GOVERNO DO ESTADO DO RIO GRANDE DO NORTE
UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN
FACULDADE DE CIÊNCIAS EXATAS E NATURAIS
DEPARTAMENTO DE INFORMÁTICA

PRICILA RAIANE NUNES DE MELO

ANÁLISE DE MÉTODOS E BASES DE COMPRESSÃO DE IMAGENS USANDO DADOS DO TRÁFEGO VEICULAR VIA INTERFACE GOOGLE MAPS

MOSSORÓ/RN
2016

PRICILA RAIANE NUNES DE MELO

ANÁLISE DE MÉTODOS E BASES DE COMPRESSÃO DE IMAGENS USANDO DADOS DO TRÁFEGO VEICULAR VIA INTERFACE GOOGLE MAPS

Monografia apresentada ao Departamento de informática, da Universidade do Estado do Rio Grande do Norte – UERN, como requisito para obtenção do grau de Bacharel em Ciência da Computação, sob a orientação do Prof. Dr. Harold Ivan Ângulo Bustos.

MOSSORÓ - RN

2016

Catalogação da Publicação na Fonte.

Universidade do Estado do Rio Grande do Norte.

Melo, Pricila Raiane Nunes De

Análise De Métodos E Bases De Compressão De Imagens Usando Dados Do Tráfego Veicular Via Interface *Google Maps* / Pricila Raiane Nunes De Melo – Mossoró, RN, 2016.

68 f.
Orientador(a): Prof. Dr. Harold Ivan Ângulo Bustos.

Monografia (Bacharelado) Universidade do Estado do Rio Grande do Norte. Curso de Ciências da Computação

1. Compressão de imagens. 2. Transformada *Wavelets*. 3. *Google Maps*. l. Bustos, Harold Ivan Angulo. II. Universidade do Estado do Rio Grande do Norte. III. Título.

UERN/ BC CDD 006

Bibliotecário: Sebastião Lopes Galvão Neto – CRB - 15/486

PRICILA RAIANE NUNES DE MELO

ANÁLISE DE MÉTODOS E BASES DE COMPRESSÃO DE IMAGENS USANDO DADOS DO TRÁFEGO VEICULAR VIA INTERFACE GOOGLE MAPS

Monografia apresentada ao Departamento de informática, da Universidade do Estado do Rio Grande do Norte – UERN, como requisito para obtenção do grau de Bacharel em Ciência da Computação, sob a orientação do Prof. Dr. Harold Ivan Ângulo Bustos, submetida à aprovação da banca examinadora composta pelos seguintes membros:

Aprovado em: 07/12/2016

BANCA EXAMINADORA

Prof. Dr. Harold Ivan Ângulo Bustos (Orientador)
Universidade do Estado do Rio Grande do Norte – UERN

Prof. Dr. Carlos Heitor Pereira Liberalino (Examinador)
Universidade do Estado do Rio Grande do Norte – UERN

Prof. Dr. Francisco Chagas de Lima Júnior (Examinador)
Universidade do Estado do Rio Grande do Norte – UERN

Dedico este trabalho ao meu Pai Francisco Leite e a minha Mãe Lúcia Luíza.

AGRADECIMENTO

Agradeço primeiramente a Deus por ter me dado saúde e força para superar todas as dificuldades

Agradeço a minha família, de uma forma muito especial aos meus pais Francisco e Lúcia, pelo amor, dedicação, apoio e por todos os sacrifícios. Aos meus irmãos Patrícia e Paulo por sempre está disposto a me ajudar. Ao meu sobrinho Kauê por todo amor.

Agradeço também a todos os meus verdadeiros amigos pelo seu encorajamento. Aos amigos que fiz fora da Universidade, pelo apoio e pelos momentos vividos durante essa jornada sempre me acrescentando muito pessoalmente. Agradeço em especial a Karol da qual a ajuda foi essencial e por todo seu apoio durante os tempos difíceis.

Agradeço ao meu Orientador Harold Ivan Ângulo Bustos, por toda a sua orientação, por me mostrar o caminho a ser seguido, pelas correções e incentivo.

E a todos que direto ou indiretamente fizeram parte da minha formação, a todos, o meu muito obrigada.

“Uma vez que as suas energias estejam todas concentradas nas raízes, surgirão novos brotos, novas folhagens, novos ramos, e você começará a mover-se para cima, em direção às estrelas”.

Osho

RESUMO

A compressão de dados tem desenvolvido um papel fundamental em aplicações de diversas áreas, principalmente no que se refere ao trabalho com imagens. Neste contexto, a técnica de *Wavelet* aparece como alternativa promissora para a redução de custos com sistemas de transmissão e de armazenamento. Neste trabalho, nos detemos a analisar os métodos e as bases de compressão de imagem, através do uso de transformada *Wavelets* com o auxílio da ferramenta *Matlab*. Buscamos, de modo geral, comparar os métodos de compressão de imagem *EZW*, *SPIHT* e *STW*, verificando qual deles é o mais eficaz na otimização da economia de espaço computacional, tanto no armazenamento como na transmissão de imagens. Para tanto, utilizamos imagens de satélite do tráfego da cidade de Mossoró/RN, obtidas a partir do *Google Maps*. Em meio as análises e tendo em vista nossos objetivos, percebemos que o método de compressão *SPHIT* na base *Haar* foi o que demonstrou um melhor desempenho, sendo, portanto, o mais eficaz. Diante disso, esta pesquisa poderá contribuir para o desenvolvimento de futuros trabalhos que visem o aprimoramento na transmissão e armazenamento de imagens.

PALAVRAS- CHAVE: Compressão de imagens, transformada *Wavelets*, *Google Maps.*

ABSTRACT

Data compression has developed a key role in various areas of application, especially with regard to work with images. In this context, the Wavelet technique appears as a promising alternative for the reduction of costs with transmission and storage systems. In this work, we study the methods and the bases of image compression, through the use of Wavelets Transform with the aid of the Matlab tool. We generally seek to compare the EZW, SPIHT and STW image compression methods by verifying which one is most effective in optimizing computational space savings, both in storage and in image transmission. To do so, we use satellite images of traffic from the city of Mossoró/RN, obtained from Google Maps. Amid the analysis and in view of our goals, we realize that SPHIT compression method in Haar base was demonstrating a better performance, and therefore more effective.Given the above, this research may contribute to the development of future works aimed at improving the transmission and storage of images.

KEYWORDS: Image compression, Wavelets Transform, Google Maps.

LISTA DE ABREVIATURAS E SIGLAS

EZW – *Wavelets* Embedded *Zerotree Wavele*

SPIHT – *Set Partitioning in Hierarchical Trees*

STW – Spatial-orientation Tree Wavelet

MSE – Mean Square Error (Erro Médio Quadrático)

PSNR – Peak Signal-to-Noise Ratio (Razão Sinal-Ruído de Pico)

TW – Transformada Wavelet

POS – Positivo Significante

NEG – Negativo Significante

IZ – Zero Isolado

ZTR – Raiz *Zerotree*

LIP – Lista de pixels insignificantes

LIS – Lista de conjuntos insignificantes

LSP – Lista de pixels significantes

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os avanços tecnológicos, principalmente na área da informática, possibilitaram que as informações contendo uma alta taxa de transmissão fossem disponibilizadas em rede de computadores. No entanto, o transporte desses dados pode gerar uma grande banda de transmissão e uma alta taxa de armazenamento.

Neste contexto, são necessários sistemas de telecomunicações e de armazenamento mais eficazes para tratar a grande quantidade de informações, aplicando técnicas de compressão de dados que facilitem seu armazenamento, transmissão e processamento. O termo “compressão de dados” consiste em reduzir o número de informações redundantes, com finalidade de obter outra especificação. Isso faz com que os dados – imagens, vídeos, transmissão de voz, entre outros – ocupem um espaço menor no seu armazenamento.

Essa compressão de dados tem desempenhando um papel crucial em aplicações de diversas áreas, principalmente no que se refere ao trabalho com imagens. Em meio a isso, a técnica de *Wavelet* aparece como alternativa promissora para a redução de custos com sistemas de transmissão e de armazenamento. Essa ferramenta exerce funções detalhadas de análise em escala para representar dados. Neste trabalho nos detemos a analisar os métodos e as bases de compressão de imagem, através do uso de transformada *Wavelets* com o auxílio da ferramenta *Matlab*, que é um software computacional mundialmente conhecido como uma ferramenta para solucionar problemas matemáticos, científicos e tecnológicos. Buscamos comparar qual desses métodos é o mais eficaz na otimização da economia de espaço computacional, tanto no armazenamento como na transmissão de imagens. Centramos nossas análises nos algoritmos *Wavelets Embedded Zerotree Wavelet (EZW)*, *Set Partitioning in Hierarchical Trees (SPIHT)* e *Spatial-orientation Tree Wavelet* (STW), e em cinco bases *Wavelets*, sendo elas *Haar*, *Daubechies*, *Symlets*, *Coiflets* e *Biortogonal*, utilizando os parâmetros de compressão, *MSE – Mean Square Error* (Erro Médio Quadrático)*, PSNR – Peak Signal-to-Noise Ratio* (Razão Sinal-Ruído de Pico) e a Relação de Compressão (Comp. Ratio). Ambos, bases e parâmetros de compressão, foram selecionados por serem mais populares e de fácil utilização.

Para tanto, utilizamos imagens de satélite do tráfego da cidade de Mossoró/RN, obtidas a partir do *Google Maps*, por ser um dos mapas mais

completos e atualizados do mundo. Essa ferramenta apresenta uma função que mostra, em tempo real, as condições de trânsito nas principais vias de algumas cidades mundiais. Além disso, o *Google Maps* exibe um histórico do tráfego com base nos dias da semana.

Diante dessas considerações, esta pesquisa se justifica, podendo contribuir para o desenvolvimento de futuros trabalhos que visem o aprimoramento na transmissão e armazenamento de imagens.

1.1 OBJETIVOS

1.1.1 Geral

Analisar os métodos de compressão de imagem *EZW*, *SPIHT* e *STW* através do uso de transformada *Wavelets* com o auxílio da ferramenta *Matlab*. Buscando comparar qual desses métodos é o mais eficaz na otimização da economia de espaço computacional, tanto no armazenamento como na transmissão de imagens.

1.1.2 Específicos

- Verificar os métodos e as bases *Wavelets* de compressão de imagem, com o auxílio da ferramenta *Matlab*.
- Comparar os métodos e as bases *Wavelets* de acordo com os parâmetros de compressão *MSE, PSNR* e *Comp. Ratio.*
- Identificar qual método e qual base *Wavelets* são mais eficazes na compressão de imagem nesta aplicação.

1.1.3 Organização do Trabalho

O trabalho esta organizado em cinco capítulos e um anexo. O Capitulo 2 apresenta primeiro uma breve introdução sobre a compressão de imagem, técnicas de compressão e as medidas de distorções. Ele apresenta também um rápido histórico sobre a *Wavelet*, uma análise sobre a Transformada *Wavelet*, as famílias *Wavelet* de *Haar, Daubechies, Symlets, Coifman, biortogonal*, e os algoritmos de compressão *EZW, SPIHT e STW* usados nos testes. O capitulo 3 apresenta os

softwares que foram usados no trabalho, ou seja, o *MatLab* utilizado para realizar todos os testes e o Excel para fazer os gráficos e as tabelas. No capitulo 4 são apresentados os testes e os resultados obtidos segundo o Erro Médio Quadrático, a Relação Sinal Ruído e a Relação de Compressão, apresentando qual a base e algoritmo fornece melhor resultado. Finalmente o Capitulo 5 apresenta as conclusões sobre os resultados obtidos.

2 FUNDAMENTOS DA COMPRESSÃO DE IMAGEM

O termo compressão de dados refere-se ao processo de redução de dados necessários para representar uma certa quantidade de informação. Considerando que os dados são o meio pelo qual a informação é transmitida, várias quantidades de dados podem ser utilizadas para representar a mesma quantidade de informação, podendo, deste modo, conter informações irrelevantes ou repetidas. Quando isso acontece, o risco de redundância de dados é maior.

A aplicação da compressão de imagem é realizada através do conjunto de técnicas e ação de algoritmos que buscam reduzir a quantidade de dados necessária para a representação da imagem, fazendo com que o código gerado pelo processo seja menor do que o código original. Além disso, tem o intuito de diminuir o custo operacional, otimizando o espaço no disco de armazenamento como também a redução do tráfico da rede. Sanches (2004) afirma que:

> A área de compressão de imagens tem por finalidade, sendo dada uma imagem em uma determinada especificação, obter uma outra especificação dessa imagem que ocupe o menor espaço possível no seu armazenamento. Os métodos de compressão visam produzir, através da eliminação da redundância dos dados, o código mais compacto que preserve as informações essenciais contidas na imagem. Dependendo diretamente do tipo de aplicação, temos a necessidade de se ter compressão reversível (codificação sem perda), permitindo a recuperação exata dos dados da imagem original, ou irreversível (codificação com perda). (SANCHES, 2004, p. 1).

Neste sentido, a compressão de imagem só é possível devido ao alto grau de coerência que elas apresentam. Em outras palavras, ao considerar qualquer pixel (elemento) de uma imagem, provavelmente a cor desse elemento e dos elementos

vizinhos será igual, porque há uma grande chance desses pixels pertencerem a um mesmo objeto da imagem. Deste modo, através da redução ou eliminação de redundância, os diversos métodos de compressão visam produzir um código mais compacto que preserve as informações contidas na imagem.

A redundância de dados é um aspecto fundamental no estudo de compressão de imagens digitais. Para ser classificado matematicamente, se n_1 e n_2 os números de unidades de informação contidas em dois conjuntos de dados, que representam a mesma informação, a redundância relativa (R_D) do primeiro conjunto de dados pode ser definida, como (equação (1)) (FILHO e NETO, 1999):

$$R_D = 1 - \frac{1}{C_R} \tag{1}$$

Em que C_R , geralmente denominada taxa de compressão, é definida por, (equação(2)):

$$C_R = \frac{n_1}{n_2} \tag{2}$$

Se $n_2 = n_1$, $C_R = 1$ e $R_D = 0$, tem-se que o primeiro conjunto de dados não contém nenhum dado redundante em relação ao segundo. Quando $n_2 << n_1$, $C_R \rightarrow \infty$ e $R_D \rightarrow 1$, tem-se compressão relevante e dados altamente redundantes. No último caso, que $n_{2 >> } n_1$, $C_R \rightarrow 0$ e $R_D \rightarrow -\infty$, podemos concluir que o segundo conjunto de dados contém muito mais dados do que o primeiro. Este ultimo caso normalmente é indesejável de expansão de dados. Em geral, C_R e R_D situa-se nos intervalos abertos (0, ∞) e (-∞, 1) respectivamente (FILHO e NETO, 1999).

São explorados três tipos de redundância de dados no mecanismo de compressão, que são identificados por: redundância de codificação, redundância interpixel e redundância psicovisual. Na redundância de código, o processo de codificação determina um código binário com variável tamanho de acordo com a probabilidade de ocorrência de tom de cinza ou cor do pixel na cena. Ou seja, atribuído um código menor (em bits), são os valores que ocorrem com mais frequência na imagem, enquanto os valores que ocorrem com menos frequência são atribuídos um código maior (GONZALES e WOODS, 2002).

Por outro lado, a redundância interpixel permite prever o valor de qualquer pixel de uma imagem pelo valor de seus vizinhos, a informação transportada por um único pixel é relativamente pequena. Uma grande parte da contribuição visual é redundante, podendo ter sido deduzida a partir de seus vizinhos (GONZALES e WOODS, 2002).

Por fim, a redundância psicovisual é diferente das anteriores, pois é associada a informações visuais quantificáveis ou reais. Ocorre quando algumas informações têm relativamente menos importância do que outras em um processamento visual normal; essas informações com menor importância podem ser consideradas redundâncias, sendo eliminadas sem afetar a qualidade da imagem na percepção visual humana. Essa eliminação traz perdas de informação de forma irreversível, o que resulta em uma compressão de dados com perdas, mas sem prejudicar a qualidade da percepção da imagem (FILHO e NETO, 1999).

É efetivada a compressão de dados quando uma ou mais dessas redundâncias são reproduzidas ou eliminadas.

2.1 MODELO DE COMPRESSÃO DE IMAGENS

Um sistema de codificação é composto por dois blocos estruturais distintos, um codificador e um decodificador. Como mostra a Figura 1, tanto o codificador como o decodificador são compostos por dois sub-blocos independentes. O codificador possui um codificador de fonte que remove as redundâncias recebidas, e também, um codificador de canal o qual aumenta a imunidade a ruídos do sinal produzido à saída do codificador de fonte (FILHO e NETO, 1999).

