finish.
submit(job);
```

```
wait(job);

%% Retrieve the Results
% Let us obtain the job results |y|, verify that all the tasks finished successfully,
% and then delete the job.
% fetchOutputs will throw an error if the tasks did not complete successfully,
% in which case we need to delete the job before throwing the error.
try
    y = fetchOutputs(job);
catch err
    delete(job);
    rethrow(err);
end

%%
% Let us format the results.  Concatenate all the cells in y into one column vector, and store it into P.
disp(y(:,:,:));
P=cat(4,y{:,1});
Result=[y{:,2}];
fprintf('Tampered happen at: \n %s\n\n', Result);
%% Store the concatenated result into Dicom file
%dicomwrite(P,'D:\MatLab\my watermarking\US used for testing multiframe\Watermarked US Images\Parallel Watermarked US Images\Recovered Parallel US Images\Recovered_Parallel_US_8.dcm');
dicomwrite(P,'D:\MatLab\my watermarking\US used for testing multiframe\Watermarked US Images\Sequential Watermarked US Images\Tampered Watermarked Seq US Images\Tampered US 4\Recovered_Par_US.dcm');
display('Complete');

%% Delete the Job
% We have now finished all the verifications, so we can delete the job.
delete(job);

%% Measure the Elapsed Time
% The time used for the distributed computations should be compared against the time it takes to perform in sequential version.
% The elapsed time varies with the underlying hardware and network infrastructure.
elapsedTime = etime(clock, startClock);
fprintf('Elapsed time is %2.1f seconds\n', elapsedTime);
```

```
%*******************************************************************
        % Function Name      : ROI-DR Watermarking Authentication Function
        % Description        : This function is to embed watermarking into ultrasound
                                images
        %                      The input argument are ultrasound image frame, its frame
                               number
        %                      and the input frame size. It will return recovered frames
                               and string
        %                       result of tampered frame as output

% *******************************************************************
        function [A, result] =
optimized_watermarking_authentication_fxn(file_name,frame_no, frame_size)
        A=[];
        result=[];
        for frm=frame_no:frame_no+frame_size-1
            % Read a DICOM file
            im=imread(file_name,frm);

            % ROI & RONI property setting for [x,y,w,h]
            L=120;H=40; % set RONI Length and Height
            roi_x=245;roi_y=71;w=160; h=240; % set ROI x,y,w and h

            % setting for RONI1 to RONI5
            roni1_x =1; roni1_y=1;roni1_w=L; roni1_h=size(im,1);
            roni2_x=size(im,2)-L+1; roni2_y=1+1; roni2_w=L; roni2_h=size(im,1)-1;
            roni3_x=L+1; roni3_y=1;roni3_w=size(im,2)-2*L; roni3_h=H;
            roni4_x=L+1; roni4_y=size(im,1)-H;roni4_w=size(im,2)-2*L; roni4_h=H;
            roni5_x=L+1; roni5_y=H+1; roni5_w=L; roni5_h=2*H; %set for RONI

            % set ROI and 4 RONI rectangle properties
            roi_rect=[roi_x, roi_y, w-1, h-1];
            roni1_rect=[roni1_x, roni1_y, roni1_w-1, roni1_h-1];
            roni2_rect=[roni2_x, roni2_y, roni2_w-1, roni2_h-1];
            roni3_rect=[roni3_x, roni3_y, roni3_w-1, roni3_h-1];
            roni4_rect=[roni4_x, roni4_y, roni4_w-1, roni4_h-1];
            roni5_rect=[roni5_x, roni5_y, roni5_w-1, roni5_h-1];

            % Crop ROI and RONI images
            roi_region = imcrop(im(:,:,:,1),roi_rect);
            roni1_region = imcrop(im(:,:,:,1),roni1_rect);
            roni2_region = imcrop(im(:,:,:,1),roni2_rect);
            roni3_region = imcrop(im(:,:,:,1),roni3_rect);
            roni4_region = imcrop(im(:,:,:,1),roni4_rect);
            roni5_region = imcrop(im(:,:,:,1),roni5_rect);

