CHAPTER 5

A SECURE FRACTIONAL FOURIER TRANSFORM BASED MEDICAL IMAGE COMPRESSION USING PASS-PARALLEL SPIHT ENCODING SCHEME

5.1. INTRODUCTION

Image compression plays a significant role in Telematics claim particularly in Telemedicine. It is used for transmission of a single or series of images through the computer network between large distances, so as to used for multiple purposes. For example, it is essential for medical images to be transmitted for consistent, better and fast medical diagnosis to be facilitated at many centers. Thus, image compression has become a significant research problem. However, the complexity in image processing applications lie in the fact that, while high compression rates are preferred, the applicability of the reconstructed images is based on some significant characteristics of the original images that are preserved after the compression process has been completed.

In the current situation, besides compression, the security of the medical image is also well thought-out as a significant feature in medical image analysis. The requirement of fast and secure diagnosis is very important in the medical field. In modern years, the transmission of medical data has become a necessary process and it is essential to spot an effectual and secure approach to transmit them over networks. On the other hand, computer networks are composite and it has an obvious risk. Hence, security issues take
place when sending medical image data through the network. As a result, encryption has become a feasible solution to afford security to the medical image data.

With the growth of multimedia applications, security has become a serious problem in communication and storage of images, and encryption has become essential to ensure security. Image encryption approaches endeavor to transform the original image into another image to keep the image secret between the users. Also, it is necessary that no one can identify the content without a key for decryption. Moreover, trusted security in storage and transmission of image data has become an essential aspect in various industries. With the intention of accomplishing such a task, many image encryption methods have been proposed.

The image encryption algorithms can be divided into three main groups, (i) position permutation based algorithm as explained by Jiun-In & Jui-Cheng (2000) (ii) value transformation based algorithm as described in Maniccam & Bourbakis (2001) and (iii) visual transformation based algorithm as discussed by Jiun-In & Jui-Cheng (2000).

Zeghid et al (2000) examined the Advanced Encryption Standard (AES) and in their image encryption technique they adjoined a key stream generator (A5/1, W7) to AES to make certain improvements in the encryption performance.

Mohammad & Aman (2008) presented a block-based transformation algorithm that depended on the grouping of image transformation and a recognized encryption and decryption algorithm known as Blowfish. The original image was separated into blocks, which would be rearranged into a transformed image by means of a Transformation algorithm and the transformed image then encrypted via the Blowfish algorithm. Their results
illustrate that the correlation between the components of image was considerably decreased. The result shows that raising the number of blocks by means of smaller block sizes may result in a lower correlation and higher entropy.

Saroj et al (2008) introduced image encryption system with the Hill cipher. They are used to produce self-invertible matrix for Hill Cipher algorithm. By means of key matrix, they encrypted gray scale on top of the colour images. Their algorithm works better for all types of gray scale in addition to colour images apart from the images with background of similar gray level or same colour.

Bibhudendra et al (2009) has presented an advanced Hill (AdvHill) cipher algorithm which utilizes an involutory key matrix for encryption. They took various images in various fields and encrypted them with original Hill cipher algorithm and the planned AdvHill cipher algorithm. It is obviously noted that the original Hill Cipher cannot appropriately encrypt the images, if the image consists of huge area enclosed with similar colour or gray level.

The previous chapter utilized Improved RSA algorithm for image data encryption. In order to provide efficient compression of the DICOM images with better security and authentication, this chapter uses Quasigroup encryption algorithm and Fractional Fourier Transform based image compression using pass-parallel SPIHT encoding scheme.

5.2. QUASI GROUP ENCRYPTION

The core idea of Quasigroup is based on the Latin square. It consists of an \( n^2 \) set of ordered triples in the form \((r_i, c_j, v_{ij})\); \( r_i, c_j, v_{ij} \in \) Integers, with the additional condition that for each \((r_i, v_{ij})\) and \((c_j, v_{ij})\), \( v_{ij} \) is unique. This relation can be made to correspond to an \( n \times n \) square matrix
with \( r_i \) and \( c_j \) being the row and column indices and \( \nu_{ij} \) is the value in the \( r_i \) th row and \( c_j \) th column. The differentiation among a Quasigroup and a Latin square is based on the operator “\( \cdot \)” on a Quasigroup. This operator is rather simple, in that it performs a table/matrix lookup from the Quasigroup. A Quasigroup equation of the form \( x \cdot y = z \) translates in a straight line to the prearranged triple \((r_x, c_y, \nu_{xy})\), where \( z := \nu_{xy} \). Thus, Quasigroup operation is jointly closed and invertible, making Quasigroup prime candidates for encoding systems.