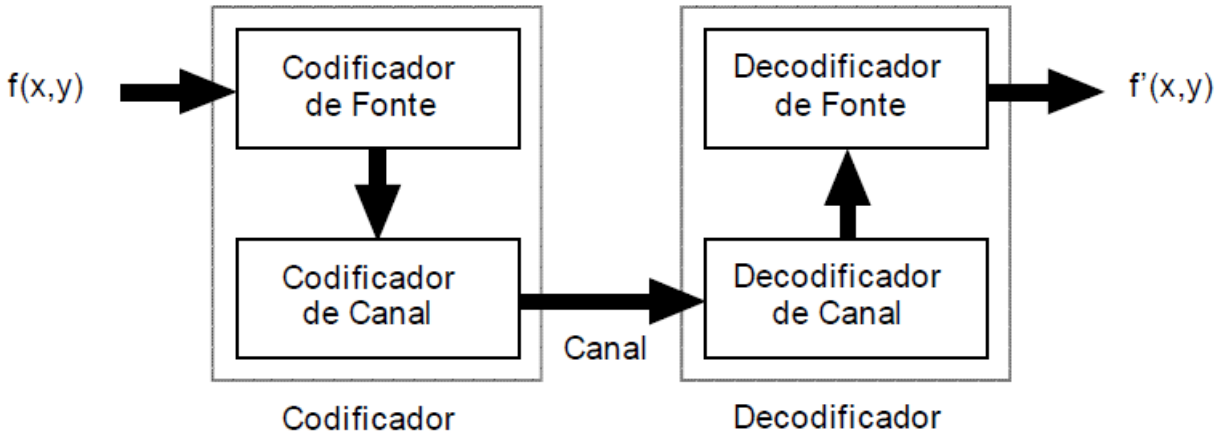
Filho e Neto (1999) descrevem da seguinte forma o sistema de compressão:

> O codificador parte de uma imagem de entrada f(x,y), a partir da qual cria um conjunto de símbolos. Após a transmissão através do canal, o sinal codificado é aplicado ao bloco decodificador, onde uma imagem de saída reconstruída f'(x,y) é produzida. A imagem recebida, f'(x,y), poderá ou não ser uma réplica exata de f(x,y). Em caso positivo, o sistema é dito imune a erros, ou seja, capaz de preservar a informação; em caso negativo, haverá um certo nível de distorção presente na imagem reconstruída.

Na etapa do decodificador é onde se encontra a situação analógica que possui um decodificador de canal, seguido por um decodificador de fonte. As etapas

codificadora e decodificadora de canal podem ser omitidas quando o canal entre o codificador e o decodificador for imune a ruídos.

Figura 1. Sistema genérico de compressão de dados

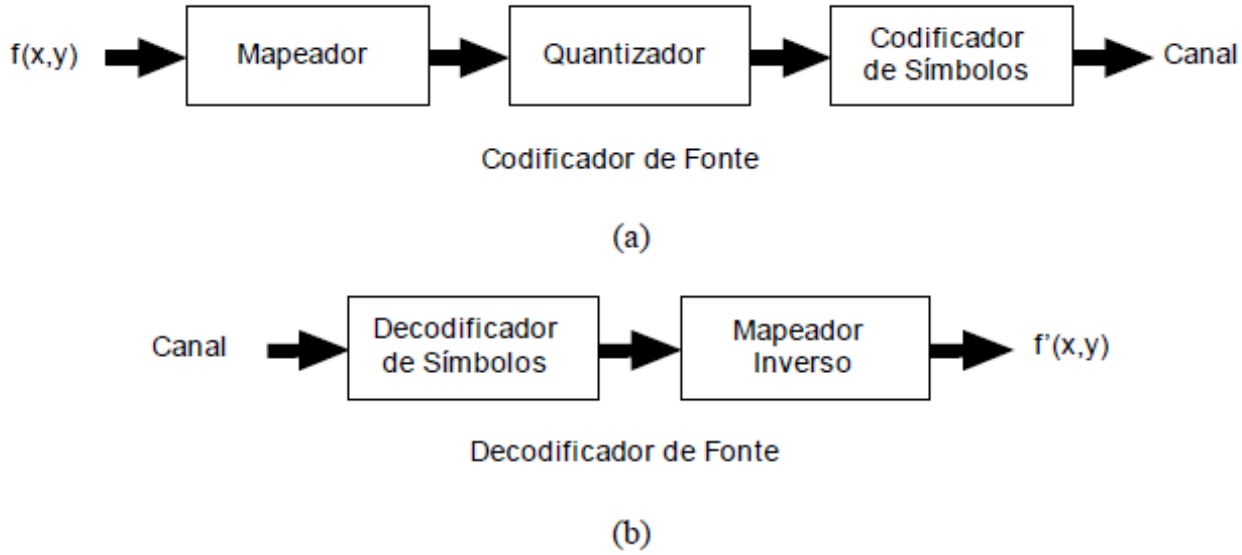


Fonte: (FILHO e NETO, 1999)

2.1.1 Codificador e Decodificador de fonte

O codificador de fonte na Figura 2 (a) está responsável pela eliminação ou redução de quaisquer redundâncias (de codificação, interpixel ou psicovisual). É composto por três estágios: no primeiro, o mapeador, essa operação pode-se considerar reversível que transforma os dados de entrada em um formato projetado para diminuir a redundância interpixel da imagem de entrada. No segundo estagio, o quantizador, as redundâncias psicovisuais da imagem original são diminuídas. Essa operação é irreversível, consequentemente ela deve ser omitida quando há o desejo de compressão livre de erros. Já no último estágio, o codificador de símbolos, é feita a criação de um código de comprimento fixo ou variável para representar a saída do quantizador e minimizar a redundância de codificação. Essa operação é reversível (GONZALES e WOODS, 2002).

Figura 2. Codificador e decodificador de Fonte



Fonte: (FILHO e NETO, 1999)

O decodificador de fonte na Figura 2 (b) contém apenas dois componentes: um decodificador de símbolos e um mapeador inverso. Esses blocos executam a sua operação em ordem inversa dos blocos mapeador e codificador de símbolo do codificador de fonte. Um bloco quantizador inverso não é incluído, porque a quantização resulta em perda irreversível.

2.1.2 Codificador e decodificador de canal

Tanto o codificador como decodificador de canal tem uma grande importância quando o canal de transmissão estiver sujeito a erro ou contaminado por ruído. Ambos têm a função de reduzir o impacto do ruído do canal através da inserção de uma forma controlada de redundância nos dados fonte codificados (FILHO e NETO, 1999).

R.W. Hamming inventou, em 1950, uma técnica de codificação de canal considerada uma das mais úteis, que garante a detecção e/ou correção de erros de múltiplos bits. Essa técnica adiciona bits suficientes aos dados que está sendo codificados para garantir que um número mínimo de bits varie entre as palavras de código válidas.

2.1.3 Técnicas de compressão

São exigidas duas grandes categorias diferentes para executar a compressão de imagens digitais, sendo elas: os métodos sem perdas e métodos com perdas. O método de compressão sem perdas resulta na eliminação somente das informações que venham a ser redundantes, permitindo a reconstrução da imagem igual a original após sua decodificação, e, como o próprio nome já sugere, não há perdas de dados durante o processo. Essa técnica é usada principalmente quando a qualidade e a fidelidade da imagem são importantes e devem ser mantidas, como, por exemplo, em documentos legais, imagens de satélites, ou imagens médicas, pois, se há perda de informações, pode comprometer a precisão do diagnóstico. Esse processo fornece baixas taxas de compressão.

Por outro lado, o método de compressão com perdas tem o objetivo de realizar uma alta taxa de compressão. Algumas informações podem ser eliminadas, pois, esse método procura tirar vantagem das limitações da visão humana que, muitas vezes, não consegue perceber algumas informações. É útil em aplicações em que certa quantidade de erro é aceitável, como por exemplo, internet, videoconferência e em transmissão de televisão (SILVA e FLÔRES, 2005). A compressão com perdas é mais utilizada na maioria das aplicações devido à busca por uma maior taxa de compressão.

2.1.4 Medidas de distorção

Para a avaliação do desempenho dos sistemas de compressão, são utilizadas as medidas objetivas. *Mean Square Error (MSE)* (em português, Média dos Erros Quadráticos), originalmente mede a fidelidade entre sinais através da avaliação de erro. O *MSE* é os dados da comparação entre a imagem original e a imagem reconstruída após o processo de compressão e descompressão.

O cálculo (3) é do *MSE*, dado por:

$$\text{MSE} = \frac{1}{M\text{x}N} \sum_{x=1}^{M-1} \sum_{y=1}^{N-1} \left| f(x,y) - f'(x,y) \right|^2 \qquad (3)$$

Onde f(x,y) é a imagem original e f'(x,y) é a imagem reconstruída, e M,N são as dimensões da imagem. Quanto menor for o valor obtido pelo MSE, menor será o erro.

No *Peak Signal to Noise Ratio (PSNR)* (pico de sinal de ruído), o calculo (representada da equação (4)) é feito em dB (decibéis). Corresponde a um ajuste de escala do *MSE*, onde quanto maior o seu valor, menor o erro.

$$PSNR_{dB} = 10 \log_{10} \left(\frac{S^2}{MSE} \right) \quad (4)$$

S é o valor máximo dos elementos na amostra. Para imagens originais representadas com 8bits/pixel, *S* = 255. *PSNR* é geralmente preferido em comparações práticas porque é uma medida logarítmica e verificou-se que os cérebros respondem algoritmicamente a mudanças na intensidade da luz. Mesmo o *PSNR* sendo objetivo, não resolve completamente o problema de especificar a melhor compressão. A avaliação irá variar dependendo da qualidade da reconstrução que é necessária para uma determinada aplicação (WALKER, 2008).

2.2 ANÁLISE DE WAVELETS

As *Wavelets* são ferramentas matemáticas para decompor funções hierarquicamente. A ideia fundamental das *wavelets* é a análise de dados de acordo com a escala. Em geral, permitem descrever através de uma função, desde o detalhe mais genérico ao detalhe mais especifico. Independente da função equivalente a uma imagem, uma curva, uma superfície, ou um sinal, as *wavelets* oferecem uma técnica sofisticada para representar os níveis de detalhes presentes.

A análise *wavelets* teve sua fundação matemática derivada do trabalho de Joseph Fourier no século XIX. *Fourier* descobriu que é possível representar funções periódicas através de funções seno e/ou cosseno, essas funções são ortogonais. As *wavelets* foram mencionadas pela primeira vez em 1909, na tese de doutorado de *Alfred Haar*. Para ele, uma propriedade da *wavelet* de *Haar* tem um suporte compacto, ela desaparece fora de um intervalo finito, a função tem valor zero. Infelizmente, *wavelets* de *Haar* não são continuamente diferenciáveis, que de alguma forma limita as suas aplicações (GRAPS, 1995).

A transformada *wavelete* é o resultado do trabalho de um grande número de pesquisadores. Ficaram no anonimato por vários anos, até que na década de 30, vários grupos de trabalho de forma independente pesquisaram a representação de funções usando uma base com variações de escala. Então, usando a base da *wavelet* de *Haar*, *Paul Lavy* investigou o movimento *Browniano*, um tipo de sinal aleatório, mostrando que as funções da base *Haar* eram melhores do que as da base de Fourier para estudar os pequenos e complicados detalhes do movimento Browniano. No início da década de 80, *Jean Morlet*, um geógrafo, junto com *Alex Grossman*, um físico, desenvolveram *wavelets* no contexto da física quântica, onde surgiu a teoria de “frames”.

Em 1985, o francês *Stephane Mallat*, através do seu trabalho em processamento digital de imagem, deu um grande impulso às *wavelets*. E inspirado nos resultados de *Mallat*, *Y. Meyer* construiu a primeira *wavelet* não-trivial (suave), que, ao contrário da *wavelet Haar*, é continuamente diferenciável, no entanto não possui suportes compactos.

Pouco tempo depois, *Ingrid Daubechies* usou o trabalho de *Mallat* para construir um conjunto de funções de bases ortogonais de *wavelets* suaves, com suportes compactos. Os trabalhos de *Daubehies* são uma base das aplicações atuais de *wavelets* (LIMA, 2003).

2.2.1 TRANSFORMADA WAVELET

Um dos métodos mais utilizados na compressão de imagens é a codificação por transformada. A transformada busca uma redução na correlação existente entre os elementos de uma imagem, e procura concentrar a maior quantidade de energia possível no menor número de coeficientes. Estes podem ser quantizados separadamente, pois tendem a apresentar correlação reduzida.

A transformada *Wavelet* (TW) destaca-se entre as transformadas, pois apresenta uma grande capacidade de concentrar energia, possui uma forte relação com o sistema visual humano, permitindo uma alta taxa de compressão com menor nível de degradação visível. Uma das propriedades mais atraentes dessa transformada é a sua flexibilidade com relação à duração e à posição das funções de base da representação (*wavelets*), o que permite que sinais com conteúdo de

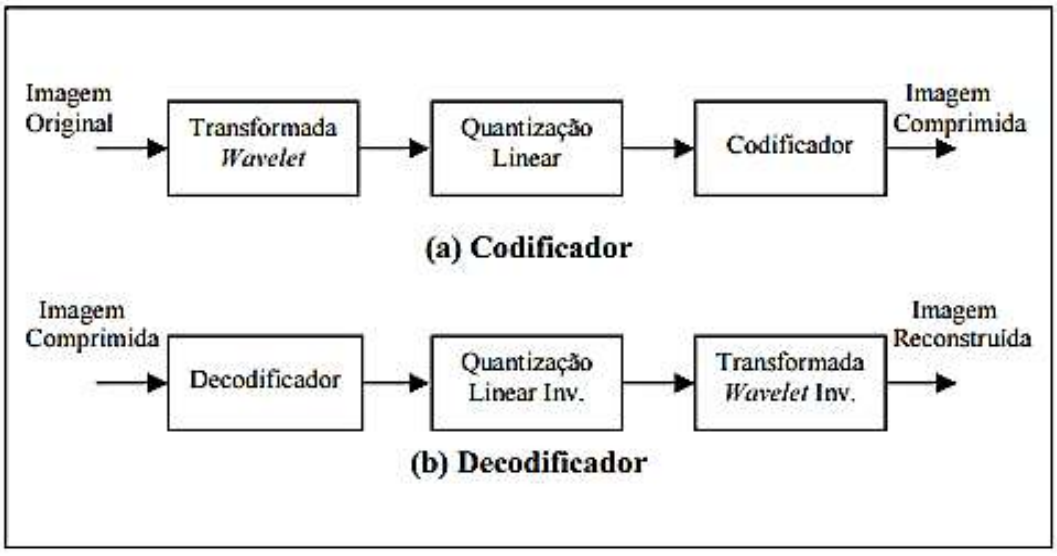
frequência variados, como imagens, possam ser bem mais representados com um número reduzido de funções (FARIAS, 1998).

Segundo *Daubechies* (1992), a transformada *wavelet* é uma ferramenta que corta dados ou funções ou operadores em componentes de frequência diferente, e em seguida estuda cada componente com uma resolução associada à sua escala.

As aplicações da transformada *wavelets* se desenvolveram nos campos da Matemática, Engenharia e da Física Quântica. Na atualidade vêm sendo incluídas por uma série de aplicações, como na geologia sísmica, processamento de imagens (compressão de dados), visão computacional e humana, radar e sonar, espectrometria, computação gráfica, predição de terremotos e maremotos, turbulências, análise de transiente em linhas de potência, fractais, entre outras aplicações.

O diagrama de blocos é um sistema básico de compressão de imagens baseado na transformada *wavelet*, representado abaixo na Figura 3 (a). O sistema pode ser dividido em três estágios, que são: a transformada *wavelet*, a quantização e a codificação de entropia (STRANG e NGUYEN, 1996). Cada um desses estágios tem importância no desempenho da compressão, levando em conta as características da imagem, as taxas de compressão desejadas e as limitações inerentes ao processo de implementação.

Figura 3. Diagrama de blocos de compressão baseado na Transformada *Wavelet*



Fonte: (FARIAS, 1998)

No primeiro estágio, calcula-se a transformada *wavelet* bidimensional da imagem, podendo ser decomposta em uma, duas ou mais camadas. Além do mais, existe uma vasta variedade de famílias de *wavelet* que podem ser utilizadas, cada uma delas apresenta suas características especificas, por exemplo, suavidade, simetria, ortogonalidade, entre outras.

Já no segundo estágio, os coeficientes da transformada são quantizados. Aqui, as distorções são introduzidas na imagem. A operação de quantização torna o processo irreversível, a imagem reconstruída difere numericamente da original, devido os erros de quantização (GUSMÃO, 2002).

No último estágio, como os coeficientes da transformada já estão quantizados, são usadas técnicas de codificação sem perdas ou entropicamente. O processo inverso é realizado no receptor para a reconstrução da imagem.

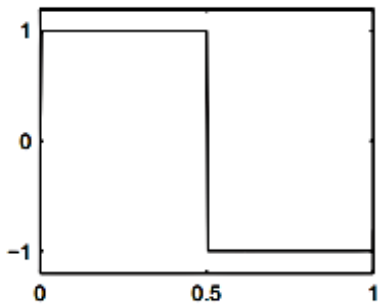
2.3 AS FAMILIAS WAVELETS

Existem diferentes tipos de famílias Wavelets. Veremos, a seguir, os vários tipos de bases Wavelets que foram usadas nesse trabalho, que são: *Haar, Daubechies, Symlets, Coiflets e Biorthogonal* usadas na análise no capitulo 3.

2.3.1 Haar

A base de Haar é a primeira e a mais simples, servindo de protótipo para as demais transformadas, é uma sequência de funções ortogonais constantes por partes. Ela é descontínua, se parece com a função degrau, como mostra na figura (4).

Figura 4. Wavelets Haar

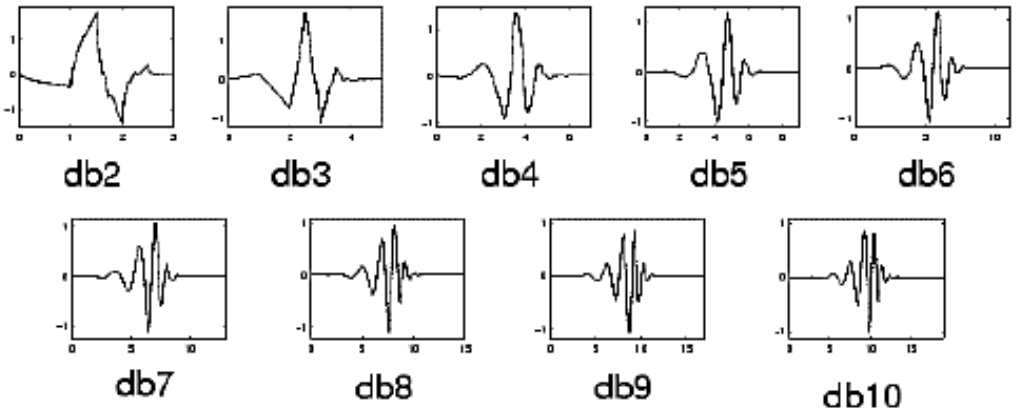


Fonte: (FONSECA, 2004)

2.3.2 Daubechies

As *wavelets Daubechies* (figura 5) levam o sobrenome de *Ingrid Daubecheis*[1], (FONSECA, 2004). Essa família é denominada por *dbN*, em que *db* é uma indicação do sobrenome e *N* é o número de sua ordem. Possui suporte compacto e gera bases ortogonais, ou seja, um erro no sinal de entrada não cresce com a transformação e a estabilidade numérica computacional é assegurada. Por outro lado não possuem fase linear.

