UNIVERSIDADE ESTADUAL PAULISTA
"Júlio de Mesquita Filho"

Pós-Graduação em Ciência da Computação

Tiago Alexandre Dócusse

Um método para melhoria de qualidade de imagens médicas utilizando a transformada *wavelet*

UNESP

2008

Tiago Alexandre Dócusse

Um método para melhoria de qualidade de imagens médicas utilizando a transformada *wavelet*

Orientador: Prof. Dr. Aledir Silveira Pereira

Dissertação de Mestrado elaborada junto ao Programa de Pós-Graduação em Ciência da Computação – Área de Concentração em Sistemas de Computação, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

UNESP

2008

Tiago Alexandre Dócusse

Um método para melhoria de qualidade de imagens médicas utilizando a transformada *wavelet*

Dissertação apresentada para obtenção do título de Mestre em Ciência da Computação, área de concentração em Sistemas de Computação junto ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

BANCA EXAMINADORA

Prof. Dr. Aledir Silveira Pereira
Professor Assistente Doutor
UNESP – São José do Rio Preto
Orientador

Prof. Dr. Roberto Marcondes Cesar Junior
Professor Doutor
Universidade de São Paulo

Prof. Dr. Norian Marranghello
Professor Assistente Doutor
UNESP – São José do Rio Preto

São José do Rio Preto, maio de 2008

À minha família, dedico.

AGRADECIMENTOS

A Deus, primeiramente, por permitir a conclusão dos meus estudos.

À minha família, pelo apoio e incentivo dados.

Ao Prof. Dr. Aledir Silveira Pereira, pela orientação, incentivo, paciência e disponibilidade na realização deste trabalho.

Ao Prof. Dr. Norian Marranghello , pelo auxílio na elaboração deste trabalho.

Ao Prof. Dr. Rodrigo Capobianco Guido, pelo auxílio na elaboração deste trabalho.

À Profa. Dra. Heloisa Helena Marino Silva, pelo incentivo dado.

À Camila Forte Bueno, pelo apoio e força nos momentos difíceis.

Aos amigos do LACE: Adriana, Alex, Henrique, Jacqueline, Jullyene, Luciana, Luciano e Otávio, pelo apoio e amizade durante os dois anos em que estivemos juntos.

Aos demais amigos do mestrado: Antônio, Dioraci, Jorge, Lucimar, Patrícia, Rodolfo, Rodrigo, entre outros.

À CAPES, pelo auxílio financeiro recebido para o desenvolvimento deste trabalho.

RESUMO

O câncer de mama é uma das doenças que mais matam mulheres com idade acima de quarenta anos no Brasil atualmente. A fim de prevenir e tratar essa doença, o exame mais indicado é a análise de mamografias, imagens obtidas da mama fazendo uso de aplicações de raios-x, que podem indicar a presença ou não de tumores. Neste trabalho é apresentado um método para melhorar o contraste da imagem das mamas, classificando o formato de microcalcificações a fim de auxiliar médicos a decidir se este tumor é maligno. O método apresentado é baseado na transformada *wavelet*, que decompõe uma imagem em bandas de diferentes freqüências, permitindo a detecção destes objetos através da característica de freqüência deles. A utilização da família Symmlets gerou melhores resultados, tanto no realce da imagem de microcalcificações quanto na classificação das bordas desses objetos.

ABSTRACT

Breast cancer is one of the diseases that kills most of women older than forty in Brazil nowadays. In order to prevent and treat it, the most appropriate exam is the analysis of mammograms, images obtained from the breast by applying x-rays on it, indicating whether or not tumors are present. In this work a method to enhance breast images is presented, classifying the format of microcalcifications in order to help doctors decide whether or not this tumor is malign. The proposed method is based on the wavelet transform, which decomposes an image into different frequency bands, allowing the detection of these elements by their frequency features. Utilization of the Symmlets family achieved the best results, on the microcalcification image enhancement and on the classification of the borders of these elements.

SUMÁRIO

LISTA DE FIGURAS viii
LISTA DE TABELAS x
LISTA DE ABREVIATURAS E SIGLAS xi
Capítulo 1 – Introdução 1
Capítulo 2 – Revisão Bibliográfica 3
2.1 Introdução sobre o câncer de mama 3
2.2 Conceitos básicos de processamento de imagens digitais 5
2.3 Transformada *wavelet* e aplicações em imagens 22
Capítulo 3 – Método proposto 40
3.1 Funcionamento do método 42
3.1.1 Etapa de pré-processamento 43
3.1.2 Etapa *wavelet* 44
3.1.3 Crescimento de região 47
3.1.4 Pós-processamento 50
3.1.5 Caracterização de bordas 51
3.2 Desenvolvimento e implementação 56
3.3 - Resultados 57
Capítulo 4 – Conclusões 74
4.1 Conclusão do trabalho 74
4.2 Trabalhos futuros 75
Referências Bibliográficas 76
GLOSSÁRIO 80
Apêndice A – Coeficientes *wavelets* utilizados 81
Apêndice B – Desenvolvimento da biblioteca WIP 84
Apêndice C – Documentação do sistema desenvolvido 91
Apêndice D – Manual de utilização do sistema desenvolvido 107

LISTA DE FIGURAS

Figura 2.1 – Exemplos de imagens de microcalcificações 4
Figura 2.2 – Exemplo de escala de cinza 7
Figura 2.3 – Exemplo de amostragem em uma cena 8
Figura 2.4 – Exemplo de quantização de uma cena amostrada 8
Figura 2.5 – Representação dos eixos em uma imagem digital 9
Figura 2.6 – Exemplos de uma imagem com diferentes valores de resolução e escala de cinza 10
Figura 2.7 – Representação visual das vizinhanças de um *pixel* 11
Figura 2.8 – Exemplos de conectividades de *pixels* em uma imagem 12
Figura 2.9 – Exemplo de aplicação da operação negação 13
Figura 2.10 – Exemplos da aplicação das operações limiarização e binarização 14
Figura 2.11 – Exemplo de aplicação da operação alargamento de contraste 15
Figura 2.12 – Exemplos de aplicação da operação filtragem por mediana para diferentes tamanhos de vizinhança 16
Figura 2.13 – Exemplo da aplicação da operação aguçamento de imagens 17
Figura 2.14 – Resposta em freqüência de filtros digitais. Adaptado de [14]. 18
Figura 2.15 – Distribuição de freqüências da transformada de Fourier 20
Figura 2.16 – Espectro de Fourier 21
Figura 2.17 – Aplicação da transformada de Fourier de tempo-curto 21
Figura 2.18 – Exemplos de dilatações e translações de uma *wavelet*-mãe e uma função de escala pai 23
Figura 2.19 – Decomposição *wavelet* de um sinal 24
Figura 2.20 – Decomposição *wavelet* em várias escalas 25
Figura 2.21 – Decomposição *wavelet* de uma imagem 27
Figura 2.22 – Distribuição de freqüências da transformada *wavelet* em uma imagem 27
Figura 2.23 – Decomposição *wavelet* de uma imagem até o nível quatro 28
Figura 2.24 – Quantidade de elementos de altas freqüências em cada nível da decomposição *wavelet* de uma imagem 29
Figura 2.25 – Exemplo de aplicação da transformada *wavelet* 30
Figura 2.26 – Aplicação de diferentes *wavelets* em uma imagem 32
Figura 2.27 – Funções de escala pai e *wavelets*-mãe de Daubechies 33
Figura 2.28 – Resposta em freqüência das *wavelets* da Daubechies 34
Figura 2.29 – Resposta em fase da *wavelet* de Haar 35
Figura 2.30 – Resposta em fase de Daub12 35
Figura 2.31 – Funções de escala pai e *wavelets*-mãe de Symmlets 36
Figura 2.32 – Resposta em freqüência das Symmlets 36
Figura 2.33 – Resposta em fase da *wavelet* Symmlets12 37
Figura 2.34 – Funções de escala pai e *wavelets*-mãe de Coiflets 37
Figura 2.35 – Resposta em freqüência das Coiflets 38
Figura 2.36 – Resposta em fase de Coif12 38
Figura 3.1 – Esquema de realce por aguçamento de imagens 40

Figura 3.2 – Aplicação de aguçamento de imagens em mamografia digital 41
Figura 3.3 – Esquema de realce pela transformada de Fourier 41
Figura 3.4 – Aplicação da transformada de Fourier em mamografia digital 42
Figura 3.5 – Esquema geral do método desenvolvido 43
Figura 3.6 – Esquema de funcionamento da etapa de pré-processamento 43
Figura 3.7 – Ajuste de resolução 44
Figura 3.8 – Esquema de funcionamento da etapa *wavelet* 44
Figura 3.9 – Esquema de funcionamento da etapa de crescimento de região 47
Figura 3.10 – Exemplo do funcionamento do algoritmo de crescimento de região 49
Figura 3.11 – Exemplo de MPLs sobrepostas 50
Figura 3.12 – Esquema de funcionamento da etapa de classificação 52
Figura 3.13 – Resultado da aplicação de *wavelets* para perturbação de bordas 52
Figura 3.14 – Exemplo de ROIs lisas e rugosas e resultado da aplicação da caracterização 54
Figura 3.15 – Resultados 1 59
Figura 3.16 – Resultados 2 60
Figura 3.17 – Resultados 3 63
Figura 3.18 – Resultados 4 65
Figura 3.19 – Número de ROIs detectadas 67
Figura 3.20 – Valor médio de área das ROIs 68
Figura 3.21 – Valor médio de nível de cinza das ROIs 69
Figura 3.22 – Classificação das ROIs detectadas em imagens contendo microcalcificações com bordas lisas 70
Figura 3.23 – Classificação das ROIs detectadas em imagens contendo microcalcificações com bordas rugosas 70
Figura 3.24 – Comparação entre o valor médio de área das microcalcificações lisas e o valor médio de área das ROIs detectadas 71
Figura 3.25 – Comparação entre o valor médio de área das microcalcificações rugosas e o valor médio de área das ROIs detectadas 72
Figura B.1 – Diagrama de classes da biblioteca WIP 85
Figura C.1 – Diagrama de casos de uso do sistema RCMW 91
Figura C.2 – Diagrama de classes do sistema RCMW 93
Figura D.1 – Tela principal do sistema 107
Figura D.2 – Barra de ferramentas 108
Figura D.3 – Imagem carregada 109
Figura D.4 – Escolha da *wavelet* utilizada 109
Figura D.5 – Exibição de imagem realçada 110
Figura D.6 – Visualização das imagens original e realçada lado a lado 111
Figura D.7 – Visualização individual de ROIs 111

LISTA DE TABELAS

Tabela 2.1 – Classificação morfológica de Le Gal 4
Tabela 2.2 – Espaço em memória necessário para armazenar uma imagem 10
Tabela 2.3 – Quantidade de elementos de altas freqüências em cada nível da decomposição *wavelet* de uma imagem 28
Tabela 2.4 – Resposta em fase das *wavelets* utilizadas 39
Tabela 3.1 – Resultado da classificação das ROIs 55
Tabela 3.2 – Descrição das legendas 57
Tabela 3.3 – Resultados 1 58
Tabela 3.4 – Resultados 2 61
Tabela 3.5 – Resultados 3 62
Tabela 3.6 – Resultados 4 64

LISTA DE ABREVIATURAS E SIGLAS

BP	:	***B**anda de **P**assagem*
BR	:	***B**anda de **R**ejeição*
CAD	:	***C**omputer **A**ided **D**iagnosis*
CMYK	:	***C**ian **M**agenta **Y**ellow Blac**k***
CoifN	:	***Coif**letsN*
DaubN	:	***Daub**echiesN*
FIR	:	***F**inite **I**mpulse **R**esponse*
HP	:	***H**igh **P**ass*
HIS	:	***H**ue **S**aturation **I**ntensity*
HSV	:	***H**ue **S**aturation **V**alue*
IIR	:	***I**nfinite **I**mpulse **R**esponse*
JAI	:	***J**ava **A**dvanced **I**maging*
JRE	:	***J**ava **R**untime **E**nvironment*
LP	:	***L**ow **P**ass*
MPG	:	***M**atriz de **P**ertinência **G**lobal*
MPL	:	***M**atriz de **P**ertinência **L**ocal*
RGB	:	***R**ed **G**reen **B**lue*
ROI	:	***R**egion **o**f **I**nterest*
STFT	:	***S**hort **T**ime **F**ourier **T**ransform*
SymN	:	***Sym**mletsN*
TF	:	***T**ransformada de **F**ourier*
TW	:	***T**ransformada **W**avelet*
UML	:	***U**nified **M**odelling **L**anguage*

Capítulo 1 – Introdução

O câncer de mama é uma doença que atinge, na maioria, pessoas do sexo feminino com idades entre 40 e 69 anos [1]. Atualmente é o segundo câncer no mundo em número de ocorrências e o primeiro entre mulheres [2]. Possivelmente tratável se detectado precocemente, o câncer de mama possui taxa de mortalidade elevada no Brasil devido principalmente à falta de cuidado das pessoas, uma vez que na maioria dos casos essa doença é diagnosticada já em estágios avançados [2].

O diagnóstico dessa doença pode ser feito através da análise de exames que utilizam aparelhos de raios-x para obter uma imagem da mama. Uma vez diagnosticada, uma biópsia – operação em que um pedaço do tumor é retirado do corpo do paciente – é feita para verificar a natureza do tumor encontrado. Apesar de ser um método eficiente, nem sempre a análise visual da imagem da mama é uma tarefa fácil de ser feita. Com esse problema em mente, diversos sistemas computadorizados de auxílio ao diagnóstico – *Computer Aided Diagnosis* (CAD) – têm sido desenvolvidos com a finalidade de auxiliar médicos a visualizar de maneira mais nítida tais imagens [3]. Elas podem ser processadas de forma a melhorar suas qualidades ou então detectar alguns elementos nela presentes, a fim de auxiliar a sua análise.

Um elemento que pode ser localizado em mamas é a microcalcificação. As microcalcificações são minúsculos sedimentos que sempre estão presentes em casos de câncer, porém, nem sempre quando eles estão presentes há realmente câncer. Ainda, esses elementos podem ser o único sinal de um câncer impossível de ser detectado por exames palpáveis [4], porém, a visualização dos mesmos é uma tarefa

difícil de ser realizada. Eles podem ser classificados em cinco diferentes tipos, sendo que as informações sobre os formatos de suas bordas podem auxiliar o médico responsável a decidir qual é o tipo da microcalcificação, o que pode auxiliá-lo a decidir se um tumor é maligno ou não sem a necessidade de realizar uma biópsia.

O objetivo deste trabalho é propor um método desenvolvido com a finalidade de auxiliar médicos na visualização de microcalcificações em imagens da mama, bem como, auxiliá-los informando o tipo de borda das microcalcificações encontradas. Isto é feito através da aplicação da transformada *wavelet*, possibilitando a separação das imagens destes elementos do restante da imagem.

O trabalho realizado utiliza a transformada *wavelet* para melhorar a visualização de microcalcificações em imagens da mama, além de utilizar essa transformada para classificar as bordas desses elementos. Uma análise de diversas *wavelets* é feita, na tentativa de verificar qual a melhor *wavelet* para essa tarefa, e a integração do método proposto com o trabalho realizado anteriormente em [7] possibilita detecção de todos os tipos de microcalcificações, o que pode auxiliar médicos no diagnóstico do câncer de mama.

A divisão do texto é feita de forma que o capítulo 1 faz uma apresentação do trabalho, o capítulo 2 apresenta uma revisão sobre processamento de imagens digitais e apresenta a transformada *wavelet*. Já o capítulo 3 apresenta o método proposto e detalhes de seu funcionamento e sua implementação. No capítulo 4 são apresentados alguns dos resultados obtidos e uma análise sobre os mesmos. O capítulo 5 apresenta as conclusões do trabalho e as propostas de trabalhos futuros.

Capítulo 2 – Revisão Bibliográfica

Este capítulo tem como objetivo apresentar os conceitos básicos necessários para a compreensão deste trabalho.

2.1 Introdução sobre o câncer de mama

O câncer de mama é o tipo de câncer com a maior incidência e número de óbitos em mulheres no mundo. Estima-se que o número de novos casos dessa doença no Brasil em 2008 seja de 49.400 casos [2]. A melhor forma de tratamento é a detecção precoce com início imediato de tratamento [4]. Como a incidência de casos de câncer de mama aumenta com o aumento da idade dos pacientes, o Ministério da Saúde recomenda, desde 2004, um exame clínico anual das mamas para mulheres entre 40 e 49 anos, além de um exame mamográfico a cada dois anos para mulheres entre 50 e 69 anos [1]. Para mulheres que estejam em grupos de risco para o câncer de mama, recomenda-se o exame clínico e mamográfico a partir dos 35 anos de idade [1].

O câncer de mama caracteriza-se pela reprodução com velocidade acima do normal das células da mama, gerando massas celulares chamadas neoplasias ou tumores. Quando as neoplasias são ditas malignas, elas podem se desenvolver em outras partes do corpo, fato esse denominado metástase [5]. Uma das características das neoplasias é que sempre que elas sempre implicam em minúsculas calcificações denominadas microcalcificações. Através de sua análise, Michèle Le Gal propôs em

[6] um esquema classificatório para as neoplasias baseado nas suas morfologias, conforme a Tabela 2.1.

Tabela 2.1 – Classificação morfológica de Le Gal

Tipo	Morfologia	Lesões malignas
I	Anelares	0 %
II	Círculos regulares	22 %
III	Granulares	40 %
IV	Círculos irregulares	60 %
V	Vermiformes	100 %

A Figura 2.1 exibe exemplos de imagens das microcalcificações classificadas segundo Le Gal, onde, em (a), tem-se microcalcificações do tipo I; em (b), tipo II; em (c), tipo III; em (d), tipo IV; e em (e), tipo V.

(a) (b) (c) (d) (e)

Figura 2.1 – Exemplos de imagens de microcalcificações

Através da classificação proposta por Le Gal é possível, dependendo do tipo da microcalcificação encontrada na mama, dizer se uma neoplasia é maligna ou não sem a necessidade de uma intervenção cirúrgica no paciente, como visto na Tabela 2.1, o que é altamente desejável, já que uma cirurgia pode resultar em complicações. A análise morfológica das microcalcificações pode ser feita através do exame mamográfico, a fim de determinar a qual tipo morfológico esses objetos pertencem.

O exame mamográfico é reconhecidamente o melhor exame para detectar precocemente o câncer de mama [7]. Este exame é realizado através da utilização do mamógrafo, um aparelho de raios-x onde a mama a ser analisada é prensada entre duas placas metálicas e é atravessada por ondas de raios-x [8]. Dependendo da resistência encontrada por essas ondas ao atravessar a mama, forma-se uma imagem da mesma em um filme fotográfico, sendo esta imagem denominada mamografia.

A análise das mamografias é feita por um radiologista – médico responsável pela análise de exames de radiográficos. Através delas, o radiologista pode analisar visualmente a imagem da mama e suas possíveis microcalcificações a fim de determinar se esses objetos estão presentes na mama e, em caso positivo, determinar sua quantidade e em quais tipos, dentre as categorias propostas por Le Gal, elas se classificam, objetivando um diagnóstico rápido e preciso. Essa análise, no entanto, não é uma tarefa simples. Devido ao pequeno tamanho das microcalcificações e de seus baixos contrastes em relação ao tecido da mama [9] [10], esses elementos podem se tornar difíceis de localizar ao analisar uma série de imagens em seqüência. Devido a esses fatores, o médico que vai analisar tais imagens pode ter cansaço visual, o que dificulta sua análise e pode comprometer o seu resultado [3]. Uma forma de minimizar esse problema é realçando a imagem desses objetos nas mamografias. Desta forma, as microcalcificações ficam com um contraste maior em relação ao tecido da mama, o que diminui a possibilidade de o radiologista não visualizá-las em casos de ele ter cansaço visual.

Através de digitalizadores específicos é possível transformar a mamografia para uma versão digital, possibilitando sua visualização em computadores e outros aparelhos eletrônicos. Existem ainda mamógrafos digitais cujas imagens de saída já estão em formato digital. Ambas imagens digitais são chamadas de *mamografias digitais*. Uma vez que a mamografia digital pode ser visualizada em computadores, abre-se a possibilidade de sua análise e tratamento pelos mesmos. Através de técnicas de processamento de imagens digitais é possível realçar as microcalcificações das mamografias, tornando-as mais fáceis de serem detectadas visualmente do que se analisadas sem realce. A seção 2.2 apresenta os conceitos básicos de processamento de imagens necessários para a compreensão de como suas técnicas podem ser utilizadas para realçar microcalcificações em mamografias digitais, bem como demais conceitos necessários para a compreensão do restante do texto.

2.2 Conceitos básicos de processamento de imagens digitais

O processamento de imagens digitais é uma área da computação que visa analisar uma imagem em formato digital e aplicar sobre ela operações para obter uma

imagem mais fácil de ser analisada. Essa área não é nova, visto que um de seus primeiros usos data do início do século 20, porém, é uma área que cresce e evolui constantemente, tornando-a desafiadora a cada nova aplicação desejada. Segundo Gonzalez [11], o interesse nessa área provém de duas principais aplicações: a melhoria da percepção visual humana de uma imagem e o processamento de imagens para uma percepção autônoma realizada por máquinas.

Diversas aplicações fazem uso de técnicas de processamento de imagens. Exemplos incluem edição de fotografias para publicação em material impresso; análise de fotografias aéreas de relevos e rios; detecção de objetos estranhos em imagens médicas; reconhecimento de caminhos e marcos em robótica, entre outros. Um detalhe a se notar é que, da mesma forma que existem diversas aplicações nessa área, existem também diversas soluções para o problema correspondente. Muito comumente, a solução encontrada para determinado problema não resolve um outro problema, o que torna a busca por soluções algo novo, embora determinadas técnicas utilizadas em um certo problema podem ser utilizadas de outra forma para gerar a solução do anterior.

Biologicamente, a luz que reflete nos objetos e penetra em nossos olhos forma uma imagem na retina e é processada pelo nosso cérebro, formando a imagem de nossa percepção do ambiente onde estamos. Se analisarmos uma imagem estática na nossa retina, podemos perceber que a imagem que visualizamos é como uma seqüência de imagens em duas dimensões. Assim, matematicamente, uma imagem pode ser modelada como uma função bidimensional, onde o valor da função em qualquer conjunto de coordenadas do espaço analisado representa a coloração da imagem naquele ponto.

Seja f uma função bidimensional e seja (x,y) um conjunto de coordenadas, tal que x represente o eixo das ordenadas (vertical) e y represente o eixo das abscissas (horizontal). Para um retângulo determinado pelos pontos $(0,0)$, $(X,0)$, $(0,Y)$ e (X,Y), onde X representa o valor máximo do eixo das ordenadas e Y representa o valor máximo do eixo das abscissas, o valor de f em qualquer ponto dentro deste retângulo corresponde à coloração da imagem no ponto analisado. Assim, $f(x,y)$ retorna a tonalidade da imagem f no ponto (x,y) sempre que $0 < x \leq X$ e $0 < y \leq Y$.

A coloração de uma imagem pode ser representada segundo vários modelos propostos na literatura. Para imagens coloridas, alguns dos mais comuns são o *Red*

Green Blue (RGB), o *Cian Magenta Yellow Black* (CMYK), *Hue Saturation Intensity* (HSI) e o *Hue, Saturation Value* (HSV). Para imagens monocromáticas, costuma-se utilizar uma escala com valores mínimo e máximo para a cor presente na imagem, variando o brilho da mesma nos intervalos entre esses valores. Como a natureza das imagens mamográficas analisadas no presente trabalho é de imagens em tons de cinza (monocromática, onde o cinza mais claro é a cor branca e o cinza mais escuro é a cor preta), esse modelo é utilizado no restante do texto.

Seja $L_{\max}$ o maior valor de intensidade possível de ser representado em uma imagem e $L_{\min}$ o menor valor de intensidade possível de ser representado na imagem mesma imagem. Cria-se então o intervalo $[L_{\min}, L_{\max}]$, onde $L_{\min}$ representa a cor preta, $L_{\max}$ representa a cor branca e os valores intermediários deste intervalo são os diversos níveis de cinza possíveis de representação em uma imagem. Este intervalo é chamado escala de cinza [11]. A Figura 2.2 exibe um exemplo de escala de cinza, onde é possível ver que a cor mais à esquerda é a cor branca, a cor mais à direita é a cor preta, e as demais cores entre elas são diferentes tonalidades de cinza.

Figura 2.2 – Exemplo de escala de cinza

Computacionalmente, uma imagem pode ser representada em formato digital através da aplicação de duas operações por um instrumento de captura de imagens: a amostragem e a quantização [11]. A amostragem consiste em discretizar as coordenadas espaciais da cena observada pelo instrumento em pontos para a formação da imagem digital, sendo que cada um destes pontos da imagem digital é chamado *pixel* (*picture element*). Para realizar a amostragem, é necessário conhecer o tamanho do *pixel* com o qual o instrumento trabalha. Para um instrumento que gere imagens com *pixels* de tamanho igual a 0,265 mm, a cada 0,265 mm da cena observada no eixo vertical e no eixo horizontal (neste caso, um *pixel* quadrado) um novo *pixel* é criado, e esse processo repete-se até que a cena seja totalmente percorrida. A Figura 2.3 mostra um exemplo da aplicação da amostragem de uma cena, onde as coordenadas espaciais dessa imagem são discretizadas com *pixels* quadrados de tamanho 0,265 mm.

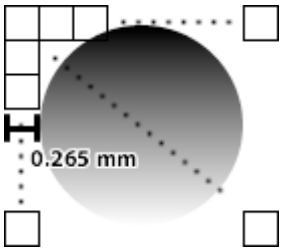


Figura 2.3 – Exemplo de amostragem em uma cena

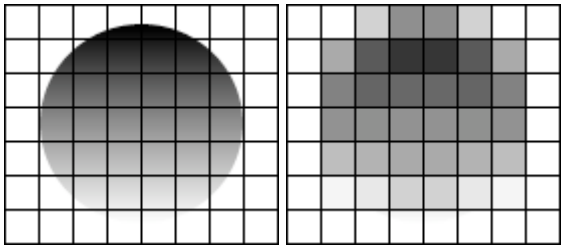


Figura 2.4 – Exemplo de quantização de uma cena amostrada

A outra operação necessária para transformar uma cena em uma imagem digital é a quantização. Esta operação consiste em discretizar o brilho da cena observada para um valor da escala de cinza utilizada. Isto é feito para todos os pontos da cena já amostrada, e ao final de sua operação a imagem digital pode ser visualizada. A Figura 2.4 mostra um exemplo da aplicação da quantização em uma cena à qual já foi aplicada a amostragem. Do lado esquerdo da Figura 2.4 é exibida a cena amostrada, e do lado direito a imagem digital formada resultante das operações de amostragem e quantização.

O resultado da aplicação dessas duas operações pode ser armazenado na forma de uma matriz, onde as posições de seus elementos representam os *pixels* da imagem digital e seus valores representam a tonalidade de cinza da imagem digital no ponto correspondente àquela posição. A Figura 2.5 mostra uma matriz vazia com as representações de seus eixos vertical e horizontal. Na Figura 2.5, a origem da imagem é representada pelo símbolo 0, o eixo das abscissas é o eixo das colunas, o eixo das ordenadas é o eixo das linhas e cada posição da matriz representa um *pixel* da imagem.

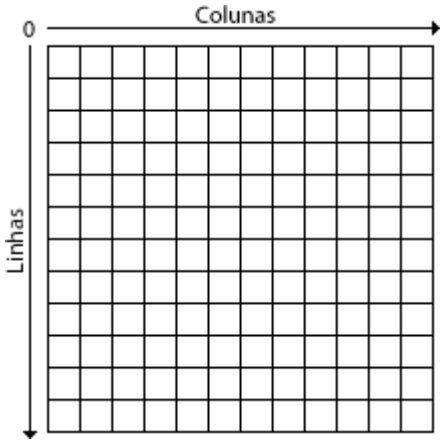


Figura 2.5 – Representação dos eixos em uma imagem digital

Seja $X_{\max}$ o índice das ordenadas do *pixel* da imagem mais distante da origem, e $Y_{\max}$ o índice das abscissas do *pixel* da imagem mais distante da origem. A resolução da imagem, também chamada tamanho (espacial) é normalmente expressa por $X_{\max}$ x $Y_{\max}$ *pixels*. É comum em processamento de imagens digitais utilizar imagens cuja resoluções sejam potências de 2, ou seja, $X_{\max} = 2^{b_x}$ e $Y_{\max} = 2^{b_y}$, onde $b_x, b_y \in \mathbb{N}$. Também é comum utilizar escalas de cinza em que o número de tonalidades $L = L_{\max} + 1$ seja potência de 2, ou seja, $L = 2^{b_t}$, onde $b_t \in \mathbb{N}$. Assim, para o armazenamento de uma imagem digital com resolução $X_{\max}$ x $Y_{\max}$ *pixels* e escala de cinza com *L* tonalidades, é necessário $G = b_x b_y b_t$ *bits* de memória.

A relação entre a resolução e a escala de cinza de uma imagem digital e o espaço em memória necessário para armazená-la é brevemente exemplificada na Tabela 2.2. Quanto maior a resolução de uma imagem, maior o seu nível de detalhes, pois menor é o tamanho de cada um de seus *pixel*, porém, maior o espaço em memória necessário para armazená-la. Quanto maior a escala de cinza de uma imagem, maior a quantidade de tonalidades exibidas na mesma, porém, maior é o espaço em memória necessário para armazená-la. Uma imagem com alta resolução e escala de cinza com grande quantidade de tons pode possuir muitos detalhes, porém, ocupa mais espaço em memória e gasta mais tempo para ser processada se comparada a uma imagem com menor resolução e com escala de cinza com menor quantidade de tons. Uma imagem com baixa resolução e escala de cinza com poucos tons, apesar de ocupar menos espaço em memória e ser mais fácil de ser processada se comparada a uma imagem com maior resolução e escala de cinza com maior

quantidade de tons, pode possuir uma quantidade inadequada de detalhes para ser analisada corretamente. Essa diferença de qualidade na imagem pode ser vista através dos exemplos exibidos na Figura 2.6, onde a mesma imagem é exibida em diferentes resoluções e com diferentes escalas de cinza. Na Figura 2.6, *A*x*B*x*C* significa que a imagem possui resolução *A*x*B pixels* e escala de cinza com *C* tonalidades possíveis.

Tabela 2.2 – Espaço em memória necessário para armazenar uma imagem

		Escala de cinza		
		4 tons	8 tons	16 tons
Resolução	16x16 *pixels*	32 *bits*	48 *bits*	64 *bits*
	32x32 *pixels*	50 *bits*	75 *bits*	100 *bits*
	64x64 *pixels*	72 *bits*	108 *bits*	144 *bits*

Figura 2.6 – Exemplos de uma imagem com diferentes valores de resolução e escala de cinza

Nota-se através da Figura 2.6 a necessidade de escolher resolução e escala de cinza adequados para a aplicação desejada, no entanto, lembrando ao mesmo tempo que quanto maiores esses valores, maiores o espaço em memória e o tempo necessário para processar essa imagem.

Uma vez que uma imagem digital pode ser tratada como uma matriz, é possível estabelecer relacionamentos entre seus *pixels*. Alguns desses

relacionamentos utilizados no desenvolvimento do presente trabalho são detalhados a seguir.

A vizinhança de *pixels* define quais *pixels* podem ser ditos vizinhos de outros *pixels* em uma imagem. Essa é uma informação importante, visto que algumas operações são baseadas na suposição de que determinado objeto é formado por *pixels* vizinhos uns aos outros. Seja *p* um *pixel* de uma imagem localizado no ponto (x,y), ele pode possuir uma das três vizinhanças apresentadas a seguir [11]:

- Vizinhança de 4: Representada por $N_4(p)$, é formada por todos os *pixels* que fazem fronteira de borda com *p* na horizontal ou na vertical. Ela pode ser definida conforme na equação (2.1).

$$N_4(p) = \{(x-1,y),(x+1,y),(x,y-1),(x,y+1)\} \tag{2.1}$$

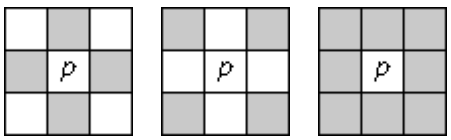
- Vizinhança diagonal: Representada por $N_D(p)$, é formada por todos os *pixels* que fazem fronteira de borda com *p* através de suas diagonais. Ela pode ser definida conforme na equação (2.2).

$$N_D(p) = \{(x-1,y-1),(x-1,y+1),(x+1,y-1),(x+1,y+1)\} \tag{2.2}$$

- Vizinhança de 8: Representada por $N_8(p)$, é formada pela união dos *pixels* vizinhos de 4 e vizinhos diagonais a *p*. Ela pode ser expressa conforme na equação (2.3).

$$N_8(p) = N_4(p) \cup N_D(p) \tag{2.3}$$

A representação visual destas vizinhanças é exibida na Figura 2.7, onde, em (a) são destacados em cinza os quatro *pixels* vizinhos de 4 de *p*; em (b) são destacados em cinza os quatro *pixels* vizinhos diagonais de *p*; e em (c) são destacados em cinza os oito *pixels* vizinhos de 8 de *p*.