            % Convert all RONI regions to binary format
            roni1_region_bin = dec2bin(roni1_region,8);
            roni2_region_bin = dec2bin(roni2_region,8);
            roni3_region_bin = dec2bin(roni3_region,8);
```

```
roni4_region_bin = dec2bin(roni4_region,8);
roni5_region_bin = dec2bin(roni5_region,8);

% Hash ROI_region with SHA-256
roi_hash_hex = hash(roi_region, 'SHA-256');
roi_hash_bin = hex2bin(roi_hash_hex);

% Retrieve ROI_hash_bin from RONI_3 region
roi_hash_bin_retrieved = retrieve_bits_from_RONI(roni3_region_bin, 256,
'h');

% Compare ROI hash bin values from both calcualted and retrieved
if isequal(roi_hash_bin,roi_hash_bin_retrieved) == 1
   disp('*** ROI Hash is Equal, Therefore NO Tampering occured!! ***');
else
    disp('*** ROI hash is NOT Equal!! Now check on whether the tamper is
occur on RONI or ROI... ***');

    % compress ROI region to jpg format
    imwrite(roi_region,'roi.jpg','jpg','Mode','lossless');
    % retrieve from file
    fid=fopen('roi.jpg');
    compressed_roi=fread(fid);
    % convert compressed ROI to binary format
    % = dec2bin(compressed_roi,8);

    % hash compressed_ROI with SHA-256
    compressed_roi_hash_hex = hash(compressed_roi, 'SHA-256');
    compressed_roi_hash_bin = hex2bin(compressed_roi_hash_hex);

    % Retrieve compressed_roi_hash_bin from RONI_4 region
    compressed_roi_hash_bin_retrieved                                =
retrieve_bits_from_RONI(roni4_region_bin, 256, 'h');

    % compare compressed_roi_hash_bin to check RONI 3 is tampered?
    if isequal(compressed_roi_hash_bin,compressed_roi_hash_bin_retrieved)
== 1
       disp('*** No Tampering occurred on ROI BUT it occured on RONI_3
instead, in which stored the ROI_hash key!!! ***');
    else
       disp('*** Tampering occurred on ROI !!! Recovery Process start...***');

       % Retrieve compressed_roi_size_bin from RONI 5 region
       compressed_roi_size_bin_retrieved                             =
retrieve_bits_from_RONI(roni5_region_bin, 16, 'h');

compressed_roi_size_retrieved=bin2dec(compressed_roi_size_bin_retrieved);

       % Reassemble compressed_ROI_binary sections into one
       m=floor(compressed_roi_size_retrieved/2);
```

```
            roi_bin_1 = retrieve_bits_from_RONI(roni1_region_bin, m, 'r');
            roi_bin_2 = retrieve_bits_from_RONI(roni2_region_bin, compressed_roi_size_retrieved-m, 'r');
            compressed_roi_bin_retrieved = vertcat(roi_bin_1, roi_bin_2);
            compressed_roi_2 = bin2dec(compressed_roi_bin_retrieved);

            % Convert compressed_roi_hash from hexa to binary format
            compressed_roi_hash_hex_2 = hash(compressed_roi_2, 'SHA-256');
            compressed_roi_hash_bin_2 = hex2bin(compressed_roi_hash_hex_2);

            % to check whether RONIs that stored ROI is tampered?
            if isequal(compressed_roi_hash_bin_2,compressed_roi_hash_bin_retrieved) == 1 %roni not tampered
              % decompress ROI_jpg and restore to its original state
                fid=fopen('temp2.jpg','w');
                fwrite(fid, compressed_roi_2);
                restored_double=imread('temp2.jpg','jpg');
                restored_uint = uint8(restored_double);
                roi_recovered=reshape(restored_uint, size(roi_region));
                roi_recovered_uint=uint8(roi_recovered);

                % replace the restored image to ROI
                im(roi_y:roi_y+h-1,roi_x:roi_x+w-1,1,1) = roi_recovered_uint;

            else
                disp('ROI recovery failed because tampering occured on RONI regions that contains ROI information!!');
            end


          end
          str_frm = num2str(frm);
          result=strcat(result,'|- Frame #',str_frm,' -| ');
        end

        A=cat(4,A, im);
    end
   end
```