Figura 5. Wavelets do tipo Daubechies



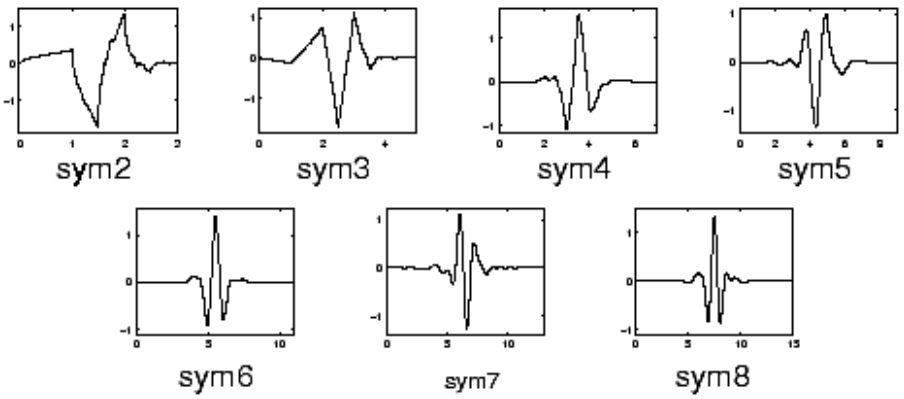
Fonte: (FONSECA, 2004)

2.3.3 Symlets

As *Symlets* (figura 6) também foram propostas por *Daubechies* como uma modificação à família *dbN*. Possuem propriedades similares e tendem a ser simétricas. É denominada por *SymN*, onde *Sym* é o nome e *N* é a ordem.

[1] Foi uma das mais importantes pesquisadoras no campo das *wavelets.* Ela descobriu as *wavelets* ortogonais com suporte compacto, tornando a análise discretas das *wavelets* possíveis.

Figura 6. Wavelets Symlet

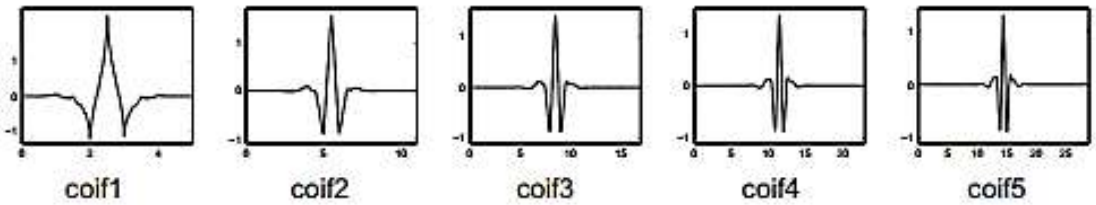


Fonte: (MISITI, MISITI, *et al.*)

2.3.4 Coiflets

Foi construída por *Daubechies* a pedido de *R. Coifman*, sendo mais simétrica do que o tipo *Symlets*. As funções *wavelet* tem 2N momentos iguais a 0 (zero) e função escalonamento com 2N-1 momentos iguais a 0 (zero). Ambas funções têm o comprimento de apoio 6N-1 (representada na figura 7).

Figura 7. Wavelet do tipo Coiflets



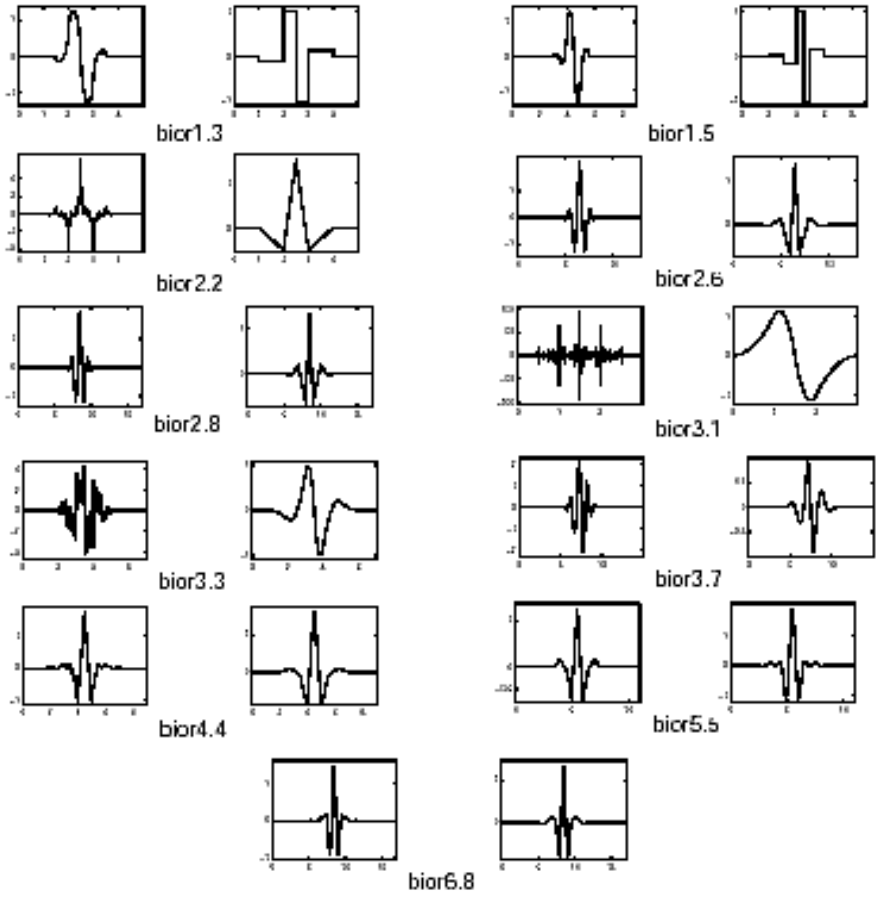
Fonte: (MISITI, MISITI, *et al.*)

2.3.5 Biortogonal

Esta família de *wavelets* tem suporte compacto e é simétrica, exibe a propriedade de fase linear, a qual é necessária para a reconstrução do sinal ou imagem. São usadas duas *Wavelets*, uma para a decomposição e a outra para a reconstrução, ao invés de apenas uma.

É denominada por *bior Nr.Nd*, onde *bior* representa o seu nome e *Nr* e *Nd* são a ordem dos filtros síntese e análise (representada na figura 8).

Figura 8. *Wavelets Biortogonal*



Fonte: **(FONSECA, 2004)**, modificado

2.4 ALGORITMOS DE COMPRESSÃO

Esta sessão descreve os aspectos dos algoritmos de compressão de imagem digital que foram utilizados nesse trabalho.

2.4.1 *Embedded Zerotree Wavelet* – (EZW)

O algoritmo de compressão *EZW* é um dos primeiros algoritmos baseado na transformada *wavelet*. Isso explica o porquê da palavra *wavelet* em seu nome. É um codificador conhecido por Zero-árvore e foi apresentado em um artigo por Shapiro (SHAPIRO, 1993). Originalmente ele foi concebido para funcionar com imagens 2D-sinais, mas, pode ser usado em outras dimensões de imagens.

O codificador *EZW* é baseado em codificação progressiva para comprimir uma imagem em um fluxo de bits com precisão cada vez maior, ou seja, quanto mais

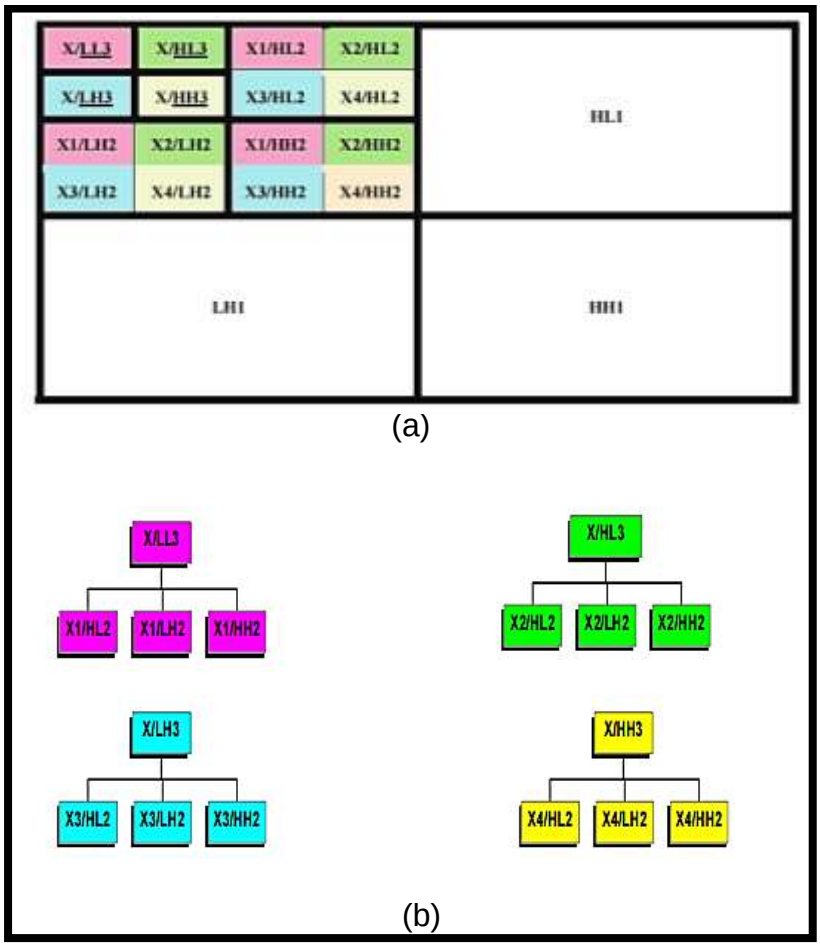
bits são adicionados à corrente, mais detalhes são inseridos na imagem recuperada. Codificação progressiva também é conhecida como codificação embutida, o que explica a palavra *embedded* em seu nome (VALENS, 1999). É usado o termo embutido para indicar que os dados comprimidos são ordenados de acordo com a importância visual. Com esse codificador, os dados comprimidos podem alcançar a taxa de bits desejada, se tratando de um esquema de compressão com perdas. Mas também é possível utiliza-lo num esquema sem perdas.

O codificador *EZW* baseia-se em duas importantes observações: 1) as imagens naturais, que têm um espectro passa-baixa. Quando é aplicada a transformada *wavelet*, a energia das sub-bandas diminui conforme aumenta a escala. Isso indica que os coeficientes *wavelet*, em média, são menores nas sub-bandas de baixas frequências, enquanto as sub-bandas de altas frequências adicionam apenas detalhes à imagem. 2) os coeficientes *wavelet* maiores são mais importantes que os coeficientes menores (VALENS, 1999).

Essas duas observações são examinadas pelo codificador *EZW*, o qual codifica coeficientes em ordem decrescente, em vários passes. A cada passo, o coeficiente da imagem é comparado com um limiar previamente estabelecido. Se um coeficiente é maior do que o limiar, ele é codificado e removido da imagem. Caso seja menor, é deixado na imagem para a próxima passagem. Após todos os coeficientes serem visitados, o limiar é reduzido e os coeficientes são novamente visitados e codificados. Esse processo se repete até que todos os coeficientes *wavelet* tenham sido completamente codificados ou outro critério seja satisfeito, como por exemplo, a taxa máxima de bits (VALENS, 1999).

A árvore é construída da seguinte forma: os coeficientes em diferentes sub-bandas estão relacionados. Exceto a sub-banda LL que tem quatro descendentes, qualquer coeficiente na menor sub-banda tem três descendentes nas próximas sub-bandas mais elevadas. Por exemplo, na Figura (9(a)), o coeficiente X/HH3 significa o coeficiente X na LL3 sub-banda. Este coeficiente possui três descendentes, os quais têm a mesma cor que este coeficiente. Estes coeficientes são X1/HL2, X1/LH2, e X1/HH2. Da mesma forma, o coeficiente de X/HL3 na sub-faixa de mais baixa frequência tem três descendentes no resto das sub-bandas do mesmo palco. Estes coeficientes são X2/HL2, X2/LH2 e X2/HH2. Além disso, o coeficiente de X/LH3 tem três filhos: X3/HL2, X3/LH2 e X3/HH2; e o coeficiente X/HH3 tem três descendentes: X4/HL2, X4/LH2 e X4/HH2. Isto é mostrado na Figura (9(b)) (ALREGIB, 2000).

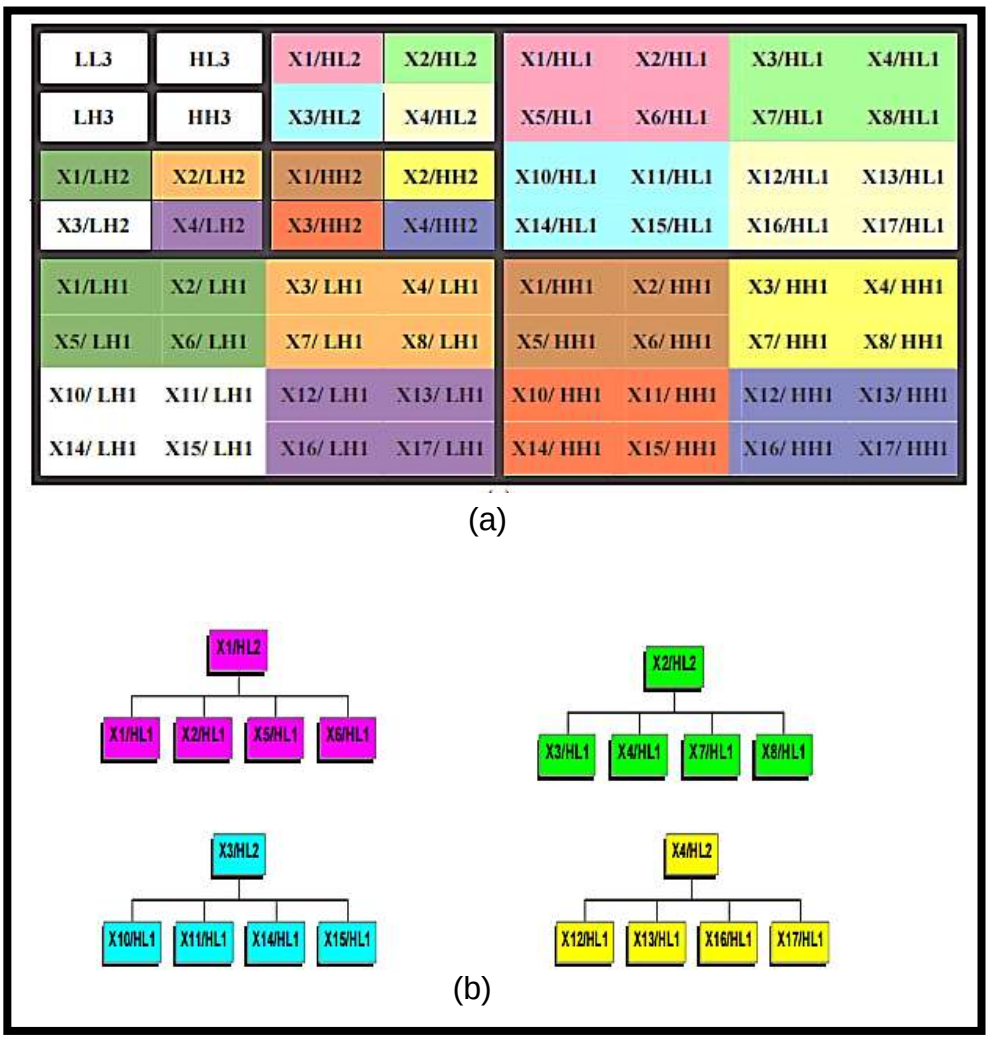
Figura 9. (a) Imagens de sub-banda. (b) A árvore para cada coeficiente na sub-faixa de menor frequência.



Fonte: (ALREGIB, 2000)

Este foi um caso especial para os coeficientes na menor sub-banda que tem apenas três descendentes. No entanto, para os outros coeficientes, cada um tem quatro descendentes. Para ilustrar isso, vamos olhar novamente para a imagem sub-banda resultante da transformada *wavelet* para uma imagem de 8×8, Figura (10(a)). Por exemplo, o coeficiente de X1/HL2 tem quatro coeficientes: X1/HL1, X2/HL1, X5/X6 HL1 e X6/HL1; o coeficiente X2/HL2 tem quatro coeficientes: X3/HL1, X4/HL1, X7/HL1 e X8/HL1; o coeficiente X3/HL2 tem quatro coeficientes: X10/HL1, X11/HL1, X14/HL1 e X15 /HL1; e o coeficiente X4/HL2 tem quatro coeficientes: X12/HL1, X13/HL1, X16/HL1 e X17/HL1. A mesma coisa se aplica para os coeficientes de descanso na sub-bandas LH2 e HH2, onde cada coeficiente tem quatro coeficientes nas sub-bandas LH1 e HH1, respectivamente. O quad-árvore para estes quatro coeficientes são apresentados na Figura (10(b)) (ALREGIB, 2000).