(a) (b) (c)

Figura 2.7 – Representação visual das vizinhanças de um *pixel*

Outro relacionamento entre *pixels,* importante de ser analisado, é a conectividade entre eles. A conectividade de *pixels* permite dizer quais *pixels* podem ser ditos conectados uns aos outros, sendo importante na localização de regiões com

semelhança na tonalidade de níveis de cinza [11]. Sejam p e q dois *pixels* de uma mesma imagem, e seja V um conjunto denominado conjunto de conectividade composto pelos níveis de cinza que os *pixels* da imagem precisam ter para definir a conectividade. Assim, podem ser definidas as seguintes conectividades [11]:

- Conectividade de 4: p e q são ditos conectados de 4 se ambos possuem seus níveis de cinza em V e se $q \in N_4(p)$.
- Conectividade de 8 : p e q são ditos conectados de 8 se ambos possuem seus níveis de cinza em V e se $q \in N_8(p)$.
- Conectividade de m (conectividade mista) : p e q são ditos conectados de m se ambos possuem seus níveis de cinza em V e se:
  - $q \in N_4(p)$ ou
  - $q \in N_8(p)$ e $N_4(p) \cap N_4(q) = \varnothing$.

A conectividade mista é uma variação da conectividade de 8 [11]. Sua diferença reside no fato de essa não permitir a formação de ciclos entre os *pixels* que aquela permite. Assim, para determinados conjuntos de *pixels* que tenham ciclos na sua conectividade de 8, a sua conectividade mista não é única, visto que um ciclo pode formar caminhos diferentes [12]. A Figura 2.8 exibe exemplos de conectividades entre *pixels* para $V = \{1\}$, onde, em (a), tem-se a imagem a ser analisada; em (b), os *pixels* conectados de 4 estão ligados por uma linha; em (c), os *pixels* conectados de 8 estão ligados por uma linha. É possível notar na Figura 2.8(c) a criação de um ciclo nesta conectividade. Na Figura 2.8(d), tem-se uma conectividade mista para imagem; e na Figura 2.8(e) tem-se outra possível conectividade mista para a mesma imagem. É importante observar que nas conectividades mistas exibidas tanto na Figura 2.8(d) quanto na Figura 2.8(e), o ciclo formado pela conectividade de 8 não existe.

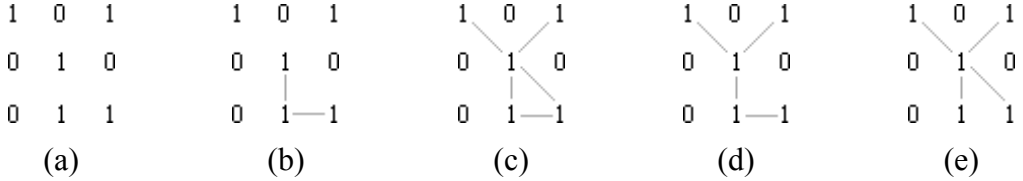


Figura 2.8 – Exemplos de conectividades de *pixels* em uma imagem

Vistos os relacionamentos básicos entre os *pixels* de uma imagem, são detalhadas agora algumas operações comumente aplicadas em imagens. Estas operações são realizadas utilizando o valor de nível de cinza dos *pixels* – $f(x,y)$, portanto, diz-se que elas trabalham no domínio espacial de uma imagem.

Uma das operações mais básicas a se aplicar em uma imagem é a operação da negação. Ela consiste em inverter os níveis de cinza de uma imagem, tornando preto os valores dos *pixels* que possuem valor branco, branco os valores dos *pixels* que possuem valor preto e assim sucessivamente para as demais cores. Uma das possibilidades de uso desta operação é quando os detalhes da imagem estão muito escuros, invertendo sua tonalidade é possível visualizar melhor esses detalhes. A operação de negação pode ser aplicada conforme na equação (2.4), onde $I(x,y)$ representa a imagem original, $G(x,y)$ representa a imagem resultante e $L_{\max}$ representa o valor máximo da escala de cinza utilizada nesta imagem.

$$G(x,y) = L_{\max} - I(x,y) \tag{2.4}$$

A Figura 2.9 exibe um exemplo de aplicação da operação negação, onde, em (a), tem-se uma imagem original e, em (b), o resultado da aplicação da negação.

(a) Imagem original [13] (b)

Figura 2.9 – Exemplo de aplicação da operação negação

Outra operação bastante utilizada é a operação de limiarização. Nesta, um limiar é escolhido e a imagem é analisada segundo ele. Um exemplo de limiarização consiste em alterar, para uma cor, o valor de todos os *pixels* que possuem nível de cinza abaixo do limiar. Já na binarização, os níveis de cinza com valores abaixo do limiar são alterados para uma cor, enquanto os valores dos outros *pixels* são alterados para outra cor. É comum utilizar a cor branca para estes *pixels*, e a cor preta para

aqueles. A Figura 2.10 mostra um exemplo de tais operações. Na Figura 2.10.(a), tem-se a imagem original; na Figura 2.10.(b), *pixels* com valores abaixo de 127 têm seus valores alterados para 0 (preto) e os valores dos demais *pixels* permanecem inalterados; e na Figura 2.10.(c), *pixels* com valores abaixo de 127 têm seus valores alterados para preto, enquanto os demais *pixels* possuem valores alterados para 255 (branco).

(a). Imagem original [13] (b) (c)

Figura 2.10 – Exemplos da aplicação das operações limiarização e binarização

A operação de alargamento de contraste, como o próprio nome diz, aumenta o contraste de uma imagem. O contraste de uma imagem é definido como a diferença de tonalidade entre o objeto e o fundo da imagem, portanto, essa operação é utilizada para melhorar a visualização de objetos em uma imagem. Para isto, uma faixa de níveis de cinza tem seu valor incrementado, enquanto duas outras faixas de níveis de cinza têm seus valores decrementados. Sejam w_1, w_2, w_3 pesos a serem aplicados em cada uma das faixas de níveis de cinza e L_1, L_2 dois limiares. Define-se então a primeira faixa de níveis de cinza como o intervalo $[0, L_1)$, a segunda como $[L_1, L_2)$ e a terceira como $[L_2, L_{\max}]$. A operação é realizada aplicando os respectivos pesos nas suas faixas correspondentes, conforme na equação (2.5), onde $I(x,y)$ representa a imagem original, $G(x,y)$ representa a imagem resultante e $L_{\max}$ representa o valor máximo da escala de cinza utilizada nesta imagem.

$$G(x,y) = \begin{cases} w_1 I(x,y), \text{ se } 0 \le I(x,y) < L_1 \\ w_2 I(x,y), \text{ se } L_1 \le I(x,y) < L_2 \\ w_3 I(x,y), \text{ se } L_2 \le I(x,y) \le L_{\max} \end{cases} \tag{2.5}$$

(a) Imagem original [13] (b)

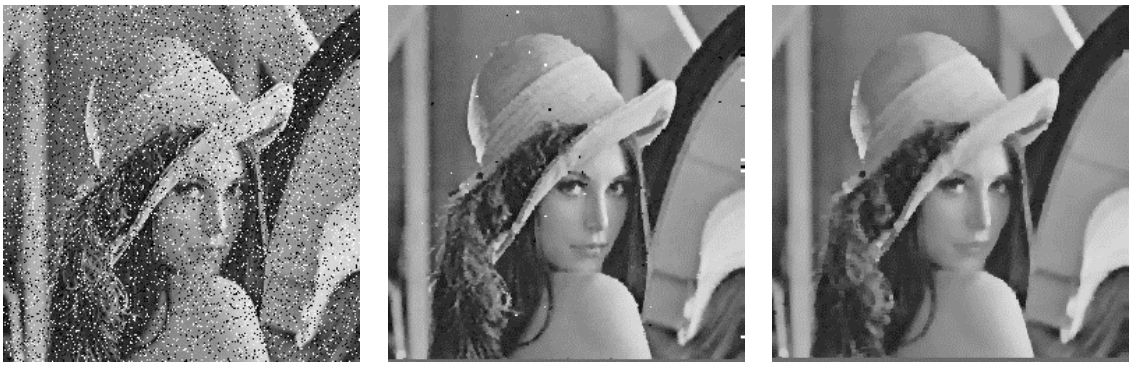
Figura 2.11 – Exemplo de aplicação da operação alargamento de contraste

A Figura 2.11 mostra um exemplo da aplicação da operação alargamento de contraste, onde foram utilizados os seguintes parâmetros: $L_1 = 60$, $L_2 = 230$, $w_1 = 0{,}7$, $w_2 = 1{,}4$ e $w_3 = 0{,}9$. É possível verificar na Figura 2.11(b) que certos elementos possuem maior contraste que na Figura 2.11(a), o que é o resultado esperado dessa operação.

As operações descritas anteriormente utilizam apenas informações de um *pixel* por vez. Essas operações são chamadas operações ponto a ponto. As operações descritas a seguir, chamadas operações sobre vizinhança de *pixels*, utilizam informações da vizinhança de cada um dos *pixels* para que a sua ação seja realizada.

Uma operação sobre vizinhança de *pixels* muito utilizada é a filtragem por mediana, que consiste em calcular o valor mediano dos valores de nível de cinza da vizinhança de um *pixel* e substituir na imagem resultante o valor desse *pixel* pelo valor mediano encontrado. Essa operação é utilizada comumente para eliminar ruídos de uma imagem, substituindo o valor do ruído por um valor próximo ao da região que circunda o *pixel*. Ela suaviza os detalhes de uma imagem e seu uso deve ser cauteloso pois ela pode eliminar tanto ruídos como pequenos detalhes da mesma. É comum nessa operação denominar vizinhanças p_A x p_A, as quais são formadas por matrizes com p_A linhas e colunas centradas no *pixel* que está sendo analisado, quando p_A é ímpar, e centradas no primeiro *pixel* da vizinhança, caso contrário [11]. A Figura 2.12 exibe exemplos de aplicações dessa operação, onde, em (a), tem-se a imagem original com ruído do tipo salpicado – ruído em que *pixels* pretos e brancos se alternam aleatoriamente pela imagem; em (b), a imagem resultante da aplicação desta operação com uma vizinhança 3x3; e em (c) a imagem resultante da aplicação

desta operação com uma vizinhança 5x5. É possível notar que na Figura 2.12(c) a quantidade de ruído é menor que na Figura 2.12(b), porém, aquela possui detalhes mais suavizados que esta. Isso ocorre pois sua vizinhança é maior, obtendo resultados mais distantes dos valores originais dos *pixels* analisados e mais próximos do valor mediano da imagem.



(a) (b) (c)

Figura 2.12 – Exemplos de aplicação da operação filtragem por mediana para diferentes tamanhos de vizinhança

Outra operação sobre vizinhança de *pixels* é o aguçamento de imagens. Essa operação tem como objetivo aguçar os detalhes de uma imagem, ao contrário da operação filtragem por mediana, que é normalmente utilizada quando a intenção é realçar os detalhes e as bordas de uma imagem. Para utilizar essa operação, pode-se usar o operador Laplaciano, definido como na equação (2.6) [11], onde $f(x,y)$ é uma imagem.

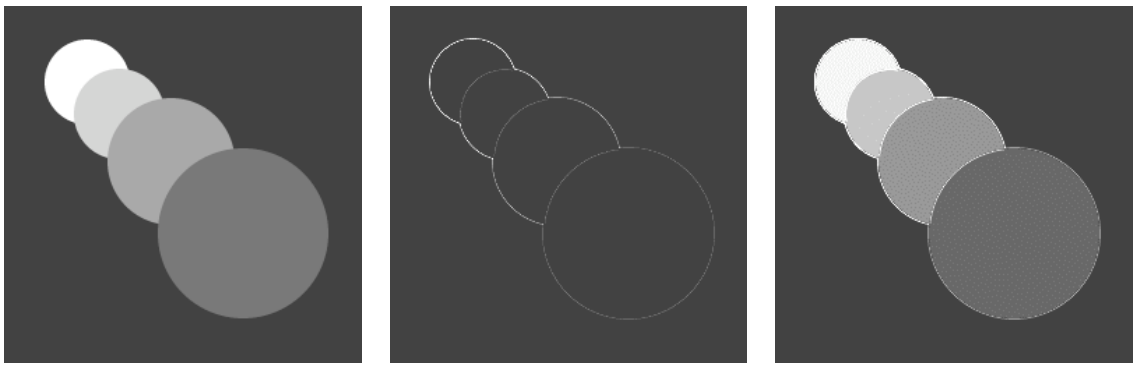
$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{2.6}$$

Se a diferenciação de uma função $g(x)$ é definida como na equação (2.7), é possível estender esta equação para uma função bidimensional $f(x,y)$. Diferenciando a função $f(x,y)$ em relação a x e a y e substituindo esses resultados na equação (2.6), obtém-se uma fórmula do operador Laplaciano aplicável a uma imagem, presente na equação (2.8).

$$\frac{\partial^2 g}{\partial x^2} = g(x+1) + g(x-1) - 2g(x) \tag{2.7}$$

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \qquad (2.8)$$

A Figura 2.13 exibe um exemplo de aplicação da operação aguçamento de imagens, onde, em (a), tem-se uma imagem original; em (b), a imagem resultante da aplicação dessa operação; e em (c) é feita a soma da imagem original com a imagem resultante, a fim de realçar as bordas e detalhes detectados através dessa operação. É possível notar na Figura 2.13(c) que as bordas da imagem na Figura 2.13(a) ficam mais realçadas se comparadas à imagem original.



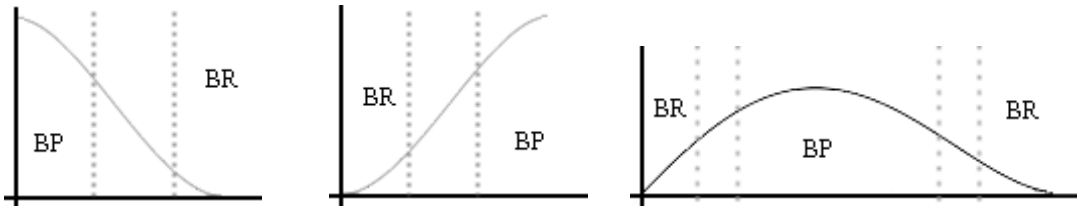
(a) (b) (c)

Figura 2.13 – Exemplo da aplicação da operação aguçamento de imagens

As operações apresentadas anteriormente possuem uma característica em comum: elas trabalham no domínio do espaço de uma imagem. Assim, o principal parâmetro de comparação dessas operações é o valor do nível de cinza dos *pixels* utilizados. Existem operações que trabalham com informações sobre a freqüência dos *pixels* de uma imagem. Para isso, é necessário transformar a imagem do domínio espacial para o domínio da freqüência, o que é feito através da aplicação de alguma transformada à imagem. Uma vez no domínio da freqüência, é possível trabalhar isoladamente com certas freqüências que ocorrem na imagem e retornar esta imagem modificada para o domínio espacial.

No domínio da freqüência é importante ressaltar que elementos de baixas freqüências são formados por zonas homogêneas de níveis de cinza da imagem, enquanto zonas com transições abruptas de níveis de cinza formam elementos de altas freqüências. Alguns exemplos de elementos de altas freqüências de uma imagem são bordas, detalhes, ruídos, entre outros.

Filtros digitais são uma série de coeficientes que podem ser aplicados a um sinal de forma a produzir um sinal modificado, de acordo com a natureza do filtro utilizado. Pode-se definir um filtro digital como uma função que permite a passagem de determinadas freqüências por ele, enquanto retém as demais freqüências. Filtros digitais podem possuir função passa-baixas, passa-altas, passa-faixas ou rejeita-faixas. Um filtro passa-baixas, como o próprio nome diz, permite que as baixas freqüências passem por ele, enquanto as altas freqüências são retidas. Tais filtros são utilizados para eliminar ruídos e também para suavizar determinadas regiões de imagens. A filtragem mediana vista anteriormente comporta-se como um filtro passa-baixas. Já um filtro passa-altas é um filtro que permite a passagem de altas freqüências, retendo as baixas freqüências. Esses filtros são utilizados para realçar detalhes ou bordas de uma imagem. A operação de aguçamento de imagens é um exemplo de filtro passa-altas. Um filtro passa-faixas é um filtro que permite a passagem de apenas determinadas faixas de freqüências, enquanto retém as demais freqüências da imagem. Esses filtros são utilizados quando se deseja realçar ou eliminar determinadas freqüências de uma imagem. A Figura 2.14 mostra exemplos da resposta em freqüência desses três tipos de filtros digitais. Dependendo do valor da freqüência (eixo das abscissas) o filtro possui uma resposta para esta (eixo das ordenadas), permitindo ou não a passagem dela pelo filtro. Na Figura 2.14(a) , tem-se um exemplo de filtro passa-baixas; na Figura 2.14(b), tem-se um exemplo de filtro passa-altas; na Figura 2.14(c), tem-se um filtro passa-faixas. Na Figura 2.14, “BR” significa banda de rejeição, “BP” significa banda de passagem, e a zona entre as linhas pontilhadas representa uma zona de transição entre a banda de rejeição e a banda de passagem.



(a) (b) (c)

Figura 2.14 – Resposta em freqüência de filtros digitais. Adaptado de [14].

Uma das transformadas mais utilizadas na literatura para fins de filtragem é a Transformada de Fourier (TF) [11]. Ela transforma um sinal, através da aplicação de

suas funções de análise – senos e cossenos – do seu domínio temporal para o domínio da freqüência. A versão original da transformada de Fourier é definida para uma função contínua unidimensional. No entanto, para aplicá-la em imagens deve-se utilizar a sua versão discretizada bidimensional. Assim, para uma imagem $f(x,y)$ de resolução *M*x*N* no domínio espacial, sua versão no domínio da freqüência $F(u,v)$ pode ser obtida aplicando-se a transformada de Fourier bidimensional direta, definida na equação (2.9) [11].

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}, \tag{2.9}$$

para $u = 0,1,2,....,M-1$, $v = 0,1,2,....,N-1$ e $j = \sqrt{-1}$.

A equação (2.9) é utilizada para transformar uma imagem do domínio espacial para o domínio da freqüência. Para transformar uma imagem do domínio da freqüência, obtida pela aplicação da equação (2.9), para o domínio espacial utiliza-se a transformada inversa de Fourier bidimensional, definida da equação (2.10) [11]:

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})}, \tag{2.10}$$

para $x = 0,1,2,....,M-1$ e $y = 0,1,2,....,N-1$.

A distribuição das freqüências na imagem no domínio da freqüência pode ser vista na Figura 2.15, onde é possível ver que as altas freqüências concentram-se no centro da imagem, e distanciando-se do centro encontra-se freqüências mais baixas. Na Figura 2.15, o menor círculo possui as maiores freqüências, o segundo menor círculo possui freqüências mais baixas em comparação ao menor círculo, e o maior círculo possui freqüências mais baixas em relação aos demais círculos. A medida em que se aumenta o raio do círculo, menor as freqüências encontradas. Em compensação, a medida em que se diminui o raio do círculo, maior as freqüências encontradas.

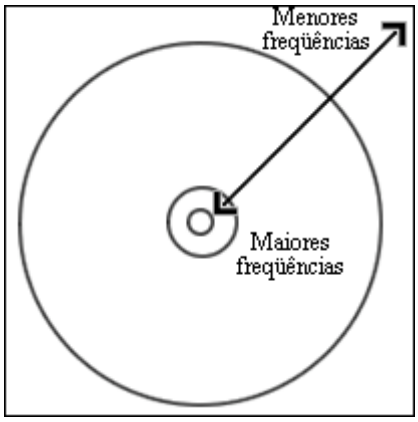


Figura 2.15 – Distribuição de freqüências da transformada de Fourier

A aplicação de filtros digitais em imagens transformadas pela transformada de Fourier baseiam-se na distribuição de freqüências mostrada na Figura 2.15. Um exemplo de um filtro passa-baixas básico consiste em zerar os elementos de altas freqüências concentrados num determinado círculo de raio *r*. Já um exemplo de filtro passa-altas básico consiste em zerar os elementos que não estejam dentro desse círculo de raio *r*. Um exemplo de filtro passa-faixa básico consiste em, dados dois círculos de raios $r_1 > r_2$, preservar os elementos que estão entre os dois círculos e zerar os demais.

Um fato importante a ser comentado sobre a transformada de Fourier é que ela possui resolução tempo-freqüência (ou espaço-freqüência) fixa [15], ou seja, uma vez que a imagem é transformada para o domínio da freqüência, não é possível saber em quais locais da imagem ocorrem as freqüências. A Figura 2.16 mostra o espectro de Fourier de uma imagem original. O espectro de Fourier é uma função comumente utilizada para visualizar imagens transformadas pela transformada de Fourier, e pode ser escrito conforme a equação (2.11) [11].

$$E(u,v) = \sqrt{\Re^2(u,v) + \Im^2(u,v)}, \tag{2.11}$$

onde $\Re(u,v)$ e $\Im(u,v)$ representam, respectivamente, as partes real e imaginária da imagem transformada.

(a) Imagem original [13] (b)

Figura 2.16 – Espectro de Fourier

É possível perceber pela Figura 2.16 que as freqüências da imagem transformada não se distribuem como os *pixels* da imagem original, não permitindo saber quais freqüências ocorrem em quais *pixels*. Tal "deficiência" da transformada de Fourier ocorre devido às suas funções de análise, que são apenas transladadas para a localização temporal a ser analisada [15].

Uma forma de descobrir quais freqüências ocorrem nos *pixels* utilizando a TF é aplicando a transformada de Fourier de tempo-curto (TFTC). Esta transformada consiste em, dada uma imagem $f(x,y)$, fatiar a imagem em pedaços menores e aplicar a transformada em cada um destes pedaços, como mostra a Figura 2.17. O resultado da aplicação da transformada em cada pedaço indica quais freqüências ocorrem neles. Assim, para saber se determinada região possui determinadas freqüências, basta verificar o resultado da aplicação desta transformada nesta fatia da imagem.

Figura 2.17 – Aplicação da transformada de Fourier de tempo-curto

A aplicação da TFTC, porém, não é simples. A principal dificuldade é na escolha do tamanho das fatias, que pode variar de aplicação para aplicação. Se for escolhido um tamanho muito grande, o problema da transformada de Fourier

permanece, impossibilitando saber quais freqüências ocorrem na região desejada. Caso seja escolhido um tamanho muito pequeno, a execução gasta tempo desnecessário para o cálculo da transformada, desperdiçando recursos computacionais.

Existem várias transformadas apresentadas na literatura para se trabalhar com imagens. A transformada *wavelet* (TW) transforma uma imagem para o domínio *wavelet*, chamado também nesse texto de domínio da freqüência, possibilitando a análise dessa através das freqüências de seus elementos. Segundo Daubechies [15], esta transformada é melhor para focar em determinadas freqüências se comparadas à transformada de Fourier. A próxima seção aborda aspectos dessa transformada, utilizada como base do trabalho desenvolvido.

2.3 Transformada *wavelet* e aplicações em imagens

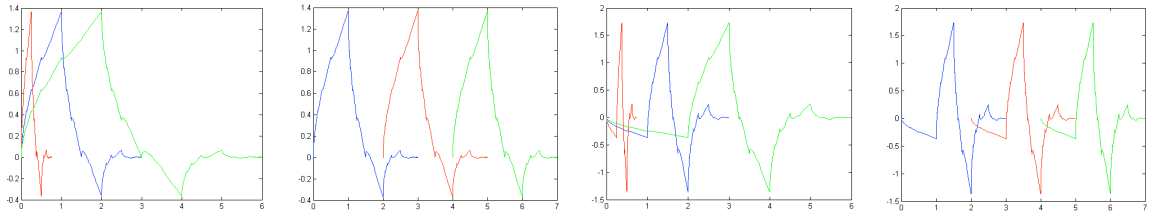
A TW é uma transformada que decompõe um sinal em bandas de diferentes freqüências; no caso da aplicação que interessa a este trabalho, este sinal é uma imagem. Essa decomposição é chamada decomposição *wavelet* [16]. O estudo dessa transformada teve início em diferentes áreas da ciência, como por exemplo, matemática, engenharia, física, entre outras, e apenas a partir de 1983 houve uma síntese de todas as abordagens diferentes, dando origem ao conhecimento atual que possuímos [17].

A decomposição *wavelet* acontece com base na aplicação de dois tipos distintos de funções: as *wavelets* e as funções de escala, sendo que as *wavelets* são filtros digitais passa-altas e as funções de escala são filtros digitais passa-baixas.

Duas operações matemáticas são utilizadas nas *wavelets* e nas funções de escala: a dilatação e a translação. A dilatação é uma operação que contrai ou dilata uma função através de multiplicações ou divisões de constantes aos valores do eixo temporal. Já a translação é uma operação que move uma função ao longo do eixo temporal através de adições ou subtrações de constantes aos valores do eixo temporal.

As *wavelet*s são obtidas a partir de dilatações e translações de uma *wavelet* original, chamada *wavelet*-mãe [16]. De forma semelhante, as funções de escala são obtidas a partir de dilatações e translações de uma outra função original, denominada

função de escala pai ou *wavelet*-pai. Essas dilatações e translações das funções originais são feitas de forma a obter funções mais adequadas ao sinal que está sendo analisado em comparação às versões originais. A Figura 2.18 mostra um exemplo dessas operações sobre uma *wavelet*-mãe e uma função de escala pai. Na Figura 2.18, em (a), tem-se uma *wavelet*-mãe dilatada e em (b), a mesma é transladada ao longo do eixo *x*; em (c), tem-se uma função de escala dilatada e em (d) a mesma é transladada ao longo do eixo *x*. Essa adequação realizada na *wavelet*-mãe e na função de escala pai torna a resolução tempo freqüência desta transformada não fixa, pois as suas funções de análise possuem comprimento temporal adaptados à suas freqüências. Assim, *wavelets* e funções de escala de altas freqüências são estreitas, enquanto *wavelets* e funções de escala de baixas freqüências são amplas [15]. Para efeito de comparação, a transformada de Fourier – que possui resolução tempo-freqüência fixa – possui suas funções de análise (senos e cossenos) apenas transladadas ao longo do sinal a ser analisado [15], evidenciando sua resolução tempo freqüência.



(a) (b) (c) (d)

Figura 2.18 – Exemplos de dilatações e translações de uma *wavelet*-mãe e uma função de escala pai

Seja $\psi(t)$ uma *wavelet* mãe, pode-se escrever uma *wavelet* derivada daquela conforme a equação (2.12) [16].

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a}), \tag{2.12}$$

onde a e b são, respectivamente, os parâmetros de dilatação e translação da *wavelet*-mãe $\psi(t)$ [16].

Seja $\phi(t)$ uma função de escala pai, pode-se escrever uma função de escala derivada daquela conforme a equação (2.13) [16].

$$\phi_{a,b}(t) = \frac{1}{\sqrt{a}}\phi(\frac{t-b}{a}), \tag{2.13}$$

onde *a* e *b* são, respectivamente, os parâmetros de dilatação e translação da função de escala pai $\phi(t)$ [16].

A aplicação de um conjunto de *wavelets* gerado a partir de uma *wavelet*-mãe em um sinal gera um novo sinal contendo os detalhes daquele, visto que funções *wavelets* possuem comportamento de um filtro passa-altas. A aplicação de um conjunto de funções de escala gerado a partir de uma função de escala pai em um sinal gera um novo sinal contendo uma versão suavizada daquele [17]. Seja $x(t)$ um sinal finito, e seja $a = a_0^m$ e $b = nb_0a_0^m$ valores discretos dos parâmetros de dilatação e translação das *wavelet*-mãe e função de escala pai. Reescrevendo as equações (2.12) e (2.13) em função dos parâmetros *n* e *m*, obtém-se, respectivamente, as equações (2.14) e (2.15).

$$\psi_{m,n}(t) = \frac{1}{\sqrt{a_0^m}}\psi\left(\frac{t - nb_0a_0^m}{a_0^m}\right), \tag{2.14}$$

$$\phi_{a,b}(t) = \frac{1}{\sqrt{a_0^m}}\phi\left(\frac{t - nb_0a_0^m}{a_0^m}\right), \tag{2.15}$$

Para obter os novos sinais de altas e baixas freqüências, basta fazer o produto das *wavelets* e funções de escalas obtidas nas equações (2.14) e (2.15) com o sinal original $x(t)$, conforme nas equações (2.16) e (2.17), respectivamente.

$$T_{m,n} = \int_{-\infty}^{\infty} x(t)\psi_{m,n}(t)dt \tag{2.16}$$

$$S_{m,n} = \int_{-\infty}^{\infty} x(t)\phi_{m,n}(t)dt, \tag{2.17}$$

onde $T_{m,n}$ e $S_{m,n}$ representam, respectivamente, o sinal de detalhe e o sinal suavizado obtidos a partir da decomposição *wavelet* do sinal $x(t)$. A Figura 2.19 ilustra esta decomposição, onde o sinal original é decomposto no sinal de detalhes e no sinal suavizado, sendo que estes possuem metade do tamanho do sinal original.

Figura 2.19 – Decomposição *wavelet* de um sinal

Tomando o sinal original como o primeiro sinal de suavização, é possível fazer a decomposição *wavelet* nos sinais suavizados encontrados. Seja a escala da decomposição *wavelet* definida por *m*, então $m = 0$ para o sinal original. Decompondo este sinal gera-se dois sinais a uma escala $m = 1$. Decompondo o sinal suavizado dessa escala gera-se dois novos sinais (um suavizado e um de detalhes) a uma escala $m = 2$, e assim sucessivamente, enquanto o sinal suavizado da última escala for divisível por dois. Esse processo também é chamado de análise multiresolução [16], pois é possível analisar as freqüências de um sinal em diferentes resoluções (escalas).

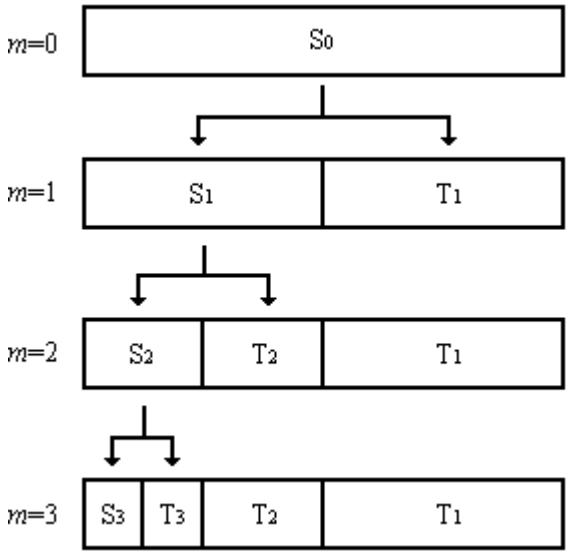


Figura 2.20 – Decomposição *wavelet* em várias escalas

A Figura 2.20 mostra um exemplo da decomposição *wavelet* realizada em várias escalas. O sinal original (S_0) é decomposto em um sinal suavizado (S_1) e um sinal de detalhes (T_1) em uma escala 1. Em seguida, o sinal suavizado da escala 1 é decomposto em um sinal suavizado (S_2) e um sinal de detalhes (T_2) a uma escala 2. O sinal suavizado da escala 2 é então decomposto em um sinal suavizado (S_3) e um sinal de detalhes (T_3) a uma escala 3, dando fim à decomposição.

Como dito anteriormente, a decomposição *wavelet* separa um sinal em novos sinais de diferentes freqüências. Assim, para o sinal exibido na Figura 2.20, pode-se dizer que T_1 possui as maiores freqüências entre os sinais decompostos, seguido por

T_2, T_3 e finalmente S_3, sendo que este possui as menores freqüências entre os sinais decompostos.

De acordo com o teorema da amostragem [18], se *s* é um sinal com freqüência de amostragem f_s então ele possui freqüência máxima $f_w = \frac{f_s}{2}$. Ao aplicar a transformada *wavelet* em um sinal são gerados dois novos sinais através da redução da taxa de amostragem do sinal original (*downsampling*), e por terem cada um a metade da taxa de amostragem do sinal original, eles possuem freqüências máximas conhecidas.

Uma vez decomposto o sinal original até a escala desejada, é possível aplicar operações sobre suas diferentes componentes. Após realizar as alterações desejadas, o sinal é recomposto para o seu domínio original, onde podem ser verificadas as alterações realizadas no domínio da freqüência.

Aplicações da decomposição *wavelet* em um sinal unidimensional, como apresentado anteriormente, podem ser utilizadas em qualquer sinal que possua tal característica, como sinais de áudio, vetores unidimensionais de caracteres, entre outras. Suas aplicações incluem, entre outras, codificação de áudio [19] e identificação patológica de sinais de voz [20].