A multi-level quasigroup accomplishment was proposed by Satti & Kak (2009). They united the execution with indices and nonce’s to look up on the potential of the encryption. Their system also focuses on a stream cipher execution. Quasigroup has also been applied to error correction and in the construction of Message Authentication Codes (MAC).

It is to be observed that the Quasigroup transformation is considered as a replacement or substitution and permutation operation. These operations form the basis of a selection of encryption systems mainly in speech encryption as discussed in Borujeni (2000). In advance, public key systems such as NTRU and elliptic curve cryptosystems contain lower power consumption than RSA. On the other hand, they are computationally more expensive than the secret key systems. Furthermore, Quasigroup does not require any computation to be performed but only needs table look up operations for encryption and decryption.

The output of the proposed encryptor is reliant in the lead of the index numbers and the orders of the matrices \((r, s)\) which are sent by the trusted influence. The encryption is also dependent on six multiplier elements that are generated by a secret algorithm based on the index numbers, the order of the matrices under contemplation and nonce (random number generated by
trusted authority). This key is rationalized by the network on a normal basis (a long time ago in every time interval that is far less that $T$, the time needed to use brute force to decrypt the key that is sent by the trusted authority). Figure 5.1 illustrates the Quasigroup encryptor. The component takes the raw data stream and randomizes it based on the encryption key (the encryption key), and the output data has popular autocorrelation properties.

![Quasi Group Encryptor](image)

**Figure 5.1 Quasi Group Encryptor**

### 5.3. TRANSFORMATION APPROACHES USED IN IMAGE COMPRESSION

The Burrows-Wheeler transform (BWT) is a data compression algorithm, introduced by Burrows and Wheeler. The main goal is to attain enhanced data compression ratio to save storage space and to permit faster data transmission through different networks. BWT based compression is nearer to the paramount known algorithm for text data. It could also be used to improve the compression performance of images.
Wavelet Transform has been a very useful tool for image processing. It permits a function which may be illustrated in terms of a common shape, with a broad range of details. The most important characteristic feature of Haar Transform lies in the actuality that it provides itself easily to simple human calculations. Modified Fast Haar Wavelet Transform (MFHWT), is one of the algorithms which can lessen the computation work in Haar Transform (HT) and Fast Haar Transform (FHT). (Amit 2013) presented the algorithm for image compression by means of MFHWT and provided better results than those obtained by any other method.

Discrete Cosine Transform (DCT) has been very important in the field of image and video compression and it is discrete time version of the Fourier Cosine series as discussed by Andrew Watson (1994). DCT has been associated with Discrete Fourier Transform (DFT) which is a system of conveying a signal into basic frequency segments.

In this approach, Fractional Fourier Transform (FrFT), which is an overview of the ordinary Fourier Transform (FT) is used for transformation studied by Namias (1980). The main characteristic and advantage of fraction Fourier domain image compression is its spare degree of freedom that is required by its fractional orders. FrFT contributes to a number of valuable properties of the normal Fourier transform and has a free parameter ‘a’ which is its fraction (Rajinder et al 2012). Hence, the fractional Fourier domain approach is used in this approach instead of wavelet domain.

To overcome these problems, an improved form of SPIHT algorithm has been used in the present research for better compression efficiency. SPIHT algorithm produces a pyramid structure based on a fractional Fourier decomposition of an image. The original SPIHT algorithm processes the transformed coefficients (Wavelet/Fourier) in a dynamic order that is based on the values of the coefficients. Therefore, it is very tedious to
process multiple coefficients in parallel and thus the throughput of the original SPIHT is degraded. In the present research Block-Based Pass-Parallel SPIHT Algorithm has been used for the compression approach to overcome the drawbacks of the original SPIHT algorithm.

5.4. METHODOLOGY

This approach comprises of the following phases, namely, Encryption, Domain Transformation, Block based Pass-Parallel SPIHT Compression, and Decoding through Inverse Block based Pass-Parallel SPIHT and Inverse Fractional Fourier Transform and finally Quasigroup decryption.