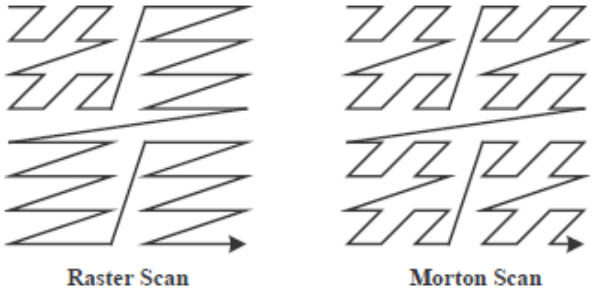
Figura 10. (a) Imagens de sub-banda resultantes da transformada wavelet para uma imagem 8 × 8. (b) A árvore para cada coeficiente nas bandas HL2, LH2 e HH2.



Fonte: (ALREGIB, 2000)

Existe duas técnicas para digitalizar os coeficientes de uma imagem de sub-banda: a primeiro é “*Raster Scan*” e a outra é “*Morton Scan*”, figura (11):

Figura 11.Técnicas de varredura



Fonte: (TENÓRIO, 2003)

Para determinar o limiar inicial T_0 (equação (5)), é escolhido tal que:

$$T_0 = 2^{\lfloor \log_2 x_{max} \rfloor} \quad (5)$$

onde x_{max} corresponde ao valor máximo na imagem.

Os coeficientes passam por um processo de Quantização por Aproximação Sucessiva (SAQ), onde são executados dois passos, o dominante e o subordinado. A passagem dominante codifica todos os coeficientes com respeito ao limiar, e a passagem subordinado quantiza todos os coeficientes identificados como significantes, ou seja, aqueles maiores que o limiar, em módulo (TENÓRIO, 2003).

A cada passagem do dominante e subordinado, o limiar é reduzido por um fator de 2 (equação (6)),

$$T_i = T_{i-1} / 2 \quad (6)$$

São usados quatros símbolos durante a passagem dominante em que todos os coeficientes são codificados:

- POS (Positivo Significante): o coeficiente é maior ou igual ao limiar atual, com sinal positivo, e não foi codificado nas iterações anteriores;
- NEG (Negativo Significante): o coeficiente é maior ou igual que o limiar atual, com sinal negativo;
- IZ (Zero Isolado): o coeficiente é menor que o limar atual, mas não é a raiz de uma *zerotree*, pois em alguns de seus desentendes existem coeficientes maiores que o limiar;
- ZTR (Raiz *Zerotree*): o coeficiente e todos os seus desentendes tem magnitude menor que o limiar, então o coeficiente é chamado de raiz *zerotree* (TENÓRIO, 2003).

Cada passagem dominante é seguida por uma passagem subordinada. Durante uma passagem subordinada, a largura do degrau do quantizador, que define o intervalo de incerteza para a magnitude real do coeficiente, é dividida pela metade. É usado um alfabeto binário para cada magnitude encontrada na lista subordinada. O símbolo “1” está indicando que o valor verdadeiro do coeficiente está

na metade superior do intervalo de incerteza ou o símbolo “0” indicando que o valor verdadeiro está na metade inferior (GUSMÃO, 2002).

O processo continua alternando entre a passagem dominante e a subordinada, enquanto o limiar vai sendo reduzido pela metade, até que uma determinada condição seja satisfeita.

2.4.1.1 Exemplo do Algoritmo EZW

Nesta seção será apresentado um exemplo simples do algoritmo *EZW*. Considerando uma Transformada *Wavelet* de três escalas simples de uma imagem de tamanho 8x8. A matriz de valores é mostrada na Figura 12. O coeficiente máximo é 63, no qual o resultado do limiar inicial é de T_0= 32.

Figura 12. Coeficiente wavelet de uma imagem 8x8

63	-34	49	10	7	-13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

Fonte: (TENÓRIO, 2003)

Figura 13. Resultado do primeiro passo dominante

POS	NEG	POS	ZTR	ZTR	ZTR	x	x
IZ	ZTR	ZTR	ZTR	ZTR	ZTR	x	x
ZTR	IZ	x	x	x	x	x	x
ZTR	ZTR	x	x	x	x	x	x
x	x	ZTR	POS	x	x	x	x
x	x	ZTR	ZTR	x	x	x	x
x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x

Fonte: (TENÓRIO, 2003)

A Figura 13 mostra o resultado do primeiro passo dominante. A magnitude 63 é positivo e é maior que o limiar T_0=32, então é gerado o símbolo positivo significante **(POS)**. A magnitude -34 é negativo e maior que o limiar inicial, assim sendo identificado como negativo significante **(NEG)**. As magnitudes -31 e 14 foram identificados como zeros isolados **(IZ)**, pois são menores que o limiar inicial, mas em sua árvore de descendência existe coeficiente maior que o limiar, que é o valor 47 que é um positivo significativo **(POS)**. Aqueles coeficientes que foram identificados como zerotree **(ZTR)** são menores que o limiar, mas sem coeficientes significantes em sua arvore de descendentes (TENÓRIO, 2003).

Durante a primeira passagem dominante, utilizando o limiar 32, foram identificados quatro coeficientes significantes, que são retirados da matriz para que não sejam novamente codificados. Esses coeficientes serão refinados na primeira passagem subordinada. Antes da primeira passagem subordinada, o intervalo de incerteza para as grandezas de todos os coeficientes significados, e [32, 64). Na primeira passagem as grandezas são refinadas no intervalo de [32,48), que será codificada com o símbolo “0”, ou no intervalo de [48,64), codificada com o símbolo “1”. Sendo assim o limite de decisão é a magnitude 48. A Tabela 1 mostra o resultado do primeiro passo subordinado e o valor de reconstrução dos coeficientes.

Tabela 1 Primeiro passo subordinado

Coeficiente Magnitude	Símbolo	Reconstrução
63	1	48
34	0	-40
49	1	48
47	0	40

O processo continua para a segunda passagem dominante, onde o limiar agora vale 16, sendo a metade do valor original, e apenas os coeficientes ainda não encontrados a ser significativos são verificados. As Figuras 14 e 15 mostram o resultado dessa passagem dominante.

Figura 14. Matriz de coeficientes após p primeiro passo dominante

0	0	0	10	7	-13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	0	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

Fonte: **(TENÓRIO, 2003)**

Figura 15.Resultado do segundo passo dominante

IZ	ZTR	x	x	x	x	x	x
NEG	POS	x	x	x	x	x	x
ZTR	ZTR	ZTR	ZTR	x	x	x	x
ZTR	ZTR	ZTR	ZTR	x	x	x	x
x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x

Fonte: (TENÓRIO, 2003)

A lista subordinada agora contém, em ordem, a magnitude (63, 34, 49, 47, 31, 23), antes de passar esta subordinado, os intervalos são de três incerteza [16,32),

[32, 48) e [48, 64), respectivamente. A Tabela 2 mostra o resultado desse segundo passo.

Tabela 2 Resultado do segundo passo subordinado

Coeficiente Magnitude	Símbolo	Reconstrução
63	1	56
34	0	-32
49	0	48
47	1	40
31	1	-24
23	0	16

O processamento continua alternando entre passes dominantes e subordinados até que toda a imagem seja codificada ou alguma condição previamente estabelecida seja alcançada.

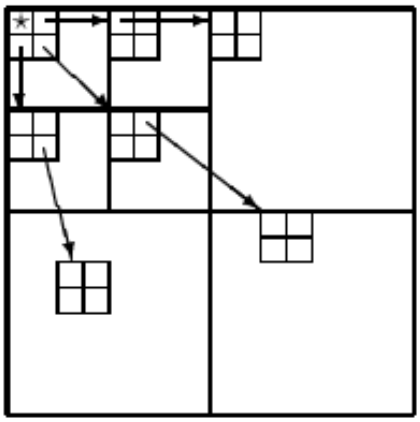
2.4.2 Set Partitioning in Hierarchical Trees – (SPIHT)

É uma extensão do algoritmo *EZW* que também utiliza o conceito de árvore particionada e foi proposto por Said e Pearlman (SAID e PEARLMAN, 1996). Há vantagens em relação ao *EZW*, pois o *SPIHT* é extremamente mais rápido, a forma como os subconjuntos de coeficientes é dividida e como a informação é transmitida são diferentes do algoritmo *EZW* e alguns dos resultados obtidos são melhores devido à sua propriedade de divisão que aumenta a sua potência de compressão (GUSMÃO, 2002).

A estrutura de dados usada é semelhante à usada no algoritmo *EZW*, embora não seja a mesma. A figura 16 mostra como a árvore é definida, que é chamada árvore espacial, define a relação espacial na pirâmide hierárquica. A árvore é definida de tal maneira que cada nó tenha 4 ou nenhum descendente, formando um grupo de 2x2 pixel adjacentes. Cada nó da árvore corresponde a um pixel, e é identificada pela coordenada deste pixel. Seus adjacentes diretos correspondem aos pixels de mesma orientação espacial no nível mais fino da pirâmide. Como mostra a Figura 16, as setas são orientadas na direção dos pais para os filhos. Como no algoritmo *EZW*, os pixels no nível mais alto da pirâmide são as raízes das árvores e também são agrupados em 2x2 pixel adjacentes, no entanto o que diferencia é como

os seus descendentes se ramificam, em cada grupo da raiz, um dos pixels (identificado pela estrela) não tem descendentes (SAID e PEARLMAN, 1996).

Figura 16. Árvore de orientação espacial
da dependência entre pai-filho



Fonte: (SAID e PEARLMAN, 1996)

As árvores são divididas em quatros tipos de conjuntos, que são coordenadas dos coeficientes (SAYOOD, 2006):

- *O* (i,j): conjunto com as coordenadas de todos os filhos do nó (i,j);
- *D* (i,j): coordenadas de todos descendentes do nó (i,j);
- *H* (i,j): coordenadas de todas as raízes da árvore (nível mais alto da pirâmide);
- *L* (i,j) = *D*(i,j) – *O*(i,j): coordenadas de todos os descendentes do nó i,j exceto a de seus filhos.

Um conjunto *D*(i,j) ou *L*(i,j) é dito significativo se qualquer um de seus coeficientes integrantes indicar magnitude maior do que o limiar. Esses limiares usados para testar a significância são potências de 2, então, o algoritmo *SPIHT* envia a representação binária do valor inteiro dos coeficientes wavelet.

Esse algoritmo faz uso de três listas: LIP (Lista de pixels insignificantes), LIS (Lista de conjuntos insignificantes) e LSP (Lista de pixels significantes). A lista LIP contém coeficiente individual. Enquanto o LIS é usada para examinar os conjuntos de descendentes insignificantes dos coeficientes da árvore. Já o LSP contém as coordenadas dos coeficientes que tem uma magnitude maior do que o limiar (SILVA, 2005). Também usa dois tipos de símbolos: O “1” que é para indicar que o coeficiente é significante e o “0” identifica o coeficiente insignificante.

O algoritmo começa por determinar o valor inicial do limiar (T_0), equação (7):

$$T_0 = 2^n \tag{7}$$

n é dado por (equação (8)) :

$$n = [\log_2 c_{max}] \tag{8}$$

Sendo que c_{max} é a magnitude máxima dos coeficientes a serem codificados.

A lista LIP é inicializada com o conjunto *H*. Os elementos de *H* que possuem descendentes também são colocados nesta lista como entrada de tipo *D*. A LSP esta inicialmente vazia. Em cada passagem, primeiro é processado os membros da LIP, depois os membros da LIS, essa etapa é essencialmente a codificação de mapa de significância, em seguida é processado os elementos de LSP na etapa de refinamento (SAYOOD, 2006).

O procedimento da LIP equivale em examinar cada coordenada contida nela. Se o coeficiente nessa coordenada for insignificante (menor do que 2^n), é transmitido “0”. Já se o coeficiente for significativo (maior ou igual a 2^n), é transmitido um “1” sendo seguido por um bit representativo do sinal do coeficiente (“0” para positivo, “1“ para negativo). Em seguindo o coeficiente será móvito para a LSP.

Depois de examinar todos os componentes da LIP, começa a examinar os conjuntos em LIS. Caso o conjunto da coordenada não é significativo (todos os elementos menores do que 2^n) é transmitido “0”, mas se for significativo (com pelo menor um elemento maior ou igual a 2^n), é transmitido “1”, e o que será feito em sequência vai depender se a o conjunto é do tipo *D* ou *L*. Se o conjunto for do tipo *D*, verifica os quadros coeficientes (filhos) da qual as coordenadas estão em *O* (i,j). Para cada coeficientes significativo transmite-se “1”, o sinal do coeficiente, logo a coordenada dele é movida para a LSP. Para o restante é transmitido “0”, e suas coordenadas movida para LIP. Depois que as coordenadas de *O* (i,j) foram removidas do conjunto, o que resta é o conjunto *L* (i,j). Se o conjunto estiver vazio é removida da lista, do contrário é movida para o fim da LIS com a etiqueta *L* (SAYOOD, 2006).

Uma vez que todos os componente da LIS foram processador, segue-se para a etapa de refinamento. Nesta etapa é examinado cada coeficiente que estava na LSP e a saída é o (n+1) bit menos significativo de $|c_{ij}|$. Os coeficientes de LSP no

passo corrente não são processador neste ponto, pois foi declarado como significativos, o decodificador foi informado do valor do (n+1) bit menos significativo de $|c_{ij}|$.

O processo de codificação pode continuar, decrementando n de 1 e repetindo os passos explicados. Ver o funcionamento deste algoritmo em um exemplo.

2.4.2.1 Exemplo do algoritmo SPIHT

Neste exemplo (figura 17) serão realizadas 3 passagens no codificador e será gerado o fluxo de bits transmitido, e achará a decodificação deste fluxo de bits.

Figura 17 Exemplo SPIHT

26	6	13	10
−7	7	6	4
4	−4	4	−3
2	−2	−2	0

Fonte: (SAYOOD, 2006)

Passo 1) C_{max} = 26. Logo, n = 4 e T_0= 16. As três listas no codificador são:
LIP: {(0,0)→ 26; (0,1)→6; (1,0)→-7; (1,1)→7}
LIS: {(0,1)*D*; (1,0)*D*; (1,1)*D*}
LSP:{ }

Inicialmente, analisando LIP, o coeficiente (0,0) é maior que 16, então é significativo, transmitindo assim “1” e “0” indicando que o coeficiente é positivo e suas coordenadas são movidas para LSP. Os 3 coeficientes seguintes de LIP são insignificantes com relação a 16, portanto é transmitido “0” para cada coeficiente e eles permanecem em LIP.

Concluída a análise de LIP, o próximo passo é examinar LIS. Observando para a descendência de (0,1) (com etiqueta (0,1)*D*), que são coeficientes 13, 10, 6 e 4, nenhum deles é significativo com relação a 16, sendo assim transmite-se um “0”. O mesmo acontece para os conjuntos (1,0)D e (1,1)D, logo, transmite-se um “0” para cada um deles. Como não há coeficientes de LSP do passo anterior, não é feito o refinamento nesse passo. É transmitido um total de 8 bits no final deste passe (10000000), e a situação das listas no fim deste primeiro passo é:

LIP: {(0,1)→6; (1,0)→-7; (1,1)→7}

LIS: {(0,1)D; (1,0)D; (1,1)D}

LSP:{(0,0)→26}

Passo 2) feito o decremento de 1 em n, n= 3 e T= 8.

Começando novamente examinando LIP, existem 3 elementos no LIP, cada um deles é insignificante em relação a esse limiar, transmitindo um “0” para cada um. Seguindo, agora o exame da LIS, o primeiro elemento é o conjunto que contém os descendentes do coeficiente (0,1)*D,* deste conjunto, o 13 e o 10 são significativos a este valor do limiar, portando o conjunto (0,1)*D* é significativo, sendo sinalizado com um “1”, agora examinando os coeficientes deste conjunto, o 13 que é significativo e positivo, será transmitido um “1” seguido por um “0”. Sendo a mesma coisa para o valor 10. Em seguida, moverá as coordenadas destes dois para LSP. Os próximos dois descendentes ambos são insignificantes a este nível, assim moverá para LIP e transmite um “0” para cada um. Observando para os outros elementos de LIS, ambos são insignificantes, portanto transmite um “0” para cada um.

No passo de refinamento é examinado o conteúdo da LSP do passe anterior, existe apenas um elemento valor 26, transmitindo um “1” e concluindo este passe. É transmitido 13 bits (0001101000001). A lista no final da segunda passagem é:

LIP: {(0,1)→ 6, (1,0)→-7, (1,1)→7, (1,2)→6, (1,3)→4}

LIS: {(1,0)D, (1,1)D}

LSP:{(0,0)→26, (0,2)→13, (0,3)→10}

Passo 3) n = 2 e T= 4

Como o limiar agora é menor, há consideravelmente mais coeficientes que são classificados significativos. Seguindo a orientação do processo, serão enviados 26 bits (10111010101101100110000010). A condição das listas no final da terceira passagem.

LIP:{ (3,0)→ 2, (3,1)→-2, (2,3)→-3, (3,2)→-2, (3,3)→0}

LIS: { }

LSP:{(0,0)→26, (0,2)→13, (0,3)→10, (0,1)→6, (1,0)→-7, (1,1)→7, (1,2)→6, (1,3)→4, (2,0)→4, (2,1)→-4, (2,2)→4}

No processo do decodificador também começa com as mesmas listas que o codificador.