A aplicação da transformada *wavelet* em imagens e outros sinais bidimensionais utiliza o mesmo conceito apresentado anteriormente. No entanto, como uma imagem possui duas dimensões espaciais, quatro novas imagens são geradas ao realizar a decomposição *wavelet*: uma imagem suavizada, uma imagem suavizada na horizontal com detalhes na vertical, uma imagem com detalhes na horizontal e suavizada na vertical e uma imagem com detalhes na horizontal e na vertical, também dita imagem com detalhes na diagonal. A decomposição *wavelet* de uma imagem pode ser visualizada conforme exibido na Figura 2.21, onde S_0 é a imagem original, S_m representa a imagem suavizada a uma escala *m*, T_m^h representa a imagem de detalhes horizontais a uma escala *m*, T_m^v representa a imagem de detalhes verticais a uma escala *m*, T_m^d representa a imagem de detalhes diagonais a uma escala *m*, LP significa *low pass* (funções de escala) e HP significa *high pass* (*wavelets*). Para gerar a imagem suavizada da próxima escala, são aplicadas à imagem suavizada da escala anterior funções de escala tanto em suas linhas quanto em suas colunas. Para gerar a imagem com detalhes horizontais da próxima escala, são aplicadas à

imagem suavizada da escala anterior funções de escala em suas linhas e *wavelets* em suas colunas. Para gerar a imagem com detalhes verticais da próxima escala, são aplicadas à imagem suavizada da escala anterior *wavelets* em suas linhas e funções de escala em suas colunas. E para gerar a imagem com detalhes diagonais, são aplicadas *wavelets* nas linhas e colunas da imagem suavizada da escala anterior.

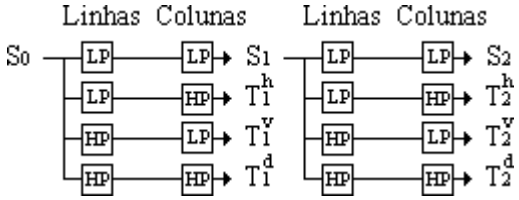


Figura 2.21 – Decomposição *wavelet* de uma imagem

A Figura 2.22 mostra a distribuição de freqüências da transformada *wavelet* em uma imagem, onde é possível observar que as imagens de freqüências diferentes ficam separadas umas das outras. Além disso, as imagens resultantes preservam a localização espacial dos *pixels* da imagem original no domínio da freqüência em todas as imagens resultantes, tornando mais fácil verificar se determinada freqüência ocorre em determinado *pixel* com o uso da transformada *wavelet* do que com a transformada de Fourier, conforme visto anteriormente.

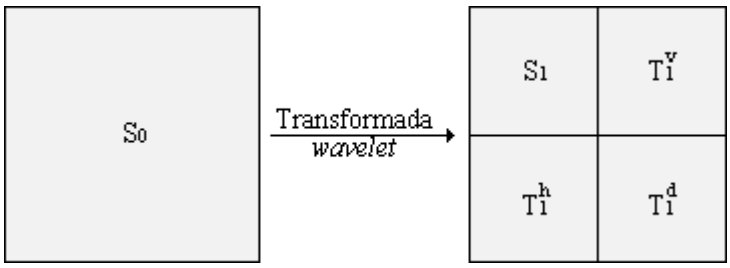


Figura 2.22 – Distribuição de freqüências da transformada *wavelet* em uma imagem

Assim como para sinais de uma dimensão – onde a transformada *wavelet* pode ser aplicada ao sinal de suavização da última escala enquanto este for divisível por dois – é possível aplicar a transformada *wavelet* em uma imagem enquanto a imagem suavizada obtida na última escala for divisível por dois. Tomando a imagem original como a primeira imagem suavizada e aplicando a transformada *wavelet* sempre na última imagem suavizada obtida, é feita a decomposição *wavelet* de uma imagem, gerando quatro novas imagens a cada iteração. A Figura 2.23 exibe um

exemplo de aplicação da transformada *wavelet* em uma imagem até o nível quatro, onde as porções mais claras das imagens representam as componentes de baixas freqüências, enquanto as porções mais escuras das imagens representam as componentes de altas freqüências.

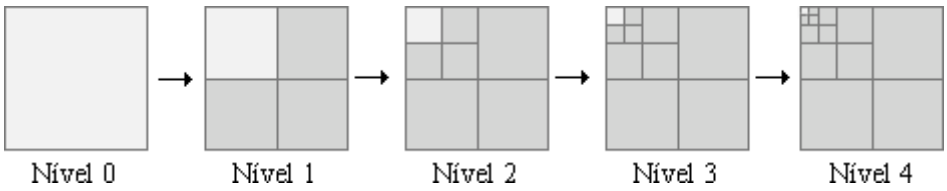


Figura 2.23 – Decomposição *wavelet* de uma imagem até o nível quatro

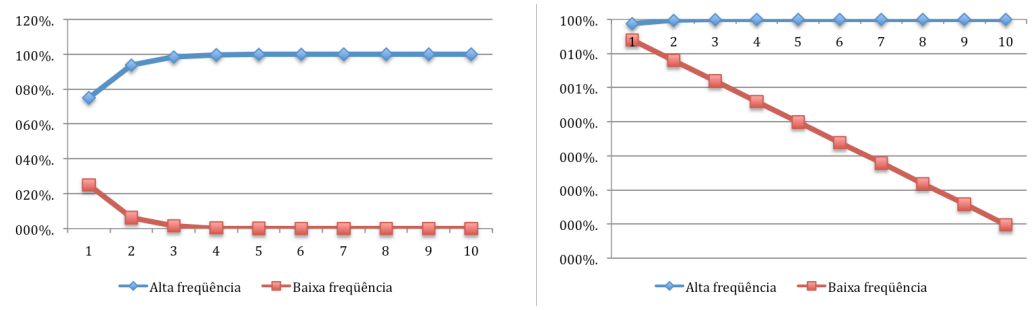
Pela Figura 2.23 é possível perceber que a quantidade de elementos de altas freqüências aumenta a cada nível da transformada *wavelet*. A Tabela 2.3 exibe a quantidade de elementos de altas freqüências presentes na imagem após a aplicação em cada nível da transformada.

Tabela 2.3 – Quantidade de elementos de altas freqüências em cada nível da decomposição *wavelet* de uma imagem

Nível	Quantidade (%)
1	75,00
2	93,75
3	98,44
4	99,61
5	99,90

Os dados da Tabela 2.3 são exibidos em forma de gráfico na Figura 2.24, sendo que, em ambos os gráficos dessa figura, a série clara representa a quantidade de elementos de baixas freqüências da imagem e a série escura representa a quantidade de elementos de altas freqüências na imagem. Na Figura 2.24(a), tem-se o gráfico em escala decimal, onde é possível notar que a partir do quarto nível, a quantidade de elementos de altas freqüências pouco aumenta em relação à sua quantidade no nível anterior, o mesmo acontecendo para os elementos de baixas freqüências. Na Figura 2.24(b), tem-se o mesmo gráfico, porém, em escala logarítmica. É possível notar por esse gráfico que, apesar da quantidade de elementos de altas freqüências pouco aumentar em relação à mesma quantidade no nível

anterior, a quantidade de elementos de baixas freqüências possui perda significativa em relação à mesma quantidade no nível anterior, pois como a quantidade de elementos de baixas freqüências está sempre diminuindo a cada nível, a perda de 75% de seus elementos a cada nível faz com que esta perda seja considerável. Ao mesmo tempo, por a quantidade de elementos de altas freqüências ser grande, a adição de poucos elementos a cada nível em relação ao nível anterior pouco altera sua porcentagem.



(a) (b)

Figura 2.24 – Quantidade de elementos de altas freqüências em cada nível da decomposição *wavelet* de uma imagem

Ao escolher até qual nível a transformada *wavelet* deve ser aplicada, é necessário avaliar tanto a quantidade de elementos de altas freqüências quanto a quantidade de elementos de baixas freqüências gerados na imagem resultante, a fim de chegar a um resultado satisfatório para a aplicação realizada. Um exemplo da importância dessa escolha é exibido na Figura 2.25, onde deseja-se detectar as bordas de uma imagem. Como as bordas de uma imagem são formadas por elementos de altas freqüências, a transformada *wavelet* é aplicada, zera-se a imagem de baixas freqüências e reconstrói-se a imagem resultante, gerando a imagem com as bordas detectadas.

Na Figura 2.25, em (a), tem-se a imagem da qual deseja-se detectar as bordas; em (b), o resultado da aplicação da operação descrita anteriormente aplicando-se a transformada *wavelet* até o nível 2; e em (c), o resultado da aplicação descrita anteriormente aplicando-se a transformada *wavelet* até o nível 8. Através desse exemplo é possível perceber que as bordas foram melhor detectadas pela aplicação da transformada *wavelet* até o nível 2, do que por sua aplicação até o nível 8. Isso deve-se ao fato de que, aplicando a transformada até o nível 2, apenas uma quantidade de elementos de altas freqüências permanecem na imagem, enquanto os

demais elementos são descartados. Ao mesmo tempo, quando se aplica a transformada até o nível 8, vários elementos de freqüências mais baixas que as freqüências das bordas permanecem na imagem, fazendo com que regiões de baixas freqüências apareçam na imagem resultante, o que prejudica a detecção de bordas, demonstrando assim a importância de se escolher de forma adequada o nível de aplicação transformada.

(a) Imagem original [13] (b) (c)

Figura 2.25 – Exemplo de aplicação da transformada *wavelet*

Até agora durante o texto foi visto que a transformada *wavelet* pode ser aplicada em um sinal unidimensional ou bidimensional utilizando duas funções: *wavelets* e funções de escala. Para a aplicação dessa transformada nesses sinais é necessário escolher um par formado por uma *wavelet*-mãe e uma função de escala pai de forma a gerar as funções *wavelets* e funções de escalas utilizadas para decompor os sinais. Nesse texto, este par é denominado *wavelet*.

Diversas *wavelets* foram propostas na literatura e suas aplicações variam dependendo do sinal a ser analisado e da finalidade de sua aplicação. As *wavelets* podem possuir características diferentes umas das outras, o que pode auxiliar a escolha da *wavelet* a ser utilizada e influenciar os resultados obtidos por cada uma delas. As características analisadas das *wavelets* neste trabalho são:

i. Resposta em freqüência;
ii. Resposta em fase;
iii. Formatos da *wavelet*-mãe e função de escala pai,

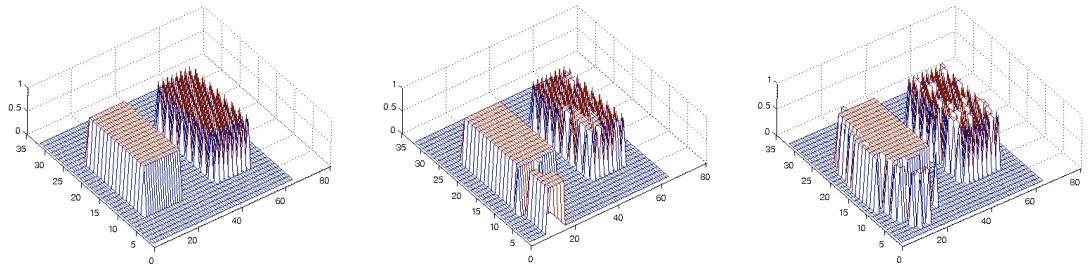
sendo que as características i e ii são importantes no momento da aplicação da transformada, enquanto a característica iii é importante no momento da reconstrução do sinal analisado [21].

A resposta em freqüência de um filtro digital, conforme visto na seção 2.2, indica a resposta do filtro às freqüências que o mesmo é submetido. Um filtro digital ideal possui resposta em freqüência à banda de passagem constante e zona de transição nula, ou seja, a transição entre a banda de rejeição e a banda de passagem é imediata. As *wavelets* utilizadas no presente trabalho possuem diferenças quanto a resposta em freqüência, que podem ocasionar diferenças no resultado de suas aplicações em um sinal.

Um filtro digital pode ser classificado em dois tipos: filtros que possuem resposta finita ao impulso – *Finite Impulse Response* (FIR) – e filtros que possuem resposta infinita ao impulso – *Infinite Impulse Response* (IIR). Define-se impulso como seqüências de amostras formadas por conjuntos contendo o valor um na sua primeira posição e valor zero nas demais [14]. Seja Q um conjunto infinito tal que $Q = \{1,0,0,0,0,...,0\}$. Um filtro digital é dito do tipo FIR se sua resposta ao impulso for finita, ou seja, ao aplicar o filtro ao impulso o resultado obtido é um sinal finito contendo os coeficientes do próprio filtro e valores zero nas posições em que o mesmo não possui coeficientes. Filtros do tipo FIR podem ter resposta em fase linear ou não-linear. Seja F um filtro digital do tipo FIR, F possui resposta em fase linear se seus coeficientes são simétricos ou anti-simétricos uns aos outros; caso contrário, F possui resposta em fase não-linear. Todas as *wavelets* utilizadas no presente trabalho são filtros digitais do tipo FIR.

O formato da *wavelet*-mãe e função de escala pai é uma característica importante na reconstrução do sinal decomposto pela transformada *wavelet*, pois ao reconstruir o sinal do domínio da freqüência para o domínio original do sinal, esse é reconstruído como uma combinação linear da *wavelet*-mãe e função de escala pai [21]. Pode-se dizer, então, que o sinal resultante da reconstrução é formado por combinações dessas duas funções de forma a representar o sinal original. Se o sinal analisado não for perturbado, a reconstrução do sinal decomposto por qualquer *wavelet* gera o sinal original. Porém, se o sinal analisado for perturbado no domínio da freqüência, então o sinal resultante da reconstrução pode ser melhor representado pelo resultado da reconstrução que utiliza uma *wavelet* com formato mais parecido com o formato do sinal original. Um exemplo disso pode ser visualizado na Figura 2.26, sendo que nessa figura em (a) tem-se a representação em três dimensões de uma imagem bidimensional, sendo a terceira dimensão uma representação da

intensidade de níveis de cinza da imagem. A imagem presente na Figura 2.26(a) possui uma região formada por um retângulo e outra formada por picos. A transformada *wavelet* é aplicada na imagem original até o quarto nível e, em seguida, a cada *pixel* um número aleatório é gerado e é multiplicado a esse *pixel* da imagem no domínio da freqüência, e a imagem resultante é então reconstruída. Na Figura 2.26(b) esse procedimento é aplicado utilizando uma *wavelet* com formatos de *wavelet*-mãe e função de escala pai que se assemelham a uma função quadrada, enquanto na Figura 2.26(c) o mesmo procedimento é aplicado utilizando uma *wavelet* com formatos de *wavelet*-mãe e função de escala pai com pontas estreitas. É possível visualizar na Figura 2.26(b) que ao aplicar a primeira *wavelet*, a imagem resultante apresenta, na região que forma um retângulo, imprecisões – devidas à multiplicação por números aleatórios – com formatos quadrados, devido ao formato da *wavelet* utilizada, sendo que o mesmo ocorre para a regiões com picos. Na Figura 2.26(c) é possível verificar que a utilização da segunda *wavelet* adiciona perturbações com formatos de picos na região que forma o retângulo da imagem, e os picos são melhores reconstruídos utilizando essa *wavelet* do que a primeira *wavelet*, devido ao formato da segunda *wavelet* ser mais parecido com picos do que a primeira.



(a) (b) (c)

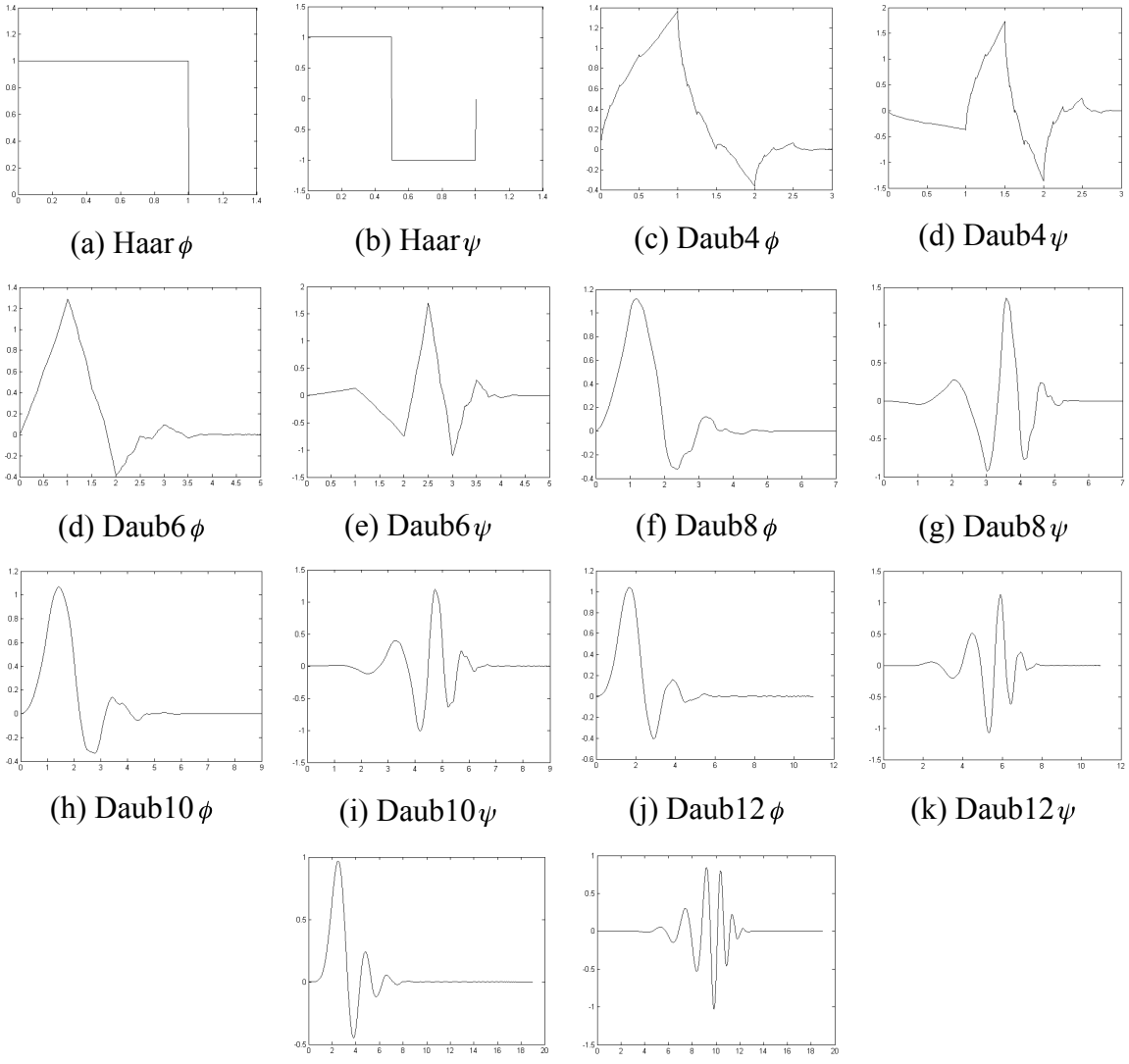
Figura 2.26 – Aplicação de diferentes *wavelets* em uma imagem

Outra característica das *wavelets* é o seu valor de suporte. Define-se o suporte de uma *wavelet* como o número de coeficientes não nulos da mesma [17]. Dessa forma, uma *wavelet* possui suporte compacto se ela possui suporte finito [15]. Todas as *wavelets* utilizadas no presente trabalho possuem suporte compacto.

As *wavelets* utilizadas no presente trabalho são descritas a seguir.

a) *Wavelets* de Daubechies

Foram propostas por Ingrid Daubechies e formam uma família de *wavelets* muito utilizada na literatura. *Wavelets* pertencentes a essa família são representadas por Daub$K$, onde K indica o valor do suporte da *wavelet* [15]. Esta família possui *wavelets* variando o suporte de 2 em 2, ou seja, Daub4, Daub6, Daub8, etc. Um caso interessante é que a Daubechies com suporte 2 também é conhecida como *wavelet* de Haar, uma das *wavelets* mais simples apresentadas na literatura, também conhecida como transformada de Haar [17]. A Figura 2.27 exibe as funções de escala pai e *wavelets*-mãe de algumas das *wavelets* presentes nessa família, onde pode ser observada a diferença entre o formato destas funções, onde ϕ representa a função de escala pai e ψ representa a *wavelet*-mãe.



(a) Haar ϕ (b) Haar ψ (c) Daub4 ϕ (d) Daub4 ψ

(d) Daub6 ϕ (e) Daub6 ψ (f) Daub8 ϕ (g) Daub8 ψ

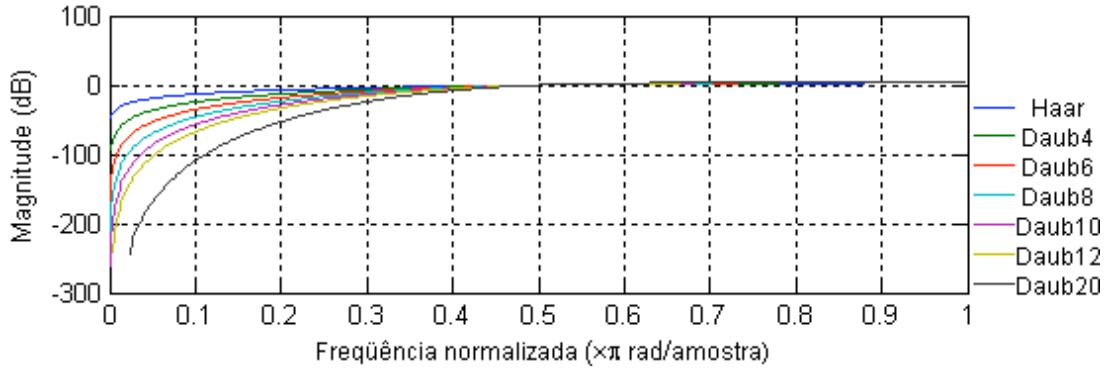
(h) Daub10 ϕ (i) Daub10 ψ (j) Daub12 ϕ (k) Daub12 ψ

(l) Daub20 ϕ (m) Daub20 ψ

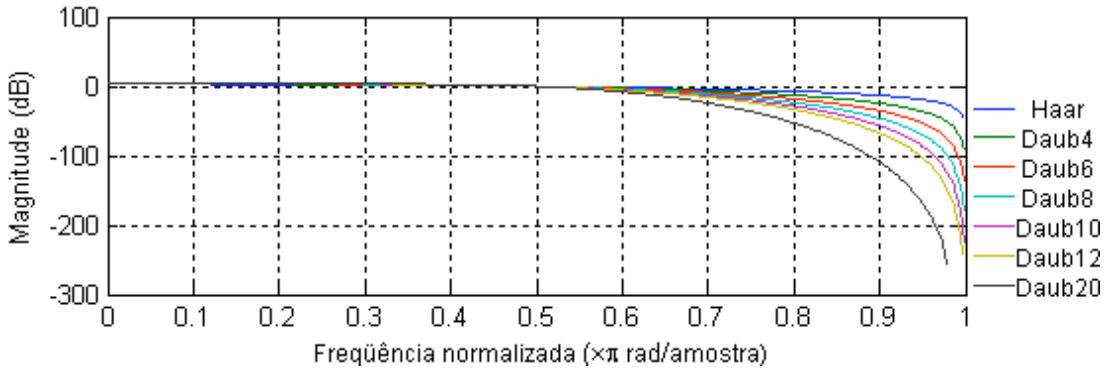
Figura 2.27 – Funções de escala pai e *wavelets*-mãe de Daubechies

Observando a Figura 2.27 pode-se notar que a *wavelet* de Haar possui formato aproximado de uma função quadrada, enquanto Daub4 possui formato com pontas estreitas. Voltando ao exemplo da Figura 2.26 , em (b) foi aplicada a *wavelet* de Haar, enquanto na em (c) foi aplicada Daub4. Assim, verifica-se a importância do formato das funções de escala pai e *wavelet*-mãe na reconstrução do sinal decomposto pela transformada *wavelet*. É possível verificar ainda pela Figura 2.27 que quanto maior o valor do suporte de uma *wavelet* desta família, mais as funções de escala pai e *wavelets*-mãe possuem formato arredondado em suas funções, o que pode prejudicar a reconstrução de um sinal que possua pontas estreitas por essas *wavelets*.

A Figura 2.28 exibe as respostas em freqüências das *wavelets* de Daubechies das *wavelets*-mãe em (a) e das funções de escala pai em (b), onde é possível verificar que *wavelets* dessa família com maior suporte possuem melhor resposta em freqüência do que *wavelets* dessa família com menor suporte.



(a)



(b)

Figura 2.28 – Resposta em freqüência das *wavelets* da Daubechies

A Figura 2.30 exibe a resposta em fase de Daub12, onde é possível observar que ela possui resposta em fase não-linear. Essa característica se estende para todas as *wavelets* de Daubechies, exceto para a *wavelet* de Haar, cuja resposta em fase linear é exibida na Figura 2.29.

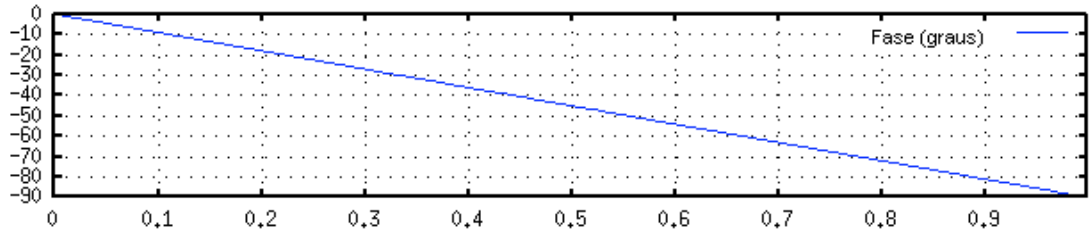


Figura 2.29 – Resposta em fase da *wavelet* de Haar

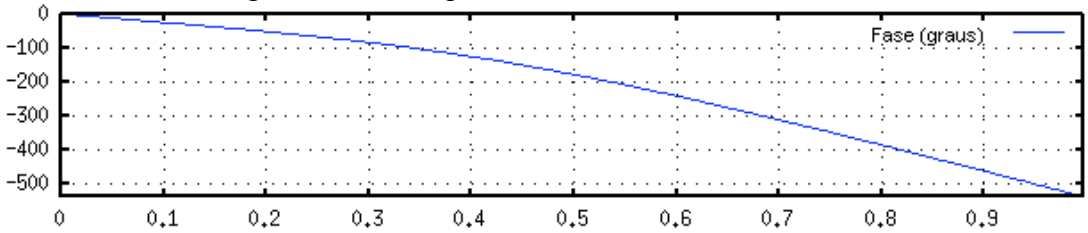
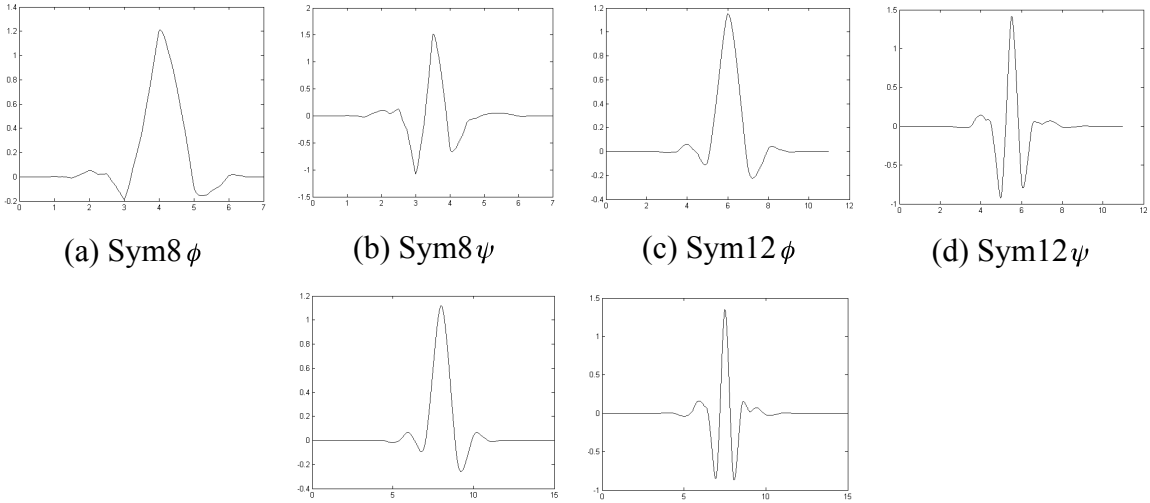


Figura 2.30 – Resposta em fase de Daub12

b) Symmlets

A família de *wavelets* Symmlets foi desenvolvida por Ingrid Daubechies como uma modificação de suas *wavelets* criadas anteriormente, de forma a torná-las mais simétricas [16]. Essa família de *wavelets* é normalmente representada por Sym*N*, onde *N* indica o suporte da *wavelet*. A Figura 2.31 exibe os formatos das funções de escala pai e *wavelets*-mãe de algumas das *wavelets* pertencentes a esta família, fator importante, como dito anteriormente, na etapa de reconstrução de um sinal decomposto pela transformada *wavelet*. É possível verificar pela Figura 2.31 que a medida que o suporte das *wavelets* pertencentes a esta família aumenta, as pontas de suas *wavelets*-mães ficam mais estreitas, porém, com menores valores de amplitude, o que pode alterar a forma e a amplitude dos elementos presentes em um sinal decomposto por estas *wavelets*.

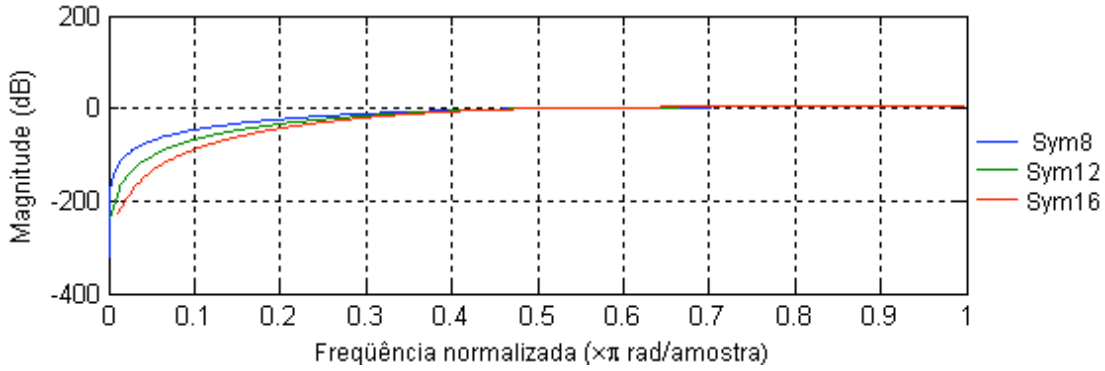


(a) Sym8 ϕ (b) Sym8 ψ (c) Sym12 ϕ (d) Sym12 ψ

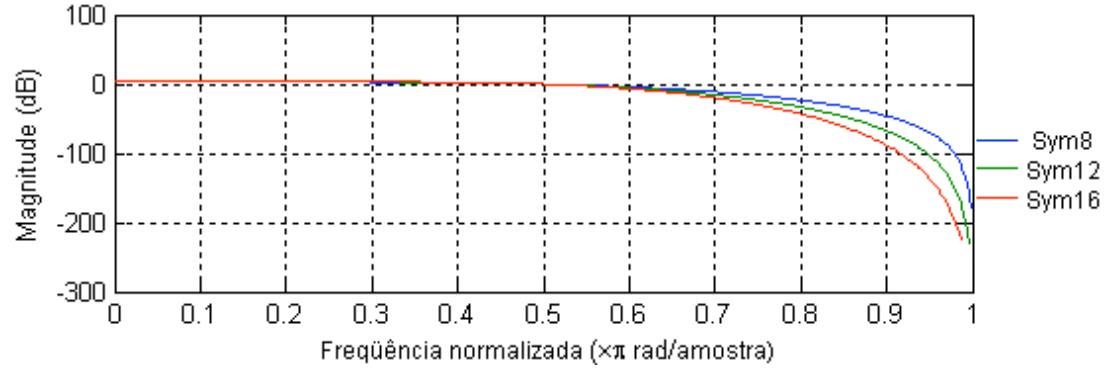
(e) Sym16 ϕ (f) Sym16 ψ

Figura 2.31 – Funções de escala pai e *wavelets*-mãe de Symmlets

A Figura 2.32 exibe as respostas em freqüências das Symmlets das *wavelets*-mãe em (a) e das funções de escala pai em (b), onde é possível verificar que quanto maior é o suporte de *wavelets* dessa família, melhor é a resposta em freqüência delas.