5.4.1. Quasi group Encryption

The original image is encrypted using Quasigroup encryption technique, which has significant data-scrambling properties and hence it has been effectively used in symmetric cryptography. The main aim of the scrambler is to enhance the entropy at the output even in the scenario where the input is constant. The massive complexity associated with the assignment of discovering scrambling transformation assures the effectiveness of the encryption procedure. The Quasigroup encryption is a development that has permutation based scrambling as its basis.
It should be that if \( Q \) is a Quasigroup such that \( a_1, a_2, a_3, \ldots, a_n \) belongs to it, then the encryption operation \( QE \), which is defined over the defined elements, maps those elements to another vector \( b_1, b_2, b_3, \ldots, b_n \) such that the elements of the resultant vector also belongs to the same Quasigroup.

The mathematical equation used for the encryption process (basic level) is defined by,

\[
E_\alpha(a_1, a_2, a_3, \ldots, a_n) = b_1, b_2, b_3, \ldots, b_n
\]

(5.1)

where \( b_1 = a \times a_1, b_i = b_{i-1} \times a_i \), \( i \) increments from two to the number of elements that have to be encrypted and \( a \) is the hidden key.

### 5.4.2. Fractional Fourier Transform

After the Quasigroup encryption, the DICOM image is encrypted and the encrypted image is given as input to the Fractional Fourier transform block.

Discrete Fractional Fourier Transform (DFrFt) is applied to the encrypted image to obtain the transformed coefficients. It is analyzed that by altering the value of fractional order “\( \alpha \)” to different values, the DFrFt can provide significant results in terms of PSNR.

In Medical image processing, compression plays a very important role. This means minimizing the dimensions of the images to a processing level. Image compression using transform coding provides significant results, with fair image quality as discussed by Yetik et al (2001). The cut-off of the transform coefficients can be tuned to bring out a negotiation between image quality and compression factor. To use this approach, an image is initially
partitioned into non-overlapped $n \times n$ (generally taken as 8x8 or 16x16) sub images. A 2D-DFrFT is applied to each block to transform the gray levels of pixels in the spatial domain into coefficients in the frequency domain. The coefficients are normalized by various scales based on the cut-off selected. At the decoder section, the process of encoding is simply reversed.

### 5.4.3. SPIHT

The image is then compressed using the SPIHT algorithm. SPIHT is the image compression technique that was studied by Hualiang et al (2010).

\[
\text{Threshold } T_n = T_{n-1}/2 \\
\text{Reconstructive Value: } R_n = R_{n-1}/2
\]

![Basic Flowchart of SPIHT Algorithm](image)

**Figure 5.3** Basic Flowchart of SPIHT Algorithm

The overall flow of the SPIHT algorithm is shown in Figure 5.3. For a given set $T$, SPIHT defines a function of significance which shows whether the set $T$ has pixels larger than a given threshold. $s_n(T)$, the significance of set $T$ in the $n$th bit-plane is defined as,

\[
s_n(T) = \begin{cases} 
1, & \max_{w(t) \in T} |w(t)| \geq 2^n \\
0, & \text{otherwise} 
\end{cases}
\]  

(5.2)

when $s_n(T)$ is “0,” $T$ is called an Insignificant set; if not, $T$ is referred as a significant set. An insignificant set can be denoted as a single-bit “0,” but a
significant set is partitioned into subsets and its significances are to be tested again. Based on the zero tree hypothesis studied by Sandeep (2011), SPIHT encodes given set T and its descendants denoted by D(T) together by verifying the significance of \( T \cup D(T) \) (the union of T and D(T)) and by denoting \( T \cup D(T) \) as a single symbol “zero” if \( T \cup D(T) \) is insignificant. On the other hand, if \( T \cup D(T) \) is significant, T is partitioned into subsets, each of which is independently tested.

In order to minimize the complexity of SPIHT, the whole depiction is decomposed into 4 \( \times \) 4 sets (sets consisting of 4 \( \times \) 4 pixels), and the significance of the union of each 4 \( \times \) 4 set and its descendants is experimented. The SPIHT algorithm encodes Fractional Fourier coefficients bit-plane by bit-plane from the most significant bit-plane to the least significant bit-plane. A SPIHT algorithm comprises of three passes, namely, Insignificant Set Pass (ISP), Insignificant Pixel Pass (IPP), and Significant Pixel Pass (SPP). Based on the results of the \((n + 1)\)th bit-plane, the \(n\)th bit of pixels are classified and processed by one of the three passes. Insignificant pixels classified by the \((n + 1)\)th bit-plane are encoded by IPP for the \(n\)th bit-plane whereas significant pixels are processed by SPP. The main aim of each pass is the formation of the suitable bitstream based on Fractional Fourier coefficient information. ISP deals with the insignificant sets. If a set in this pass is categorized as a significant set in the \(n\)th bit-plane, it is then decomposed into smaller sets until the smaller sets are insignificant or they correspond to single pixels. If the smaller sets are insignificant, they are handled by ISP. If the smaller sets correspond to single pixels, they are handled by either IPP or SPP based on their significance.