LIP: {(0,0); (0,1); (1,0); (1,1)}

LIS: {(0,1)*D*; (1,0)*D*; (1,1)*D*}

LSP:{ }

Sabe-se que o t_0= 16, ao receber os resultados da primeira passagem (10000000), pode-se ver que o primeiro elemento da LIP é significativo e positivo e nenhum outro coeficiente é significativo a este nível. Usando o mesmo procedimento de reconstrução que em *EZW*, reconstroem-se os coeficientes nesta fase como:

Figura 18 Exemplo - decodificador SPIHT

24	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Fonte: (SAYOOD, 2006)

E as listas podem ser atualizadas como:

LIP: {(0,1), (1,0), (1,1)}

LIS: {(0,1)D, (1,0)D, (1,1)D}

LSP:{(0,0)}

Para a segunda passagem, decrementamos n por 1 e examinamos o fluxo de bits transmitido: 0001101000001. Os 3 bits primeiros são “0” e existem apenas três entradas no LIP, todas as entradas no LIP são ainda insignificantes. Os próximos 9 bits dá informações sobre os conjuntos em LIS. O quarto bit recebido é “1”, isto significa que o conjunto com raiz na coordenada (0,1) é significativo. Uma vez que este conjunto é do tipo D, os bits seguintes referem-se aos seus descendentes. A sequência 101000 indica que os dois primeiros descendentes são significativos neste nível e positivos e os dois últimos são insignificantes. Portanto, movem-se os dois primeiros descendentes para LSP e os dois últimos para LIP. Também podem aproximar estes dois coeficientes significativos em nossa reconstrução por 1.5×2^3 =

12, removendo (0,1) D de LIS. Os dois bits seguintes ambos são “0”, que indica os dois conjuntos restantes insignificantes. O bit final corresponde ao passo de refinamento, “1”, então atualiza a reconstrução do coeficiente (0,0) para 24 + 2^{n-1} = 28. A reconstrução nesta da terceira passagem fica, (figura 19):

Figura 19 Reconstrução da terceira passagem dominante

28	0	12	12
0	0	0	0
0	0	0	0
0	0	0	0

Fonte: (SAYOOD, 2006)

E as listas são as seguintes:

LIP:{ (0,1), (1,0), (1,1),(1,2),(1,3)}

LIS: {(0,1)D, (1,1)D}

LSP: {(0,0),(0,2),(0,3)}

Para a terceira passagem novamente decrementamos n, que agora é 2, dando um valor limiar de 4. Decodificando o fluxo de bits gerado durante a terceira passagem (10111010101101100110000010), reconstrução final recuperado (figura 20) e as listas:

Figura 20 Reconstrução final

26	6	14	10
−6	6	6	6
6	−6	6	0
0	0	0	0

Fonte: (SAYOOD, 2006)

LIP:{ (3,0), (3,1)}

LIS: {}

LSP: {(0,0),(0,2),(0,3),(0,1),(1,0),(1,1),(1,2),(2,0),(2,1),(3,2)}

Nesta fase, não tem quaisquer conjuntos deixados em LIS e simplesmente é atualizado os valores dos coeficientes.

2.4.3 Spatial-orientation Tree Wavelet - (STW)

O algoritmo *STW* conhecido como algoritmo de Árvore de Orientação Espacial *Wavelet*.

É essencialmente parecido com o algoritmo *SPIHT*, o que diferencia é que o *SPHIT* é mais cuidadoso no que se refere a sua organização de codificação de saída na tabela 3, de modo que apenas um bit seja emitido de cada vez. Por exemplo, pra a transição $I_R \rightarrow S_R$, que é codificada como 1 0 na tabela, *SPIHT* emite um 1 primeiro e depois emite um 0. Mesmo se o orçamento de bit for esgotado antes que o segundo bit possa ser saída, O primeiro bit de 1 indica que há um novo valor significativo (WALKER, 2000).

Tabela 3. Código para Transições De Estado, ● indica que $S_V \rightarrow S_V$, a transição é certo

Old / New	I_R	I_V	S_R	S_V
I_R	00	01	10	00
I_V		0		1
S_R			0	
S_V				●

Fonte: (WALKER, 2000)

A diferença entre *STW* e *EZW* é que *STW* usa uma abordagem diferente para codificar a informação de árvore zero. *STW* usa um modelo de transição de estado. De um limiar para o outro, as localizações de valores de transformação passam por transições de estado (KUMAR e CHAUDHARY, 2013).

3 SOFTWARES UTILIZADOS

Nesta seção apresentados os principais softwares que foram utilizados para a realização deste trabalho.

3.1 MATLAB

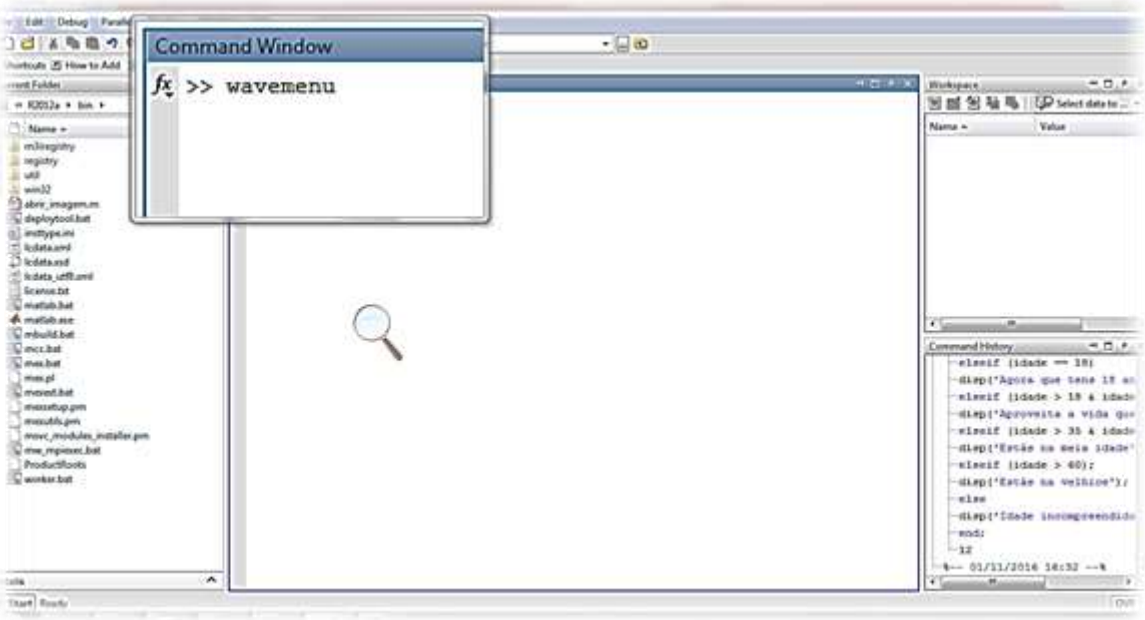
O *Matlab* do inglês (*Matrix Laboratory),* desenvolvido pelo *“The Mathworks, Ins”*, fundado em 1984 em *Natick, Massachusett*, é um *software* computacional mundialmente conhecido como uma ferramenta para solucionar problemas matemáticos, científicos e tecnológicos. Inicialmente era apenas um software que realizava operações matemáticas sobre matrizes, que ao longo dos anos foi se tornando um programa de sistema computacional flexível e bastante útil para resolver problemas técnicos (NOBRE, 2005).

A sua linguagem de programação é escrita em linguagem matemática diferentes dos outros softwares, assim dificilmente seus programas serão executados em um ambiente que seja fora do *Matlab*. Possui uma grande quantidade de bibliotecas auxiliares (*Toolboxes*) que essa ferramenta proporciona a otimização do tempo gasto na realização das tarefas, na qual o usuário poderá utilizar muitas funções já definidas (BECKER, SILVA, *et al.*, 2010). Essas funções podem ser usadas para resolver problemas como: processamento de imagens em especial compressão de imagens, manipulação algébrica, redes neurais, processamento de sinais, entre outros.

3.1.1 Ambiente Matlab

A Figura 21 mostra a tela unicial do Matlab. *Command Window* é o local onde são inserios os comandos, utilizando a tecla *Enter*. Nesse trabalho é utilizado o comando *Wavemenu*.

Figura 21.Tela inicial MatLAb



Fonte: Print screen do programa MatLab

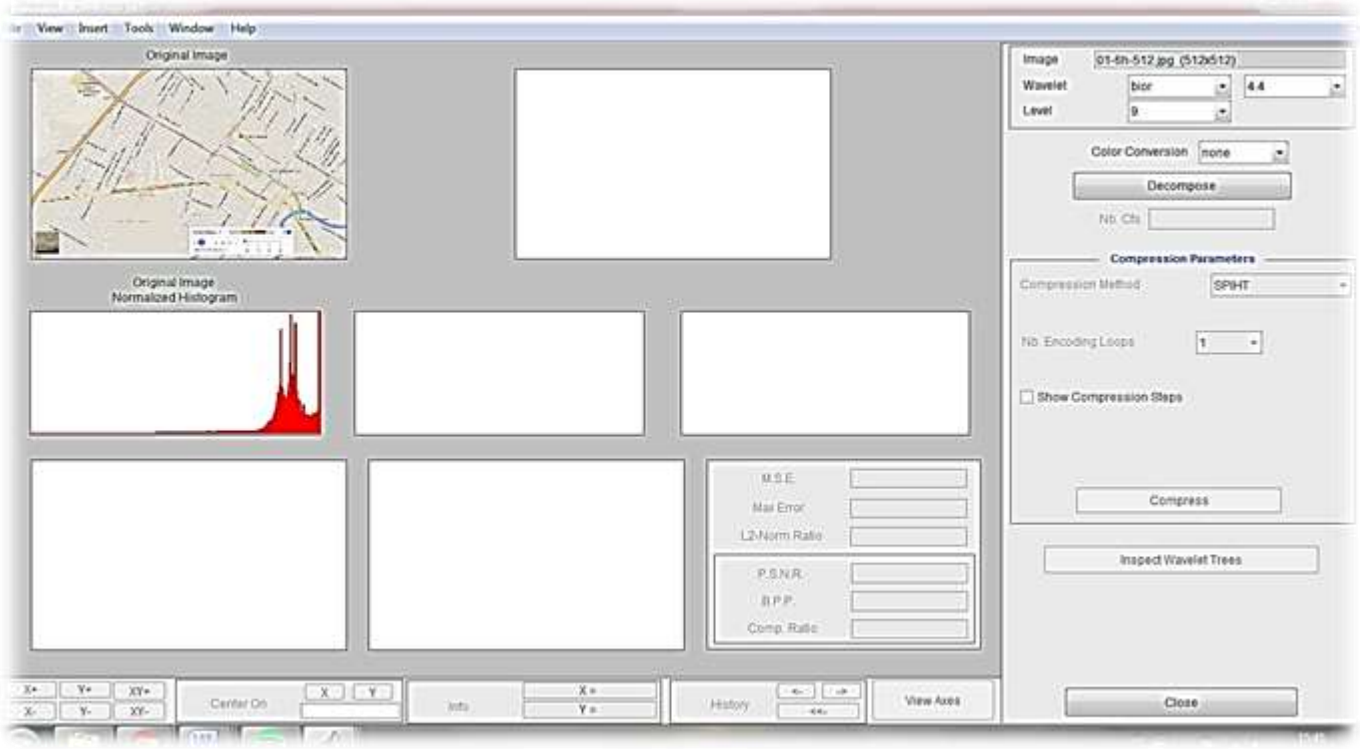
Ao digitar o comando (*wavemenu*) aparecerá uma nova janela *Menu* (Figura 22) com uma lista de comandos com vários tipos de funções envolvendo *wavelet*.

Figura 22. Menu principal da caixa de ferramentas Wavelet

Fonte: Print screen do programa MatLab

A opção *True Compression 2-D* direciona ao ambiente onde serão realizadas as análises para o desenvolvimento do trabalho com a imagem selecionada.

Figura 23. Ambiente para ser realizada a compressão de imagem



Fonte: Print screen do programa MatLab

Nesse ambiente (Figura 23) são feitos os testes com as famílias *Wavelets* e os algoritmos de parâmetros de compressão utilizados nesse trabalho.

3.2 EXCEL

O Excel é um dos programas de planilhas de dados eletrônica mais populares do mundo que foi desenvolvido pela empresa *Microsolf*. Este programa é usado para a realização de inúmeras tarefas como: Cálculos simples e complexos, estatísticas, elaboração de gráficos, relatórios e entre outras atividades relacionadas à rotina doméstica, administrativa e empresarial.

Neste trabalho a ferramenta Excel foi utilizada para a organização em tabelas dos resultados obtidos com os testes realizados. Como também, para a construção dos gráficos gerados a partir desses testes.

4 ESTES E DISCUSSÕES DOS RESULTADOS

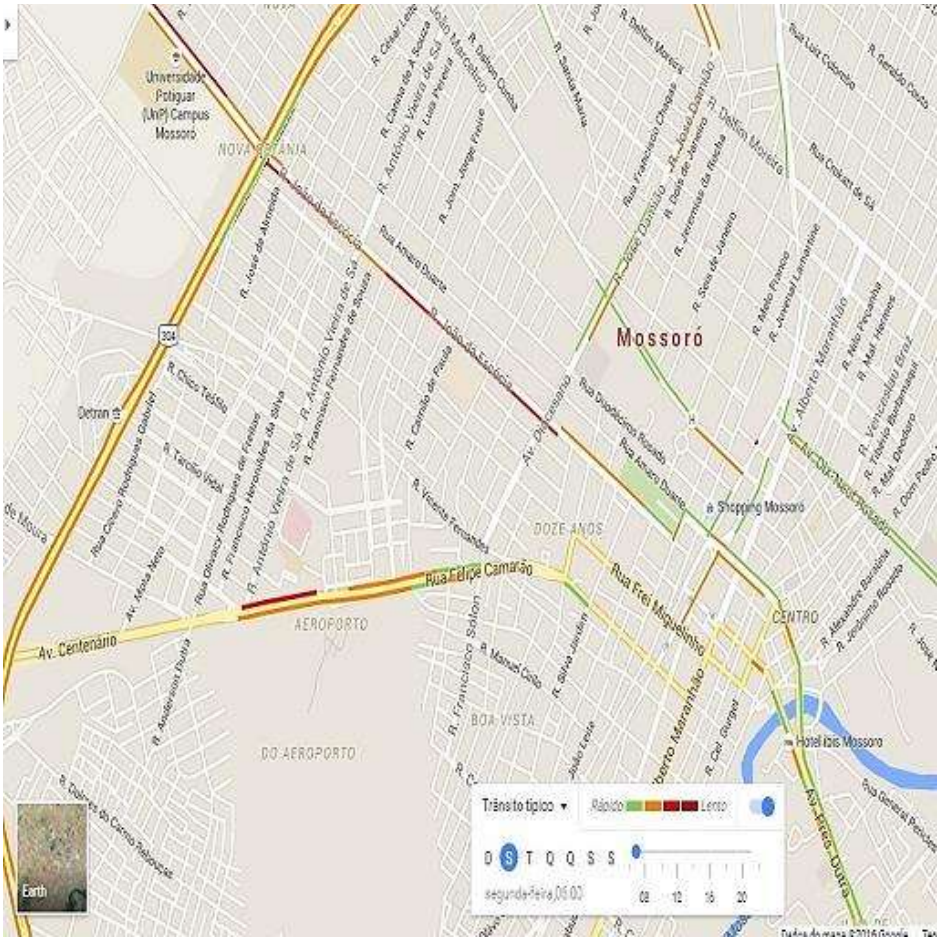
Neste capítulo, apresentaremos com base em nossos objetivos a análise dos métodos e das bases de compressão de imagem, através do uso de transformada *Wavelets* com o auxílio da ferramenta *Matlab*. É importante salientar que utilizamos a opção *Nb.Encoding Loops*[2] referente a 15, para uma resolução equilibrada na qualidade da imagem. Deste modo, mostraremos inicialmente a análise dos métodos *EZW, SPIHT* e *STW*, nas cinco bases *Wavelets*, de acordo com as métricas de erros *MSE, PSNP* e *Comp. Ratio* para estabelecer uma base formal de comparação entre as bases de *waveletes* e também entre os algoritmos de compressão. Em seguida, faremos a comparação de ambos na tentativa de identificar qual método e qual base *Wavelets* são mais eficazes na compressão de imagem nesta aplicação.

3.1 IMAGEM DE TESTE

A Figura 24 representa a imagem escolhida para a realização dos testes, com dimensão 512x512 pixels. A imagem obtida foi capturada do *google maps*, que é um serviço gratuito fornecido pela empresa *Google*, possui uma interface rica e intuitiva, tem um enorme acesso a uma base de dados contendo inúmeras imagens de satélite, mapas de cidades, bairros, ruas e avenidas do mundo.

[2] Define o número de codificação. O aumento leva a uma melhor recuperação, mas a taxa de compressão piora.