(a)



(b)

Figura 2.32 – Resposta em freqüência das Symmlets

A Figura 2.33 exibe a resposta em fase de Sym12, onde é possível observar que a mesma possui resposta em fase quase-linear. Essa característica se estende para todas as Symmlets.

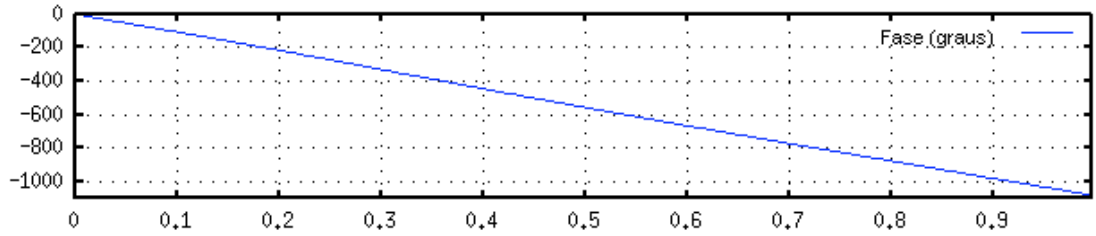
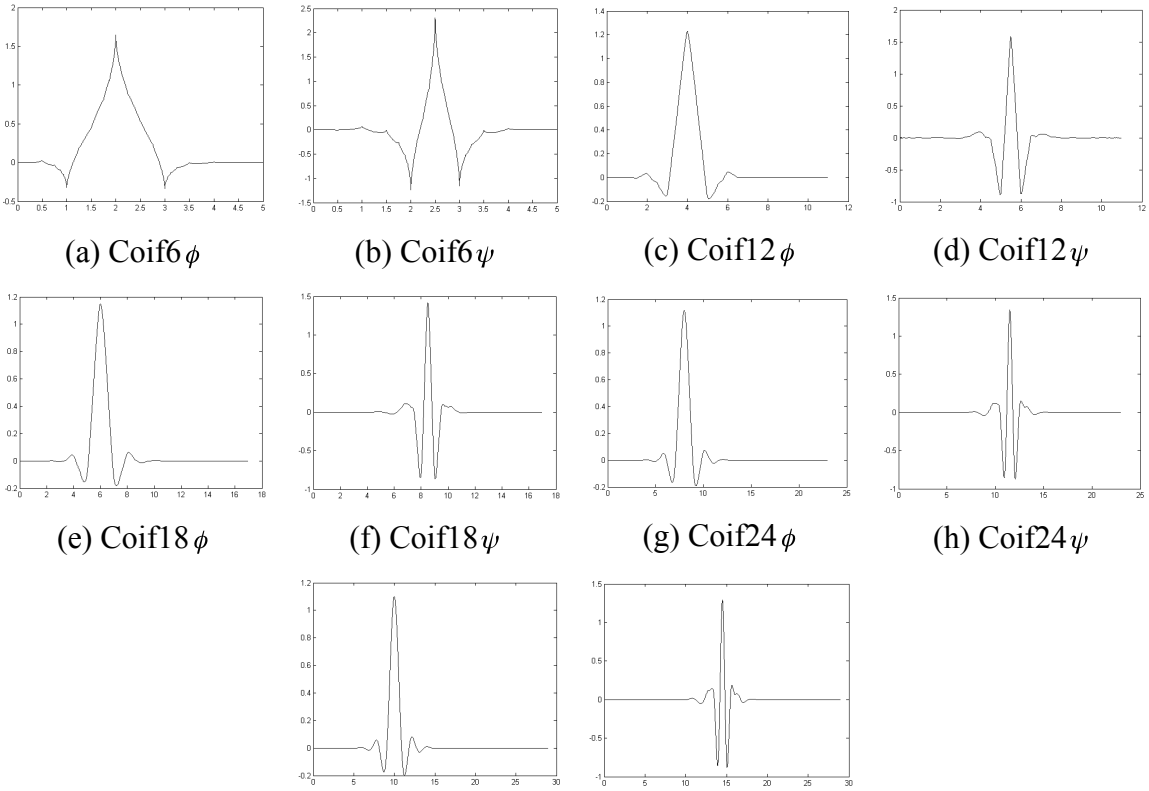


Figura 2.33 – Resposta em fase da *wavelet* Symmlets12

c) Coiflets

As Coiflets formam outra família de *wavelets* criada por Ingrid Daubechies visando encontrar *wavelets* mais simétricas que as de Daubechies [16]. Elas são normalmente representadas por Coif$K$, onde K é o suporte da *wavelet*. A Figura 2.34 exibe as funções de escala pai e *wavelets*-mãe de algumas das *wavelets* pertencentes a essa família, onde é possível observar que quanto maior o suporte das Coiflets, menos estreitas ficam as pontas dessas funções e menor é a sua amplitude.



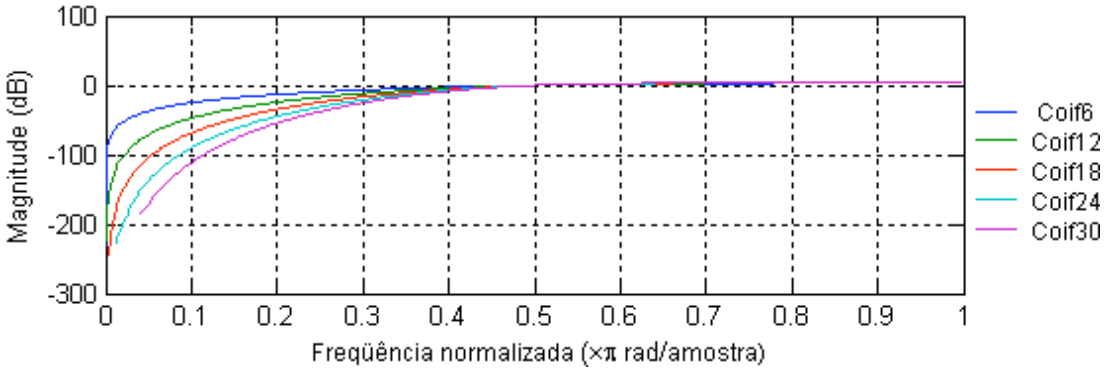
(a) Coif6 ϕ (b) Coif6 ψ (c) Coif12 ϕ (d) Coif12 ψ

(e) Coif18 ϕ (f) Coif18 ψ (g) Coif24 ϕ (h) Coif24 ψ

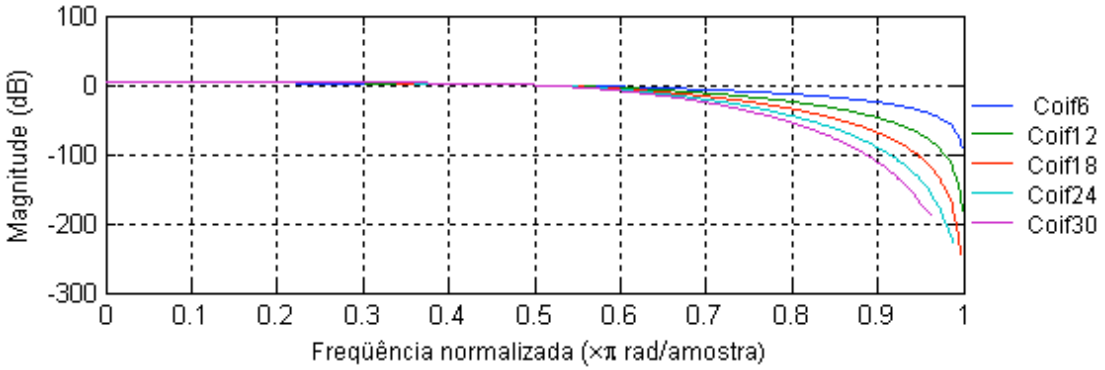
(i) Coif30 ϕ (j) Coif30 ψ

Figura 2.34 – Funções de escala pai e *wavelets*-mãe de Coiflets

A Figura 2.35 exibe as respostas em freqüências das Coiflets das *wavelets*-mãe em (a) e das funções de escala pai em (b), onde é possível verificar que quanto maior o suporte da Coiflet, melhor a resposta em freqüência dela.



(a)



(b)

Figura 2.35 – Resposta em freqüência das Coiflets

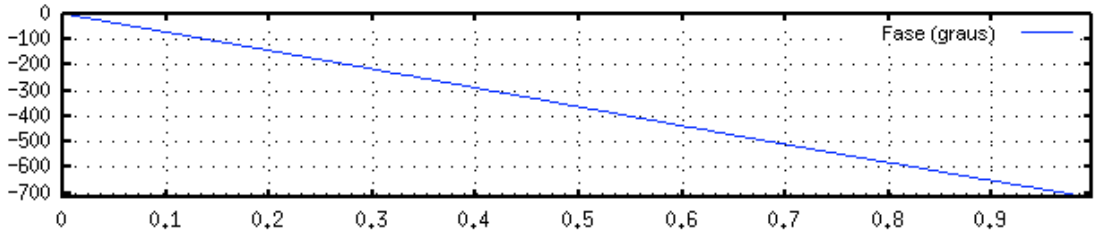


Figura 2.36 – Resposta em fase de Coif12

A Figura 2.36 exibe a resposta em fase de Coif12, onde é possível observar que ela possui resposta em fase quase linear. Essa característica se estende para todas as Coiflets. A Tabela 2.4 exibe a resposta em fase das *wavelets* utilizadas no presente trabalho.

Tabela 2.4 – Resposta em fase das *wavelets* utilizadas

Wavelet	Resposta em fase
Haar	Linear
Daubechies	Não-linear
Symmlets	Quase linear
Coiflets	Quase-linear

Foi visto nesse capítulo a fundamentação teórica necessária para o entendimento do trabalho desenvolvido. O próximo capítulo apresenta o método proposto e o projeto desenvolvido.

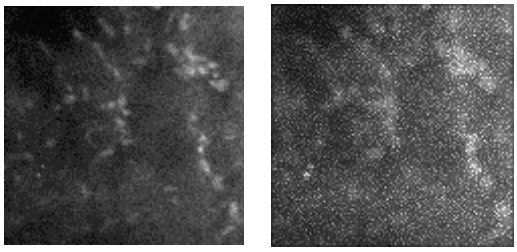
Capítulo 3 – Método proposto

Este capítulo apresenta o funcionamento do método proposto, bem como detalhes de sua implementação, os resultados obtidos e outras características suas. As imagens de microcalcificações são elementos de altas freqüências presentes em uma imagem [22], pois são regiões claras com certo grau de diferença de tonalidade com o tecido da mama, embora essa diferença não seja tão grande a ponto de tornar estes elementos fáceis de serem visualizados a olho nu. Assim, pode-se utilizar um filtro digital do tipo passa-altas para filtrar a imagem desses elementos de uma mamografia digital. Somando a imagem original com a imagem de altas freqüências encontrada, é possível realçar a imagem de elementos de altas freqüências, o que pode realçar as imagens das microcalcificações. Um dos filtros digitais passa-altas mais simples que pode ser utilizado é a operação de aguçamento de imagens, vista no capítulo 2. Uma esquematização de uma abordagem utilizando essa operação pode ser vista na Figura 3.1.

Figura 3.1 – Esquema de realce por aguçamento de imagens

Um exemplo da aplicação dessa abordagem pode ser visto na Figura 3.2, onde tem-se em (a) uma imagem de um corte de mamografia e em (b) o resultado da operação de aguçamento de imagens. É possível notar pela Figura 3.2 que apesar de

realçar as microcalcificações, aparecem muitos ruídos com a aplicação dessa operação, o que pode dificultar – ao invés de auxiliar – a análise da imagem resultante.



(a) (b)

Figura 3.2 – Aplicação de aguçamento de imagens em mamografia digital

Outra forma de aplicar essa abordagem é utilizando a transformada de Fourier. Um esquema de aplicação dessa abordagem pode ser visto na Figura 3.3.

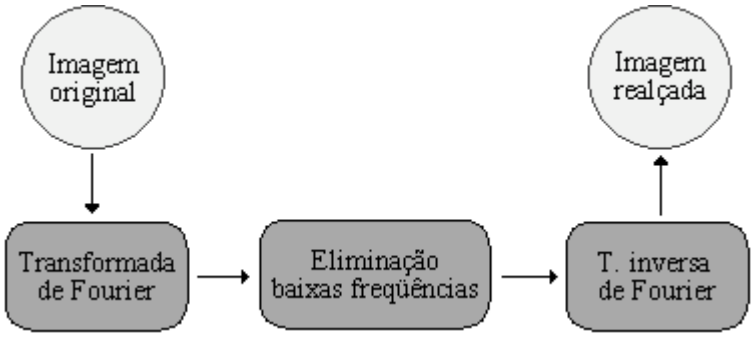
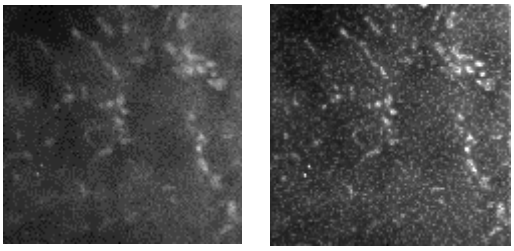


Figura 3.3 – Esquema de realce pela transformada de Fourier

Um exemplo da aplicação dessa abordagem pode ser visto na Figura 3.4, onde, em (a), tem-se uma imagem de um corte de mamografia digital e, em (b), o resultado da aplicação da abordagem utilizando a transformada de Fourier. É possível notar que, apesar da melhora obtida na visualização da imagem realçada obtida por essa transformada, ainda há adição de ruídos na imagem, o que compromete a visualização da imagem resultante.



(a) (b)

Figura 3.4 – Aplicação da transformada de Fourier em mamografia digital

Devido ao fato da qualidade da primeira abordagem não contribuir para auxiliar um médico a analisar essas imagens e, da segunda abordagem, por calcular senos e cossenos tanto na transformação direta quanto na inversa, da transformada *wavelet* possuir resolução tempo-freqüência não fixa, da utilização de um operador de realce de bordas apresentado em [23] e da possibilidade de classificar as bordas das microcalcificações utilizando-a, foi escolhida a transformada *wavelet* como base do método desenvolvido, apresentado na próxima seção.

3.1 Funcionamento do método

O princípio do método parte do fato de que as microcalcificações são elementos de altas freqüências em uma imagem. Inicialmente, a imagem de entrada é pré-processada, e a transformada *wavelet* é aplicada à imagem resultante dessa etapa, quando são separadas as baixas das altas freqüências. Um algoritmo de crescimento de região é aplicado à imagem de altas freqüências. A imagem resultante dessa operação e a imagem de baixas freqüências são as entradas da etapa de pós-processamento, resultando na imagem realçada. Ainda, a imagem de altas freqüências obtida após a aplicação da etapa de crescimento de região é processada por um caracterizador de bordas a fim de caracterizar as microcalcificações já realçadas. Um esquema geral do funcionamento do método desenvolvido, seguindo os passos mencionados, pode ser visualizado na Figura 3.5.

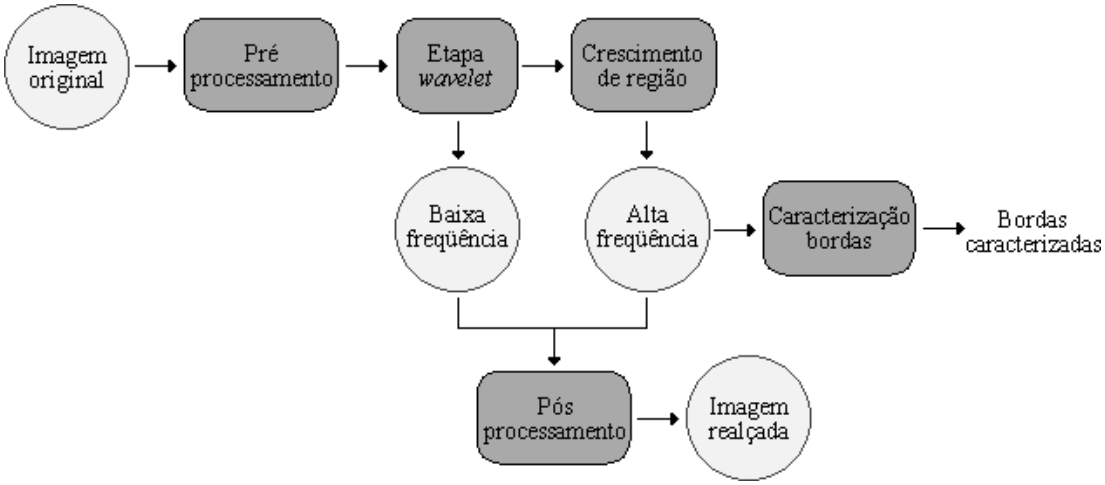


Figura 3.5 – Esquema geral do método desenvolvido

São detalhadas, nas sub-seções seguintes, cada uma das etapas que formam o método, iniciando pela etapa de pré-processamento.

3.1.1 Etapa de pré-processamento

Essa etapa tem como objetivo preparar a imagem original para o seu posterior processamento. A Figura 3.6 exibe um esquema do funcionamento dessa etapa.

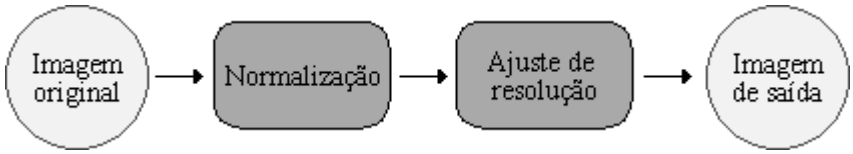


Figura 3.6 – Esquema de funcionamento da etapa de pré-processamento

Pela Figura 3.6 pode-se ver que, inicialmente, a imagem original é lida e a ela é aplicada uma operação de normalização de níveis de cinza, para distribuí-los melhor pela imagem. Essa operação é realizada utilizando-se a equação (3.1).

$$G(x,y) = \frac{[I(x,y) - \min(I)]^2}{\max(I) - \min(I)} + \min(I), \qquad (3.1)$$

onde $I(x,y)$ é a imagem original, $G(x,y)$ é a imagem resultante dessa operação e $\min(I)$ e $\max(I)$ são, respectivamente, o menor e o maior valores de I.

Após a normalização dos níveis de cinza da imagem, ela possui sua resolução ajustada de forma a evitar complicações ao aplicar a transformada *wavelet*. Como essa transformada divide a imagem em quatro novas imagens com metade da resolução da imagem original, caso em algum nível de decomposição a imagem a ser

dividida possua resolução ímpar, não é possível dividir essa imagem em quatro partes iguais. Assim, caso a imagem resultante da operação de normalização não possua resoluções horizontal e vertical que sejam números derivados da potenciação de 2 – ou seja, 2^r, tal que $r \in \mathbb{N}$ – ela é adequada inserindo linhas e colunas contendo o valor zero onde for necessário, formando uma nova imagem com resoluções horizontal e vertical idênticas e com o valor da potência de 2 mais próximo da maior resolução da imagem. A Figura 3.7 exibe um exemplo dessa operação, onde uma imagem que não possui resolução com valor potência de 2 possui ambas resoluções alteradas para o valor potência de 2 mais próximo de sua maior resolução, e suas novas posições são preenchidas com zeros.

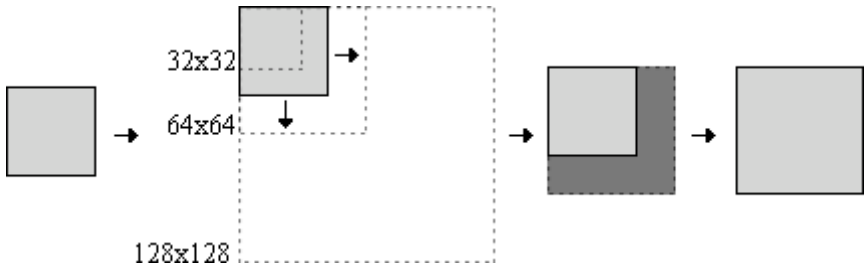


Figura 3.7 – Ajuste de resolução

Ao final do ajuste de resolução encerra-se a fase de pré-processamento. A imagem com resolução ajustada é a saída dessa etapa, dando início à etapa *wavelet*.

3.1.2 Etapa *wavelet*

Esta etapa possui como imagem de entrada a imagem de saída da etapa de pré-processamento. Um esquema do funcionamento dela pode ser visto na Figura 3.8.

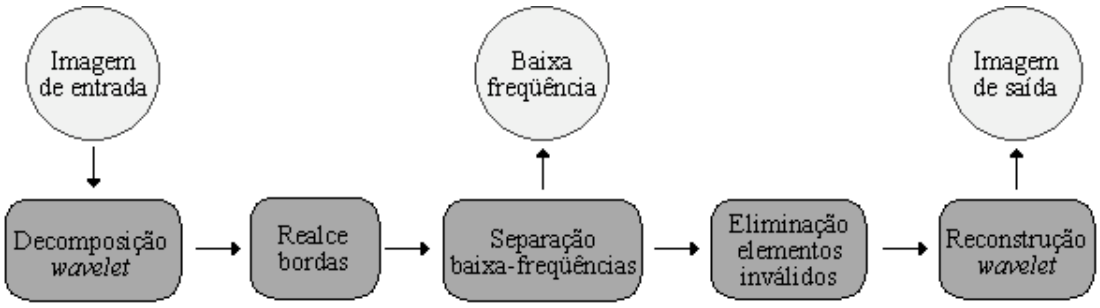


Figura 3.8 – Esquema de funcionamento da etapa *wavelet*

Inicialmente, a imagem de entrada é decomposta até o nível quatro da transformada *wavelet* pois, conforme visto no capítulo 2, nesse nível mais de 99%

dos elementos são de altas freqüências. Ainda, as imagens resultantes da aplicação da transformada *wavelet* até esse nível possuem mais elementos de altas freqüências que em relação à decomposição até níveis inferiores, e mais elementos de baixas freqüências em relação a níveis superiores, o que auxilia no caso do presente trabalho, conforme exemplo da Figura 2.25. No presente trabalho, são utilizadas diversas *wavelets* para verificar qual delas proporciona melhores resultados para o objetivo proposto. São utilizadas as *wavelets* de Daubechies, Symmlets e Coiflets, devido ao formato de suas *wavelets*-mãe e funções de escala pai, fator importante na reconstrução da imagem, conforme visto no capítulo 2. Uma vez no domínio da freqüência, é aplicado um operador de realce de bordas que opera nesse domínio, apresentado em [23]. Seja $I(x,y)$ uma imagem no domínio da freqüência e $H(x,y)$ a imagem resultante da aplicação desse operador, então para se obter a imagem com bordas realçadas aplica-se a equação (3.2).

$$H_j^i(x,y) = \begin{cases} I(x,y) - (G-1)mean_j^i(I), \text{ se } I(x,y) < -mean_j^i(I) \\ G * I(x,y), \text{ se } I(x,y) < mean_j^i(I) \\ I(x,y) + (G-1)mean_j^i(I), \text{ se } I(x,y) > mean_j^i(I) \end{cases}, \qquad (3.2)$$

onde $H_j^i(x,y)$ representa o valor de H na escala j e banda i e $mean_j^i(I)$ é a mediana dos valores absolutos de I na escala j e banda i. $G \geq 1$ é um parâmetro de realce de bordas, cujo valor foi definido como $G = 2$ após testes variando-o de 1 a 10. Nesses testes, quando $G = 1$, nenhum realce é obtido e, quando $G \geq 3$, as imagens resultantes tiveram o realce de vários falsos positivos.

Após o realce das bordas da imagem, a mesma é separada em duas imagens através da remoção de suas baixas freqüências. Uma cópia da imagem contendo tanto as baixas quanto as altas freqüências é feita, e é denominada imagem de baixas freqüências. A outra versão da imagem possui seus elementos de baixas freqüências do quarto nível zerados, de forma a isolar os elementos de altas freqüências. Após essa separação, alguns elementos inválidos da imagem são eliminados, para reduzir o número de falsos positivos da imagem realçada. Para isto, verifica-se em cada matriz da imagem decomposta se seus elementos contribuem com valores significantes no domínio da freqüência. Essa operação é descrita pela equação (3.3).

$$H_j^{Th,Tv,Td}(x,y) = \begin{cases} 0, \text{ se } I_j^{Th}(x,y) < 0{,}5 \\ 0, \text{ se } I_j^{Tv}(x,y) < 0{,}5, \\ 0, \text{ se } I_j^{Td}(x,y) < 0{,}5 \end{cases} \quad (3.3)$$

onde $H_j^{Th,Tv,Td}(x,y)$ representa a imagem resultante na escala *j* e bandas horizontal, vertical e diagonal, e $I_j^i(x,y)$ representa a imagem no domínio da freqüência na escala *j* e banda *i*. Assim, para a banda *j*, caso o ponto $(x.y)$ possua valor em umas das imagens de altas freqüências inferior a 0,5, esse ponto é marcado como falso positivo. Esse valor de limiar foi escolhido após a análise de valores variando entre 0 e 2, e obteve melhores resultados na remoção de ruídos na imagem resultante. Nesses testes, com o limiar igual a zero não houve diferença na imagem resultante e com limiar superior ou igual a 1, as imagens resultantes possuíam algumas microcalcificações não realçadas.

A eliminação de componentes inválidos prossegue ao analisar os valores dos *pixels* da imagem de entrada dessa etapa com os valores da imagem que está sendo processada no domínio da freqüência. Esse procedimento tem como objetivo eliminar da imagem elementos de altas freqüências contendo valores com baixos níveis de cinza, detectados nessa imagem por serem vizinhos de ruídos. A aplicação dessa operação é feita com base nas equações (3.4), (3.5) e (3.6).

$$H_j^{Th}(x,y) = 0, \text{se } I(x,y) < 0{,}25 * avg(I) \text{ e } H_j^{Th}(x,y) > 2000 \quad (3.4)$$

$$H_j^{Tv}(x,y) = 0, \text{se } I(x,y) < 0{,}25 * avg(I) \text{ e } H_j^{Tv}(x,y) > 2000 \quad (3.5)$$

$$H_j^{Td}(x,y) = 0, \text{se } I(x,y) < 0{,}25 * avg(I) \text{ e } H_j^{Td}(x,y) > 2000 \quad (3.6)$$

Nas três últimas equações, $I(x,y)$ é a imagem resultante da aplicação da equação (3.3), $avg(I)$ é a média da imagem I e H_j^{Th}, H_j^{Tv} e H_j^{Td} representam, respectivamente, as imagens de altas freqüências horizontal, vertical e diagonal, a uma escala *j*. Os limiares 0,25 e 2000 foram escolhidos após testes realizados com diferentes valores para os mesmos. Nesses testes, valores abaixo de 0,25 para o primeiro limiar não demonstraram diferença na aplicação desse procedimento e a utilização de valores acima desse limiar resultaram em imagens com microcalcificações não realçadas. A utilização de valores do segundo limiar abaixo de 2000 resultam em imagens com microcalcificações não realçadas, enquanto a

utilização de valores desse limiar acima de 2000 resultam em imagens sem a eliminação de alguns falsos positivos.

Ao final da eliminação dos componentes inválidos no domínio da freqüência, a imagem é então reconstruída utilizando a *wavelet* escolhida e essa imagem resultante é a imagem de saída dessa etapa, dando início à etapa de crescimento de região.

3.1.3 Crescimento de região

Esta etapa utiliza a imagem de saída da etapa anterior como imagem de entrada. Seu objetivo é utilizar os pontos de altas freqüências detectados na etapa anterior como sementes para a localização de microcalcificações, de forma a reduzir o número de falsos positivos gerados na imagem realçada. Esse procedimento é muito utilizado para detecção de objetos em mamografias [24]. Um diagrama do funcionamento dessa etapa pode ser visto na Figura 3.9.

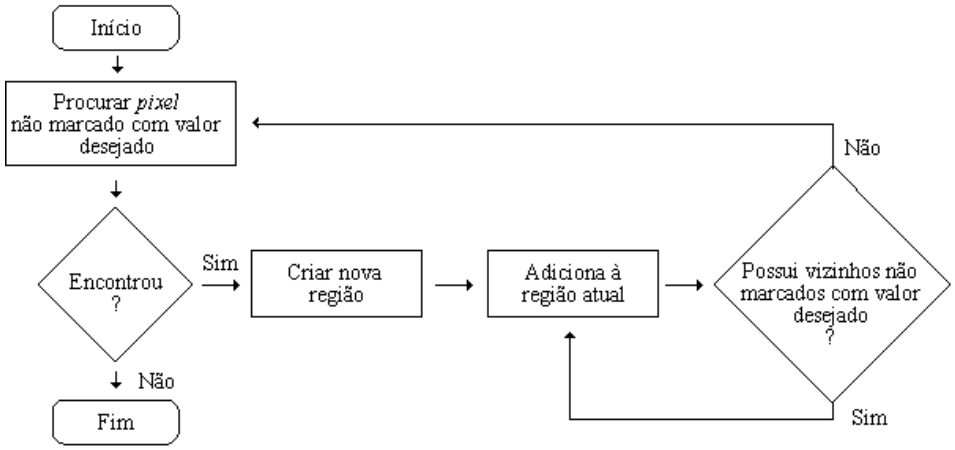


Figura 3.9 – Esquema de funcionamento da etapa de crescimento de região

Regions of Interest (ROIs) são áreas da imagem original que podem conter as imagens de possíveis microcalcificações presentes nas mamas. Essa etapa localiza ROIs na imagem, e logo após determina se elas são falsos positivos ou microcalcificações através de informações como suas áreas e seus tamanhos. Para isso, são utilizadas matrizes lógicas de pertinência para indicar quais *pixels* pertencem a cada uma das ROIs encontradas. Cada ROI possui apenas uma matriz de pertinência, denominada Matriz de Pertinência Local (MPL), que possui tamanho variável, definido de acordo com o tamanho da ROI que ela representa. Essas matrizes são utilizadas para determinar a qual ROI cada um dos pontos detectados

pertence. Existe ainda uma única matriz para a imagem inteira, denominada Matriz de Pertinência Global (MPG), que possui tamanho igual à resolução da imagem original. Essa matriz indica quais *pixels* já foram marcados como pertencentes a uma ROI, evitando que um mesmo ponto seja detectado como pertencente a mais de uma ROI, ou então, evitando que uma mesma ROI possua mais de uma MPL.

Inicialmente, a matriz é percorrida em busca de "sementes" para o início do algoritmo de crescimento de região. Seja $J(x,y)$ a imagem de entrada dessa etapa, uma semente para esse algoritmo é definida como qualquer *pixel p* tal que $J(p) \geq 10$, condição essa denominada como condição de pertinência. Esse valor de limiar utilizado foi escolhido após testes variando-o entre 0 e 30 de 5 em 5 unidades, sendo que o valor 10 foi o que obteve melhores resultados. Nesses testes, a utilização de valores de limiar abaixo de 10 resultou em imagens com grande quantidade de falsos positivos e, a utilização de valores desse limiar acima de 10, resultou em imagens com algumas microcalcificações não realçadas. Encontrado o primeiro *pixel* que satisfaça essa condição, verifica-se se ele está marcado na MPG: caso ele não esteja marcado, uma nova MPL deve ser criada para o *pixel* encontrado; caso ele esteja marcado, esse *pixel* já faz parte de uma ROI, e um novo *pixel* deve ser procurado. No primeiro caso, a posição desse *pixel* na MPG é marcada como verdadeiro, e verifica-se então se os *pixels* vizinhos de oito daquele *pixel* já detectado satisfazem a condição de pertinência e se estão marcados como falso na MPG. Em caso afirmativo, tais *pixels* são adicionados na ROI atual, seus valores na MPL da ROI atual são marcados como verdadeiros, juntamente com seus valores na MPG. Para cada um desses novos *pixels* marcados inicia-se uma nova pesquisa por *pixels* vizinhos não marcados que satisfaçam a condição de pertinência, e esse processo se repete enquanto houver *pixels* não marcados conectados a *pixels* dessa ROI que satisfaçam a condição de pertinência. Terminado esse processo, essa ROI é inserida em uma lista de ROIs juntamente com informações sobre a mesma, como tamanho da ROI, valor médio de nível de cinza, entre outros. Logo após, busca-se um novo *pixel* que não esteja marcado na MPG e que satisfaça a condição de pertinência, e o processo de crescimento de região desse *pixel* inicia-se. Esse processo de buscar novas regiões continua até que não existam mais na imagem *pixels* não marcados que satisfaçam a condição de pertinência. Um exemplo de funcionamento do algoritmo de crescimento de região pode ser visto na Figura 3.10, onde em (a), têm-se uma

imagem com uma ROI a ser detectada, com seus *pixels* escuros na imagem; em (b), o primeiro *pixel* da ROI é detectado; em (c), o símbolo "?" indica os vizinhos do *pixel* detectado que serão examinados; em (d), o símbolo "x" indica os *pixels* que não passaram na verificação; em (e), o símbolo "o" indica que o *pixel* foi detectado como pertencente a essa ROI, e os seus vizinhos também passam a ser analisados. Repare que os *pixels* que já foram verificados por serem vizinhos do primeiro *pixel* verificado da ROI não são verificados novamente, devido ao uso da MPG; em (f), tem-se a imagem da ROI já totalmente identificada, com os *pixels* detectados agora claros na imagem.