If the most significant bit-plane is a zero bit-plane (that is all coefficients have their most significant bit equal to 0), the bit-plane is not encoded, and as a result, the number of encoded bit-planes is decreased. The following significant bit-planes are not encoded if they are also a zero bit-
plane. Therefore, the SPIHT algorithm starts from the First Non-Zero Bit (FNZB) plane as explained by Said & Pearlman (1996).

The original SPIHT algorithm consists of three linked lists for processing namely ISP, SPP, and IPP, respectively. In each pass, the entries in the linked list are processed in the First-In-First-Out (FIFO) order. This FIFO order causes a huge overhead slowing down the computational speed of the SPIHT algorithm.

To speed up the algorithm, sets and pixels are visited in the Morton order and processed by the suitable pass. This modified algorithm, called Morton order SPIHT, is relatively easy to implement in hardware with a slight degradation of the compression efficiency than the original SPIHT as explained in Wheeler & Pearlman (2000). The processing order of the for-loops in each pass differentiates the original SPIHT from the Morton order SPIHT algorithm.

5.4.4. Limitations of SPIHT Coding Scheme

- Slow processing speed due to its dynamic processing order that depends on the image contents

5.4.5. High Throughput Image Coding

This section presents a modified SPIHT algorithm, called the Block-based Pass-parallel SPIHT (BPS). The proposed algorithm mainly concentrates to speed up both the encoding and decoding times.

A. Block-Based Pass-parallel SPIHT

BPS processes each bit-plane from the most significant bitplane just like the original SPIHT algorithm. However, the processing order of the pixels in each bit-plane is not the same as the original SPIHT algorithm. BPS
first decomposes the whole bitplane into 4 × 4 bit blocks and processes each 4 × 4-bit block at a time. After one 4 × 4-bit block is processed, the next 4×4 bit block is processed in the Morton scanning order as explained in Algazi & Estes (1995).

The encoded stream in the original SPIHT comprises of three kinds, namely, sorting bit, magnitude bit and sign bit. The sorting bit is the result of the significance test for a 2 × 2 or 4 × 4 set showing whether the set is significant or not. The magnitude and sign bits indicate the magnitude and sign of each pixel, respectively. The magnitude and sign bits output in IPP and SPP are called “refining bit,” but the magnitude and sign bits output in ISP are called the “first refining bit” as these bits are the refining bits formed initially for each pixel. The proposed BPS algorithm is formulated for a single 4 × 4-bit block. The 4 × 4-bit block is represented by \( H \) that is decomposed into four 2 × 2 blocks.

BPS comprises of three passes. They are output refining bits, sorting bits, and first refining bits. Based on the type of generated bits, these three passes are called Refinement Pass (RP), Sorting Pass (SP), and First Refinement Pass (FRP), respectively.

The RP is a integration of the IPP and SPP from the original SPIHT and visits each 2 × 2 block which is significant in the previous bit-plane (i.e., \( S_n(Q) = 1 \) as the condition in line 2 of the algorithm. RP then outputs the \( n^{th} \) magnitude bit of the significant 2 × 2 bit block. Moreover, the sign bit of a pixel is the output if the pixel becomes significant in the \( n^{th} \) bit-plane (i.e. \( S_{n+1}(w(l)) + S_n(w(l)) = 0 \) \& \( S_n(w(l)) = 1 \). The order of pixels processed in BPS is different from that in the original SPIHT as the two passes IPP and SPP from the original SPIHT algorithm are integrated as a single pass RP in the proposed Block based Pass parallel SPIHT algorithm.
The ISP pass in the original SPIHT is decomposed into SP and FRP passes in BPS. The SP categorizes a block as either significant or insignificant and transmits the sorting bits. The initial step of the SP is to transmit and produce the significance of the 4 × 4-bit block. This is processed when two constraints are met. The initial condition is that the 4×4-bit block is insignificant in the \((n + 1)\)th bit-plane (i.e., \(S_{n+1}(H \cup D(H)) = 0\)). The second condition \(\sim (parent(H) \wedge S_n(parent(H))) = 0\) implies that it is not mandatory to construct the significance of the set if the 4×4-bit block has a parent whose descendants are insignificant as the insignificance of the parent already shows that the 4 × 4-bit block is insignificant. SP is the only pass that processes a 4×4-bit block. The other two passes RP and FRP process a 2 × 2 bit block as the processing unit.

The remaining operation of the SP is based on the significance of the 4×4-bit block. If the block is significant, it is decomposed into four 2 × 2-bit blocks. The significance of each 2 × 2 block is generated if it is insignificant in the \((n + 1)\)th bit-plane. According to its significance, each 2 × 2-bit block is classified either as an insignificant block to be processed by the SP for the \((n−1)\)th bit-plane or as a significant block to be processed by the FRP pass in the current bit-plane. To be processed by the FRP, a 2 × 2 block requires its significance \(S_n(Q)\) to be set to 1. The significant block processed by the FRP is called the new-significant block. When \((H \cup D(H))\) is insignificant, all four 2 × 2-bit blocks in \(H\) are categorized as insignificant blocks for the \((n − 1)\)th bit-plane. The FRP pass processes the new-significant 2×2-bit blocks categorized by the SP. FRP outputs the \(n\)th magnitude bit of the pixels in the new-significant blocks. If the magnitude bit is significant in the FRP, this shows that the magnitude bit is significant in the first time for the pixel. Therefore, the sign bit is also an output. It is to be noted that ISP in the original SPIHT is decomposed into SP and FRP in the BPS algorithm. The separation of SP and FRP facilitates each pass to be
processed in a single cycle. It should be noted that the process of FRP is based on the results of SP, so that parallel execution is not possible. In the implementation, FRP is delayed by one cycle, thus it can be executed in parallel with the RP and SP of the next 4 × 4-bit block. Parallel execution is possible as the FRP in the current 4 × 4-bit block is not dependent on the RP and SP of the next 4 × 4-bit block. Thus, for each cycle, the bitstream of a single 4 × 4-bit block for a given bit-plane is produced.

The compression efficiency can be obtained by a small adjustment in the selection of FNZB. When the size of the Fractional Fourier transformed image is relatively small (e.g., 16 × 16), the root pixel(s) has a much better absolute value than the other pixels in the image. Thus, only the root pixel(s) is significant for several essential bit-planes. Therefore, the FNZB is obtained from the pixels excluding the root pixel(s). The bit-plane coding then initiates from this FNZB. As a result, the number of encoded bitplanes can be minimized. For the root pixel(s), the value from MSB to FNZB-1 is stored in the header.

Initialization before the algorithm is essential for the FNZBth bit-plane which is the most significant bit-plane to be processed. Initially, the 2 × 2 set that comprises the root pixel(s) is categorized as a significant block. All the other blocks are categorized as insignificant. For any 2×2 set $Q$, the parameter $d_{sig}$ is derived. This parameter is used to compute the significance $S_n(Q \cup D(Q))$. Moreover, for any 4×4 set $H$, significance $S_n(Q \cup D(Q))$ is also computed in advance. The initial derivation of $d_{sig}$ makes the significance evaluation simple such that it can be processed in a single cycle.

**B. Bitstream Generation for a Fast Decoder**

In this scenario, improving the speed of a decoder is more complicated than improving that of an encoder. Since RP and SP are
independent, encoder can process them in parallel. However, in decoder, parallel execution of RP and SP is not possible as their independency of each other is not sufficient for parallel execution. Another constraint for parallel execution is the precalculation of the start bit of each pass in the bitstream. This constraint is very evident as a decoder cannot start to process a pass till the start bit of the pass is known prior to the start of the pass. It is very difficult for a decoder to identify the start bit of each pass as the length of each pass is not constant, and the length can be identified by the decoder only after the pass is completely decoded. Thus, to facilitate parallel execution of multiple passes in a decoder, the bitstream should be formatted in such a way that it should look ahead for the length of the bitstream for each pass.

Thus, before start of the RP, the end bit of the RP magnitude (and the start bit of the next SP sorting) is identified, this, in turn facilitates the decoding of both RP magnitude bits and SP sorting bits in parallel. Alternatively, the number of sign bits in the RP can be identified only after the magnitude bits of the equivalent RP are decoded. Therefore, the sign bits of RP are decoded in one cycle later than the decoding of the equivalent magnitude bits. For FRP, the number of sorting bits transmitted by the SP is known only after SP is completed. Therefore, FRP magnitude bits can be decoded one cycle later than SP. The number of magnitude bits from FRP is identified by the outcome of the SP in the same bit-plane. Therefore, the length of the FRP can be computed in advance before the FRP begins. This shows that the start bit of the RP of the next 4×4-bit block is also identified before the FRP starts. Therefore, both the RP and SP of the next 4×4-bit block is carried out in the same cycle as FRP. Hence, the RP and SP can be processed in parallel with the FRP of the previous 4×4-bit block.

Sign bits are stored from the right of the bitstream. The length of the sign bits transmitted by RP is not known before the RP is completed as it
is identified by the RP based on the magnitude bit. Thus, the sign bits are processed by the decoder one cycle after the equivalent magnitude bits. The sign bits of FRP can initiate only after the magnitude bits of the FRP are decoded. Thus, the decoding of FRP sign bits is carried out one cycle after the decoding of FRP magnitude bits. It is to be observed that the FRP sign bits can be processed in the same cycle as the RP sign bits of the next 4 × 4-bit block.

5.4.6. Decompression

This process is the reverse of the compression technique. After SPIHT, it is necessary to transform the data to the original domain (spatial domain), to do this, the Inverse Fractional Fourier Transform is first applied in the columns and then in the rows.

5.4.7. Quasi Group Decryption

This process is highly similar to the process of encryption which has just been discussed. The key point to be considered is the construction of the inverse matrix. The left inverse ‘\‘ is used for the Quasigroup decryption. The fundamental equation for encryption is,

$$D(a_1, a_2, a_3 \ldots, a_n) = e_1, e_2, e_3, \ldots, e_n$$  \hspace{1cm} (5.3)

where \(e_1 = \frac{a}{a_1}\) and \(e_i = \frac{a_i-1}{a_i}\)

To carry out the process of decryption, the inverse matrix of a given Quasigroup has to be constructed and the mapping procedure executed.

Thus, the entire process of DICOM image compression is carried out with Fraction Fourier Transform domain and BSP SPIHT approach.
5.5. EXPERIMENTAL RESULTS

The experiment is carried out in MATLAB 7.0. The same set up used in the previous chapter has been used here too.

![Figure 5.4 DICOM Lung Test Images](image)

**Figure 5.4 DICOM Lung Test Images**

<table>
<thead>
<tr>
<th>Standard Images</th>
<th>Modulus (bits)</th>
<th>Encryption Time (seconds)</th>
<th>Decryption Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>RSA</td>
<td>Quasigroup Encryption</td>
</tr>
<tr>
<td>Lung 1</td>
<td>2048</td>
<td>3.105</td>
<td>2.01</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>1.965</td>
<td>1.12</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>1.510</td>
<td>0.92</td>
</tr>
<tr>
<td>Lung 2</td>
<td>2048</td>
<td>2.901</td>
<td>1.85</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>1.789</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>1.443</td>
<td>0.79</td>
</tr>
<tr>
<td>Lung 3</td>
<td>2048</td>
<td>2.936</td>
<td>1.625</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>1.839</td>
<td>0.945</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>1.493</td>
<td>0.602</td>
</tr>
</tbody>
</table>
It is observed from the Table 5.1 that the proposed Quasigroup Encryption consumes lesser encryption and decryption time than the traditional RSA approach. This is mainly due to the matrix formation of the Quasigroup encryption. The evaluation is carried out for three different bits such as 2048, 1024 and 512 bits. For all the three Lung images considered, the encryption and decryption time of the proposed Quasigroup encryption is lesser.

Table 5.2 Comparison of PSNR of Fractional Fourier Transform with Pass-Parallel SPIHT

<table>
<thead>
<tr>
<th>Standard Images</th>
<th>Bit Per Pixel (Bpp)</th>
<th>Wavelet Transform with SPIHT</th>
<th>D2 Wavelet Transform with Modified SPIHT</th>
<th>Fractional Fourier Transform with Pass-Parallel SPIHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lung 1</td>
<td>0.5</td>
<td>29.48</td>
<td>31.89</td>
<td>36.6</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>33.56</td>
<td>34.33</td>
<td>37.10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>36.62</td>
<td>37.29</td>
<td>38.20</td>
</tr>
<tr>
<td>Lung 2</td>
<td>0.5</td>
<td>20.10</td>
<td>21.7</td>
<td>23.97</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>27.69</td>
<td>29</td>
<td>32.85</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>33.11</td>
<td>35.76</td>
<td>37.19</td>
</tr>
<tr>
<td>Lung 3</td>
<td>0.5</td>
<td>20.3</td>
<td>21.75</td>
<td>24.26</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>27.83</td>
<td>29.36</td>
<td>33.17</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>33.81</td>
<td>35.88</td>
<td>38.16</td>
</tr>
</tbody>
</table>

Table 5.2 shows the comparison of the PSNR value of the proposed DFrFT with that of the existing approaches such as Wavelet Transform with SPIHT and D2 Transform Modified SPIHT. It is observed that the proposed approach provides better PSNR value than the existing techniques.
Figure 5.5  PSNR Evaluation of the Image Compression Techniques for DICOM Images for 0.5 (Bpp)

Figure 5.5 is drawn for PSNR Evaluation of the Image Compression Techniques for DICOM Images for 0.5 (Bpp). From the figure, it is observed that the PSNR value of the proposed DFrFT algorithm approach is much higher than that of the existing transformation approaches.

Figure 5.6  PSNR Evaluation of the Image Compression Techniques for DICOM Images for 1 (Bpp)
Figure 5.6 is drawn for PSNR Evaluation of the Image Compression Techniques for DICOM Images for 1 (Bpp). From the figure, it is observed that the PSNR value of the proposed DFrFT algorithm approach is much higher than that of the existing transformation approaches.

![Graph showing PSNR evaluation for various compression techniques.

Test Images for 2 bpp

Lung 1
Lung 2
Lung 3

Wavelet Transform with SPIHT
D2 Wavelet Transform with Modified SPIHT
Fractional Fourier Transform with BSP SPIHT]

Figure 5.7  PSNR Evaluation of the Image Compression Techniques for DICOM Images for 2 (Bpp)

Figure 5.7 is drawn for PSNR Evaluation of the Image Compression Techniques for DICOM Images for 2 (Bpp). From the figure, it is observed that the PSNR value of the proposed DFrFT algorithm approach is much higher than that of the existing transformation approaches.

The comparison of MSE value is shown in Table 5.3.
Table 5.3 Comparison of MSE Value of Fractional Fourier Transform with Pass-Parallel SPIHT

<table>
<thead>
<tr>
<th>Standard Images</th>
<th>Bit Per Pixel (Bpp)</th>
<th>Wavelet Transform with SPIHT</th>
<th>D2 Wavelet Transform with Modified SPIHT</th>
<th>Fractional Fourier Transform with BSP SPIHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lung 1</td>
<td>2</td>
<td>115.68</td>
<td>35.11</td>
<td>28.41</td>
</tr>
<tr>
<td>Lung 2</td>
<td>2</td>
<td>114.52</td>
<td>33.14</td>
<td>27.56</td>
</tr>
<tr>
<td>Lung 3</td>
<td>2</td>
<td>113.54</td>
<td>32.54</td>
<td>26.14</td>
</tr>
</tbody>
</table>

Figure 5.8 MSE Evaluation of the Image Compression Techniques for DICOM Images for 2 (Bpp)

The comparison of the MSE value of the various techniques is shown in Figure 5.6. It is observed from the figure that the MSE value of the proposed DFrrFT approach is much lesser than that of the existing transformation approaches.
Figure 5.9 Evaluated DICOM Lung Images using the Proposed Image Compression Approach

Figure 5.9 shows the output images of the proposed Image compression technique which uses Quasigroup encryption and pass parallel SPIHT encoding algorithm.
5.6. SUMMARY

This chapter clearly discusses the proposed Fractional Fourier Transform Based Medical Image Compression Using Pass-Parallel SPIHT Encoding Scheme. The overall methodological flow of the proposed technique has been discussed in this chapter. It evaluates the performance of the proposed approach. The performance of the proposed approach has been compared with that of the various image compression techniques. It is observed from the experimental results that the proposed Fractional Fourier Transform and Pass-Parallel SPIHT Encoding Scheme provide the best results.