Figura 24 Tráfego de transito de Mossoró/RN



Fonte: (GOOGLE MAPS)

4.1.1 Análise do método EZW

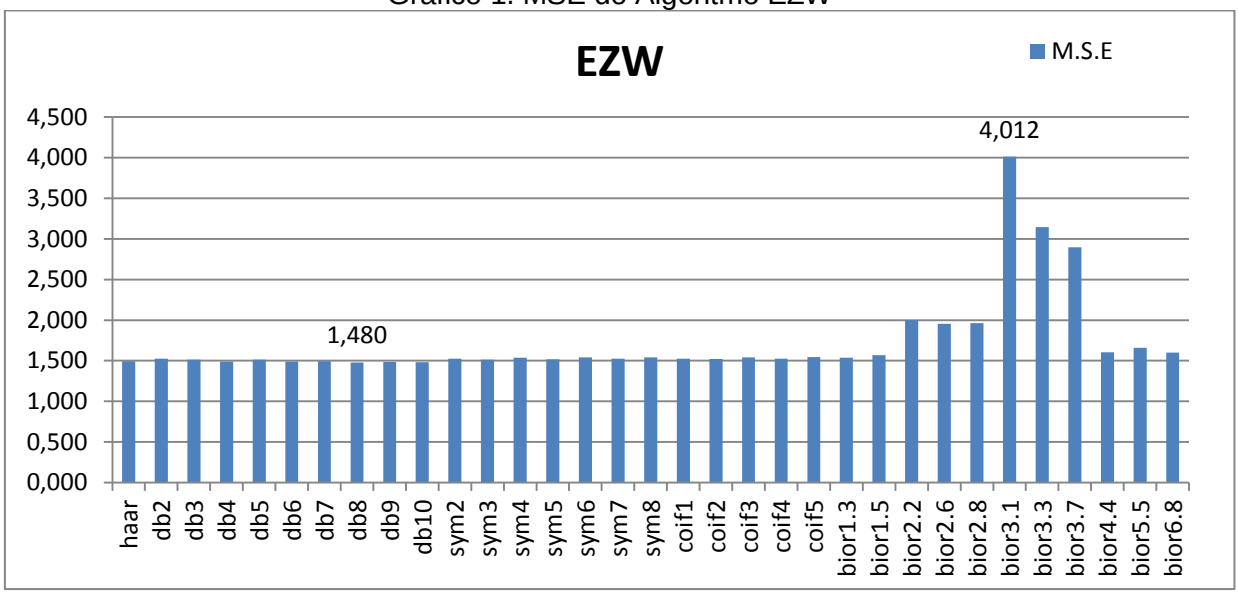
Nesta seção apresentaremos a análise do método *EZW*, em todas as bases *wavelets*, de acordo com os parâmetros de compressão.

4.1.1.1 Resultado Média dos Erros Quadráticos (MSE)

O MSE nos diz que quanto maior o valor obtido, pior será o desempenho da base *Wavelet*, pois será maior o Erro Médio Quadrático. Assim sendo, quanto menor o Erro, melhor será o desempenho da base.

Podemos observar no Gráfico 1, que o valor do *MSE* máximo é de 4,012 obtido com a base bior3.1, alcançando assim um pior desempenho. O valor mínimo é de 1,480 que foi obtido com a base *bd8*, mostrando que alcançou o melhor desempenho. Lembramos que os resultados completos estão nas tabelas em Anexos.

Gráfico 1. MSE do Algoritmo EZW



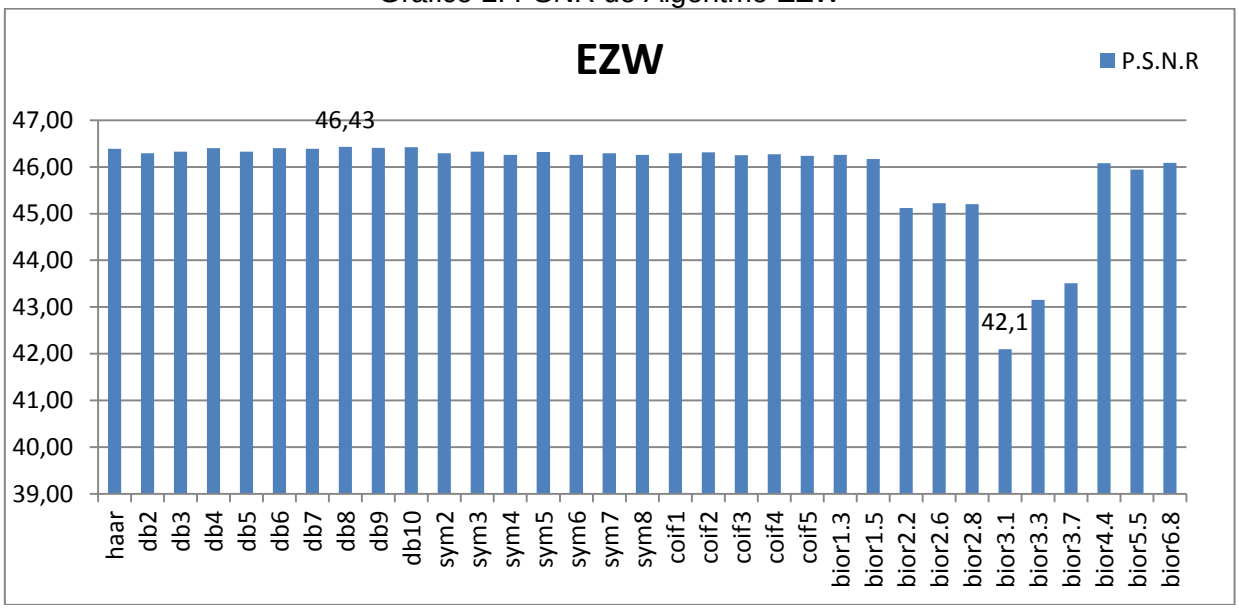
Fonte: elaborado pelo autor

4.1.1.2 Resultado Relação Sinal Ruído de Pico (PSNR)

O PSNR nos diz que quanto maior o valor obtido melhor será o desempenho da base *Wavelet*. Assim sendo, quanto menor o valor obtido pior será o desempenho da base *Wavelet*.

Podemos observar, no Gráfico 2, o valor de *PSNR* máximo foi 46,43 que pertence a base db8, alcançando um melhor desempenho da base. O valor mínimo obtido foi de 42,1, da base *bior3*.1, obtendo assim o pior desempenho. Lembrando que os resultados completos estão nas tabelas em Anexo.

Gráfico 2. PSNR do Algoritmo EZW



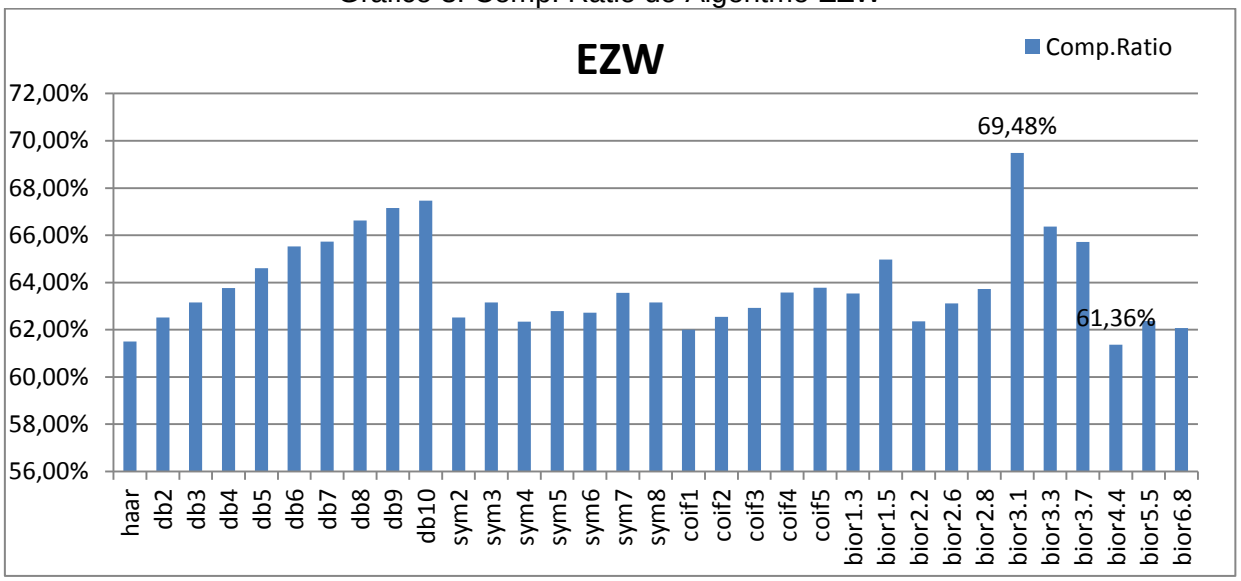
Fonte: elaborado pelo autor

4.1.1.3 Resultados da Relação de compressão

A relação de compressão indica que a imagem comprimida é armazenada utilizando uma porcentagem do tamanho inicial de armazenamento. Assim quanto menor for o resultado, menor será o tamanho ocupado pela imagem.

Podemos observar no Gráfico 3 que a base *bior4.4* utiliza apenas 61,36% do tamanho inicial de armazenamento alcançando, assim, um melhor resultado. Enquanto a base *bior3.1* utilizou 69,48% do tamanho inicial de armazenamento, obtendo um pior desempenho.

Gráfico 3. Comp. Ratio do Algoritmo EZW

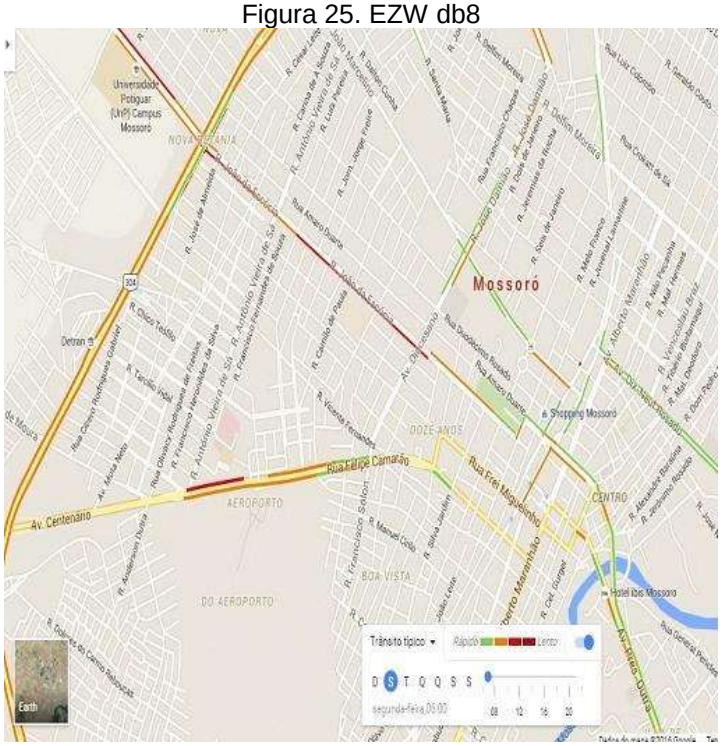


Fonte: elaborado pelo autor

Ao analisar o método *EZW*, nas cinco bases *wavelets*, de acordo com os parâmetros *MSE*, *PSNR* e *comp*. *Ratio*, pudemos perceber que a melhor base é a *bd8* que pertence a família da base *Daubechies*.

Na sequência, a Figura 25 mostra a imagem reconstruída após o processo de compressão e descompressão. Mesmo com a relação de compressão um pouco elevada.

Figura 25. EZW db8



4.1.2 Análise do método SPIHT

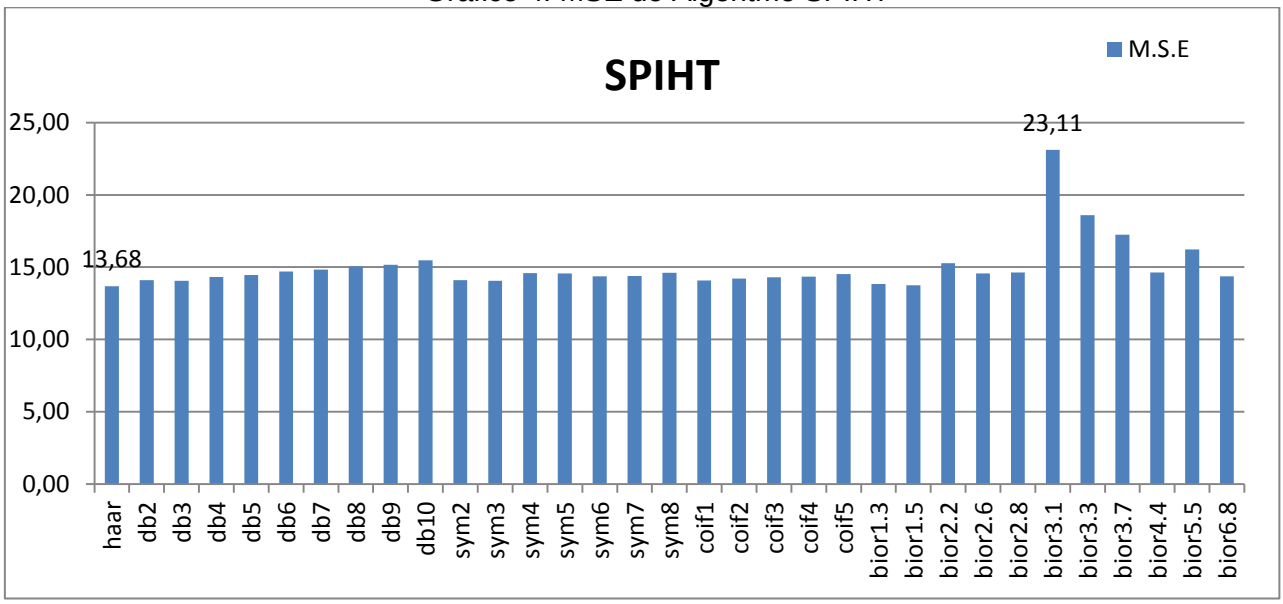
Nesta seção apresentaremos a análise do método *SPIHT*, em todas as bases *wavelets*, de acordo com os parâmetros de compressão.

4.1.2.1 Resultado da Média dos Erros Quadráticos (MSE)

O *MSE* nos diz que quanto maior o valor obtido, pior será o desempenho da base *Wavelet*, pois será maior o Erro Médio Quadrático. Assim sendo, quanto menor o Erro, melhor será o desempenho da base.

Podemos observar no Gráfico 4, que o valor do *MSE* máximo é de 23,11 obtido com a base *bior3.1*, alcançando assim um pior desempenho. O valor mínimo é de 13,68 que foi obtido com a base *Haar*, mostrando que alcançou o melhor desempenho.

Gráfico 4. MSE do Algoritmo SPIHT



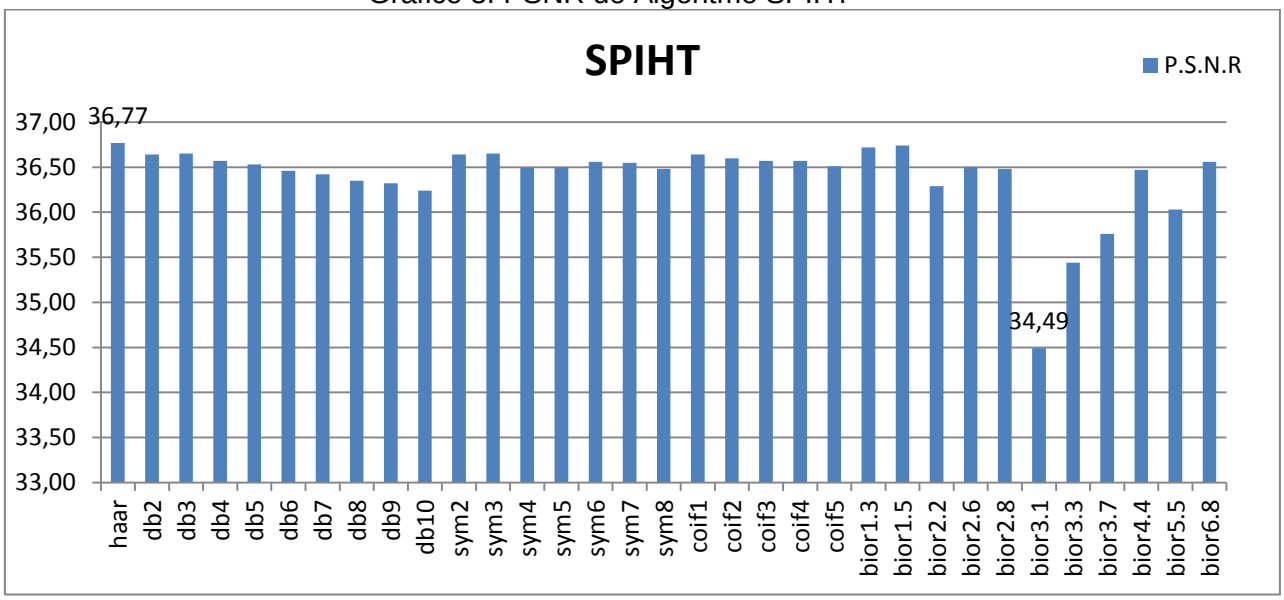
Fonte: elaborado pelo autor

4.1.2.2 Resultado Relação Sinal Ruído de Pico (PSNR)

O PSNR nos diz que quanto maior o valor obtido melhor será o desempenho da base *Wavelet*. Assim sendo, quanto menor o valor obtido pior será o desempenho da base *Wavelet*.

Podemos observar, no Gráfico 5, que o valor de *PSNR* máximo é de 36,77 da base *Haar*, alcançando assim o melhor desempenho da base *Wavelet*. O valor mínimo obtido é de 34,49 da base *bior3.1*, tendo assim o pior desempenho.