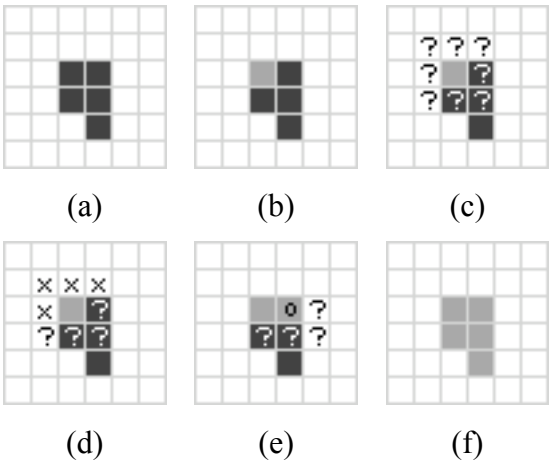


Figura 3.10 – Exemplo do funcionamento do algoritmo de crescimento de região

A Figura 3.11 mostra um exemplo de duas ROIs A e B cujas MPLs se sobrepõem. Devido ao uso da MPG, as duas ROIs são detectadas isoladamente, mesmo com suas MPLs se sobrepondo. Na Figura 3.11(a), tem-se a imagem com as duas ROIs (em escuro) a serem detectadas; na Figura 3.11(b), tem-se destacada a ROI A, juntamente com a região de sua MPL; na Figura 3.11(c), tem-se destacada a ROI B, juntamente com a região de sua MPL; na Figura 3.11(d), tem-se uma imagem exibindo a localização das MPLs de A e B sobrepostas; na Figura 3.11(e), tem-se os valores da MPG dessa imagem; na Figura 3.11(f), tem-se os valores da MPL da ROI A; e na Figura 3.11(g) tem-se os valores da MPL da ROI B.

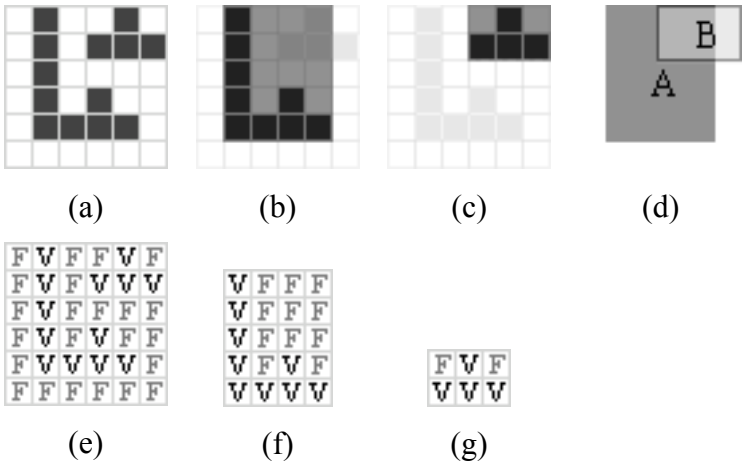


Figura 3.11 – Exemplo de MPLs sobrepostas

Para determinar quais ROIs são falsos positivos, após a detecção de todas as ROIs presentes na imagem, verifica-se quais delas possuem valor de área inferior a 5 ou valor médio de nível de cinza inferior a 50. Caso alguma ROI tenha qualquer verificação positiva, ela é descartada por ser um falso positivo, tanto em relação à sua área [25] quanto em relação ao seu valor médio de nível de cinza, onde o valor de limiar foi escolhido após testes variando seu valor, e o valor utilizado foi o que obteve melhores resultados. Nesses testes, a utilização de valores de limiar abaixo de 50 resultou em imagens com maior quantidade de falsos positivos, enquanto que, a utilização de valores de limiar acima de 50, resultou em imagens com algumas microcalcificações não realçadas. ROIs descartadas nesse processo são excluídas da lista de ROIs e seus *pixels* são marcados como falsos na MPG.

Ao final da verificação de todas as ROIs encontradas, a lista de ROIs contém apenas as que passam na validação de falso positivo, e a MPG possui verdadeiro somente nas posições das ROIs presentes na lista de ROIs. A imagem de saída dessa etapa, chamada imagem de altas freqüências, é obtida ao percorrer a imagem de entrada e tornar zero os *pixels* que possuem valor falso na MPG. Dessa forma, a imagem de altas freqüências possui valor diferente de zero apenas nos *pixels* que foram validados na verificação de ROIs.

3.1.4 Pós-processamento

Esta etapa consiste em unir a imagem de baixas freqüências com a imagem de altas freqüências, a fim de gerar a imagem realçada. Para atingir esse objetivo, a transformada *wavelet* é aplicada à imagem de altas freqüências, e seus valores são

adicionados à imagem da baixas freqüências, que ainda está no domínio da freqüência. Essa operação tem a finalidade de realçar os elementos de altas freqüências, ao mesmo tempo em que mantém os níveis de cinza das regiões de baixas freqüências da imagem. A imagem resultante é reconstruída, e a ela é aplicada a equação (3.7).

$$G(x,y) = \begin{cases} 0, \text{ se } I(x,y) < 0 \\ 255, \text{ se } I(x,y) > 255 \\ I(x,y), \text{ caso contrário} \end{cases}, \tag{3.7}$$

onde $I(x,y)$ é a imagem de entrada dessa etapa e $G(x,y)$ é a imagem realçada. Essa operação é realizada, pois, como a imagem foi perturbada no domínio da freqüência, alguns valores, após a reconstrução *wavelet*, podem possuir valores fora da escala de níveis de cinza, o que pode prejudicar a visualização dessa imagem. Ao aplicar essa operação elimina-se esse problema, uma vez que a imagem resultante dessa etapa possui *pixels* com valores dentro da escala de níveis de cinza utilizada. Esse é o final dessa etapa, cuja imagem de saída é a imagem realçada desejada.

3.1.5 Caracterização de bordas

Esta etapa tem como objetivo caracterizar as bordas das ROIs encontradas na etapa de crescimento de região em três classes diferentes: lisas, rugosas e indefinidas, a fim de auxiliar um médico a decidir a qual tipo uma microcalcificação pertence, uma vez que a diferença entre alguns tipos é a rugosidade das bordas das microcalcificações. Essa etapa pode ser realizada logo após a etapa de crescimento de região, podendo ser executada em paralelo com a etapa de pós-processamento, uma vez que ela utiliza apenas a imagem de altas freqüências gerada pela etapa de crescimento de região.

A abordagem dessa etapa se baseia, novamente, na freqüência dos elementos a serem analisados. Já que bordas são definidas como elementos de altas freqüências, utiliza-se a transformada *wavelet* para perturbá-las no domínio da freqüência e, após a reconstrução *wavelet*, decidir a qual tipo de borda ela pertence. Esse procedimento utiliza as MPLs de cada uma das ROIs como entrada. Um esquema de funcionamento dessa etapa pode ser visto na Figura 3.12, onde são exibidas as fases dessa etapa.

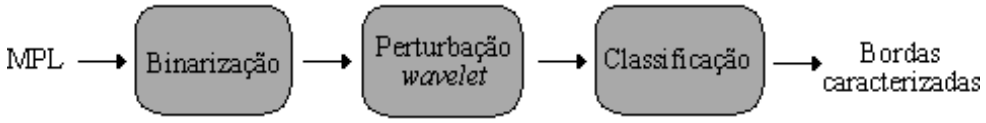


Figura 3.12 – Esquema de funcionamento da etapa de classificação

Inicialmente, são criadas imagens binarizadas a partir dos valores de cada uma das MPLs, sendo que os *pixels* dessas imagens possuem valor 0, para pontos com valor falso nas MPLs, e valor 255, para pontos com valor verdadeiro nas MPLs. É criada uma imagem binarizada para cada MPL, de forma que elas são analisadas separadamente.

A fase "perturbação *wavelet*" tem como objetivo perturbar a imagem da ROI no domínio da freqüência para permitir a sua classificação. Para diferenciar regiões lisas de rugosas da imagem, aplica-se a transformada *wavelet* até o primeiro nível, e a banda de componentes diagonais – por possuir maiores freqüências – é zerada, e a imagem é reconstruída. Foram testadas algumas *wavelets* nesse processo, para tentar identificar qual apresenta o resultado mais fácil de ser analisado após a perturbação. Exemplos da aplicação desse procedimento por diferentes *wavelets* podem ser vistos na Figura 3.13.

(a) Original (b) Haar (c) Daub4 (d) Daub20 (e) Sym8 (f) Coif6

Figura 3.13 – Resultado da aplicação de *wavelets* para perturbação de bordas

Na Figura 3.13 os *pixels* com valores diferentes na imagem resultante da imagem original são coloridos com cinza. Através da Figura 3.13 é possível observar que a aplicação da *wavelet* de Haar adiciona menos *pixels* cinzas às imagens resultantes, e estes localizam-se nas regiões rugosas da imagem da ROI, por serem áreas de mudanças abruptas de níveis de cinza. A aplicação das outras *wavelets* testadas gera um número maior de *pixels* cinzas, tanto nas regiões rugosas da imagem da ROI quanto em outras regiões da imagem. Devido ao fato da imagem resultante da aplicação de Haar resultar em uma imagem mais fácil para analisar a perturbação causada do que a imagem resultante da aplicação das outras *wavelets* testadas, foi utilizada essa *wavelet* nessa etapa de classificação. Esse fato pode ser

causado devido à pior resposta em freqüência de Haar em relação às outras *wavelets*, assim, poucos elementos são detectados como de altas freqüências na banda de detalhes diagonais da primeira decomposição da transformada, resultando em menos elementos cinzas nas imagens resultantes.

Após a binarização, então, aplica-se a transformada *wavelet* em cada umas das imagens binarizadas até o nível 1, utilizando a *wavelet* de Haar. Após a decomposição, a imagem com detalhes diagonais do primeiro nível é eliminada, tornando zero o valor de todos os seus elementos. Em seguida, a imagem é reconstruída, e todos os *pixels* dessa imagem que possuem valor diferente de zero possuem valores alterados para 255. Verifica-se então a imagem binarizada da MPL antes da perturbação *wavelet* com a imagem binarizada da MPL depois da perturbação. *Pixels* dessa nova imagem que possuem valores idênticos aos daquela, permanecem com seus valores inalterados, enquanto *pixels* da nova imagem cujos valores são diferentes naquela, têm seus valores alterados para 127. Exemplos de imagens binarizadas de ROIs lisas e rugosas, juntamente com o resultado da aplicação desse procedimento, pode ser visto na Figura 3.14, onde são exibidos exemplos de ROIs lisas, da Figura 3.14(a) a Figura 3.14(e); resultados da aplicação do procedimento descrito anteriormente em ROIs lisas, da Figura 3.14(f) a Figura 3.14(j); exemplos de ROIs rugosas, da Figura 3.14(k) a Figura 3.14(o); e resultados da aplicação do procedimento descrito anteriormente em ROIs rugosas, da Figura 3.14(p) a Figura 3.14(t).

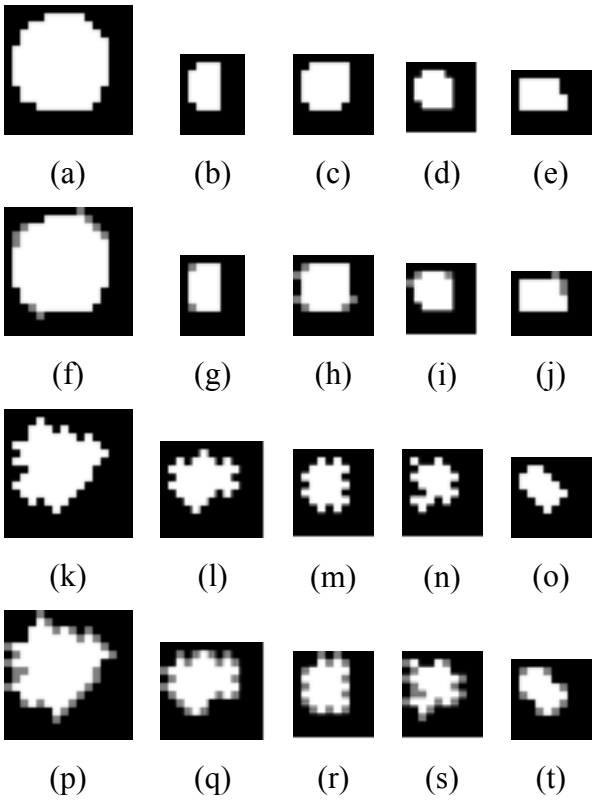


Figura 3.14 – Exemplo de ROIs lisas e rugosas e resultado da aplicação da caracterização

É possível notar, pela Figura 3.14, que regiões lisas das bordas das microcalcificações possuem poucos *pixels* cinzas em sua vizinhança, enquanto que as regiões rugosas possuem mais *pixels* cinza do que aquelas. ROIs com bordas lisas, então, possuem poucos *pixels* cinza em sua volta, pois são formadas predominantemente por regiões lisas, enquanto ROIs com bordas rugosas possuem mais *pixels* cinza em sua volta, pois são formados predominantemente por regiões rugosas. Assim, para classificar uma ROI como lisa, rugosa ou indefinida, calcula-se a quantidade de *pixels* de borda da ROI antes e depois da perturbação *wavelet*. No segundo caso, se o *pixel* de borda não possuir *pixel* cinza em toda sua vizinhança de 4, ele é classificado como um *pixel* liso. Calcula-se então o índice de lisura *S* dessa ROI, segundo a equação (3.8).

$$S = \frac{L(I_{pert})}{N(I)}, \tag{3.8}$$

onde I é a imagem formada pela binarização da MPL, I_{pert} é a imagem de I perturbada no domínio da freqüência, $L(I_{pert})$ é a quantidade de *pixels* lisos em I_{pert} e

$N(I)$ é a quantidade de *pixels* de borda de *I*. Esse valor indica a porcentagem de *pixels* lisos da ROI verificada.

Para classificar a borda da ROI, verifica-se o valor do índice de lisura *S*. Caso ela possua maior quantidade de *pixels* lisos, ou seja, $S > 0{,}5$, a borda dessa ROI é classificada como lisa; caso ela possua maior quantidade de *pixels* rugosos, ou seja, $S < 0{,}5$, a borda dessa ROI é classificada como rugosa e caso $S = 0{,}5$, a borda dessa ROI é classificada como indefinida, pois a quantidade de *pixels* lisos e rugosos é a mesma. A Tabela 3.1 mostra o resultado da classificação das ROIs apresentadas na Figura 3.14, e o índice *S* obtido para cada uma delas.

Tabela 3.1 – Resultado da classificação das ROIs

ROI	Classificação	*S*
Figura 3.14(a)	Lisa	0.70
Figura 3.14(b)	Lisa	0.71
Figura 3.14(c)	Lisa	0.64
Figura 3.14(d)	Lisa	0.69
Figura 3.14(e)	Lisa	0.80
Figura 3.14(k)	Rugosa	0.10
Figura 3.14(l)	Rugosa	0.00
Figura 3.14(m)	Rugosa	0.00
Figura 3.14(n)	Rugosa	0.05
Figura 3.14(o)	Rugosa	0.00

É possível verificar pela Tabela 3.1 que ambos os tipos de bordas foram classificados corretamente. Para o desenvolvimento do algoritmo foram utilizadas 20 imagens diferentes para cada tipo, sendo que todas essas imagens foram classificadas corretamente. A utilização da *wavelet* de Haar é fator importante nesse procedimento, pois ao remover a imagem de detalhes diagonais no domínio *wavelet* ela perturba apenas as bordas rugosas, preservando as bordas lisas. Foram feitos testes com outras *wavelets*, porém estas não obtiveram resultados satisfatórios, pois deformaram tanto bordas lisas quanto rugosas. Também se testou a eliminação de outras bandas da imagem, o que também não gerou resultados satisfatórios.

A saída dessa etapa é o tipo de borda de cada ROI detectada previamente. As informações de cada uma das ROIs são atualizadas, de forma a guardar detalhes sobre a sua borda, permitindo auxiliar um radiologista no momento de decidir a qual tipo determinada microcalcificação pertence.

Essa seção apresentou informações sobre o funcionamento do método desenvolvido. A seção seguinte apresenta detalhes de sua implementação e desenvolvimento.

3.2 Desenvolvimento e implementação

Um dos objetivos no desenvolvimento do método proposto foi a sua implementação em uma linguagem livre para desenvolvimento e multi-plataforma. Assim, a linguagem escolhida foi Java[1], que é gratuita para desenvolvimento e pode ser executada em qualquer sistema operacional que possua uma máquina virtual Java instalada.

Para permitir uma fácil utilização, o método foi desenvolvido na forma de um *software* visual denominado RCMW. A documentação do *software* pode ser encontrada no Apêndice C, e um manual de utilização correspondente pode ser encontrado no Apêndice D. Para o desenvolvimento desse *software* foi aproveitada uma biblioteca em Java, criada por Guido e Guilherme em [26], que calcula a transformada *wavelet*. Uma biblioteca básica de processamento de imagens denominada WIP foi desenvolvida para facilitar a utilização de operações básicas pelo *software*. Mais detalhes sobre ela podem ser encontrados no Apêndice B.

O programa desenvolvido requer que o computador no qual ele será executado tenha instalada a Java *Runtime Environment* (JRE) [27].

Essa seção apresentou pequenos detalhes sobre a implementação do método desenvolvido e as condições para que ele possa ser executado. Informações mais detalhadas sobre o mesmo podem ser encontradas nos Apêndices B, C e D. A seção seguinte apresenta os resultados obtidos na aplicação do método e uma discussão sobre os mesmos.

[1] http://java.sun.com

3.3 - Resultados

Esta seção apresenta alguns resultados obtidos na aplicação do método desenvolvido utilizando as *wavelets* descritas na seção 3.2. As imagens utilizadas são imagens de cortes de mamografias de resolução 120x120 *pixels* cada, com 256 possíveis níveis de cinza, cedidas pela Dra. Selma de Pace Bauab. Também foram utilizados cortes de mamografias gentilmente cedidas pelo Hospital da USP de Ribeirão Preto, com tonalidade de 16 *bits* convertida para 8 *bits*. Foram utilizadas as imagens dos dois grupos descritos, sendo que as imagens da Hospital da USP de Ribeirão Preto possuem maior qualidade, uma vez que foram obtidas cerca de dez anos após as demais. A Tabela 3.2 mostra a descrição das legendas utilizadas da Figura 3.15 até a Figura 3.18.

Tabela 3.2 – Descrição das legendas

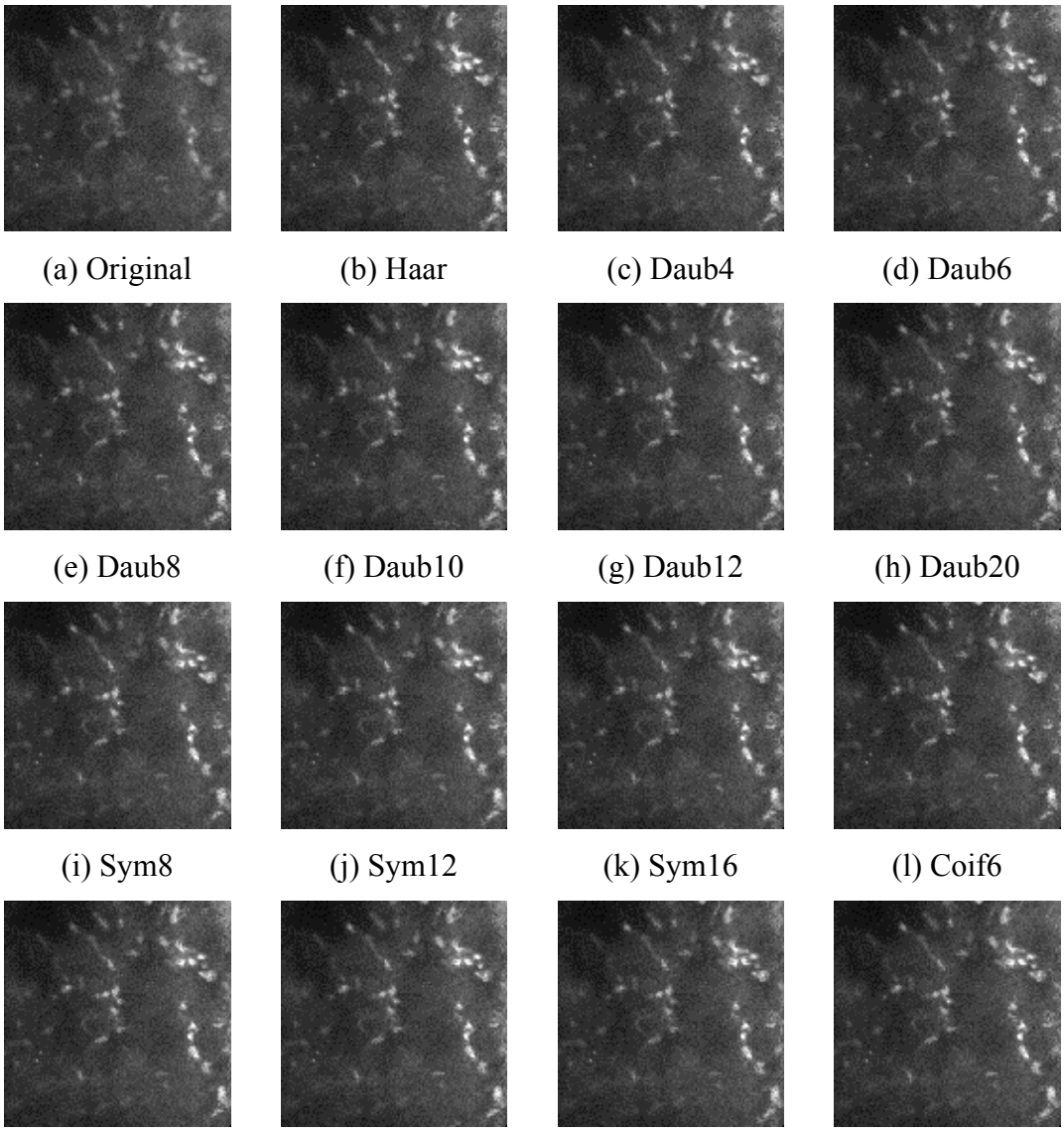
Legenda	Descrição	Legenda	Descrição
a	Imagem original	i	Realce por Sym8
b	Realce por Haar	j	Realce por Sym12
c	Realce por Daub4	k	Realce por Sym16
d	Realce por Daub6	l	Realce por Coif6
e	Realce por Daub8	m	Realce por Coif12
f	Realce por Daub10	n	Realce por Coif18
g	Realce por Daub12	o	Realce por Coif24
h	Realce por Daub20	p	Realce por Coif30

A primeira imagem analisada pode ser vista na , onde é possível ver que todas as *wavelets* utilizadas realçam, de certa forma, as microcalcificações presentes na imagem original. No entanto, a detecção das microcalcificações é visualmente diferente utilizando cada uma das *wavelets*. É possível perceber que na reconstrução dessa figura, a utilização da *wavelet* de Haar adiciona artefatos – objetos que não estão presentes na imagem original – de formato quadrada na imagem. A aplicação das *wavelets* de Daubechies gera diferentes resultados visuais para cada uma delas. Em relação ao formato das microcalcificações realçadas, Daubechies4 gera microcalcificações mais próximas das encontradas na imagem original, enquanto

Daubehies20 gera microcalcificações com maiores deformações em relação às demais *wavelets* de Daubechies. De forma geral, nessa figura, quanto maior o suporte da *wavelet* de Daubechies (excluindo a *wavelet* de Haar), maior a deformação causada no formato da microcalcificação. Em relação à adição de artefatos nessa imagem, *wavelets* dessa família com menor suporte geram mais artefatos em relação a *wavelets* dessa família com maior suporte. A aplicação de cada uma das Symmlets nessa imagem gera microcalcificações semelhantes em todas elas, juntamente com adição de artefatos na imagem, sendo que a aplicação de *wavelets* dessa família gera menos artefatos a medida que o suporte aumenta. A aplicação das Coiflets gera deformações menores para as *wavelets* com menor suporte, enquanto as com maior suporte geram maiores deformações nas microcalcificações. Em relação à adição de artefatos, a aplicação de *wavelets* com menor suporte gera mais artefatos em relação à aplicação de *wavelets* com maiores suportes. A Tabela 3.3 exibe os resultados do processamento da Figura 3.15(a), onde é possível verificar o número de ROIs detectadas e os valores médios de área e nível de cinza das ROIs.

Tabela 3.3 – Resultados 1

Wavelet	Total de ROIs	Média de área	Média de nível de cinza
Haar	26	41,2692	109,8513
Daub4	33	24,4848	110,0061
Daub6	29	22,8966	112,7653
Daub8	33	25,1515	109,7617
Daub10	30	30,1333	110,3727
Daub12	26	32,9231	112,1403
Daub20	30	30,0333	107,9721
Sym8	29	27,1724	114,0961
Sym12	30	29,2000	110,7458
Sym16	32	27,5000	109,1177
Coif6	35	22,6286	111,2584
Coif12	30	28,1333	113,3336
Coif18	26	37,1538	108,0417
Coif24	28	28,6071	109,5189
Coif30	33	23,4545	109,9238



(a) Original (b) Haar (c) Daub4 (d) Daub6

(e) Daub8 (f) Daub10 (g) Daub12 (h) Daub20

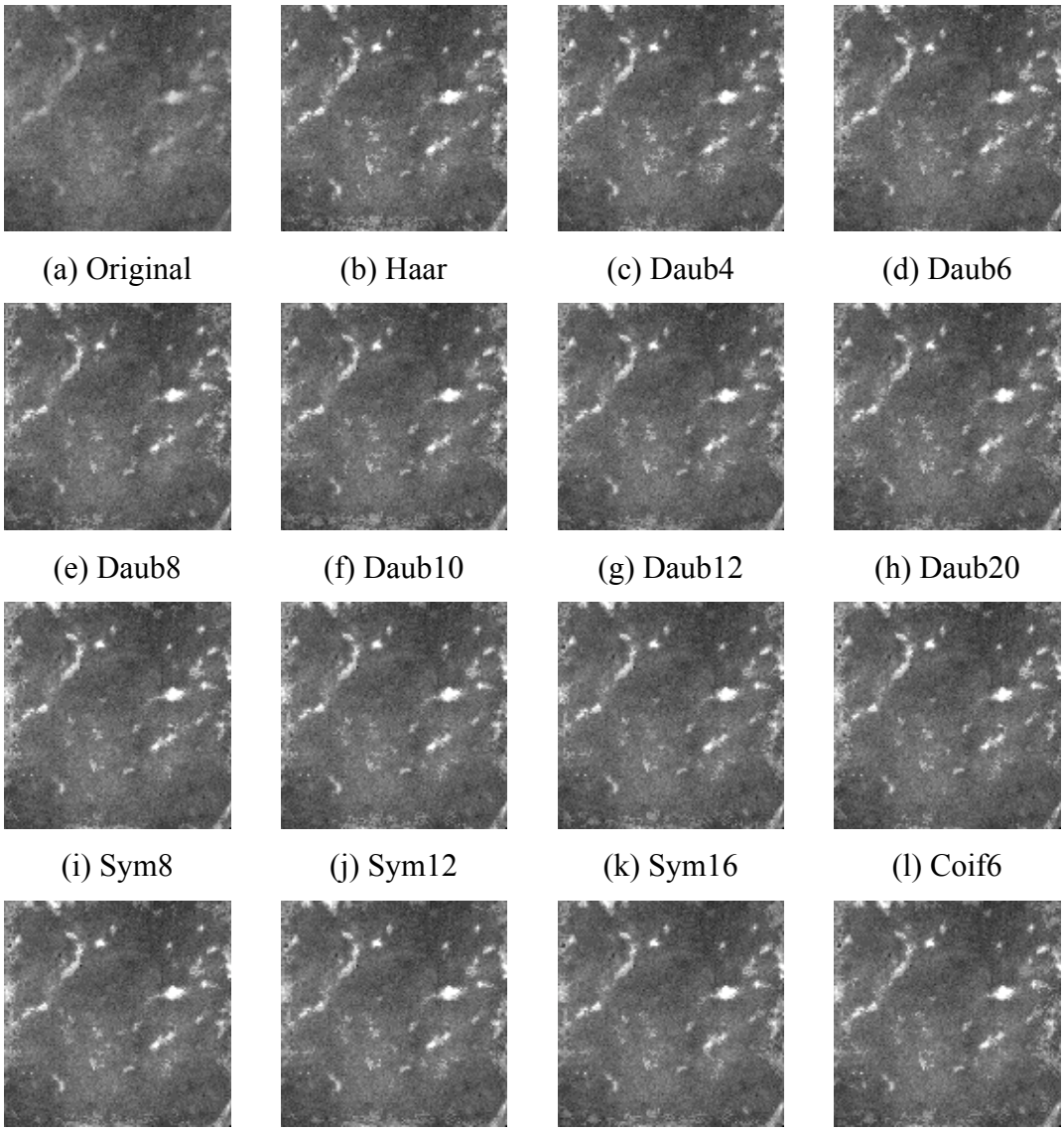
(i) Sym8 (j) Sym12 (k) Sym16 (l) Coif6

(m) Coif12 (n) Coif18 (o) Coif24 (p) Coif30

Figura 3.15 – Resultados 1

Outro exemplo de aplicação do método pode ser visto na Figura 3.16, onde é possível observar também que a aplicação *wavelet* de Haar adiciona artefatos de formato quadrado na imagem. A aplicação das *wavelets* de Daubechies produz imagens com artefatos em quantidade semelhante em todas as *wavelets*, e os formatos das microcalcificações são mais próximos às da imagem original utilizando *wavelets* dessa família com menor suporte. A aplicação das Symmlets gera imagens com formatos das microcalcificações semelhantes, independentemente do suporte da *wavelet*, enquanto a aplicação das Coiflets produz microcalcificações com deformações em seu formato em todas as *wavelets* dessa família, sendo que as

wavelets com menor suporte geram deformações que diminuem a área das ROIs, enquanto *wavelets* com maior suporte geram deformações que aumentam a área das ROIs. A Tabela 3.4 exibe os resultados da aplicação das *wavelets* escolhidas na imagem presente da Figura 3.16(a), exibindo o número de ROIs detectadas e os valores médios de área e de níveis de cinza dessas.