Gráfico 5. PSNR do Algoritmo SPIHT



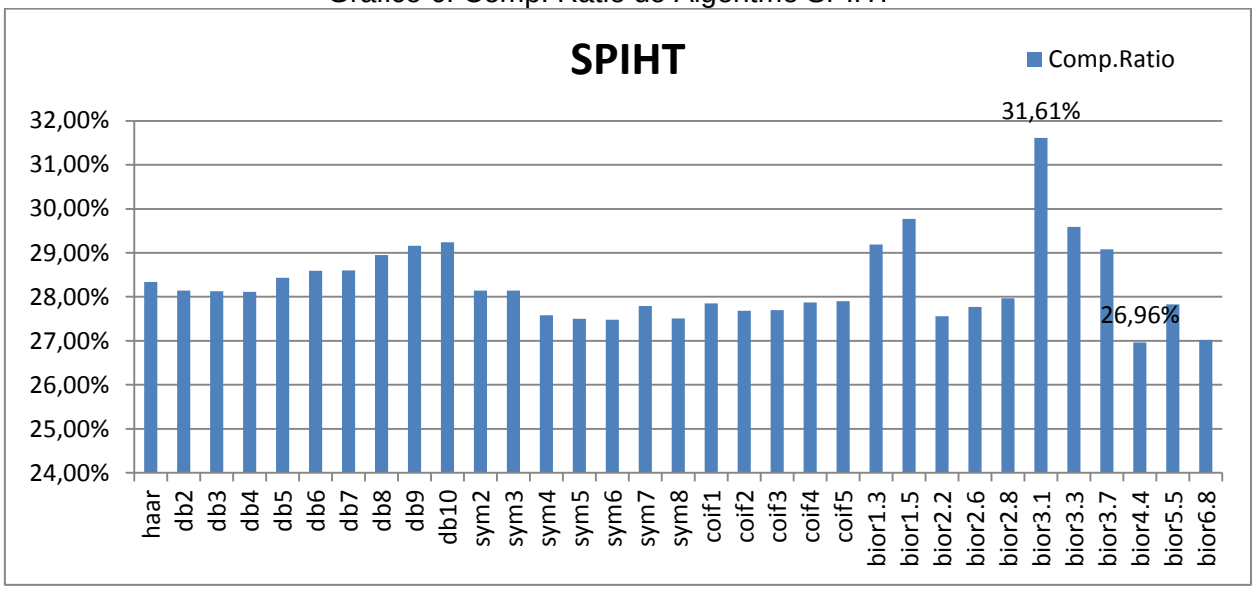
Fonte: elaborado pelo autor

4.1.2.3 Resultados da Relação da Compressão

A relação de compressão indica que a imagem comprimida é armazenada utilizando uma porcentagem do tamanho inicial de armazenamento. Assim, quanto menor for o resultado, menor será o tamanho ocupado pela imagem.

Podemos perceber, no Gráfico 6, que a base *bior4.4* utiliza apenas 26,96% do tamanho inicial de armazenamento, alcançando com isso um melhor resultado. Enquanto a base *bior3.1* utilizou 31,61% do tamanho inicial de armazenamento, obtendo um pior desempenho.

Gráfico 6. Comp. Ratio do Algoritmo SPIHT

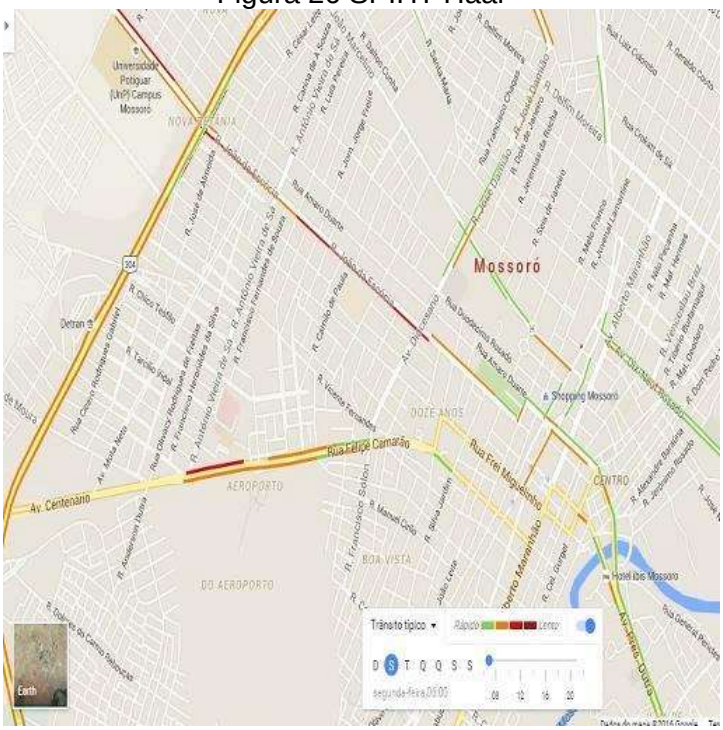


Fonte: elaborado pelo autor

Ao analisar o método *SPIHT* nas cinco bases *wavelets*, de acordo com os parâmetros *MSE, PSNR* e *Comp. Ratio*, podemos perceber que a melhor base é a *Haar*.

Abaixo, segue a Figura 26, com a imagem reconstruída após o processo de compressão e descompressão, mesmo com a relação de compressão um pouco elevada.

Figura 26 SPIHT Haar



4.1.3 Análise do método STW

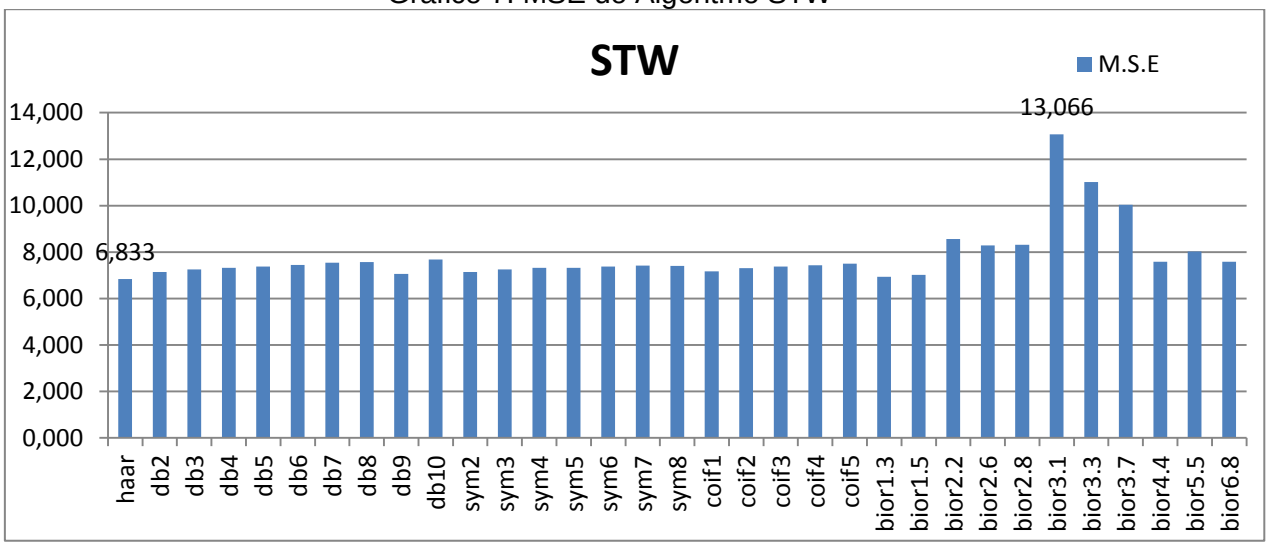
Nesta seção apresentaremos a análise do método *STW*, em todas as bases *wavelets*, de acordo com os parâmetros de compressão.

4.1.3.1 Resultado da Média dos Erros Quadráticos (MSE)

O *MSE* nos diz que quanto maior o valor obtido, pior será o desemprenho da base *Wavelet*, pois será maior o Erro Médio Quadrático. Assim sendo, quanto menor o Erro, melhor será o desempenho da base.

Percebemos, no Gráfico 7, que o valor do MSE máximo é de 13,066 obtido com a base bior3.1, alcançando assim um pior desempenho. O valor mínimo é de 6,833 que foi obtido com a base *Haar*, mostrando que alcançou o melhor desempenho.

Gráfico 7. MSE do Algoritmo STW



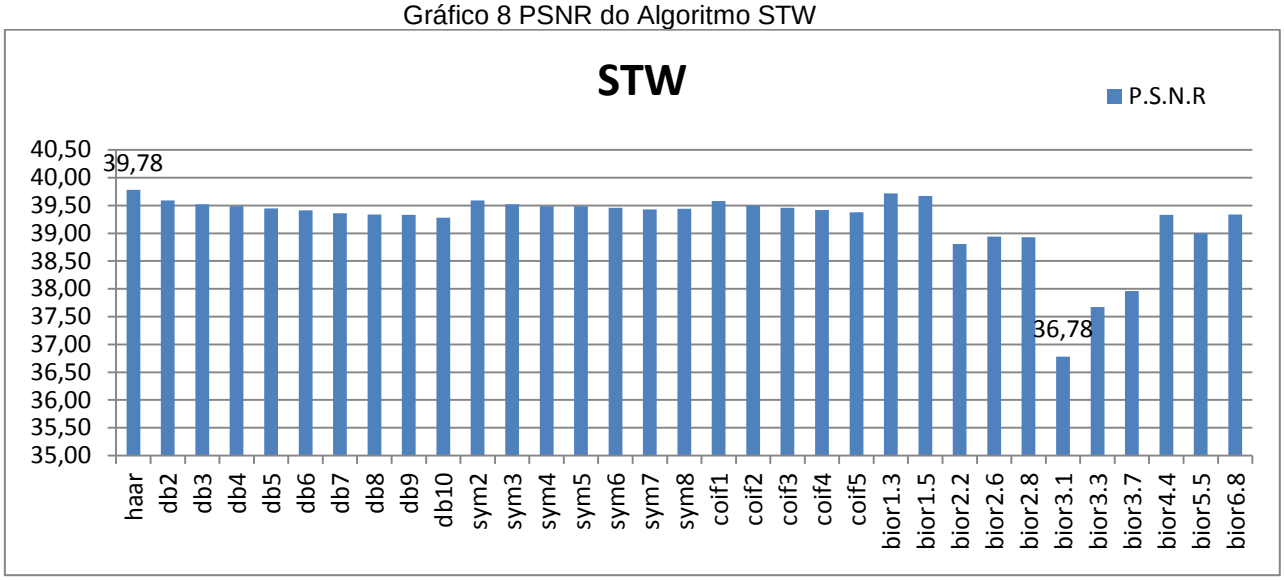
Fonte: elaborado pelo autor

4.1.3.2 Resultado Relação Sinal Ruído de Pico (PSNR)

O gráfico nos diz que quando maior o valor obtido melhor será o desempenho da base *Wavelet*. Assim sendo, quando menor o valor obtido pior será o desempenho da base *Wavelet*.

Podemos observar no Gráfico 8, o valor de PSNR máximo é de 39,78 da base *haar* alcançando assim o melhor desempenho da base *Wavelet*. O valor mínimo obtido é de 36,78 da base *bior3.1*, obtendo assim o pior desempenho. Lembramos que os resultados completos estão nas tabelas do Anexo I.

Gráfico 8 PSNR do Algoritmo STW



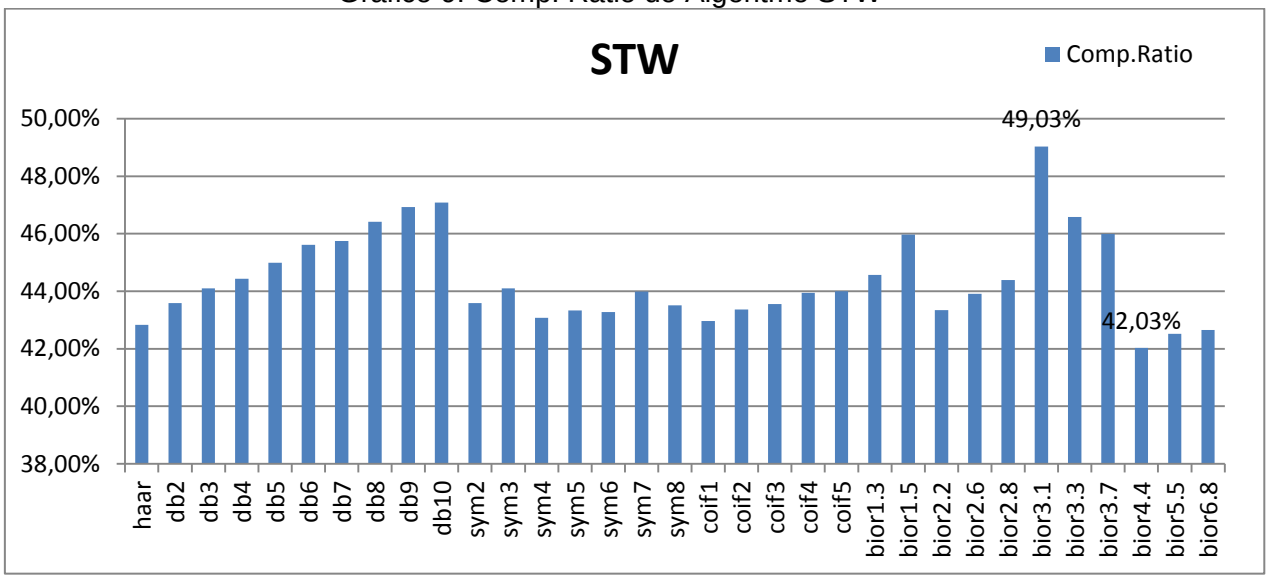
Fonte: elaborado pelo autor

4.1.3.3 Resultado da Relação de Compressão

A relação de compressão indica que a imagem comprimida é armazenada utilizando uma porcentagem (%) do tamanho inicial de armazenamento. Assim, quanto menor for o resultado menor será o tamanho ocupado pela imagem.

Observamos, no Gráfico 9, que a base *bior4.4* utiliza apenas 42,03% do tamanho inicial de armazenamento, alcançando um melhor resultado. Enquanto a base *bior3.1* utilizou 49,03% do tamanho inicial de armazenamento, obtendo um pior desempenho.

Gráfico 9. Comp. Ratio do Algoritmo STW

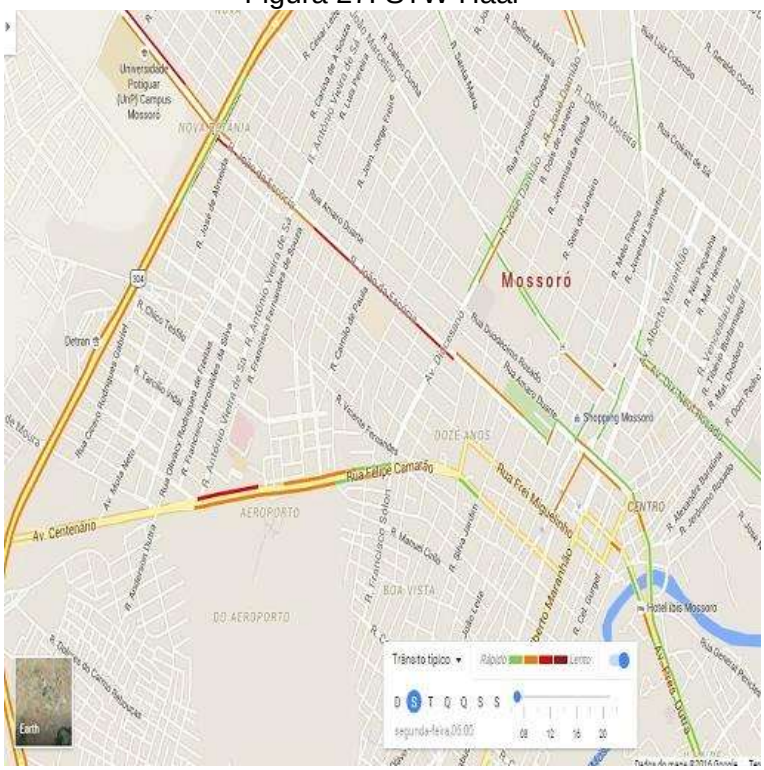


Fonte: elaborado pelo autor

Ao analisar o método STW nas cinco bases *wavelets*, de acordo com os parâmetros MSE, PSNR e Comp. Ratio, podemos perceber que a melhor base é a Haar.

A Figura 27 mostra a imagem reconstruída após o processo de compressão e descompressão, mesmo com a relação de compressão um pouco elevada.

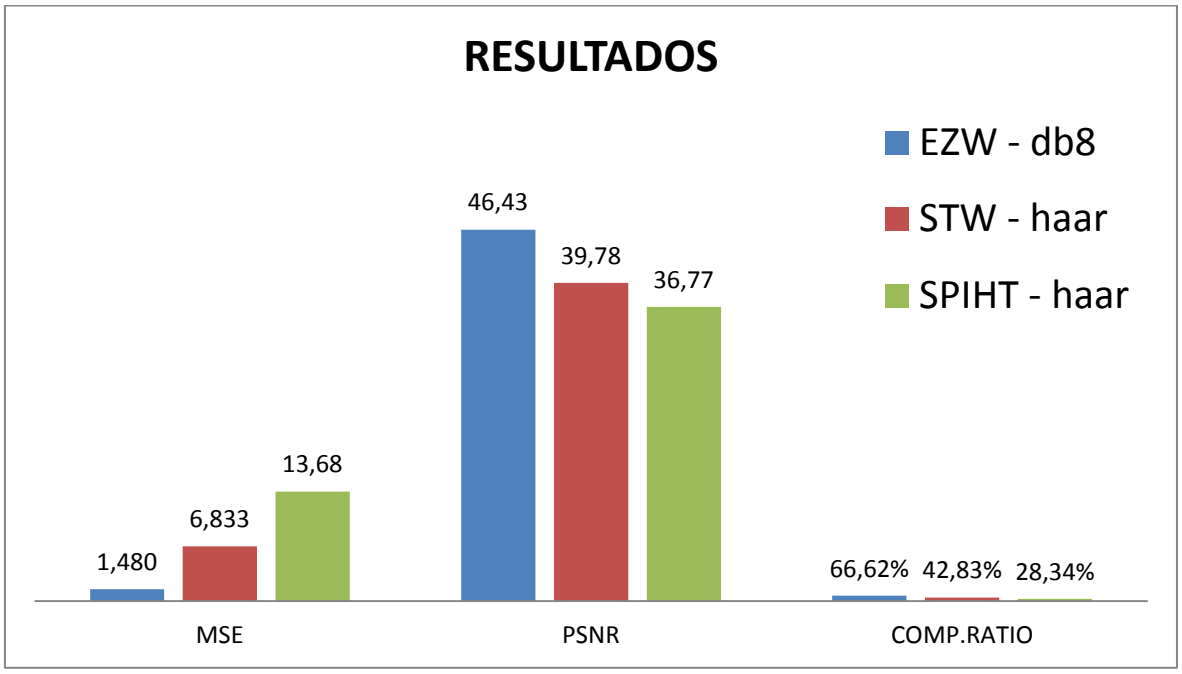
Figura 27. STW Haar



4.2 RESULTADOS DOS TESTES

Diante dos testes realizados, percebemos que ao comparar os métodos de compressão *EZW*, *SPIHT* e *STW*, o que se mostrou mais eficaz foi o *SPIHT* na base *Haar* como podemos observar, abaixo, no Gráfico 10.

Gráfico 10. Resultados



Fonte: elaborado pelo autor

Observamos no Gráfico 10, que os parâmetros de compressão *MSE* e o *PSNR* nos métodos *EZW* e *STW* têm valores que poderiam ser considerados eficazes vistos separadamente. No entanto, levando em consideração o conjunto da análise dos parâmetros e os propósitos desse trabalho, o que demonstrou um melhor desempenho foi o *SPHIT* na base *Haar*. Em outras palavras, como mencionamos anteriormente, o parâmetro *Comp. Ratio*, neste caso, torna-se o mais eficaz já que influencia no tamanho inicial de armazenamento, indicando que quanto menor for a porcentagem menor o tamanho ocupado pela imagem.

Considerando a imagem de teste e a imagem com melhor compressão, *SPHIT* na base *Haar*, podemos perceber uma considerável diferença no seu tamanho de armazenamento: inicialmente a imagem ocupava o total de 123Kb, posteriormente passou a ocupar um total de 54kb (Figura 29).

Figura 28. Imagem Original

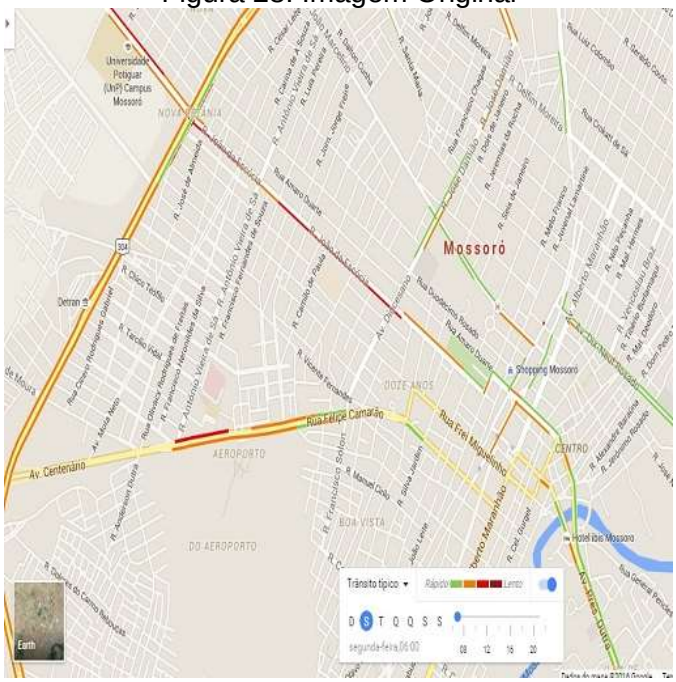
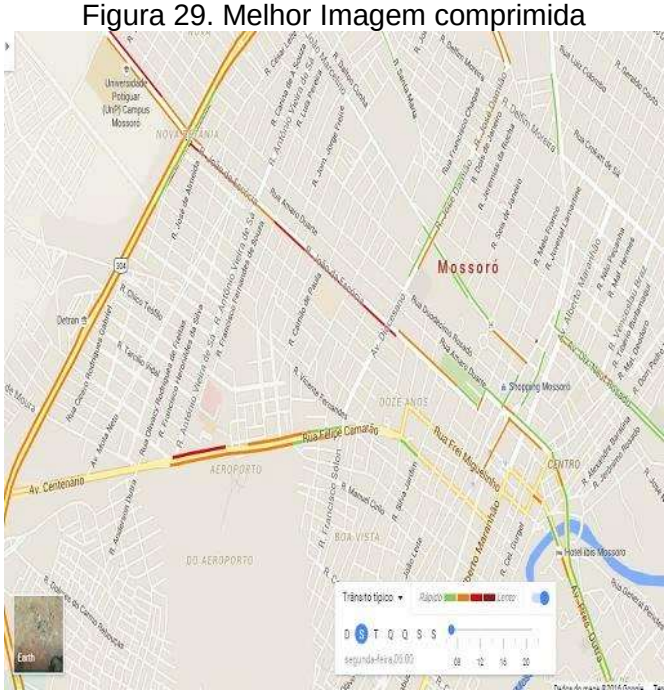


Figura 29. Melhor Imagem comprimida



Portanto, o método de compressão *SPHIIT* na base Haar mostrou-se o mais satisfatório em relação aos demais testados, apresentando uma porcentagem inferior, correspondente a 28,34%.

Os resultados foram obtidos utilizando um computador com processador Intel Core i3 2.1GHz, 3Gb de memória *RAM* e sistema operacional *Windows 7 Home Premium.*

5 CONCLUSÃO

Considerando o contexto tecnológico atual que exige sistemas de armazenamento e transmissão de informações mais eficazes, a compressão de dados se faz cada vez mais necessária. Pensando nisso, nos detemos a analisar os métodos e as bases de compressão de imagem, especificamente, através do uso de transformada *Wavelets* com o auxílio da ferramenta *Matlab*. Deste modo, buscamos comparar os métodos *EZW, SPIHT e o STW* no intuito de identificar qual deles é o mais eficaz na otimização da economia de espaço computacional, tanto no armazenamento como na transmissão de imagens.

Considerando os testes dos métodos *EZW, SPIHT e o STW*, nas cinco bases *wavelet*s, percebemos que: ao analisar o método *EZW*, nos parâmetros MSE, PSNR e *Comp. Ratio*, observamos que a melhor base foi a db8, pertencente à família da base *Daubechies*; já na análise do método SPIHT, de acordo com os parâmetros *MSE, PSNR* e *Comp. Ratio*, podemos perceber que a melhor base foi a Haar; por fim, ao analisar o método *STW*, nos parâmetros *MSE, PSNR* e *Comp. Ratio*, notamos que a melhor base foi também a *Haar*.

Em meio a esses resultados e tendo em vista nossos objetivos, observamos que o método de compressão *SPHIT* na base *Haar* foi o que demonstrou um melhor desempenho, sendo, portanto o mais eficaz. Além disso, o parâmetro Comp. Ratio foi o mais habilitado por influenciar no tamanho de armazenamento da imagem. Neste sentido, a imagem com melhor compressão, *SPHIT* na base *Haar*, apresentou uma considerável diferença no seu tamanho de armazenamento em relação à imagem de teste.

Percebemos, contudo, que esta pesquisa poderá ajudar em futuros trabalhos que visem questões voltadas à compressão de dados, principalmente no trato com imagens, melhorando as técnicas de armazenamento e transmissão desse tipo de mídia. Uma possível aplicação desta pesquisa poderá acarretar no desdobramento em outras áreas de pesquisas correlacionadas a serem desenvolvidas pelo professor Haroldo Bustos, estando ligada a área de logística operacional como também em sistemas de transporte usando interface do *Google Maps*, cujo objetivo é desenvolver novos algoritmos de aprendizado de maquina para estimação e previsão dinâmica, com proposito que seja possível fazer previsões sobre o trânsito

em determinado lugar de forma mais rápida, melhorando assim o tráfego de dados já que uma imagem comprimida tende a ser transmitida com maior velocidade.

6 REFERÊNCIAS

AL-REBIB, G. **EZW Coding Tutorial**. ECE/ Georgia Tech. Atlanta, GA, p. 15. 2000.

ALREGIB, G. **Embedded Zerotree Wavelet Encoding (EZW) Based on Shapiro's Paper**. Georgio INstitute of Technology. Atlanta, GA. 2000.

BECKER, A. J. et al. **Noções Básicas de Programação**. Santa Maria: [s.n.]. 2010.

DAUBECHIES, I. **Ten lectures on wavelets**. Philadelphia: SIAM, 1992.

FARIAS, M. C. Q. D. Aplicação da Transformada Wavelet na Compressão de Imagens. **Dissertação de Mestrado - Universidade Estadual de Campinas**, Campinhas - SP, 1998.

FILHO, O. M.; NETO, H. V. **Processamento Digital de Imagens**. Rio de Janeiro: Brasport, 1999.

FONSECA, M. S. Um Estudo sobre a Influência das Famílias Wavelets na Compressão de Imagem. **Dicertação de Mestrado da Universidade Federal Fluminense**, Niterói - RJ, Março 2004.

GONZALES, R. C.; WOODS, R. E. **Digital Image Precessing**. 2ª. ed. [S.l.]: Prentice Hall, 2002.

GOOGLE MAPS. Disponivel em: <https://www.google.com.br/maps/>. Acesso em: 01 março 2016.

GRAPS, A. An Introduction to Wavelets. **IEEE Computational Science and Engineering**, USA, v. 2, 1995.

GUSMÃO, A. A. Método robusto e simples de compressão de imagens baseado no Algoritmo EZW. **Dissertação de Mestrado - Universidade Estadual de Campinas**, Campinas-SP, 2002.

KUMAR, T.; CHAUDHARY, D. Compression Study Between 'ezw', spiht, stw,wdr, aswdr and spiht_3d. **International Journal of Scientific & Engineering Research**, v. 4, 2013.

LAUAND, B. P.; OLIVEIRA, J. Inferindo as Condições de Trânsito através da Análise de Sentimentos no Twitter. **iSys - Revista Brasileira de Sistemas de Informação**, Rio de Janeiro – RJ, 2014.

LIMA, P. C. Wavelets: uma introdução. **Departamento de Matemática - ICEX - UFMG**, Minas Gerias , 29 Julho 2003.

MISITI, M. et al. **Wavelet Toolbox™ 4:** User's Guide. [S.l.]: MatLab.

NOBRE, M. **O uso do software Matlab para o estudo de alguns tópicos de álgebra linear**. Universidade Católica de Brasília. Brasilia: [s.n.]. 2005. p. http://repositorio.ucb.br/jspui/handle/10869/1858.

OLIVEIRA, H. M. **Análise de sinais para engenheiros:** uma abordagem via wavelets. Rio de Janiero: Brasport, 2007.

SAID, A.; PEARLMAN, W. A. A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 6, Junho 1996.

SANCHES, I. J. **Compressão sem perdas de projeções de Tomografia computadorizada usando a transformada Wavelet**. Universidade Federal do Paraná, Dissertação de Mestrado. Curitiba. 2001.

SANCHES, Y. C. Computação Gráfica: Quantização e Compressão de Imagens, Presidente Prudente - SP, 2004. Disponivel em: "http://www.sbmac.org.br/cnmacs/2004/cd_cnmac/files_pdf/10091a.pdf".

SAYOOD, K. **Introduction to Data Compression**. 3. ed. [S.l.]: [s.n.], 2006.

SHAPIRO, J. M. EMBEDDED IMAGE CODING USING ZEROTREES OF WAVELET COEFFICIENTS. **IEEE Transactions on Signal Processing**, v. 41, n. 12, 1993.

SILVA, F. S. **Procedimentos para tratamento e compressão de imagens e vídeo utilizando tecnologia fractral e transformadas Wavelet**. Tese Doutorado - Universidade Estadual de Campinas. Campinas, SP. 2005.

SILVA, J. M.; FLÔRES, E. L. Introdução à compressão de images utilizando fractais, Uberlândia - MG, 2005.

STOLLITZ, E. J.; DEROSE, T. D.; SALESIN, D. H. Wavelets for Computer Graphics: A Primer Part 1. **IEEE Computer Graphics and Applications**, 1995.

STRANG , G.; NGUYEN , T. **Wavelets and Filter Banks**. [S.l.]: Wellesley-Cambride Press, 1996.

TENÓRIO, A. G. **Implementação de um compressor de imagens na Linguagem Python usando a Transformada Wavelet**. Dicertação mestrado - Faculdade de Engenharia Elétrica e de Computação. Campinas, SP: [s.n.]. 2003.

VALENS, C. EZW Encoding, 1999. Disponivel em: <http://polyvalens.pagesperso-orange.fr/clemens/ezw/ezw.html>.

WALKER, J. S. Wavelet-based Image Compression. In: RAO, K. R.; YIP, P. C. **The Transform and Data Compression Handbook**. Boca Raton, FL: [s.n.], 2000. Cap. 6, p. 286-330.

WALKER, J. S. **A Primer on WAVELETS and Their Scientific Applications**. 2ª. ed. EUA: Chapman e Hall/CRC, 2008.

ANEXO

RESULTADOS APÓS A COMPRESSÃO E DESCOMPRESSÃO DA IMAGEM DE TESTE, UTILIZANDO AS FAMÍLIAS WAVELETS E OS ALGORITMOS DE COMPRESSÃO

Tabela de teste –

Base	EZW		
	M.S.E	P.S.N.R	Comp.Ratio
haar	1,494	46,39	61,50%
db2	1,527	46,29	62,52%
db3	1,515	46,33	63,16%
db4	1,491	46,4	63,77%
db5	1,515	46,33	64,60%
db6	1,49	46,4	65,53%
db7	1,494	46,39	65,73%
db8	1,48	46,43	66,62%
db9	1,486	46,41	67,15%
db10	1,482	46,42	67,47%
sym2	1,527	46,29	62,52%
sym3	1,515	46,33	63,16%
sym4	1,537	46,26	62,34%
sym5	1,519	46,32	62,79%
sym6	1,54	46,26	62,72%
sym7	1,527	46,29	63,56%
sym8	1,540	46,26	63,15%
coif1	1,527	46,29	62,00%
coif2	1,521	46,31	62,55%
coif3	1,54	46,25	62,92%
coif4	1,525	46,27	63,58%
coif5	1,544	46,24	63,78%
bior1.3	1,539	46,26	63,54%
bior1.5	1,57	46,17	64,97%
bior2.2	1,999	45,12	62,36%
bior2.6	1,955	45,22	63,11%
bior2.8	1,962	45,2	63,72%
bior3.1	4,012	42,1	69,48%
bior3.3	3,146	43,15	66,37%
bior3.7	2,896	43,51	65,72%
bior4.4	1,603	46,08	61,36%
bior5.5	1,658	45,94	62,37%
bior6.8	1,599	46,09	62,07%

Base	STW		
	M.S.E	P.S.N.R	Comp.Ratio
haar	6,833	39,78	42,83%
db2	7,141	39,59	43,59%
db3	7,255	39,52	44,10%
db4	7,316	39,49	44,43%
db5	7,376	39,45	44,99%
db6	7,444	39,41	45,61%
db7	7,534	39,36	45,75%
db8	7,573	39,34	46,41%
db9	7,059	39,33	46,93%
db10	7,674	39,28	47,08%
sym2	7,141	39,59	43,59%
sym3	7,255	39,52	44,10%
sym4	7,32	39,49	43,08%
sym5	7,314	39,49	43,33%
sym6	7,372	39,46	43,28%
sym7	7,417	39,43	43,99%
sym8	7,405	39,44	43,51%
coif1	7,163	39,58	42,96%
coif2	7,303	39,5	43,36%
coif3	7,368	39,46	43,55%
coif4	7,433	39,42	43,94%
coif5	7,494	39,38	44,00%
bior1.3	6,931	39,72	44,57%
bior1.5	7,012	39,67	45,97%
bior2.2	8,559	38,81	43,34%
bior2.6	8,291	38,94	43,91%
bior2.8	8,315	38,93	44,39%
bior3.1	13,066	36,78	49,03%
bior3.3	11,012	37,67	46,58%
bior3.7	10,04	37,96	45,99%
bior4.4	7,581	39,33	42,03%
bior5.5	8,018	39	42,52%
bior6.8	7,578	39,34	42,65%

Base	SPIHT		
	M.S.E	P.S.N.R	Comp.Ratio
haar	13,68	36,77	28,34%
db2	14,11	36,64	28,14%
db3	14,07	36,65	28,13%
db4	14,33	36,57	28,11%
db5	14,46	36,53	28,43%
db6	14,69	36,46	28,59%
db7	14,83	36,42	28,60%
db8	15,06	36,35	28,95%
db9	15,16	36,32	29,16%
db10	15,47	36,24	29,24%
sym2	14,11	36,64	28,14%
sym3	14,07	36,65	28,14%
sym4	14,6	36,49	27,58%
sym5	14,57	36,5	27,50%
sym6	14,37	36,56	27,48%
sym7	14,39	36,55	27,79%
sym8	14,62	36,48	27,51%
coif1	14,08	36,64	27,85%
coif2	14,21	36,6	27,68%
coif3	14,31	36,57	27,70%
coif4	14,34	36,57	27,87%
coif5	14,52	36,51	27,90%
bior1.3	13,84	36,72	29,19%
bior1.5	13,76	36,74	29,77%
bior2.2	15,28	36,29	27,56%
bior2.6	14,56	36,5	27,77%
bior2.8	14,63	36,48	27,97%
bior3.1	23,11	34,49	31,61%
bior3.3	18,6	35,44	29,59%
bior3.7	17,25	35,76	29,08%
bior4.4	14,64	36,47	26,96%
bior5.5	16,23	36,03	27,83%
bior6.8	14,37	36,56	27,02%