(a) Original (b) Haar (c) Daub4 (d) Daub6

(e) Daub8 (f) Daub10 (g) Daub12 (h) Daub20

(i) Sym8 (j) Sym12 (k) Sym16 (l) Coif6

(m) Coif12 (n) Coif18 (o) Coif24 (p) Coif30

Figura 3.16 – Resultados 2

Tabela 3.4 – Resultados 2

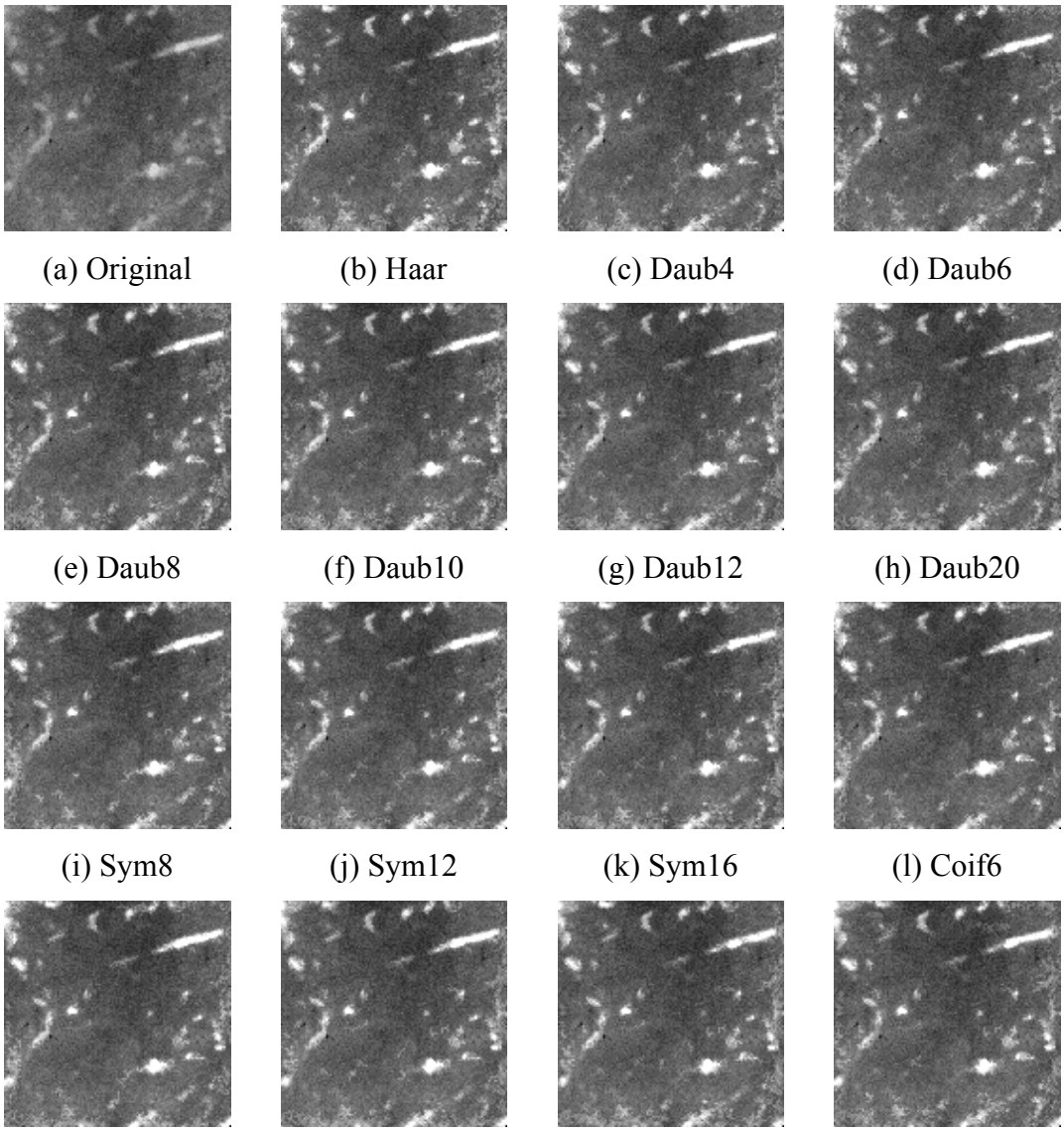
Wavelet	Total de ROIs	Média de área	Média de nível de cinza
Haar	57	35,1930	124,4214
Daub4	49	37,7143	127,4780
Daub6	56	31,7143	124,1381
Daub8	53	36,6792	121,2202
Daub10	51	42,4706	124,8137
Daub12	43	47,9767	124,1755
Daub20	63	30,7937	122,2265
Sym8	48	38,6667	126,0267
Sym12	44	44,7955	126,1112
Sym16	54	33,8333	125,5295
Coif6	49	36,7755	127,0491
Coif12	52	37,8846	123,1191
Coif18	48	42,0000	127,3437
Coif24	52	37,6346	126,0416
Coif30	55	37,6727	122,2019

Outro exemplo da aplicação do método pode ser visto na Figura 3.17, onde é possível ver que a aplicação de Haar adiciona artefatos com formato quadrado na imagem. Ao aplicar Daubechies, ocorre adição de artefatos com todas as *wavelets* utilizadas dessa família, porém, quanto menor o suporte da *wavelet*, menor a deformação no formato das bordas das microcalcificações. A aplicação das Symmlets gera resultados visualmente parecidos, com menor adição de artefatos nas *wavelets* dessa família com maior suporte. A aplicação das Coiflets gera imagens com menor deformação e maior adição de artefatos para *wavelets* dessa família com menor suporte. A Tabela 3.5 exibe os resultados obtidos, tais como total de ROIs encontradas e valores médios de área e nível de cinza das ROIs, para as imagens presentes nessa figura.

Tabela 3.5 – Resultados 3

Wavelet	Total de ROIs	Média de área	Média de nível de cinza
Haar	45	58,0222	127,4581
Daub4	52	42,0769	125,0573
Daub6	52	41,6538	128,2919
Daub8	48	53,8542	124,6999
Daub10	42	64,4762	126,7935
Daub12	40	64,4750	124,3771
Daub20	47	54,5957	124,5685
Sym8	43	53,7209	127,7473
Sym12	43	53,0930	126,4512
Sym16	36	66,6389	127,1043
Coif6	45	46,4444	128,5473
Coif12	41	54,6341	126,0661
Coif18	43	56,7907	125,0411
Coif24	45	54,4889	125,1609
Coif30	55	46,2182	123,4200

Outro exemplo da aplicação do método pode ser visto na Figura 3.18, onde a aplicação de Haar, como nos exemplos anteriores, adiciona artefatos com formato quadrado na imagem. A aplicação de Daubechies gera imagens visualmente semelhantes, porém *wavelets* dessa família com menor suporte geram microcalcificações mais próximas às da imagem original, porém, com maior adesão de ruídos. A aplicação das Symmlets nessa imagem também gera imagens semelhantes, porém, com menor adição de artefatos para as *wavelets* dessa família com maior suporte. A aplicação das Coiflets nessa imagem gera maior quantidade de artefatos adicionados à imagem nas *wavelets* dessa família com menor suporte, porém, estas geram microcalcificações com bordas mais próximas às da imagem original. Ainda, a aplicação de Coiflets com menor suporte não preenche totalmente o centro de algumas ROIs. A Tabela 3.6 exibe a quantidade de ROIs detectadas e os valores médios de área e níveis de cinza resultantes da aplicação do método nessa imagem.



(a) Original (b) Haar (c) Daub4 (d) Daub6

(e) Daub8 (f) Daub10 (g) Daub12 (h) Daub20

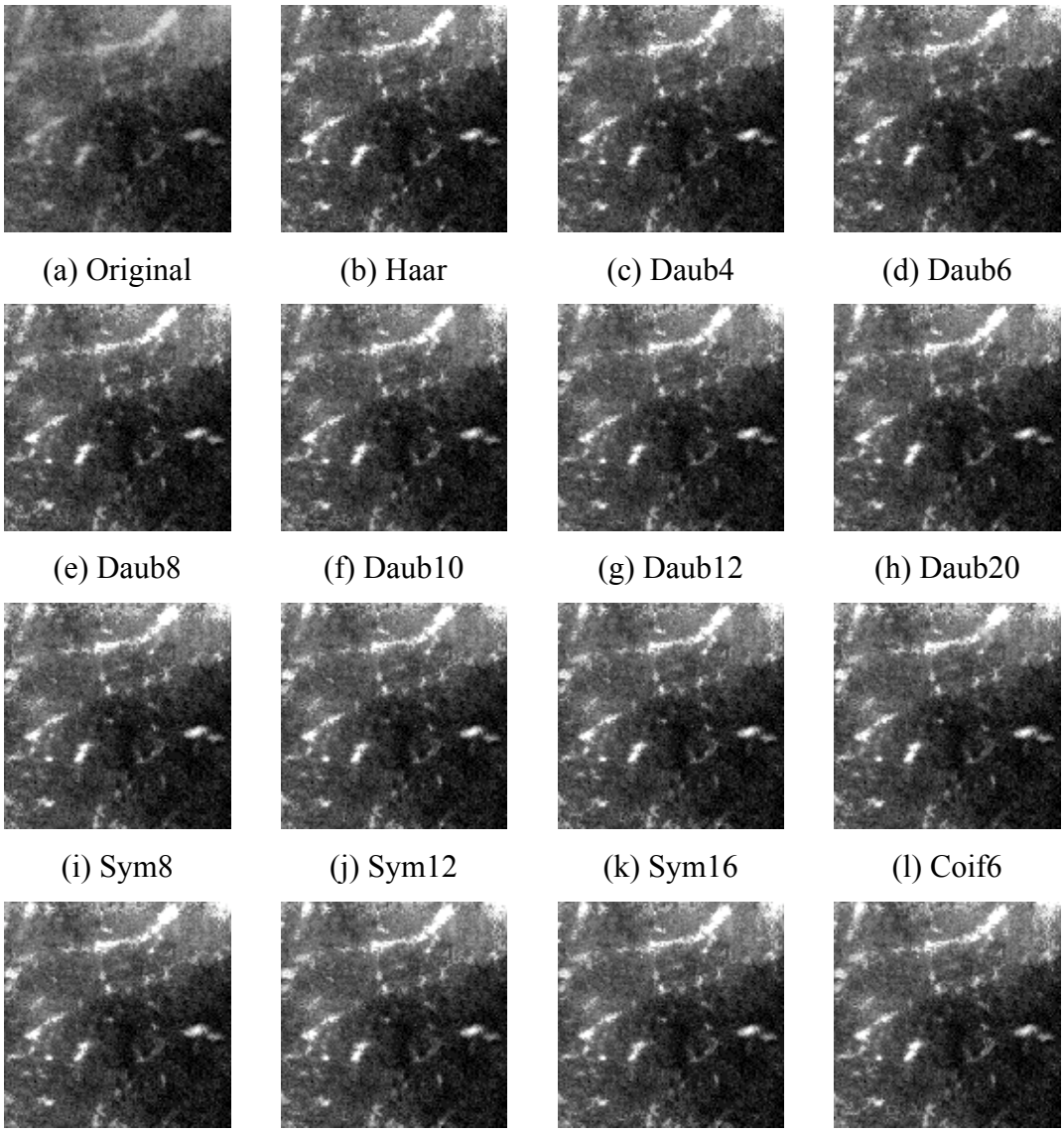
(i) Sym8 (j) Sym12 (k) Sym16 (l) Coif6

(m) Coif12 (n) Coif18 (o) Coif24 (p) Coif30

Figura 3.17 – Resultados 3

Tabela 3.6 – Resultados 4

Wavelet	Total de ROIs	Média de área	Média de nível de cinza
Haar	59	30,4576	116,1763
Daub4	51	36,8431	118,4319
Daub6	50	40,3800	122,5081
Daub8	57	36,1930	118,6821
Daub10	47	40,5319	121,2684
Daub12	49	42,4898	119,8091
Daub20	51	39,4706	113,3315
Sym8	45	46,3111	122,4092
Sym12	43	43,9767	118,6833
Sym16	56	35,3036	121,5360
Coif6	39	53,2051	123,8270
Coif12	48	39,5208	121,6282
Coif18	51	38,0196	118,6575
Coif24	62	31,3226	114,9318
Coif30	43	47,5116	117,6035



(a) Original (b) Haar (c) Daub4 (d) Daub6

(e) Daub8 (f) Daub10 (g) Daub12 (h) Daub20

(i) Sym8 (j) Sym12 (k) Sym16 (l) Coif6

(m) Coif12 (n) Coif18 (o) Coif24 (p) Coif30

Figura 3.18 – Resultados 4

A partir dos resultados obtidos ao aplicar o método ao conjunto de figuras utilizadas é possível fazer alguns comentários sobre o funcionamento geral do mesmo, para cada uma das *wavelets* utilizadas. A *wavelet* de Haar gera, na maioria das imagens resultantes, artefatos de formato quadrado, o que ocorre devido ao formato de suas *wavelet*-mãe e função de escala pai. Como essas funções possuem formatos semelhantes a ondas quadradas, ao reconstruir um sinal com a transformada *wavelet*, o mesmo é reconstruído como uma combinação linear daquelas funções, conforme visto no Capítulo 2. Além disso, as imagens resultantes da aplicação de Haar possuem mais artefatos que as demais *wavelets*, realçando áreas maiores que as microcalcificações, devido à sua resposta em freqüência que está longe do ideal. As

demais *wavelets* de Daubechies geram resultados semelhantes na maior parte das imagens analisadas. De forma geral nos testes realizados, conforme o suporte das *wavelets* dessa família aumenta, também aumenta a deformação do formato das bordas das microcalcificações, de forma que a aplicação de *wavelets* com maior suporte gera resultados com microcalcificações com bordas mais suavizadas que as microcalcificações obtidas através da aplicação de *wavelets* dessa família com menor suporte. Esse fato ocorre devido ao formato das funções de escala pai e *wavelet*-mãe dessas *wavelets* pois, como visto no Capítulo 2, as Daubechies com maior suporte possuem formato mais suavizadas que as Daubechies de menor suporte. Ainda sobre essa família de *wavelets*, ocorre na aplicação delas a adição de artefatos na imagem. Essa adição não prejudica tanto a visualização dessas imagens se comparada com a visualização resultante da aplicação da *wavelet* de Haar. Além disso, *wavelets* dessa família com menor suporte geram alguns artefatos a mais que *wavelets* com maior suporte na aplicação do método nas imagens testadas, o que pode ser resultado da resposta em freqüência dessas *wavelets*. Em relação à visualização, a aplicação de todas as *wavelets* testadas realça as microcalcificações, com vantagem para as *wavelets* dessa família com menor suporte.

A aplicação das Symmlets gera resultados visualmente semelhantes à aplicação das Daubechies. Alguns artefatos são adicionados nas imagens resultantes, porém, não prejudicam tanto a visualização das microcalcificações se comparadas à imagem obtida através da aplicação de Haar, ocorrendo mais freqüentemente nas *wavelets* dessa família com menor suporte, o que deve ocorrer devido à resposta em freqüência delas. Em relação à deformação das bordas, não há diferenças perceptíveis visualmente em relação à *wavelet* utilizada. A aplicação das Coiflets gera resultados visualmente semelhantes aos resultados obtidos por Daubechies e Symmlets. Na aplicação dessa família de *wavelets* ocorre, também, inclusão de artefatos na imagem resultante. Em geral, nos testes realizados, quanto menor o suporte da *wavelet* dessa família utilizada, maior a quantidade de artefatos que aparece na imagem resultante, porém, eles não comprometem tanto a visualização das microcalcificações realçadas quando comparadas com Haar. Em relação à deformação do formato das bordas das microcalcificações, Coif6 gera bordas deformadas mais próximas às da imagem original, enquanto as demais *wavelets*

dessa família, com maior suporte, possuem bordas deformadas, aumentando a área das microcalcificações.

Alguns comentários gerais sobre o desempenho das *wavelets* testadas podem ainda ser feitos. Em relação à resposta em freqüência, *wavelets* com melhor resposta em freqüência geraram imagens com menor quantidade de artefatos adicionados, porém, em algumas imagens, algumas microcalcificações não foram realçadas. Em relação à resposta em fase das *wavelets*, não foram observadas diferenças visuais ao utilizar *wavelets* com resposta linear, não-linear e quase-linear. Em relação ao formato das funções de escala pai e *wavelets*-mãe, *wavelets* com essas funções mais suavizadas geram maior deformação nas bordas das ROIs pois, conforme visto no capítulo 2, a imagem resultante é reconstruída como uma combinação linear dessas duas funções. Assim, as regiões rugosas da imagem são suavizadas ao utilizar *wavelets* com essa característica, o que pode causar erros na etapa de classificação das ROIs.

A Figura 3.19 mostra um gráfico comparando o número de ROIs encontradas nos exemplos da Figura 3.15 até a Figura 3.18. Através do gráfico presente na Figura 3.19 é possível verificar uma variação na quantidade de ROIs detectadas por cada *wavelet* utilizada, o que ocorre devido à etapa de crescimento de região. Nessa etapa, dependendo da *wavelet* utilizada, uma ou mais ROIs podem ser detectadas como uma ou mais regiões, alterando a quantidade total de ROIs detectadas.

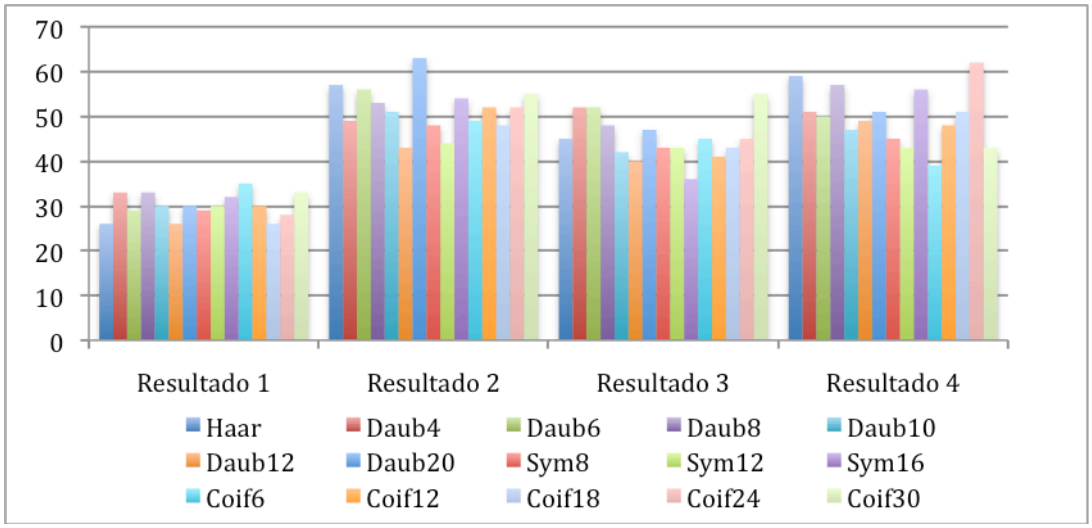


Figura 3.19 – Número de ROIs detectadas

A Figura 3.20 exibe o valor médio de área das ROIs encontrada da Figura 3.15 até a Figura 3.18, utilizando cada uma das *wavelets* propostas. Através do

gráfico presente na Figura 3.20 é possível verificar que na maioria dos casos, os valores encontrados no gráfico correspondem com os valores dos gráficos presentes na Figura 3.19, sendo que nos casos em que um pequeno número de ROIs foi detectado, o valor médio de área é geralmente alto, e em casos em que um maior número de ROIs foi detectado, o valor médio de área é geralmente baixo. Isso ocorre pois, em alguns casos, diversas ROIs são detectadas como uma única ROI, e em outros casos, essas ROIs são detectadas separadamente. Assim, vê-se que a etapa de crescimento de região está diretamente ligada ao número de ROIs detectadas e ao valor médio de área das mesmas.

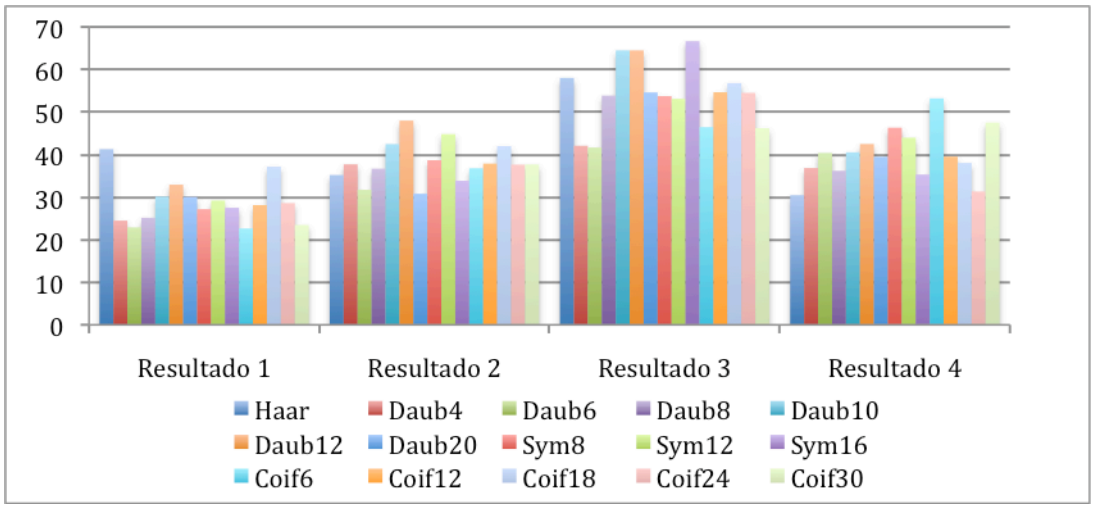


Figura 3.20 – Valor médio de área das ROIs

A Figura 3.21 exibe um gráfico comparando os valores médios de nível de cinza encontrados ao aplicar as diferentes *wavelets* propostas da Figura 3.15 até a Figura 3.18. É possível verificar que ocorre pouca variação nesses valores, o que pode indicar uma determinada faixa de valores de níveis de cinza na qual as microcalcificações ocorrem. No geral, em todo o conjunto de imagens analisado, o valor médio de nível de cinza das microcalcificações detectadas foi 115.

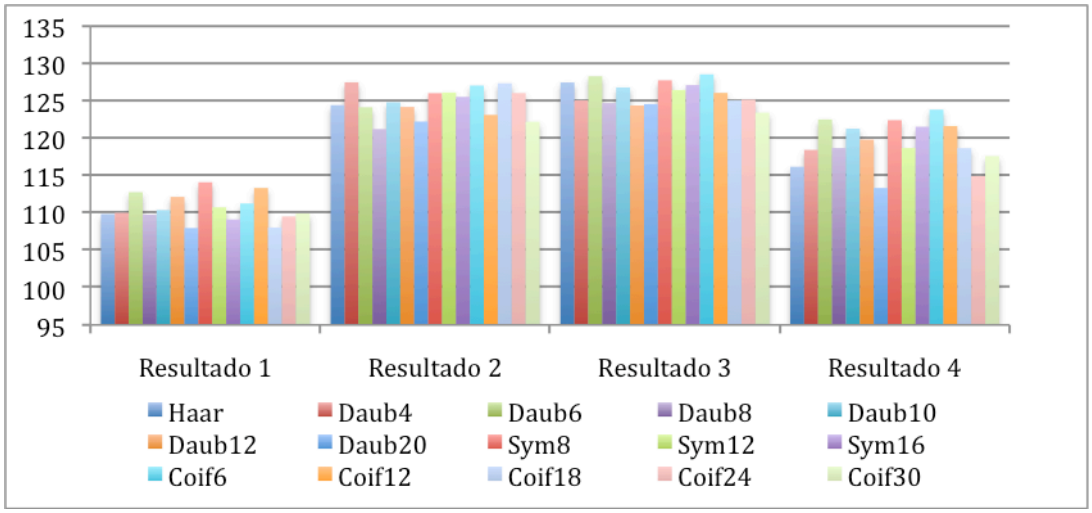


Figura 3.21 – Valor médio de nível de cinza das ROIs

Um fato importante a ser comentado está relacionado com a aplicação de um algoritmo de crescimento de região. Através da utilização desse algoritmo, é possível eliminar falsos positivos de forma a melhorar a qualidade da imagem resultante. Para todas as imagens analisadas, a redução média de falsos positivos foi de 86,17%.

Para analisar a eficácia da classificação das bordas obtidas na aplicação do método, foram utilizadas imagens de microcalcificações com a natureza de suas bordas previamente conhecidas. O método foi então aplicado nessas imagens utilizando as *wavelets* propostas, e os resultados obtidos permitem uma discussão sobre a eficiência obtida por cada *wavelet*. A Figura 3.22 exibe um gráfico mostrando a porcentagem de ROIs lisas, rugosas e indefinidas detectadas aplicando cada uma das *wavelets* a ROIs com bordas lisas. Pelo gráfico presente na Figura 3.22 é possível verificar que as *wavelets* que obtiveram melhores resultados foram, respectivamente, Sym16, Sym8, Coif6 e Sym12.

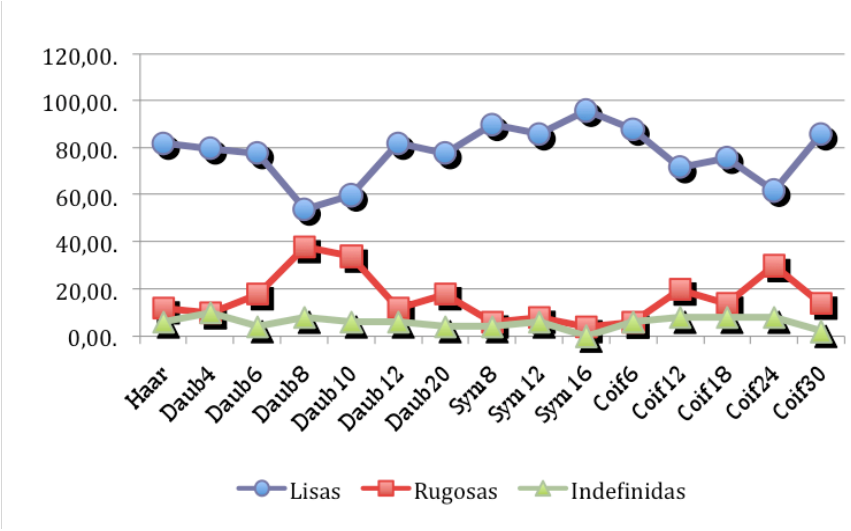


Figura 3.22 – Classificação das ROIs detectadas em imagens contendo microcalcificações com bordas lisas

A Figura 3.23 exibe um gráfico mostrando a porcentagem de ROIs lisas, rugosas e indefinidas detectadas aplicando cada uma das *wavelets* a ROIs com bordas rugosas. A partir do gráfico presente na Figura 3.23 é possível perceber que as *wavelets* que obtêm melhor resultado são, respectivamente, Sym16, Sym12, Daub4, Coif6 e Sym8.

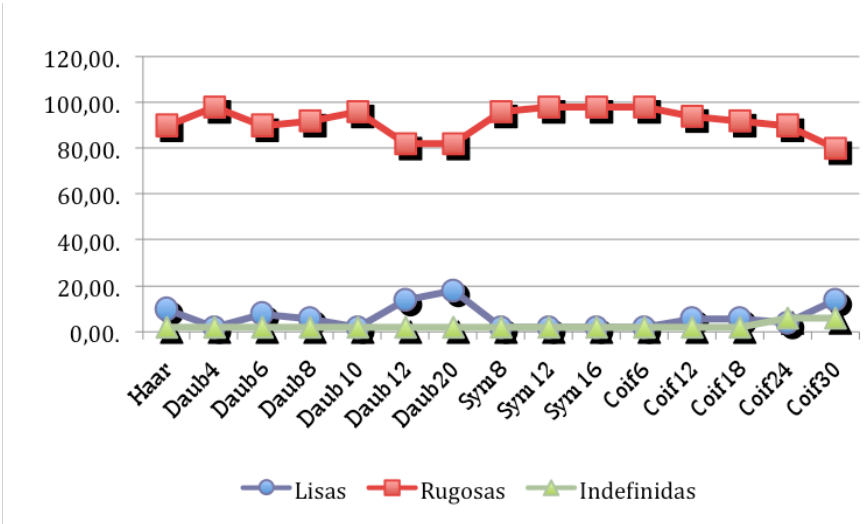


Figura 3.23 – Classificação das ROIs detectadas em imagens contendo microcalcificações com bordas rugosas

A partir dos resultados visualizados da Figura 3.22 até a Figura 3.23 é possível verificar que, ao contrário dos resultados visuais, onde a resposta em fase parece não ser fator importante, ela tem importância na caracterização das bordas das

microcalcificações, juntamente com o formato das *wavelets*-mãe e funções de escala pai. Os melhores resultados obtidos na classificação do formato das bordas das microcalcificações são de *wavelets* com resposta em fase quase-linear e que possuem *wavelet*-mãe e função de escala pai com pontas estreitas, o que pode indicar que a aplicação de *wavelets* com essas características gera resultados mais precisos do que outras *wavelets*.

A Figura 3.24 exibe um gráfico comparando o valor médio de área das microcalcificações analisadas com borda lisa e o valor médio de área detectado na aplicação de cada uma das *wavelets* propostas. Os resultados do gráfico presente na Figura 3.24 incluem os artefatos encontrados na aplicação dessas, assim, quanto maior a adesão de artefatos na imagem gerada através da aplicação das *wavelets*, maior a diferença entre a área média real das ROIs e as áreas médias detectadas. Através da Figura 3.24 é possível verificar que as *wavelets* que obtêm melhor resultados são, respectivamente, Sym8, Sym12, Coif6 e Sym16.

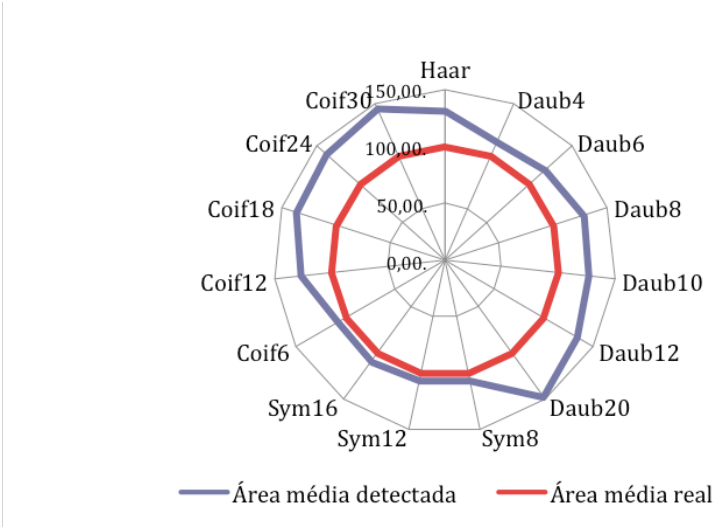


Figura 3.24 – Comparação entre o valor médio de área das microcalcificações lisas e o valor médio de área das ROIs detectadas

De forma similar, a Figura 3.25 exibe um gráfico comparando o valor médio de área das ROIs com bordas rugosas e o valor médio de área das ROIs detectadas aplicando as diferentes *wavelets* utilizadas. Nesse caso, as *wavelets* que produzem melhores resultados são, respectivamente, Sym16, Coif6, Sym12, Sym8 e Daub4.

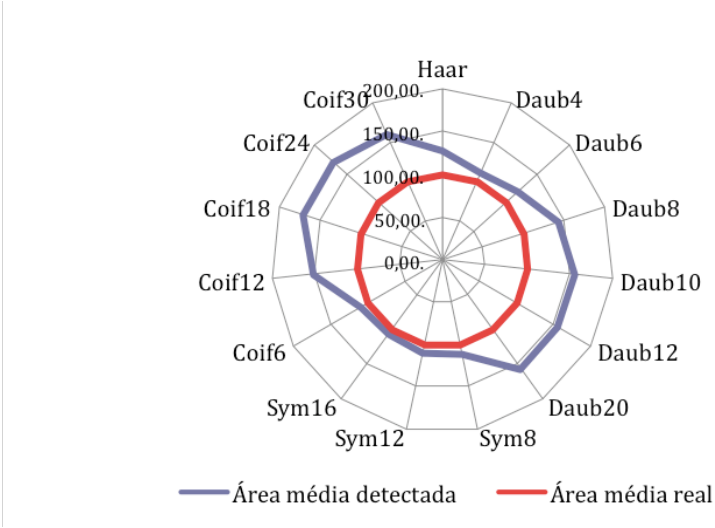


Figura 3.25 – Comparação entre o valor médio de área das microcalcificações rugosas e o valor médio de área das ROIs detectadas

Os resultados vistos nos gráficos exibidos da Figura 3.24 até a Figura 3.25 mostram, novamente, a importância da resposta em fase das *wavelets* utilizadas, pois os melhores resultados são obtidos com *wavelets* que possuem resposta em fase quase-linear e que possuem *wavelets*-mãe e funções de escala pai que possuem pontas estreitas.

De forma geral, a diferença encontrada por cada um dos três fatores verificados para as *wavelets* utilizadas nesse trabalho foram as seguintes: para a resposta em freqüência, quanto pior ela é, maior a adesão de ruídos na imagem, porém, se ela for muito diferente da ideal, algumas microcalcificações não são encontradas; para a resposta em fase, as *wavelets* com resposta em fase quase-linear obtiveram vantagens na classificação das bordas das ROIs se comparadas às *wavelets* com resposta linear e não-linear; em relação aos formatos das *wavelets*-mãe e funções de escala pai. *wavelets* que possuem essas funções com formatos contendo pontas estreitas geraram menor quantidade de ruídos nas imagens e menor deformação no formato das bordas das microcalcificações, porém, visualmente, *wavelets* contendo pontas muito estreitas, como Coif6, geraram imagens com microcalcificações que não foram realçadas por inteiro.

Em relação à qualidade das imagens geradas, a aplicação do método em imagens com pior qualidade resultou em microcalcificações mais fáceis de serem percebidas visualmente, porém, devido à baixa qualidade delas, alguns ruídos foram também realçados nas imagens resultantes. A aplicação do método em imagens com

maior qualidade resultou em microcalcificações mais nítidas, sem o aparecimento de ruídos na imagem resultante. Em ambos os casos, as microcalcificações ficaram mais fáceis de serem percebidas visualmente, o que auxilia a visualização das mesmas pelo médico responsável por analisar essas imagens.

Esse capítulo apresentou detalhes sobre o método proposto, seu funcionamento e sua implementação, bem como os resultados obtidos e uma discussão sobre o mesmo. O capítulo seguinte apresenta as conclusões do presente trabalho.

Capítulo 4 – Conclusões

Neste capítulo são feitas as conclusões do presente trabalho e são apresentadas algumas propostas de trabalhos futuros.

4.1 Conclusão do trabalho

Nesse trabalho foi apresentado um método desenvolvido com o objetivo de auxiliar radiologistas a visualizar de forma mais clara microcalcificações presentes em mamas, e auxiliá-los na tarefa de classificar esses objetos nos possíveis tipos em que os mesmos podem ser classificados. Ao realçar a imagem desses elementos em mamografias digitais, diminui-se a possibilidade de um médico ter cansaço visual ao analisar diversas dessas imagens em seqüência, e a classificação das bordas desses elementos fornece uma indicação sobre a natureza do objeto analisado.

A utilização da transformada *wavelet* foi fundamental, pois tornou possível separar as imagens das microcalcificações do restante da imagem da mama e classificar as bordas de acordo com sua natureza. Com base nos resultados obtidos podemos afirmar que, para as imagens analisadas nesse trabalho, a família *wavelet* Symmlets apresentou melhores resultados em relação às demais *wavelets* analisadas, devido à combinação de fatores analisados nesse trabalho: resposta em freqüência, resposta em fase e formato de *wavelets*-mãe e funções de escala pai. A utilização de um algoritmo de crescimento de região permitiu eliminar falsos positivos da imagem realçada, melhorando sua qualidade. O desenvolvimento do sistema em uma

linguagem gratuita e multi-plataforma permite o seu uso em qualquer computador que possua uma máquina virtual Java instalada, tornando-o assim de fácil acesso e independente de sistema operacional.

4.2 Trabalhos futuros

Como propostas de trabalhos futuros podem ser sugeridos algoritmos focados na melhoria do desempenho do método, tais como a paralelização do código e a utilização de duas frentes de programação, uma em Java – para manter o aspecto multi-plataforma do método – e uma em uma linguagem estruturada, como C, de forma a melhorar o desempenho do método, para um sistema operacional particular ou uso em tempo real em uma plataforma dedicada. Ainda, em relação ao desempenho do método, podem ser testados algoritmos de detecção de componentes conexos, para verificar se esses geram um ganho de desempenho em relação à etapa de crescimento de região. Em relação às *wavelets*, podem ser testadas outras *wavelets* para verificar seus resultados e até mesmo desenvolver uma família de *wavelets* específica com o objetivo de melhorar a classificação.

Referências Bibliográficas

1. **CONTROLE do Câncer de Mama:** Documento de Consenso, abril de 2004. Rio de Janeiro: Instituto Nacional de Câncer (INCA). Disponível em < http://www.inca.gov.br/publicacoes/Consensointegra.pdf>. Acesso em: dez. 2007.

2. **ESTIMATIVAS 2008:** Incidência de Câncer no Brasil, 2007. Rio de Janeiro: Instituto Nacional de Câncer (INCA). Disponível em < http://www.inca.gov.br/estimativa/2008/versaofinal.pdf>. Acesso em: dez. 2007.

3. REZAI-RAD, G.; JAMARANI, S. Detecting microcalcification clusters in digital mammograms using combination of wavelet and neural network. In: INTERNATIONAL CONFERENCE ON COMPUTER GRAPHICS, IMAGING AND VISION: NEW TRENDS, 2005, Beijing – China. **Proceedings of the international conference on computer graphics, imaging and visualization**, 2005. p. 197-201.

4. HEINLEIN, P.; DREXL, J.; SCHNEIDER, W. Integrated Wavelet for enhancement of microcalcifications in digital mammography. **IEEE Transactions on medical imaging**, v. 22, n. 3, 2003. p. 402-413.

5. **SISTEMA de Informações Ambulatoriais:** Manual de Bases Técnicas Oncologia, 2006. Brasília: Secretaria de Atenção à Saúde (SAS). Disponível em <

http://dtr2004.saude.gov.br/sas/documentos/manual_onco%20_211106.pdf>. Acesso em: dez. 2007.

6. LE GAL, M. apud PEREIRA, A. S. **Processamento de imagens médicas utilizando transformada de Hough**. 1995. Tese (Doutorado em Física Computacional) – Universidade de São Paulo, São Paulo, 1995.

7. PEREIRA, A. S. **Processamento de imagens médicas utilizando transformada de Hough**. 1995. Tese (Doutorado em Física Computacional) – Universidade de São Paulo, São Paulo, 1995.

8. **CÂNCER:** a informação pode salvar vidas, 2007. Rio de Janeiro: Instituto Nacional de Câncer (INCA). Disponível em <http://www.inca.gov.br/eventos/dncc/2007/folder_cancer_mama_0727(web_20071120).pdf>. Acesso em: dez. 2007.

9. STRICKLAND, R. N.; HAHN, H. I. Wavelet transform matched filters for the detection and classification of microcalcification in mammography. In: INTERNATIONAL CONFERENCE ON IMAGE PROCESSING, 1995, Washingthon – EUA. **Proceedings of the 1995 international conference on image processing**, 1995. v. 01, p. 422-425.

10. LADO, M. J.; TAHOCES, P. G.; et al. A wavelet-based algorithm for detecting clustered microcalcifications in digital mammograms. **Medical Physics**, v. 26, n. 7, 1999. p. 1294-1305.

11. GONZALES, R. C.; WOODS, R. E. **Digital image processing**. 2nd ed. Upper Saddle River: Prentice-Hall, 2002. 793 p.

12. DIESTEL, R. **Graph theory**. 3rd ed. New York: Springer-Verlag Heidelberg, 2005. 410 p.

13. WIKIMEDIA FOUNDATION, INC. **Standard Test Image – Wikipédia, the free encyclopedia**. Disponível em <http://en.wikipedia.org/wiki/Standard_test_image>. Acesso em: 10 dez. 2007.

14. VIEIRA, L. C. **Conversão de voz baseada na transformada wavelet**. 2007. Dissertação (Mestrado em Física Computacional) – Universidade de São Paulo, São Carlos, 2007.

15. DAUBECHIES, I. **Ten lectures on wavelets**. Philadelphia: SIAM, 1992. 355 p.

16. ADDISON, P. S. **The illustrated wavelet transform handbook:** Introductory Theory and Applications in Science, Engineering, Medicine and Finance. Edinburgh: IoP Publishing, 2002. 353 p.

17. WALKER, J. S. **A primer on wavelets and their scientific applications**. Washingthon: Chapma and Hall/CRC, 1999. 155 p.

18. HAYKIN, S.; VEEN, B. V. **Sinais e sistemas**. Porto Alegre: BOOKMAN, 2002.

19. GUIDO, R. C.; VIEIRA, L. S.; et al. A matched FIR filter bank for audio coding. In: IEEE INTERNATIONAL SYMPOSIUM ON MULTIMEDIA, 2005, Irvine – Canadá. **Proceedings of the 7th IEEE international symposium on multimedia**, v. 1, p. 796-801, 2005.

20. FONSECA, E. S.; GUIDO, R. C.; et al. Discrete wavelet transform and support vector machine applied to pathological voice signals identification. In: IEEE INTERNATIONAL SYMPOSIUM ON MULTIMEDIA, 2005, Irvine – Canadá. **Proceedings of the 7th IEEE international symposium on multimedia**, v. 1, p. 785-789, 2005.

21. GUIDO, R. C.; MACIEL, C. D.; et al. A study on the best wavelet for audio compression. In: IEEE ASILOMAR INTERNATIONAL CONFERENCE ON SIGNALS, SYSTEMS AND COMPUTERS, 40th, 2006, Pacific-Grove, EUA.

Proceedings of the 40th IEEE asilomar international conference on signals, systems and computers, v. 1, p. 2115-2118, 2005.

22. AKAY, M. **Time frequency and wavelets in biomedical signal processing**. New York: IEEE Press in Biomedical Engineering, 1998. 739 p.

23. SCHARCANSKI, J.; JUNG, C. R.; Denoising and enhancing digital mammographic images for visual screening. **COMPUTERIZED MEDICAL IMAGING AND GRAPHICS**. 2006. v. 30, n. 4. p. 243-254.

24. CHENG, H. D.; SHI, X. J.; et al. Approaches for automated detection and classification of masses in mammograms. **PATTERN RECOGNITION**. 2006. v. 39, n. 4. p. 646-668.

25. ADODZ, T.; KURDZIEL, M.; et al. Detection of clustered microcalcifications in small field digital mammography. **COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE**. 2006. v. 81, n. 1. p. 56-65.

26. GUIDO, R. C.; GUILHERME, M. B. A. **Java-based implementation of the discrete wavelet transform**. São Carlos: Speechlab, Instituto de Física de São Carlos. Universidade de São Paulo, 2007. Disponível em <http://speechlab.ifsc.usp.br>. Acesso em: 02 fev. 2007.

27. SUN MICROSYSTEMS. **Download Java software from Sun Microsystems**. Disponível em <http://www.java.com/en/download/manual.jsp>. Acesso em: 13 jul. 2007.

GLOSSÁRIO

Transformada *wavelet* – Transformada que decompõe um sinal em bandas de diferentes freqüências, utilizando funções de escala e *wavelets*.

Função de escala pai – Filtro passa-baixas original da transformada *wavelet*.

Função de escala – Filtro obtido a partir de uma função de escala pai através de operações de dilatação e translação.

***Wavelet*-mãe** – Filtro passa-altas original da transformada *wavelet*.

Wavelet – Filtro obtido a partir de uma *wavelet*-mãe através de operações de dilatação e translação. Também é utilizado para denominar o par formado por uma *wavelet*-mãe e uma função de escala pai.

Apêndice A – Coeficientes *wavelets* utilizados

Este apêndice exibe os coeficientes das *wavelets* utilizados referentes aos coeficientes dos filtros passa-baixas.

Haar

[0.7071067, 0.7071067]

Daubechies4

[0.482962913144, 0.836516303737, 0.224143868042, -0.129409522551]

Daubechies6

[0.33267055295, 0.806891509311, 0.459877502118, -0.13501102001, -0.085441273882, 0.0352262918857]

Daubechies8

[0.230377813308, 0.714846570552, 0.630880767929, -0.0279837694168, -0.187034811719, 0.0308413818355, 0.0328830116668, -0.010597401785]

Daubechies10

[0.160102397974, 0.603829269797, 0.724308528437, 0.138428145901, -0.242294887066, -0.0322448695846, 0.07757149384, -0.00624149021279, -0.012580751999, 0.00333572528547]

Daubechies12

[0.11154074335, 0.494623890398, 0.751133908021, 0.315250351709, -0.226264693965, -0.129766867567, 0.0975016055873, 0.0275228655303, -0.0315820393174, 5.53842201161E-4, 0.00477725751094, -0.0010773010853]

Daubechies20

[0.0266700579005, 0.188176800077, 0.527201188931, 0.688459039453, 0.28117234366, -0.249846424327, -0.195946274377, 0.127369340335, 0.0930573646035, -0.0713941471663, -0.0294575368218, 0.0332126740593, 0.00360655356695, -0.0107331754833, 0.00139535174705, 0.00199240529518, -6.85856694959E-4, -1.16466855129E-4, 9.358867032E-5, -1.32642028945E-5]

Symmlets8

[0.032223100604, -0.012603967262, -0.099219543577, 0.297857795606, 0.803738751807, 0.497618667633, -0.029635527646, -0.075765714789]

Symmlets12

[-0.007800708325, 0.001767711864, 0.044724901771, -0.021060292512, -0.072637522786, 0.337929421728, 0.78764114103, 0.491055941927, -0.048311742586, -0.117990111148, 0.003490712084, 0.015404109327]

Symmlets16

[0.001889950333, -3.02920515E-4, -0.014952258337, 0.003808752014, 0.049137179674, -0.027219029917, -0.051945838108, 0.364441894835, 0.777185751701, 0.481359651258, -0.061273359068, -0.143294238351, 0.007607487325, 0.031695087811, -5.42132332E-4, -0.003382415951]

Coiflets6

[-0.072732619513, 0.337897662458, 0.852572020212, 0.384864846864, -0.072732619513, -0.015655728135]

Coiflets12

[0.016387336464, -0.041464936782, -0.067372554722, 0.386110066823, 0.81272363545, 0.417005184424, -0.076488599079, -0.059434418647, 0.023680171946, 0.005611434819, -0.001823208871, -7.20549445E-4]

Coiflets18

[-0.003793512864, 0.007782596427, 0.023452696142, -0.065771911282, -0.061123390003, 0.40517690241, 0.793777222626, 0.428483476378, -0.071799821619, -0.082301927107, 0.034555027573, 0.015880544864, -0.009007976137, -0.002574517689, 0.001117518771, 4.6621696E-4, -7.0983303E-5, -3.4599773E-5]

Coiflets24

[8.92313669E-4, -0.001629492013, -0.007346166328, 0.016068943965, 0.026682300156, -0.081266699681, -0.056077313317, 0.41530840703, 0.782238930921, 0.434386056491, -0.066627474263, -0.096220442034, 0.039334427123, 0.025082261845, -0.015211731528, -0.005658286687, 0.003751436157, 0.001266561929, -5.89020756E-4, -2.59974552E-4, 6.2339034E-5, 3.1229876E-5, -3.25968E-6, -1.784985E-6]

Coiflets30

[-2.1208084E-4, 3.58589688E-4, 0.002178236358, -0.004159358782, -0.010131117521, 0.023408156788, 0.028168028974, -0.091920010569, -0.052043163181, 0.421566206733, 0.77428960373, 0.437991626216, -0.062035963969, -0.105574208714, 0.041289208754, 0.03268357427, -0.019761778945, -0.009164231163, 0.006764185449, 0.002433373213, -0.001662863702, -6.38131343E-4, 3.02259582E-4, 1.4054115E-4, -4.1340432E-5, -2.1315027E-5, 3.734655E-6, 2.063762E-6, -1.67443E-7, -9.5177E-8]

Apêndice B – Desenvolvimento da biblioteca WIP

A biblioteca WIP é uma biblioteca básica de processamento de imagens digitais desenvolvida em Java, que tem como objetivo aumentar a simplicidade de um código escrito e aumentar a produtividade ao reutilizar a biblioteca, evitando assim a reescrita de código já presente nela. Apesar de a linguagem Java contar com uma biblioteca própria para tratamento de imagens, a JAI (Java *Advanced Imaging*), a liberdade oferecida por ela não é tão grande ao tentar abrir uma imagem como uma matriz. Portanto, o desenvolvimento dessa biblioteca visa aumentar a liberdade do programador fornecendo-lhe uma ferramenta que faça as funções mais básicas do processamento de imagens.

A Figura B.1 exibe o diagrama de classes UML (*Unified Modelling Language*) da biblioteca desenvolvida, onde é possível visualizar as classes desenvolvidas, juntamente com seus atributos e métodos públicos. Esse diagrama deve ser utilizado como um guia de classes ao desenvolver algum *software* que utilize essa biblioteca.

A linguagem Java possui um utilitário de documentação denominado Javadoc, que permite a criação de documentação de código a partir de informações digitadas no próprio código. A biblioteca WIP é documentada utilizando o Javadoc, e sua documentação é apresentada a seguir.

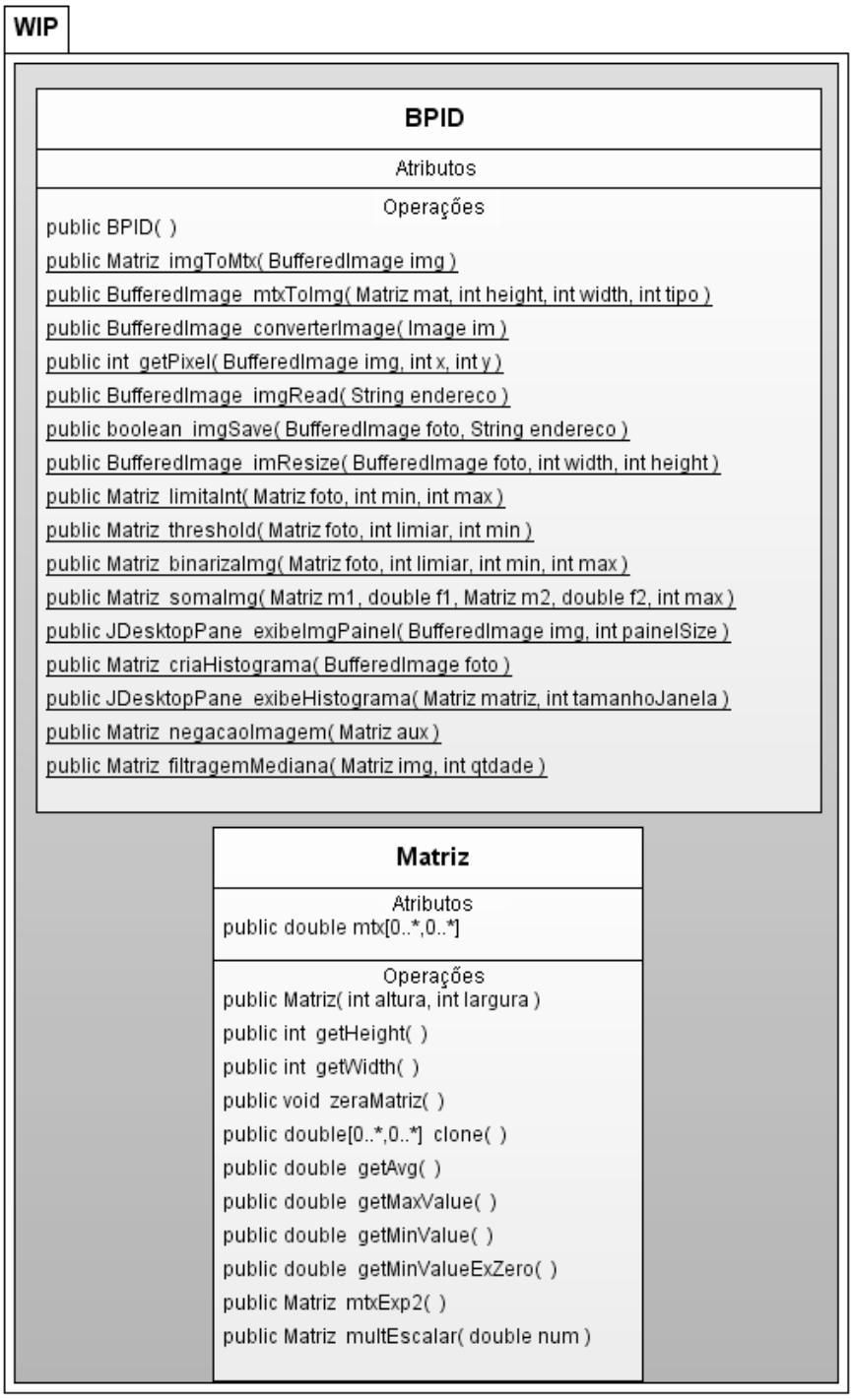


Figura B.1 – Diagrama de classes da biblioteca WIP

Nome do pacote: WIP
Nome da classe: BPID
Definição: public class BPID extends java.lang.Object
Descrição: classe que realiza processamento de imagens em Java.

Construtores
public BPID() Construtor da classe.
Métodos
public static Matriz imgToMtx (BufferedImage img) Método que abre uma imagem a partir de um endereço. **Parâmetros:** endereco - Endereço da imagem no computador. **Retorna:** Imagem desejada. Retorna null se a imagem não estiver no local informado.
public static BufferedImage mtxToImg (Matriz mat, int height, int width, int tipo) Método que retorna uma imagem do tipo BufferedImage formada pelos *pixels* presentes em uma matriz. **Parâmetros:** mat - Matriz contendo os valores de níveis de cinza. height - Altura da imagem. width - Largura da imagem. tipo - Tipo da imagem. **Retorna:** Imagem do tipo BufferedImage equivalente a matriz mat.
public static BufferedImage converterImage (Image im) Método que converte uma imagem do tipo Image para uma imagem do tipo BufferedImage. **Parâmetros:** im - Imagem do tipo Image que será convertida. **Retorna:** Imagem do tipo BufferedImage.
public static int getPixel (BufferedImage img, int x, int y) Método que retorna o valor de um *pixel* isolado de uma imagem. **Parâmetros:** img - Imagem da qual deseja-se obter o valor do *pixel*. x - Número da linha do *pixel* desejado. y - Número da coluna do *pixel* desejado. **Retorna:** Valor do *pixel* na posição desejada.
public static BufferedImage imgRead (String endereco) Método que carrega uma imagem a partir de um endereço. **Parâmetros:**

endereco - Endereço da imagem no computador.

Retorna:

Imagem desejada. Retorna null se a imagem não estiver no local informado.

public static boolean imgSave (BufferedImage foto, String endereco)

Método que armazena uma imagem no endereço especificado.

Parâmetros:

foto - Imagem a ser armazenada.

endereco - Local onde a imagem deve ser armazenada.

Retorna:

True, se a imagem for armazenada corretamente. False, se a imagem não pode ser salva.

public static BufferedImage imResize (BufferedImage foto, int width, int height)

Método que redimensiona uma imagem e retorna a imagem resultante.

Parâmetros:

foto - Imagem a ser redimensionada.

width - Nova largura da imagem.

height - Nova altura da imagem.

Retorna:

Imagem redimensionada.

public static Matriz limitaInt (Matriz foto, int min, int max)

Método que limita uma imagem entre os valores mínimo e máximo especificados, associando o valor mínimo para os valores abaixo do mínimo, e o valor máximo para os valores acima do máximo.

Parâmetros:

foto - Imagem à qual será aplicado o método.

min - Valor mínimo.

max - Valor máximo.

Retorna:

Matriz com os valores limitados.

public static Matriz threshold (Matriz foto, int limiar, int min)

Método que faz a limiarização em uma imagem a partir do valor especificado.

Parâmetros:

foto - Imagem a ser limiarizada.

limiar - Valor de limiar.

min - Valor mínimo que será atribuído aos *pixels* abaixo do valor de limiar.

Retorna:

Imagem limiarizada.

Static Matriz binarizaImg(Matriz foto, int limiar, int min, int Max)

Método que realiza a binarização em uma imagem a partir de um limiar, associando o valor

mínimo aos valores inferiores ou iguais ao limiar e o valor máximo aos valores superiores ao limiar.

Parâmetros:

foto - Imagem a qual será aplicada a binarização.

limiar - Limiar a ser utilizado.

min - Valor mínimo.

max - Valor máximo.

Retorna:

Imagem binarizada.

public static Matriz somaImg (Matriz m1, double f1, Matriz m2, double f2, int max)

Método que soma duas imagens de acordo com os pesos especificados.

Parâmetros:

m1 - Primeira imagem.

f1 - Peso a ser aplicado à primeira imagem.

m2 - Segunda imagem.

f2 - Peso a ser aplicado à segunda imagem.

max - Valor máximo da soma.

Retorna:

Imagem somada.

public static JDesktopPane exibeImgPainel(BufferedImage img, int painelSize)

Método que retorna um painel para a exibição de uma imagem na tela.

Parâmetros:

img - Imagem a ser exibida.

painelSize - Tamanho do painel que deve ser criado para exibir a imagem.

Retorna:

Painel contendo a imagem.

public static Matriz criaHistograma(BufferedImage foto)

Método que cria o histograma de uma imagem com o tamanho informado.

Parâmetros:

foto - Imagem da qual será criado o histograma.

Retorna:

Matriz contendo o histograma da imagem desejada.

public static JDesktopPane exibeHistograma(Matriz matriz, int tamanhoJanela)

Método que exibe um histograma do tamanho desejado.

Parâmetros:

matriz - Histograma na forma de matriz.

tamanhoJanela - Tamanho da janela a ser exibida.

Retorna:

JDesktopPane contendo o histograma desejado.
public static Matriz filtragemMediana(Matriz img, int qtdade) Método que realiza a filtragem da mediana em uma imagem. **Parâmetros:** img - Imagem da qual será realizada a filtragem da mediana. qtdade - Vizinhança a ser utilizada. 3 - 3x3, 5 - 5x5, etc. **Retorna:** Imagem filtrada pela mediana.

Nome do pacote: WIP
Nome da classe: Matriz
Definição: public class Matriz extends java.lang.Object
Descrição: Classe que define uma matriz.
Campos
public double[][] mtx Matriz de valores.
Construtores
public Matriz(int altura, int largura) Construtor da classe. Cria uma nova matriz com altura e largura determinadas. **Parâmetros:** altura - Altura da matriz. largura - Largura da matriz.
Métodos
public int getHeight () Método que retorna a altura da matriz. **Retorna:** Altura da matriz.
public int getWidth () Método que retorna a largura da matriz. **Retorna:** Largura da matriz.
public void zeraMatriz () Método que zera a matriz.
public double[][] clone() Método que realiza a cópia da matriz para outra. Sobrescreve java.lang.Object.clone(). **Retorna:**

Cópia da matriz.
public double getAvg() Método que retorna o valor médio da imagem. **Retorna:** Valor médio dos *pixels* da imagem.
public double getMaxValue() Método que retorna o valor máximo da imagem. **Retorna:** Valor máximo da imagem.
public double getMinValue() Método que retorna o valor mínimo da imagem. **Retorna:** Valor mínimo da imagem.
public double getMinValueExZero() Método que retorna o menor valor da matriz, exceto o valor zero. **Retorna:** Menor valor da imagem, exceto o valor zero.
public Matriz mtxExp2 () Método que transforma uma imagem para uma imagem quadrada com dimensões em potência de 2. **Retorna:** Imagem com dimensões em potência de 2.
public Matriz multEscalar (double num) Método que multiplica uma matriz por um número escalar. **Parâmetros:** num - Número escalar que será multiplicado à matriz. **Retorna:** Matriz com os valores atualizados.

Apêndice C – Documentação do sistema desenvolvido

Este apêndice tem como objetivo documentar o sistema desenvolvido. Para isso, são utilizados alguns diagramas de forma a facilitar a compreensão, como no caso do diagrama de casos de uso, bem como servir como um guia de como o sistema foi desenvolvido, no caso do diagrama de classes. O diagrama de casos de uso, exibido na Figura C.1, mostra como um usuário pode interagir com o sistema.

Figura C.1 – Diagrama de casos de uso do sistema RCMW

Cada um dos círculos presentes na Figura C.1 representa uma operação que o usuário pode realizar. A descrição dessas operações encontra-se a seguir.

Abrir imagem: o usuário indica o endereço completo da imagem a ser carregada. Caso o endereço exista, o sistema carrega a imagem e exibe-a na tela.
Salvar imagem: o usuário indica a imagem a ser armazenada e o endereço completo da mesma. O sistema a armazena no local especificado.
Exibir/ocultar informações: exibe ou oculta informações da imagem exibida na tela, tais como nome, resolução e o histograma da imagem.
Salvar histograma: armazena o histograma da imagem escolhida.
Solicitar processamento de imagem: solicita o processamento da imagem escolhida, indicando a *wavelet* e o nível da transformada desejados.
Marcar ROI como falso positivo: marca a ROI escolhida como falso positivo.
Alterar tipo de borda da ROI: Altera o tipo da borda da ROI escolhida para o tipo escolhido pelo usuário.
Imprimir relatório: Imprime um relatório básico com informações sobre o processamento de uma imagem.

O diagrama de classes do sistema construído encontra-se na Figura C.2. Esse diagrama serve como um guia para verificar como as classes foram construídas, a definição de seus construtores e quais são seus campos e métodos públicos. Em seguida encontra-se a documentação das classes presentes no diagrama.

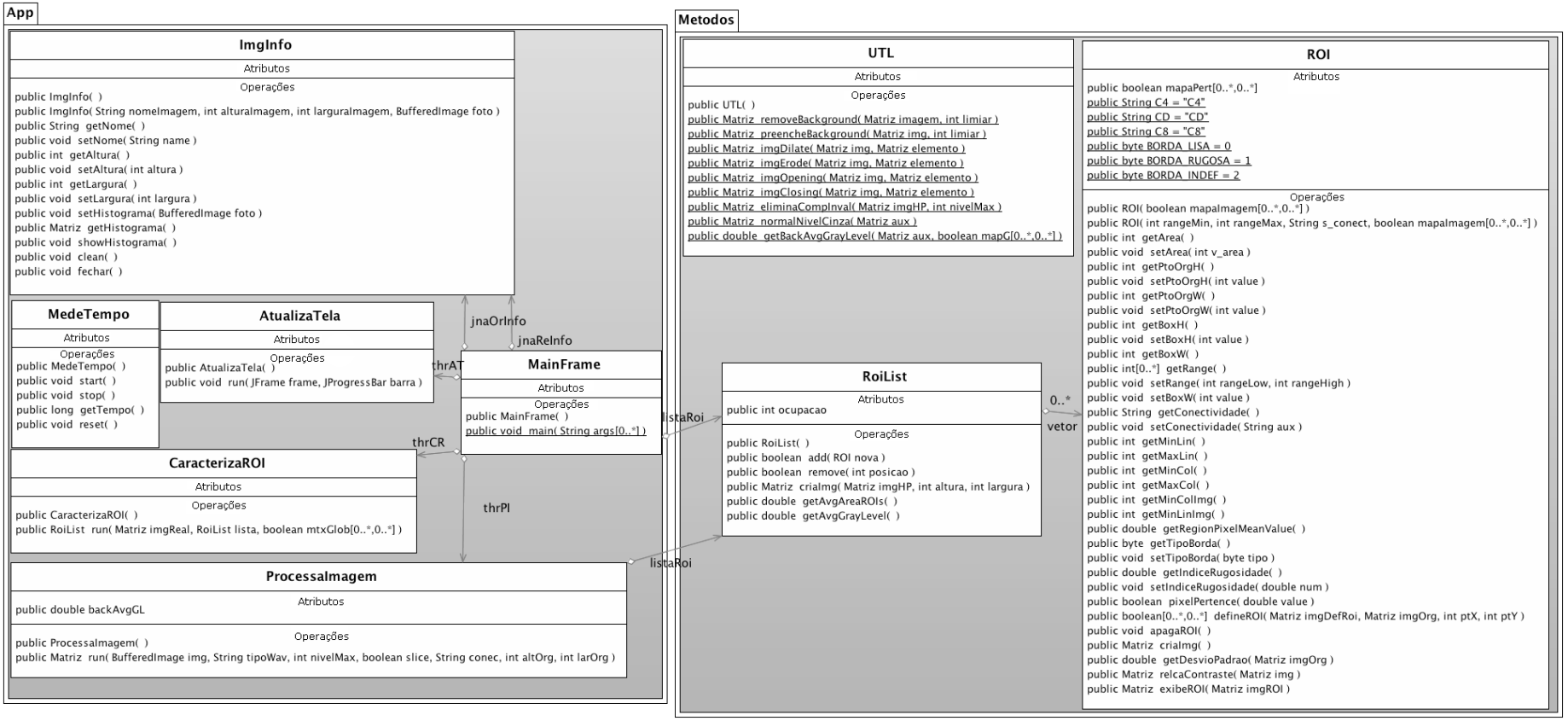


Figura C.2 – Diagrama de classes do sistema RCMW

Nome do pacote: App
Nome da classe: AtualizaTela
Definição: public class AtualizaTela extends Thread
Descrição: Classe que representa um Thread para atualizar a tela.
Construtores
public AtualizaTela() Cria uma nova instância desta classe.
Métodos
public void run(JFrame frame, JProgressBar barra) Método que executa a Thread. **Retorna:** frame - Frame a ser atualizado. barra - Barra de progresso que será atualizada.

Nome do pacote: App
Nome da classe: CaracterizaROI
Definição: public class CaracterizaROI extends Thread
Descrição: Classe que representa um Thread para caracterizar as ROIs.
Construtores
public CaracterizaROI() Cria uma nova instância desta classe.
Métodos
public RoiList run(Matriz imgReal, RoiList lista, boolean[][] mtxGlob) Método que executa a caracterização das ROIs. **Parâmetros:** imgReal - Imagem realçada. lista - Lista contendo as ROIs detectadas. mtxGlob - Matriz de pertinência global das ROIs na imagem. **Retorna:** Lista contendo informações das ROIs atualizadas.

Nome do pacote: App
Nome da classe: ImgInfo
Definição: public class ImgInfo extends JinternalFrame

Descrição: Classe que representa uma janela de informações da imagem.
Construtores
public ImgInfo() Construtor da classe. Cria um novo objeto desta classe.
public ImgInfo(String nomeImagem, int alturaImagem, int larguraImagem, BufferedImage foto) Construtor da classe. Cria uma nova janela com o nome, altura e largura fornecidos. **Parâmetros:** nomeImagem - Nome da imagem exibida. alturaImagem - Altura da imagem exibida. larguraImagem - Largura da imagem exibida. foto - Imagem da qual será criado o histograma.
Métodos
public String getNome() Método que retorna o nome da imagem. **Retorna:** Nome da imagem.
public void setNome(String name) Método que altera o nome exibido. **Parâmetros:** name - Novo nome exibido.
public int getAltura() Método que retorna a altura da imagem. **Retorna:** Altura da imagem.
public void setAltura(int altura) Método que altera o valor de altura exibido. **Parâmetros:** altura - Novo valor de altura exibido.
public int getLargura() Método que retorna a largura da imagem. **Retorna:** Largura da imagem.
public void setLargura(int largura) Método que altera o valor de largura exibido. **Parâmetros:**

largura - Novo valor de largura exibido.
public void setHistograma(BufferedImage foto) Método que altera a imagem do histograma. **Parâmetros:** foto - Imagem d o histograma.
public Matriz getHistograma() Método que retorna o histograma. **Retorna:** Histograma.
public void showHistograma() Método que exibe o histograma na janela.
public void clean() Método que limpa os valores da janela.
public void fechar() Método que fecha uma janela.

Nome do pacote: App
Nome da classe: MainFrame
Definição: public class MainFrame extends javax.swing.JFrame
Descrição: Classe que representa o Frame principal da aplicação.
Campos
public RoiList listaRoi Lista de ROIs encontradas durante a fase de crescimento de região.
Construtores
public MainFrame() Construtor da classe.
Métodos
public static void main(String args[]) Método que inicia a execução da classe. **Parâmetros:** args - Argumentos de comando de linha.

Nome do pacote: App
Nome da classe: MedeTempo
Definição: public class MedeTempo extends java.lang.Object

Descrição: Classe utilizada na medição do tempo de processamento de blocos.
Construtores
public MedeTempo() Cria um novo objeto da classe e zera seus valores.
Métodos
public void start() Método que inicia a medição de tempo.
public void stop() Método que finaliza a medição de tempo.
public long getTempo() Método que retorna o tempo gasto. **Retorna**: Tempo gasto.
public void reset() Método que zera os valores da medição de tempo.

Nome do pacote: App
Nome da classe: ProcessaImagem
Definição: public class ProcessaImagem extends Thread
Descrição: Classe que representa um Thread para processar as imagens.
Campos
public double backAvgGL Valor médio de nível de cinza do fundo da imagem.
Construtores
public ProcessaImagem() Construtor da classe.
Métodos
public Matriz run(BufferedImage img, int tipoWav, int nivelMax, boolean slice, String conec, int altOrg, int larOrg) Método que executa a Thread. **Parâmetros:** img - Imagem a ser processada. tipoWav - *Wavelet* escolhida. nivelMax - Nível de decomposição da transformada. slice - Indica se a imagem a ser processada é uma fatia ou não.

conec - Conectividade utilizada na fase de crescimento de região. altOrg - Altura da imagem original (sem ajustes). larOrg - Largura da imagem original (sem ajustes). **Retorna**: Imagem reconstruída.

Nome do pacote: Metodos
Nome da classe: ROI
Definição: public class ROI extends java.lang.Object
Descrição: Classe que representa uma ROI de uma mamografia.
Campos
public boolean[][] mapaPert Matriz que define se um *pixel* pertence a uma ROI ou não.
public boolean[][] globalMp Matriz de pertinência global da imagem.
public static String C4 = "C4" String que representa conectividade de 4.
public static String CD = "CD" String que representa conectividade diagonal.
public static String C8 = "C8" String que representa conectividade de 8.
public static byte BORDA_LISA = 0 Número que representa borda lisa.
public static byte BORDA_RUGOSA = 1 Número que representa borda rugosa.
public static byte BORDA_INDEF = 2 Número que representa borda indefinida.
Construtores
public ROI(boolean[][] mapaImagem) Construtor da classe. Inicia a ROI com valores nulos. **Parâmetros:** mapaImagem - Matriz de pertinência global da imagem.
public ROI(int rangeMin, int rangeMax, String s_conect, boolean[][] mapaImagem) Construtor da classe. Inicia o ROI com os valores especificados. **Parâmetros:**

rangeMin - Valor mínimo do intervalo utilizado para definir conectividade. rangeMax - Valor máximo do intervalo utilizado para definir conectividade. s_conect - Conectividade utilizada. mapaImagem - Matriz de pertinência global da imagem.
Métodos
public int getArea() Retorna a área da ROI. **Retorna**: Área da ROI.
public void setArea(int v_area) Altera a área da ROI. **Parâmetros:** v_area - Novo valor de área da ROI.
public int getPtoOrgH() Retorna a altura da ROI na imagem original. **Retorna**: Altura da ROI na imagem original.
public void setPtoOrgH(int value) Altera a altura da ROI na imagem original. **Parâmetros:** value - Nova altura da ROI na imagem original.
public int getPtoOrgW() Retorna a largura da ROI na imagem original. **Retorna**: Largura da ROI na imagem original.
public void setPtoOrgW(int value) Altera a altura da ROI na imagem original. **Parâmetros:** value - Nova largura da ROI na imagem original.
public int getBoxH() Retorna a altura da menor matriz na qual a ROI se encaixa. **Retorna**: Altura da box da ROI.
public void setBoxH(int value) Altera a altura da menor matriz na qual a ROI se encaixa. **Parâmetros:** value - Nova altura da box.

public int getBoxW()

Retorna a largura da menor matriz na qual a ROI se encaixa.

Retorna:

Largura da box da ROI.

public void setBoxW(int value)

Altera a largura da menor matriz na qual a ROI se encaixa.

Parâmetros:

value - Nova largura da box.

public int[] getRange()

Método que retorna o intervalo que define a conectividade entre *pixels*.

Retorna:

Intervalo de conectividade.

public void setRange(int rangeLow, int rangeHigh)

Método que altera o intervalo que define a conectividade entre *pixels*.

Parâmetros:

rangeLow - Valor mínimo do intervalo.

rangeHigh - Valor máximo do intervalo.

public String getConectividade()

Método que retorna a conectividade utilizada em uma ROI.

Retorna:

Conectividade da ROI.

public void setConectividade(String aux)

Método que altera a conectividade utilizada em uma ROI.

Parâmetros:

aux - Nova conectividade.

public int getMinLin()

Método que retorna o valor mínimo de linha utilizado na matriz temporária de pertinência.

Retorna:

Valor mínimo de linha.

public int getMaxLin()

Método que retorna o valor máximo de linha utilizado na matriz temporária de pertinência.

Retorna:

Valor máximo de linha.

public int getMinCol()

Método que retorna o valor mínimo de coluna utilizado na matriz temporária de pertinência.

Retorna:

Valor mínimo de coluna.

public int getMaxCol()

Método que retorna o valor máximo de coluna utilizado na matriz temporária de pertinência.

Retorna:

Valor máximo de coluna.

public int getMinColImg()

Método que retorna o valor mínimo de coluna da ROI na imagem original.

Retorna:

Valor mínimo de coluna.

public int getMinLinImg()

Método que retorna o valor mínimo de linha da ROI na imagem original.

Retorna:

Valor mínimo de linha.

public double getRegionPixelMeanValue()

Método que retorna o valor médio da imagem na região da ROI.

Retorna:

Média da imagem na região da ROI.

public byte getTipoBorda()

Método que retorna o tipo de borda da ROI.

Retorna:

Tipo da borda da ROI.

public void setTipoBorda(byte tipo)

Método que altera o tipo de borda da ROI.

Parâmetros:

tipo - Tipo de borda da ROI.

public double getIndiceRugosidade()

Método que retorna o índice de rugosidade/lisura da ROI.

Retorna:

Índice de rugosidade/lisura da ROI.

public void setIndiceRugosidade(double num)

Método que altera o índice de rugosidade/lisura da ROI.

Parâmetros:

num - Novo valor.

public boolean pixelPertence(double value)

Método que verifica se um determinado valor pertence ao conjunto de conectividade.

Parâmetros:

value - Valor a ser verificado.

Retorna:

True - se o valor pertence ao conjunto. False, caso contrário.
public boolean[][] defineROI(Matriz imgDefRoi, Matriz imgOrg, int ptX, int ptY) Define a ROI baseada nos dados informados. **Parâmetros:** imgDefRoi - Imagem na qual será definida a ROI. imgOrg - Imagem original. ptX - Ponto no eixo X na imagem original onde se iniciará o processo de crescimento. ptY - Ponto no eixo Y na imagem original onde se iniciará o processo de crescimento. **Retorna**: Matriz de pertinência global da imagem.
public void apagaROI () Método que apaga uma ROI.
public Matriz criaImg () Método que cria uma imagem a partir da matriz de pertinência da ROI. **Retorna**: Matriz contendo branco se o *pixel* pertence àquela ROI, ou preto caso contrário.
public double getDesvioPadrao(Matriz imgOrg) Método que retorna o desvio padrão da imagem original na região da ROI. **Parâmetros:** imgOrg - Imagem original. **Retorna**: Desvio padrão da imagem na região da ROI.
public Matriz relcaContraste(Matriz img) Método que realça o contraste da imagem na região da ROI. **Parâmetros:** img - Imagem a ser realçada. **Retorna**: Imagem realçada.
public Matriz exibeROI(Matriz imgROI) Método que cria uma imagem exibindo uma ROI. **Parâmetros:** imgROI - Imagem original. **Retorna**: Imagem com a ROI exibida.

Nome do pacote: Metodos
Nome da classe: RoiList

Definição: public class RoiList extends java.lang.Object
Descrição: Classe que define uma lista de ROIs.
Campos
public ROI[] vetor Vetor contendo as ROIs.
public int ocupação Ocupação da lista.
Construtores
public RoiList() Construtor. Cria uma nova lista de ROIs.
Métodos
public boolean add(ROI nova) Método que adiciona uma nova ROI na lista. **Parâmetros:** nova - Nova ROI a ser adicionada na lista. **Retorna**: True - se a ROI for adicionada com sucesso, False - caso contrário.
public boolean remove(int posicao) Método que remove uma ROI da lista. **Parâmetros:** posicao - Posição da ROI a ser removida da lista. **Retorna**: True - se a ROI for removida com sucesso, False - caso contrário.
public Matriz criaImg (Matriz imgHP, int altura, int largura) Método que cria uma imagem baseada na lista de ROIs. **Parâmetros:** imgHP - Imagem original. altura - Altura da imagem. largura - Largura da imagem. **Retorna**: Imagem resultante.
public double getAvgAreaROIs() Método que retorna o valor médio de área das ROIs. **Retorna**: Valor médio de área das ROIs.
public double getAvgGrayLevel()

Método que retorna o valor médio de níveis de cinza das ROIs. **Retorna**: Valor médio de níveis de cinza das ROIs.

Nome do pacote: Metodos
Nome da classe: UTL
Definição: public class UTL extends java.lang.Object
Descrição: Classe com os métodos utilizados no algoritmo.
Construtores
public UTL() Cria uma nova instância de UTL.
Métodos
public static Matriz removeBackground(Matriz imagem, int limiar) Método que remove o fundo de uma mamografia baseado na D4. **Parâmetros:** imagem - Imagem da qual será removida o fundo. limiar - Limiar utilizado para verificar se o *pixel* pertence ao fundo da imagem. **Retorna**: Matriz com o fundo em branco (255).
public static Matriz preencheBackground(Matriz img, int limiar Método que preenche o fundo de uma imagem com zeros (preto). **Parâmetros:** img - Imagem a ter o fundo preenchido. limiar - Limiar utilizado para definir o fundo da imagem. **Retorna**: Imagem com o fundo preenchido.
public static Matriz imgDilate(Matriz img, Matriz elemento) Método que faz a dilatação em uma imagem. **Parâmetros:** img - Imagem a ser dilatada. elemento - Elemento estruturante a ser utilizado na dilatação. **Retorna**: Imagem dilatada.
public static Matriz imgErode(Matriz img, Matriz elemento) Método que faz a erosão em uma imagem. **Parâmetros:**

img - Imagem a ser erodida. elemento - Elemento estruturante a ser utilizado na erosão. **Retorna**: Imagem erodida.
public static Matriz imgOpening(Matriz img, Matriz elemento) Método que realiza a operação de abertura em uma imagem. **Parâmetros:** img - Imagem a qual será aplicada a abertura. elemento - Elemento estruturante a ser utilizado na abertura. **Retorna**: Imagem resultante da aplicação da operação abertura.
public static Matriz imgClosing(Matriz img, Matriz elemento) Método que realiza a operação de fechamento em uma imagem. **Parâmetros:** img - Imagem a qual será aplicada o fechamento. elemento - Elemento estruturante a ser utilizado no fechamento. **Retorna**: Imagem resultante da aplicação da operação fechamento.
public static Matriz eliminaCompInval (Matriz imgHP, int nivelMax) Método que elimina componentes inválidos da imagem. **Parâmetros:** imgHP - Imagem a ser analisada. nivelMax - Nível máximo da decomposição *wavelet*. **Retorna**: Imagem processada.
public static Matriz normalNivelCinza (Matriz aux) Método que realiza a normalização de níveis de cinza. **Parâmetros:** aux - Imagem a ser normalizada. **Retorna**: Imagem normalizada.
public static double getBackAvgGrayLevel(Matriz aux, boolean[][] mapG) Método que retornar o valor médio de nível de cinza dos *pixels* da imagem que fazem parte do fundo. **Parâmetros:** aux - Imagem a ser processada. mapG - Matriz de pertinência da imagem. **Retorna**:

Valor médio de nível de cinza dos *pixels* da imagem que pertencem ao fundo da mesma.

Apêndice D – Manual de utilização do sistema desenvolvido

Este apêndice tem como objetivo ensinar como utilizar o sistema desenvolvido de forma rápida e intuitiva. Ao abrir o sistema, a tela principal do mesmo é exibida, conforme pode ser visto na Figura D.1.

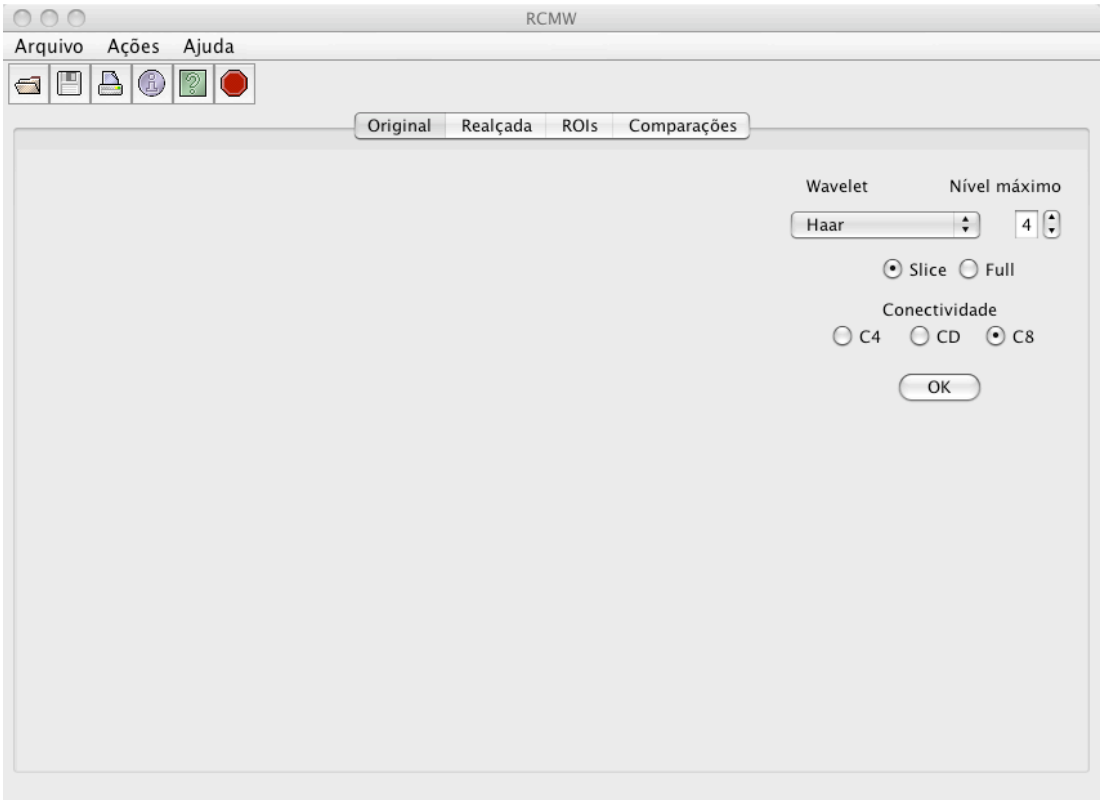


Figura D.1 – Tela principal do sistema

As principais funcionalidades básicas do sistema podem ser acessadas a partir da janela principal do mesmo. Ele contém um menu para a leitura, escrita e fechamento de imagens, assim como opções para exibir/ocultar informações sobre a imagem lida e uma opção para salvar o histograma da imagem. O sistema ainda possui uma barra de ferramentas para acessar suas principais funções de forma simplificada através do *mouse*. Uma imagem contendo a barra de ferramentas pode ser vista na Figura D.2. Os botões da barra de ferramentas representam, respectivamente, as opções de abrir uma imagem, salvar uma imagem, imprimir um relatório com dados sobre o processamento de uma imagem, exibir as informações de uma imagem, exibir a ajuda e encerrar o programa.

Figura D.2 – Barra de ferramentas

O primeiro passo na utilização do sistema consiste em carregar uma imagem. Para isso, basta acessar “Arquivo/Abrir” ou então selecionar o botão correspondente na barra de ferramentas. Uma nova janela de escolha de arquivos é aberta, onde a imagem desejada deve ser selecionada e o botão abrir selecionado. Ao realizar esse procedimento, a imagem é carregada e exibida na aba denominada “Original”. Caso a imagem possua tamanho maior que a janela do sistema, ela é redimensionada para ser exibida de forma apropriada. Assim que a imagem é lida, informações sobre a mesma são exibidas em uma sub-janela no canto inferior direito da aba atual, contendo o nome, a resolução e o histograma da mesma. Essa sub-janela pode ser ocultada selecionando o botão apropriado na barra de ferramentas ou indo em “Ações/Exibir informações”. O histograma da imagem carregada pode ser armazenado como uma imagem. Para isso, basta acessar “Ações/Salvar histograma”, e escolher o local onde o arquivo deve ser armazenado. Um exemplo do estado do sistema no momento em que uma imagem é carregada pode ser visto na Figura D.3.

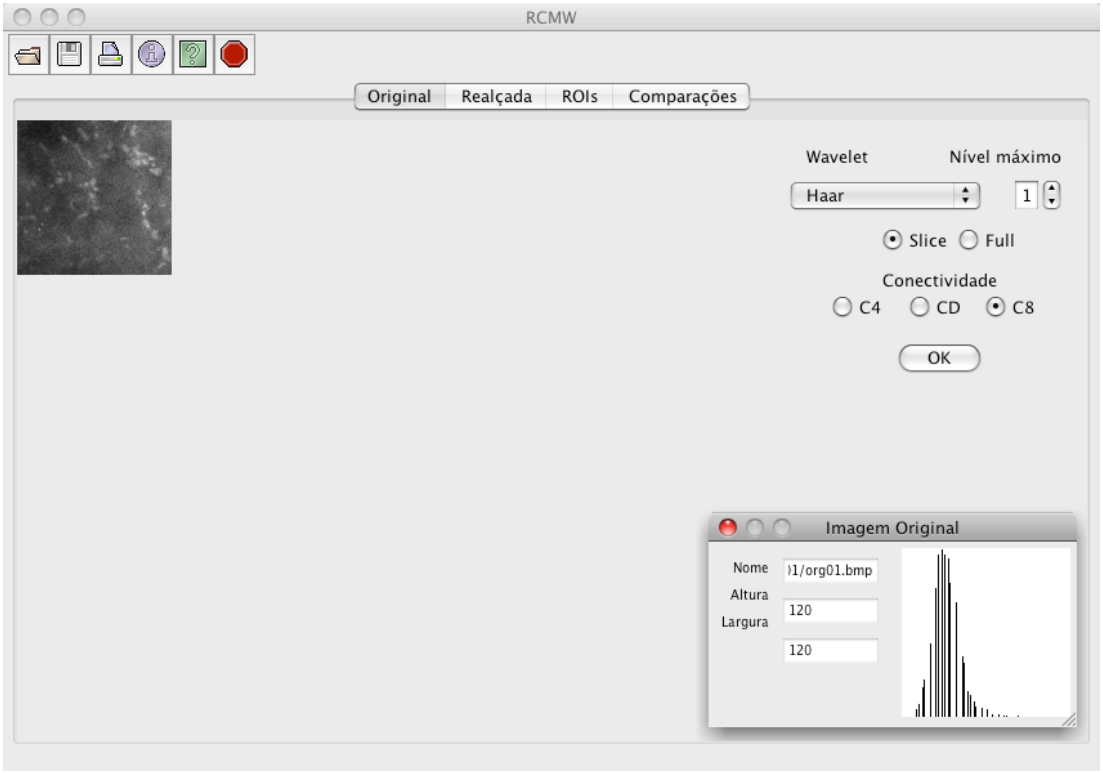


Figura D.3 – Imagem carregada

Após carregar a imagem desejada, é possível aplicar o método a ela. Basta selecionar a *wavelet* desejada no menu "Wavelet" e o nível de decomposição desejado (no método é utilizado o nível quatro), conforme pode ser visto na Figura D.4.

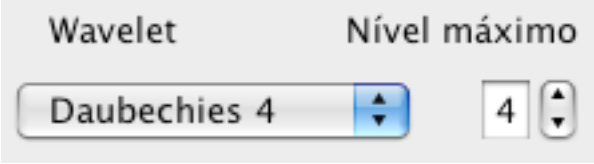


Figura D.4 – Escolha da *wavelet* utilizada

Apos a escolha da *wavelet*, ao selecionar o botão "OK", o processo se inicia e, uma vez terminado, a aba "Realçada" é exibida, contendo a imagem realçada obtida pela aplicação do método, o tempo gasto de processamento em segundos e a janela de informações dessa imagem, contendo sua resolução e seu histograma, conforme pode ser visto na Figura D.5.

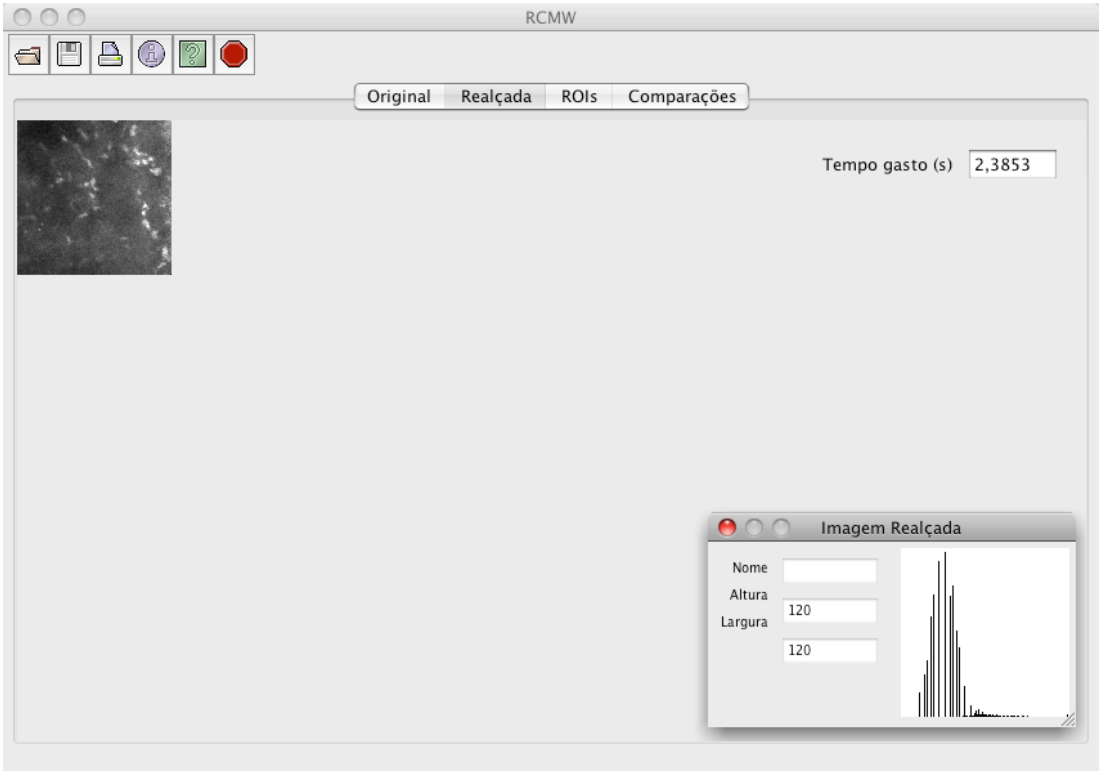


Figura D.5 – Exibição de imagem realçada

Da mesma forma que a imagem original, a imagem realçada pode ser armazenada no local desejado indo em “Arquivo/Salvar” ou selecionando o botão correspondente na barra de ferramentas. O histograma da mesma também pode ser armazenado da mesma forma como o histograma da imagem original.

A aba “Comparações" exibe as imagens original e realçada uma do lado da outra, juntamente com seus respectivos histogramas. Um exemplo da exibição dessa aba pode ser visto na Figura D.6.

A aba “ROIs” exibe informações sobre as microcalcificações detectadas. Ao selecionar essa aba, é possível observar o número de ROIs encontradas, bem como o número de ROIs lisas, rugosas e indefinidas, o valor médio de área e o valor médio de nível de cinza delas. Ao ser exibida, essa aba exibe uma versão da imagem original destacando apenas a primeira ROI encontrada. Para selecionar uma ROI diferente, basta escolher uma entre elas na lista “ROI #”, ou então percorrer uma a uma selecionando os botões “Anterior” e “Próximo”. Um exemplo da exibição dessa aba pode ser visto na Figura D.7.

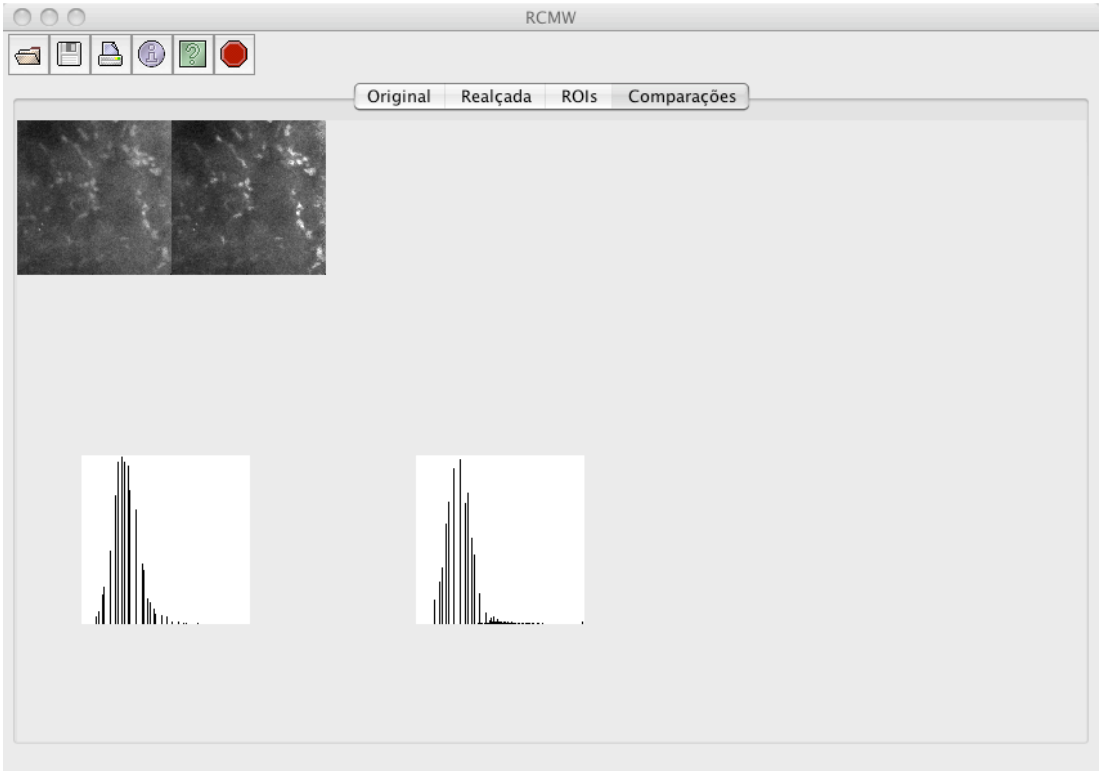


Figura D.6 – Visualização das imagens original e realçada lado a lado

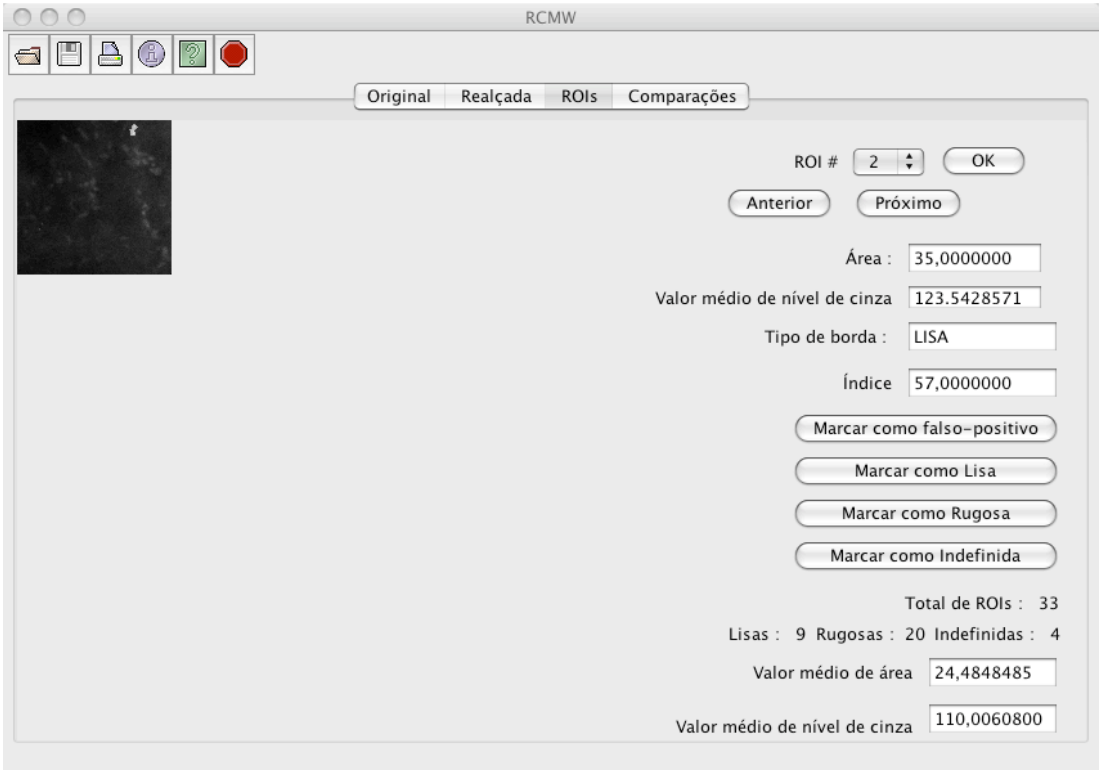


Figura D.7 – Visualização individual de ROIs

Ao selecionar uma ROI, além da imagem desta ser destacada na imagem original, são exibidas as seguintes informações sobre a mesma: valor de área, valor médio de nível de cinza, tipo de borda e índice de rugosidade/lisura da borda. O usuário pode ainda marcar uma ROI como falso positivo. Nesse caso, a ROI é removida da lista de ROIs, e a região da mesma não fica mais realçada na imagem exibida na aba “Realçada" e na imagem final. A lista de ROIs é atualizada, e as informações como número total de ROIs, número de determinado tipo de ROI e valores médios de área e nível de cinza também são atualizados. É possível, ainda, alterar o tipo de borda da ROI, marcando-a como lisa, rugosa ou indefinida. Nesse caso, o tipo da borda da ROI é alterado com índice máximo para o tipo escolhido e as informações sobre a quantidade de cada um dos tipos de ROIs são atualizadas. É possível ainda imprimir um pequeno relatório com informações sobre o processamento da imagem. Basta acessar “Arquivo/Imprimir” ou então selecionar o botão correspondente na barra de ferramentas.

Após a análise da imagem, o usuário pode voltar o sistema para o estado inicial, sem imagens carregadas, bastando, para isso, acessar “Arquivo/Fechar”. O usuário pode ainda carregar outra imagem, ou então encerrar o programa, indo em “Arquivo/Sair”, ou então selecionando o botão correspondente na barra de ferramentas.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Administração
Baixar livros de Agronomia
Baixar livros de Arquitetura
Baixar livros de Artes
Baixar livros de Astronomia
Baixar livros de Biologia Geral
Baixar livros de Ciência da Computação
Baixar livros de Ciência da Informação
Baixar livros de Ciência Política
Baixar livros de Ciências da Saúde
Baixar livros de Comunicação
Baixar livros do Conselho Nacional de Educação - CNE
Baixar livros de Defesa civil
Baixar livros de Direito
Baixar livros de Direitos humanos
Baixar livros de Economia
Baixar livros de Economia Doméstica
Baixar livros de Educação
Baixar livros de Educação - Trânsito
Baixar livros de Educação Física
Baixar livros de Engenharia Aeroespacial
Baixar livros de Farmácia
Baixar livros de Filosofia
Baixar livros de Física
Baixar livros de Geociências
Baixar livros de Geografia
Baixar livros de História
Baixar livros de Línguas

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo