Introdução

Atualmente um campo de pesquisa para o processamento de imagens é o que envolve a segmentação de imagens mamográficas. Aplicam-se algoritmos em mamogramas digitalizados com o intuito de auxiliar no diagnóstico médico.

Desta forma inicialmente digitaliza-se um mamograma. De posse desta imagem, convertida num arquivo de computador podemos manipular via software , este arquivo que contém a imagem em questão.

Neste relatório o objetivo foi o de apresentar uma proposta de algoritmo de segmentação, chamado algoritmo de Water shed, além de se fazer um estudo teórico das bases teóricas das operações de segmentação..

Resumo deste relatório

Neste relatório inicialmente apresenta-se uma discussão teórica à respeito do pré processamento de imagens mamográficas, e imagens digitais em geral.

Desta forma inicialmente é apresentado um resumo sobre as técnicas de pré processamento e filtragem de imagens. Este compreende a explanação sobre certas operações básicas para os algoritmos de segmentação.

Em seguida apresentamos um outro resumo sobre as técnicas de pré- processamento mais utlizadas para o a segmentação de mamogramas digitais.

Para finalizar apresenta-se a descrição do algoritmo de watershed, que foi implementado e testado.

Este teste é apresentado à seguir após estas considerações teóricas.

I Parte Teórica

1- Considerações teóricas sobre as técnicas de pré- processamento e filtragem de imagens digitais.

1.1 Introdução

O objetivo de utilizarmos as técnicas de filtragem de imagens digitais (que consistem em técnicas de pré- processamento) em mamogramas digitalizados é de poder termos uma melhoria na visualização das microcalcificações existentes nos mamogramas.

Na literatura encontramos diversos filtros, mas podemos dividi-los em dois grupos: filtros de borda (**filtros espaciais**) e **filtros morfológicos.** Entretanto podemos ter ainda, como uma outra técnica de pré- processamento, a equalização do histograma gerado pela digitalização da imagem.

Os filtros espaciais são aqueles, que utilizam **máscaras de convolução** para atuar sobre a imagem. Incluem-se neste tipo os filtros de detecção e melhoria de borda e os filtros de média e mediana.

Já os filtros morfológicos consistem naqueles que utilizam de operações matemáticas, chamadas de morfológicas para atuar sobre a imagem. As duas operações morfológicas básicas são a dilação e a erosão, que podem ser combinadas para constituir as operações de abertura e fechamento da imagem.

Além destes dois tipos, podemos efetuar a equalização o histograma da imagem. Quando digitalizamos uma imagem podemos obter seu histograma de distribuição de *pixels*. Computacionalmente podemos equalizar este histograma, isto é fazermos uma nova distribuição de *pixels* de maneira a termos uma imagem melhor.

1.2- Representação das imagens, histogramas e operações com o histograma

1.2.1- Representação das imagens digitais.

As imagens digitais são representadas por matrizes(ver pag. 5 do relatório anterior). Os elementos dessa matriz são números binários. Numa imagem binária os elementos da matriz são 0 ou 1. Já numa imagem com 256 níveis de cinza, temos os elementos da matriz compostos por números binários de oito *bits*, tratados porém como, níveis 0, 1, 2, 3, 4, ...253, 254, 255. Estas matrizes podem ser alteradas por operações computacionais, que modificam as matrizes e portanto as imagens por elas representadas.

1.2.2- Histograma de uma imagem digital.

Consideremos uma imagem digital com 256 níveis de cinza. Segundo [Myler] um histograma dessa imagem descreve a porcentagem da ocorrência dos 256 níveis de cinza que constituem a imagem em questão.

Os valores de nível de cinza podem ser representados por um número inteiro Gi, cujo índice i representa o iésimo nível de cinza. Esta imagem possui nt pixels e ni é o número de pixels que possuem valor de nível de cinza igual a Gi.

Define-se o histograma da imagem como:

$$hi = ni / nt \text{ para } i=0 \text{ até } (M-1) \qquad (1)$$

Onde M é o número total de valores de nível de cinza de uma imagem. Para o caso de uma imagem com 256 níveis, M= 256.

Deste modo o número hi representa a porcentagem sobre o total nt de pixels, de um determinado nível de cinza com ni pixels na imagem. Assim devemos ter:

$$\sum_{i=0}^{M-1} hi = 1 \qquad (2)$$

Podemos ter informação sobre uma imagem, apenas analisando seu histograma. Considere a figura1 abaixo.

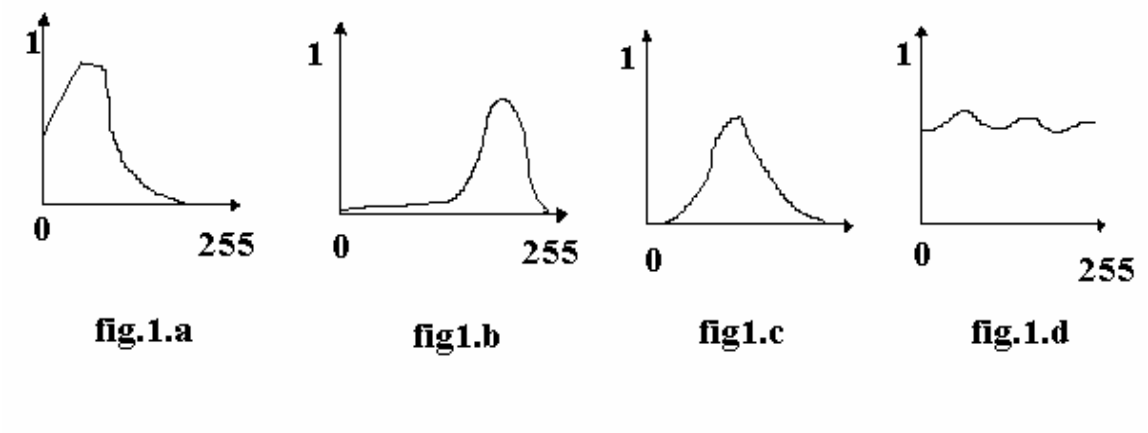

Figura1- Exemplos de histogramas de várias imagens digitais
Porcentagem do nível de cinza por cada nível

Na figura 1.a, temos uma imagem com maior número de *pixels* próximos ao zero (referência de preto). Desta forma temos uma imagem mais escura, já que os há predominância de pixels com valores de

níveis de cinza mais baixos. Na fig1.b, temos o contrário. O nível 255 é a referência de branco, portanto há um maior número de pixels com valores próximos a 255. Assim a imagem tem o aspecto geral mais claro. Na figura 1.c temos uma imagem aonde predominam os valores de nível de cinza médios. Assim a imagem não possui predominância de escuro ou claro. Na figura 1.d temos uma distribuição normal de níveis de cinza. O livro, **Computer Image Recipes** (de **Myler, H.R, Weeks**, PTR Prentice Hall, 1993 pag. 56), nos mostra um algoritmo em linguagem C para a obtenção desse histograma.

1.2.3- Equalização do Histograma de uma imagem digital

Segundo [Low], as vezes é interessante encontrar uma função que vá melhorar o contraste geral de uma imagem, espalhando a distribuição dos níveis de cinza, para que tenhamos o mesmo número de pixels para cada nível de cinza. Este processo é chamado de equalização do histograma. Porém na prática nào conseguimos esta distribuição, pois ela é idealizada.

Este processo nos proporciona uma melhoria para a visualização de uma imagem digital. Na fig.2 abaixo há um exemplo de uma imagem equalizada ao lado da imagem original.

Figura 2- Imagem original à esquerda e Imagem cujo histograma foi equalizado à direita

Para equalizarmos uma imagem de acordo com [Myler] devemos considerar dois histogramas, o proveniente da imagem original hfi, e um de uma imagem que já tenha sido equalizada, hgi. Nosso objetivo é então encontrar uma transformação que nos permita equalizar o histograma hfi ou seja:

$$g = P[f] \qquad (3)$$

Em um histograma uniforme cada pixel tem a mesma probabilidade de ocorrer. Ou seja:

$$hgi = 1/256 \qquad (4)$$

E uma transformação que equalize o histograma de uma imagem é simplesmente, a soma dos elementos do histograma da imagem original. Assim:

$$gi = P(fi) \sum_{j=0}^{i} hfj \qquad (5)$$

Onde m é número de níveis de cinza na imagem original e i é o iésimo nível de cinza na imagem equalizada. Como exemplo, utilizando a equação (5):

- o nível zero da imagem equalizada é igual a g0= (m-1).h0.
- o nível três da imagem equalizada é igual a g3= (m-1) (ho+h1+h2+h3)
- o nível 100 da imagem equalizada é dado por

$$g100 = (m-1)\sum_{j=0}^{100} hfj$$

1.3- Filtros Espaciais

1.3.1- Convolução e máscaras de convolução

De acordo com [Gonzales 87] podemos definir a convolução entre duas funções f(x) e g(x) pode ser definida pela integral:

$$\int_{\infty}^{-\infty} f(\alpha).g(x-\alpha)d\alpha, \qquad (1)$$

Suponhamos agora que ao invés de termos funções contínuas , f(x) e g(x) são discretizadas em vetores de dimensões A e B respectivamente:
{ f(0), f(1), f(2),……., f(A-1)}, e {g(0), g(1), g(2),……..g(B-1)} . De acordo com [Gonzales 87] as funções discretas devem ser supostas M periódicas . A convolução de ambas deve ser também M periódica. Desta forma podemos definir a convolução discreta entre duas funções como sendo :

$$f(x) * g(x) = \sum_{m}^{M-1} f(m)g(x-m) \qquad (2)$$

para x= 0,1, 2,……., M-1. A função de convolução é um vetor discreto, periódico de dimensão M , com os valores x=0,1,2,3…….M-1 e descrevem um período inteiro de f(x)*g(x).

A convolução bi- dimensional é análoga a (1):

$$f(x,y) * g(x,y) = \int_{-\infty}^{\infty} \int f(\alpha,\beta)g(x-\alpha, y-\beta)d\alpha d\beta \qquad (3)$$

Fazendo f(x,y), g(x,y) serem discretizadas por matrizes de tamanho AxB e CxD respectivamente , e considerando que estas matrizes são M periódicas (na direção X) e N periódicas (na direção Y). A convolução bi- dimensional de f(x,y) e g(x,y) discretas é dada por :

$$f(x,y) * g(x,y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) g(x-m, y-n) \qquad (4)$$

Para x=0,1,2,3…….,M-1 e y=0,1,2,3…….,N-1.

Em processamento de imagens, a convolução discreta representa a convolução de duas matrizes, uma delas representa a imagem a ser processada e outra representa uma máscara a ser convoluída da imagem. Por exemplo, os filtros de detecção de borda são matrizes que convoluímos da imagem que desejamos alterar.

Segundo RUSS J.C em **The Image Processing Handbook** [Russ 1995], na convolução, os valores centrais da máscara, é multiplicado por cada pixel e sua vizinhança em uma pequena região, os resultados são somados e armazenados no local onde o pixel original estava localizado. Isto é aplicado para todos os pixels na imagem. Em todos os casos os valores dos pixels originais são utilizados na multiplicação e adição, e os novos valores alterados são utilizados para produzir uma nova imagem. Consideramos a máscara abaixo:

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

Na prática a mais rápida implementação seria multiplicar o pixel e os seus 8 pixels vizinhos imediatos pelos inteiros 1,2, ou 4 , os produtos seriam somados e então o total dividido por 16. Neste caso, ao utilizar os inteiros , que são potências de dois, permitimos que as operações matemáticas sejam rápidas .

De acordo com [Myler, pag.99] um algoritmo em linguagem C que calcula a convolução discreta entre uma Imagem de tamanho MxM e uma máscara 3x3 esta listado abaixo:

```
for (i= 1; i< M-2;i++)
        for(j= 1; j< M-2; ++j)
                Result[i][j] =  Image[i-1] [j-1] * mask[0] [0] +
```

```
Image[i-] [j+1]  * mask[0] [1] +
Image[i-1] [j]    * mask[0] [2] +
Image[i] [j-1]    * mask[1] [0] +
Image[i] [j]       * mask[1] [1] +
Image[i] [j+1]    * mask[1] [2] +
Image[i+1][j-1]  * mask[2] [0] +
Image[i+1][j]      *mask[2][1]+
Image[i+1][j+1]   *mask[2][2];
```

1.3.2- Filtros de detecção de bordas

As bordas em imagens digitais são consideradas como informação de altas freqüências. Deste modo, filtros de detecção de borda são considerados como filtros passa - alta . Considere o exemplo abaixo (extraído de [Myler])para melhor entendimento do que são bordas em uma imagem.

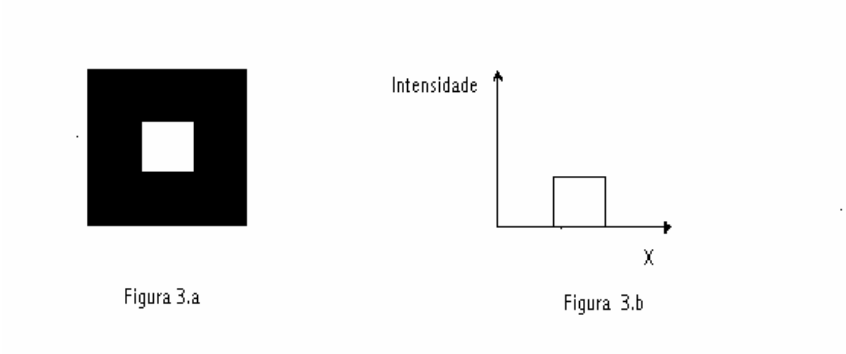

Figura 3- Imagem de um quadrado branco em fundo escuro e seu gráfico de intensidade de luz na direção x

Na figura 3.a acima temos uma imagem de um quadrado branco sobre um fundo escuro . Quando caminhamos na figura 3.a da esquerda para à direita na foto na direção x, inicialmente encontramos uma região totalmente escura. Isto se reflete no gráfico da figura 3.b , aonde podemos verificar que há uma região totalmente escura antes do quadrado. Há então, na figura 3.a, uma região que abruptamente passa do escuro para o claro. Como pode ser visto na figura 3.b, esta região clara é o comprimento do próprio quadrado. As bordas são então, delimitadoras da região que sabemos ser um quadrado.

Para detectarmos bordas verticais no lado esquerdo do quadrado da figura 3.a, podemos uma máscara de convolução que funciona como um filtro passa alta espacial de freqüência. Este filtro segundo [Myler] pode ser :

-1	0	1
-2	0	2

-1	0	1

Figura 4- Máscara de convolução para detecção de bordas verticais

Nas regiões da imagem aonde hajam valores de *pixels* iguais, a soma dos produtos será igual a zero. Desta forma as regiões homogêneas da imagem são removidas. Uma vez que a máscara de convolução atue sobre a imagem, quando os valores dos pixels, passam de nível baixo para alto (como numa borda vertical, a máscara fará com que tenhamos como saída o seu maior valor. Estes valores na imagem resultante, estarão nas bordas à esquerda.

Se transpormos verticalmente a matriz da figura 4, obtemos um filtro para detecção de bordas verticais à direita, como mostra a figura 5.

1	0	-1
2	0	-2
1	0	-1

Figura 5- Máscara para detecção de bordas verticais á esquerda

Este último filtro irá fazer com que as bordas à direita sejam ressaltadas. Da mesma forma podemos imaginar máscaras para detectar filtros horizontais abaixo ou acima na imagem. Este filtro bastante utilizado é o filtro de *Sobel.*

Podemos combinar os dois processos, de detecção de bordas verticais e através da fórmula abaixo, termos um novo processo para filtrar uma imagem:

$$S = \sqrt{V^2 + H^2} \qquad (7)$$

Neste caso aplicamos os filtros para encontrar as bordas horizontais e verticais, obtendo assim duas imagens, H e V , e em seguida aplicamos (7) para cada elemento das imagens.

Estes filtros são considerados operadores direcionais, pois a direção é importante a direção em que estão sendo aplicados. Abaixo na figura 6 apresentamos 3 filtros diferente aplicados à mesma foto.

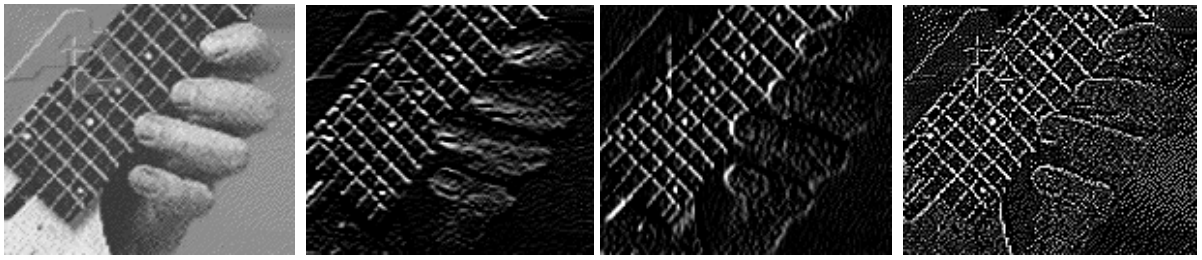

Figura 6.a *Figura 6.b* *Figura 6.c* *Figura 6.d*

Figura 6- Imagem original e imagens resultantes da aplicação de diversos filtros

Na figura 6.a temos a imagem original. Na figura 6.b foi aplicado uma máscara para encontrar as bordas horizontais. Nas figuras 6.c e 6.a foram aplicados respectivamente filtros para encontrar bordas verticais, e o filtro ***Sobel.***

Há um outro importante filtro para detecção de bordas é o filtro Laplaciano**.** Ele difere dos filtros anteriores por não ser um filtro direcional, isto é ele não depende do gradiente (taxa de mudança da imagem), porém ele suscetível a ruídos. Um exemplo, aplicado nos experimentos é o filtro ***Laplaciano*** 5x5 explicitado abaixo:

-1	-1	-1	-1	-1
-1	-1	-1	-1	-1
-1	-1	24	-1	-1
-1	-1	-1	-1	-1
-1	-1	-1	-1	-1

Figura 7 - Máscara laplaciana 5x5

E ao aplicarmos o filtro Laplaciano à imagem original 6.a, obtemos o resultado da figura 8.

Figura 8- Efeito do filtro Laplaciano

1.3.3 Filtros mediana e média

O filtro de média, é aquele que substitui um pixel pela média dos pixels vizinhos. Um exemplo segundo [Myler] , de um filtro de média esta abaixo na figura 9.

1	1	1	1	1	
1	1	1	1	1	
1	1	1	1	1	*1/25
1	1	1	1	1	

1 1 1 1 1

O objetivo deste filtro é eliminar regiões de pixels em que haja diferenças entre o restante da imagem. Isto contrapõe o objetivo dos filtros detetores de bordas, cujo objetivo é encontrar e acentuar as diferenças. Este filtro de média, suaviza a imagem como um todo como pode ser visto na figura 10.

Figura 10- Efeito do filtro de média aplicado sobre uma imagem

A imagem original pode ser vista na figura 2.

Já o filtro de mediana não pode ser computado usando-se uma máscara de convolução. Para que possamos aplicar este filtro devemos proceder conforme [Myler]. Como exemplo consideremos a seguinte região de uma imagem:

7 7 7
7 16 7
7 7 7

cuja mediana é 7. É possível converter a vizinhança dos pixels em um conjunto A= {7,7,7,716,7,7,7,7}. Ordenando A, ficamos com {7,7,7,7,7,7,7,7,16}. Daí obtemos a mediana no valor do meio deste conjunto.

Este procedimento, de mediana é eficaz quando desejamos eliminar um sinal ou pixel que sejam muito diferentes dos outros pixels da imagem. Ou seja, eliminar aéreas estranhas na imagem.

Abaixo temos a figura original 2, processada com um filtro de mediana.

Figura 11- Efeito do filtro de mediana sobre a Imagem

1.4 - Filtros Morfológicos

As duas operações morfológicas mais importantes são a dilação e a erosão. Ambas são baseadas na subtração e adição de Minkowski . Consideremos o exemplo abaixo, visto em [Myler].

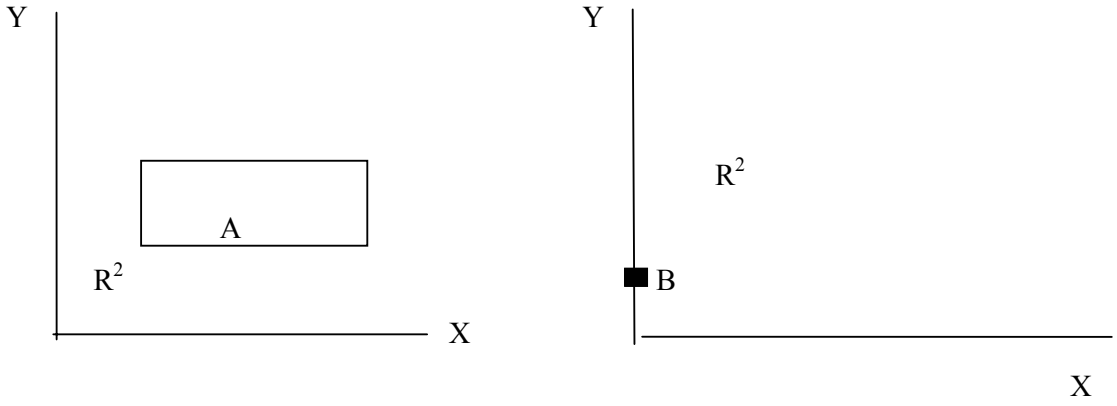

Figura 12- Dois objetos em um espaço R^2

Podemos definir a soma de Minkowski do objeto A com o objeto B, pelo símbolo $\pm$. Esta operação é definida como sendo:

$$A \oplus B = \{t \in R^2 : t = a + b, a \in A, b \in B\} \qquad (8)$$

A equação acima nos diz que, todos os elementos, a, do objeto A são somados à todos elementos, b, do objeto B, para formar um novo conjunto de elementos t. Devemos apenas considerar os contornos dos objetos para determinar a forma do novo objeto. No caso da figura 12, a adição de A com B, resulta na translação vertical de A da distância que o ponto B dista da origem, como verificamos na figura abaixo:

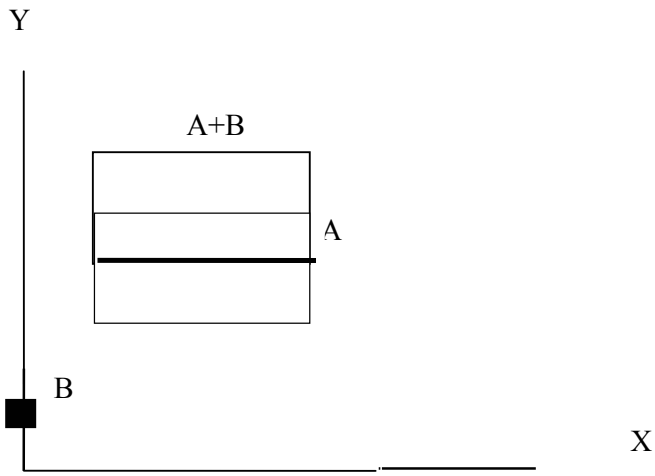

Figura 13- Objeto A somado com objeto B

Podemos também efetuar a subtração de Minskowski, dos dois objetos da figura 14. A subtração de Minskowski é definida como sendo:

$$A - B = (A^c \oplus B)^c \qquad (9)$$

Onde A^c e B^c são respectivamente o complemento de A e o complemento de B.

Na figura 12, o complemento de A são todos os pontos fora do quadrado.

Considere agora o retângulo da figura 12, e a figura abaixo:

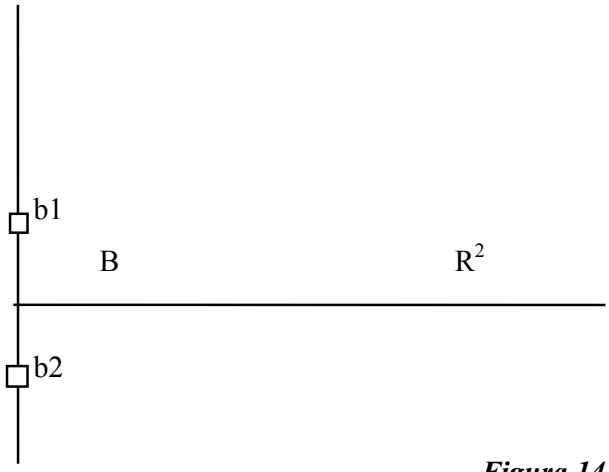

Figura 14- Objeto B

Quando realizamos a operação de subtração de A com B, utilizamos a equação (9).

Primeiro encontramos o complemento do objeto A, como dito acima. A adição ou(dilação), do complemento de A com o objeto B, seguido do complemento deste resulatdo resulta em :

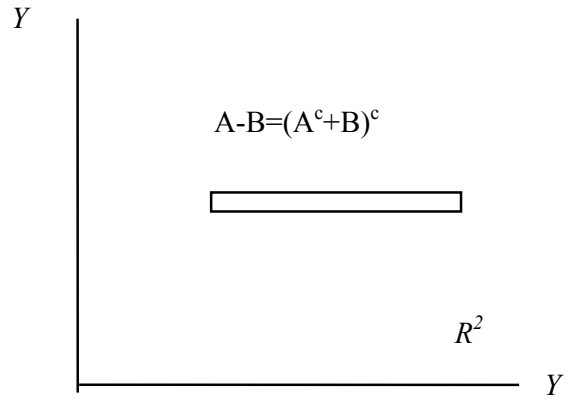

Figura 15- Objeto A subtraído do Objeto B

Podemos combinar as duas operações de dilação e erosão para criar duas novas operações, a abertura da imagem e o fechamento da imagem.

Na figura abaixo temos exemplos da aplicação da dilação e erosão sobre a imagem original da figura 6.a

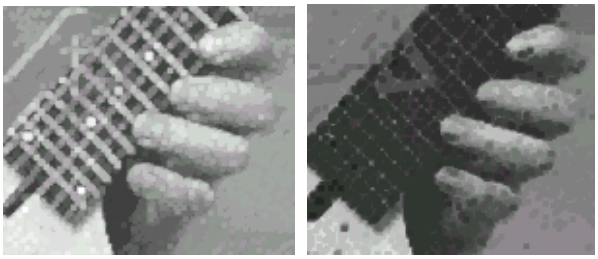

Figura 16- Dilação(à direita) e Erosão(esquerda) da foto original na figura 6.a

2 Estudo teórico das técnicas de segmentação utilizadas para separar microcalcificações.

2.1- Introdução

A segmentação é definida por GONZALES, R.C; WINTZ P em **Digital Image Processing,** [Gonzales 87] como sendo um processo pelo qual a imagem é dividida em suas partes constituintes. Para isso utilizam-se critérios de similaridade. No caso de imagens digitais de 8 bits o critério utilizado é o nível de cinza.

A segmentação é baseada em dois princípios: descontinuidade e similaridade. A **descontinuidade**, consiste em procurar por mudanças abruptas nos níveis de cinza. As técnicas mais utilizadas são as de detecção de bordas, linhas e pontos isolados. Estes tipos de técnicas nos possibilitam encontrar as formas dos objetos procurados. A **similaridade,** onde estão incluídas as técnicas de "thresholding" e crescimento de região, procura por pixels que tenham uma determinada característica em comum.

Neste relatório descreveremos as seguintes técnicas de segmentação: Imagem Diferença, Crescimento por região, Threshold e utilizando operadores morfológicos. O método de watershed será decrito no próximo relatório porque será implementado no segundo semestre.

2.2 Técnicas utilizando imagem diferença.

Esta técnica visa encontrar microcalcificações em mamografias, utilizando-se um sistema automático auxiliado por computador. Ela foi apresentada por CHAN H.P; DOI K; GALLOTRA S; VYBORNY CJ; JOKICH P.M [Chan 87] e seu objetivo é suprimir os ruídos da imagem e ao mesmo tempo eliminar os ruídos existentes na imagem.

Esta técnica consiste na aplicação de filtros baseados em máscaras de convolução. Inicialmente aplica-se uma máscara 3x3 (tipo passa- alta) sobre a imagem com o objetivo de realçar as microcalcificações obtendo a imagem 1. Em seguida é aplicado um filtro passa- baixa sobre a imagem original, com o objetivo de eliminar os ruídos. Este filtro é denominado filtro de contraste reverso. Ele consiste em combinar dois filtros conforme a fórmula abaixo:

$$F = (2\,F1 - 1)F2 \qquad (1)$$

Onde: F é o filtro de constraste reverso; F1 é um filtro de passa-baixa de suavização; F2 é um filtro para eliminar os ruídos de altas frequências obtendo a imagem2.

A imagem final é obtida subtraindo-se a imagem2 da imagem1.

Em CHAN H.P; DOI K; VYBORNY C.J; LAM K.L; SCHIMDT R.A em **Computer aided detection of microcalcifications in mammograms: methodology and preliminary clinical studys,** [Chan 88], é proposto um aperfeiçoamento desta técnica. Aplica-se aqui um novo filtro para eliminar os ruídos, denominado de "box rim". Este filtro elimina os ruídos de alta freqüência sem que haja a perda dos sinais procurados.

As máscaras utilizadas nesta técnica são mostradas nas figuras 1 e 2.

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0.75	0.75	0.75	0	0
0	0	0.75	1	0.75	0	0
0	0	0.75	0.75	0.75	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Figura 17- Filtro passa- alta convoluído da imagem original, para obter a imagem1.

0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	11	1	1	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0

Figura 18- Filtro "Box Rim" aplicado sobre a imagem original, para obter a imagem2.

Após as duas máscaras acima serem aplicadas sobre a imagem original, é subtraído a imagem2 da imagem1.

2.3 Técnicas utilizando crescimento de região.

Nesta técnica é preciso inicialmente definir quais são os pixels que formam uma região. Desta forma é possível visualizar uma região considerando todos os pixels agrupados que possuam o mesmo nível de cinza. Determinam-se em seguida as sementes. Estas consistem em pixels pertencentes à região.

Após ser escolhida uma semente, deve-se avaliar os pixels vizinhos para verificar se eles satisfazem uma certa condição de similaridade. Os pixels vizinhos que satisfaçam esta condição lhe são adicionados , caso contrário são descartados. O processo é repetido para cada novo pixel incluído a região. Este processo é interrompido conforme um critério de parada estabelecido inicialmente. No entanto há problemas para para escolher as sementes, para definir os critérios de parada e os critérios de similaridade.

Para encontrar estes critérios de similaridade [Fam 88] apresenta uma análise estatística das calcificações em mamogramas digitalizados. O valor médio de um nível de cinza destas calcificações encontradas na pesquisa foi de 162 (8 bits , 256 níveis de cinza). 94 % das calcificações possuiam pixels com valores entre 80 e 255.

Após determinar todos os pixels que pertencem a uma região, seleciona-se um ponto como semente, e então aplica-se o crescimento de região. Este processo continuará até que todos os pixels válidos tenham sido agregados.

Em FAM B.W; OLSONS H; WINTWR P.F; SCHOLZ F.J em **Algoritm for the detecttion of the fine clustered calcifications on the film mammomagrams,** [Fam 88] estabeleceu como limites para o tamanho das microcalcificações o intervalo de 0.1 a 1mm , e o contraste em relação a vizinha em torno de 1.0 a 1.59.

Em SHEN L; RANGAYYAN R.M; DESAUTELS J.E.L em **Application of shape analysis to mammographic calcifications** [Shen 94] apresenta uma variante deste método. Neste processo é escolhido um pixel manualmente e em seguida são testados os 4 pixels conectados ao semente para verificar as condições de similaridade.

Estas condições de similaridade são dadas pela fórmula abaixo:

$$(1+\pi)(Fmax + Fmin)/2 >= p(i,j) >= (1-\pi)(Fmax + Fmin)/2 \qquad (2)$$

E são dadas numa faixa de valores de níveis de cinza em torno dos valores máximo e mínimo da região. Esta faixa é dada pela equação (2) onde:

Fmax : valor máximo dos pixels da região atual.

Fmin: valor mínimo dos pixels da região atual.

π: uma valor entre 0 e 1 que indica a tolerância da variação do valor dos pixels vizinhos em relação a região.

O pixel em questão é integrado a região somente se ele satisfaz a condição de similaridade. Para cada novo ponto inserido são testados também seu 4 pixels conectados vizinhos. Isto é feito até que todos os pixels conectados que satisfaçam a equação (2) sejam encontrados. O contorno da região e a última camada encontrada.

Como pode ser observado na equação (2) a eficiência do método esta ligada ao valor de π. Na pesquisa o valor de π variou de 0.04 e 0.30.

Para aumentar a eficiência do método, em SHEN L; RANGAYYAN R.M; DESAUTELS J.E.L em **Detection and classification of mammographic calcifications.** [Shen 93] os autores calculam a média da imagem inteira, e então escolhem como sementes pixels com valores muito mais altos do que esta média. Em seguida eles aplicam o algoritmo com valores de tolerância entre 0.01 e 0.4 com um passo

calculado pelo inverso o pixel. Em cada tolerância testada, eles calculam o centro de gravidade e a compactividade da forma e o número de pixels.

Uma região é considerada uma microcalcificação somente se seu tamanho em pixels estiver na fixa de5 a 2500 pixels e o seu contraste maior que 0.2.

Não é preciso nesta técnica estabelecer os valores da tolerância e não é necessário escolher manualmente as sementes.

2.4 Técnicas de segmentação por Threshold

Nesta técnica um ou mais valores de nível de cinza são estabelecidos para separar os tipos de objetos da imagem.

De acordo com MASCARENHAS, N.D.A; VELASCO F.R.D em **Processo Digital de Imagens**. Editora Kapelusz[Mascarenhas 89] um tipo comum de threshold ocorre quando utilizamos esta técnica para obter a binarização da imagem. Escolhido um valor T, e em seguida todos os pixels com valores abaixo deste valor são considerados zero, e os pixels com valores acima de T são considerados com nível 255 (para imagem com 256 níveis de cinza).

Este tipo de técnica pode ser utilizado para encontrar microcalcificações em mamografias. Entretanto isto depende do valor T escolhido.

Em WOODS K.S; SOLKA J.L; PRIEBE, C.E; KEGELMEYER W.P em **Comparative evaluation of pattern recognition techniques for detectition of microcalcifications in Mammograms** [Woods 94] é utilizada um tipo de segmentação por Threshold. Neste método os autores inicialmente subtraiam de cada pixel o valor médio de uma região quadrada (15 pixels x 15 pixels) . A imagem resultante é chamada de imagem contraste local. Em seguida se o máximo desta imagem for 15 então utiliza-se T= 10. Caso contrário, T=5. O segundo passo é aplicar um algoritmo de crescimento de região para agrupar os pixels remanescentes em objetos. E em seguida um Threshold que elimine 97% dos objetos segmentados com menor contraste.

Outro exemplo de Threshold foi proposto por CHAN H.P; DOI K; GALLOTRA S; VYBORNY CJ; JOKICH P.M em **Image feature analysis and computer- aided diagnosis on digital radiology**[Chan 87] . Neste método são aplicados dois Thresholds sobre uma imagem segmentada conforme um algoritmo de imagem diferença. O primeiro Threshold (de toda a imagem) mantém os pixels de maior valor da imagem. O segundo Threshold (local) mantém o pixel se ele for maior que o pixel médio de sua área mais um múltiplo desvio padrão local. A região considerada pelos autores tinha um tamanho de 51 pixels x 51 pixels.

Em DAVIES D.H; DANCE D.R em **Automatic computer detection of clustered calcifications in digital mammograms** [Davies 90] os autores propõem um outro tipo de threshold para segmentação das microcalcificações . Neste método eles processam a imagem inicialmente para obter um fundo

uniforme no mamograma. Em seguida eles calculam o histograma da imagem para encontrar o mínimo deste histograma. Este mínimo é considerado então como o valor de Threshold da imagem. Em seguida os autores aplicam um filtro modal para zerar os pixels centrais de regiões quadradas, com valor modal igual a zero.

Em seguida eles dividem a imagem em sub- imagens quadradas. Cada histograma sofre a ação de um filtro mediana. Se o histograma da sub-imagem for bimodal, o valor deste é utilizado como valor de Threshold. Se o histograma da sub-imagem for unimodal, interpolam-se os thresholds das imagens vizinhas. Só haverá a segmentação do pixel se o valor dele for maior que o valor de threshold um determinado número de vezes (threshold de superposições). Os autores concluiram que os melhores tamanhos de sub imagem são de 32x32 pixels e o número de threshold de superposições igual a 3.

Em seguida [Davies 90] procura características para determinar se as microcalcificações são falsas ou não. Tais características são: área, nível médio de cinza, parâmetro W (razão da área para o quadrado da dimensão linear máxima), parâmetro S (de forma), e parâmetro de força de borda R (valor médio do gradiente de Robert dos pixels que compõe o perímetro do objeto.

O parâmetro S é dado pela fórmula abaixo:

$$S= P^2/ 4 \text{ Pi } A \qquad (3)$$

Onde P é o perímetro do objeto, e A é a área do objeto. Em seguida, após selecionar as microcalcificações, estas são agrupadas de modo que cada agrupamento tenha 3 ou mais calcificações e que a distância entre cada calcificação seja menor do que 0.5 mm. Os autores aplicam então um threshold com número de threshold de superposições iguais a 1, para segmentar as estruturas de fundo e comparam esta imagem com a primeira imagem processada . Onde há a correspondência entre as estruturas de fundo da segunda imagem e as calcificações segmentadas no primeiro processamento, estas estruturas são eliminadas.

2.5 Técnicas utilizando operadores morfológicos.

As operações morfológicas descritas no relatório anterior podem ser utilizadas para segmentar imagens.

Em DENGLER J; BEHREN S; DESAGA; J.F em **Segmentation of microcalcifications in mammograms** [Dengler 93] há uma proposta de segmentação de microcalcificações utilizando-se morfologia matemática. Um filtro Gaussiano Ga de largura a é aplicado sobre a imagem, e depois subtraído da imagem original. As partes negativas da nova imagem são consideradas como zero. Deste modo as estruturas de fundo são eliminadas.

Utilizando a fórmula abaixo, detectam –se sinais:

$$I2\ (x,y) = (w.Ga+ Ga-)*I1 \qquad (4)$$

onde:

I1: imagem após a eliminação das estruturas de fundo

I2: imagem após a detecção dos sinais.

Ga+:Núcleo negativo da largura esperada dos sinais

Ga-:Núcleo negativo com largura equivalente distância esperada dos sinais.

w: peso para o filtro de núcleo positivo.Define o nível de contraste necessário de um sinal em relação a sua vizinhança.

Após a aplicação deste filtro aplica-se um threshold sobre a imagem com valor de threshold igual a um múltiplo do desvio padrão global do ruído da imagem.

Porém segundo [Dengler 93] os filtros Guassianos distorcem a forma original das microcalcificações. Os autores utilizam então uma composição de operações morfológicas, compostas de uma erosão seguida de uma dilação. Esta operação é chamada de abertura da imagem, dada pela fórmula abaixo:

$$(M\ opening\ I) = M\ Dil\ (M\ Ero\ I) \qquad (5)$$

onde:

I: imagem original

M: elemento estruturante. Um disco de diâmetro igual a 13 pixels.

Dil: operação de dilação

Ero:operação de erosão.

O objetivo de [Dengler 93] é de suprimir as microcalcificações, preservando as estrtuturas de fundo. A imagem resultante da equação 5 é subraída da imagem original, de acordo com a fórmula (6).

$$Ir = I – (M\ Opening\ B) \qquad (6)$$

Onde:

Ir: Imagem final

I: Imagem Original

M: elemento estruturante

Sobre esta imagem é aplicado o mesmo tipo de threshold da técnica inicial.

Esta combinação de operações, entre threshold é resíduo da abertura morfológica é conhecido como transformação Top- Hat. Apesar desta operação preservar melhor os sinais que representam as microcalcificações, ela é bastante sensível a ruídos

Deste modo uma solução encontrada pelos autores foi de combinar os resultados dos dois métodos, com filtros gaussianos e operações morfológicas, utilizando a equação abaixo:

(M1,M2) Cthickening X Y = Y interseção (X união ((M1 Ero X) inter(M1 Ero X^c))) (7)

onde:
M1, M2: é o par de elementos estruturais da operação.
X: É o resultado do processo de segmentação utilizando-se filtros gaussianos.
Y:É o resultado da transformação Top Hat.

.

3 Considerações teóricas sobre as técnica de Watershed

3.1Introdução

Quando aplica-se um determinado algoritmo de segmentação numa imagem mamográfica, tem-se como objetivo principal destacar as estruturas que são conhecidas como microcalcificações. Entretanto é possível aglutinar, ou juntar duas regiões que deveriam ser consideradas em separado.

Como exemplo, muitas das técnicas de segmentação fazem uso da operação morfológica de dilação. Em certos casos este "crescimento" ou aumento de pixels sobre uma região, pode fazer com que tenhamos duas microcalcificações sobrepostas, dificultando a visão correta do objeto.

O algoritmo de Watershed , que significa "divisor de águas" faz esta diferenciação, e permite determinar se dois objetos estão sobrepostos e separá-los.

De acordo com [Parker 94], este algoritmo funciona para um conjunto de regiões convexas conectadas. A idéia básica é separar as estruturas conectadas.

3.2 Descrição do algoritmo.

Considere a região abaixo onde aparecem duas estruturas superpostas.

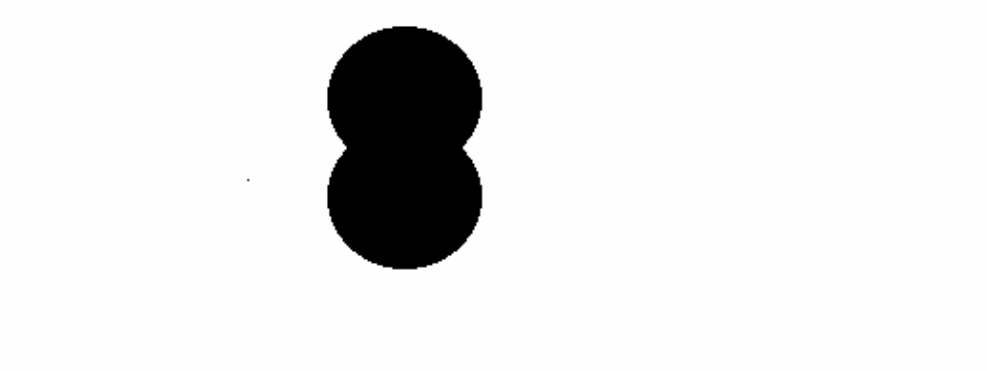

figura 19. Exemplo de duas estruturas sobrepostas.

Se a distância de cada célula a um pixel no fundo da imagem (a menor distância) é determinada, é possível determinar uma figura tri- dimensinal, onde a distância de cada pixel ao fundo da imagem, seria o eixo que nos dá a altura da figura. Quando procede-se desta maneira obtém-se duas regiões que se assemelham a duas montanhas com dois picos distintos ligadas por um cume. Este cume no meio dos dois picos representa a máxima distância dos picos. O cume será localizado , e alguns dos pixels serão apagados, para separa as duas regiões, obtendo-se assim duas regiões.

Inicialmente calcula-se a transformada da distância de cada pixel até o fundo da imagem. Isto é feito em dois passos através da imagem.

No primeiro passo, começando do pixel superior a direita (0,0), e movendo-se nas linhas da esquerda para a direita, os pixels objeto são identificados e recebem um valor numérico igual a distância de cada pixel a um pixel na fronteira da imagem.

O segundo passo, começa do pixel mais inferior e a direita. Move-se então da direita para a esquerda através das linhas e de baixo para cima na direção das colunas. É calculado a distância de cada pixel objeto a uma borda inferior ou a direita. Se esta nova distância for menor que o valor existente, ele é armazenado como um novo pixel. Quando completo este procedimento, cada pixel representa a distância dos pixels ao fundo da imagem.

Notam-se dois picos, cada um tendo um valor 8, e representando o centro da célula. Estes são então considerados como os pixels semente para uma dilação. Se camadas de pixels são adicionadas a estes simultaneamente , as duas regiões aumentadas , encontrar-se-ão no cume. Deve-se entretanto tomar

cuidado para que duas regiões distintas sejão conectadas no processso. Como isto é feito é descrito posteriormente.

Inicialmente, todos os pixels com valor máximo são marcados, e todos os pixels vizinhos com valor abaixo do máximo de um dígito ou menos são também marcados se, e somente se, eles não conectam duas regiões.

A condição de não conexão é testada usando máscaras , ou templates Por exemplo o pixel central em cada um dos casos da figura 2 não seria marcado pois conectaria duas regiões.

*	*	o	*	o	*	*	o	o
o	.	o	*	.	*	o	.	o
o	*	*	o	o	o	*	o	*

Figura 20

Desde que um pixel tem 256 possíveis máscaras, para testar a regra de "não conectar novas regiões" é preciso utilizar uma "look up table" , LUT. Esta é indexada por um inteiro no intervalo de 0 - 255. O índice para a tabela é calculado atribuindo-se uma posição de bit para cada um dos 8 pixels vizinhos formando um inteiro de 8 bits da região 3x3. O pixel central , ou alvo, não é usado na construção do índice.

Aplica-se a máscara abaixo para este processo:

128	64	32
16	*	8
4	2	1

A matriz 3x3 dos pixels é aplicada a matriz 3x3 de pesos comos seria feito
para uma máscara de convolução. Neste caso, desde que o peso é uma potência única de
2, o resultado pode ser transformado em um número de 8 bits (inteiro) e usado como índice para uma tabela. Assim, por exemplo, considerando a região 3x3 abaixo:

1	1	0
0	*	0
0	1	1

O índice da LUT será:

128 + 64 +0
+0 +0
+0 +2 +1= 128+64+0+0+0+0+2+1=194= (11000011), representação binária.

Esta condição resultaria em 2 regiões conectadas .A entrada da LUT deverá ser 0, ou falsa, significando que "não adicione um pixel aqui". Desde que existem somente 256 possíveis padrões , é possível codifica-los em um curto período de tempo

O algoritmo pode ser resumido como abaixo:

➔ Calcular a transformada da distância
➔ Localizar os valores mais altos da matriz D proveniente do passo anterior e que são aqueles objetos que estão mais distantes do fundo da imagem. Estes pixels terão valor K.
➔ Agora começando nos picos de D, marca-se aqueles pixels que :
i) São adjacentes a um pixel com valor K
ii) tem valor K-1
iii) não conectar duas regiões marcadas mas não conectadas.

O passo iii) é implementado usando a tabela de LUT como discutido. Agora o valor de K é decrementado em 1 e o processo se repete até que K torna-se 0.

II Parte Prática

1 Materiais e métodos

1.1 Introdução

Os trabalhos realizados neste item consistiram em adaptar o algoritmo de Watershed descrito em [Parker 94] para linguagem **Delphi**, testar este algoritmo e verificar sua eficiência aplicando o algoritmo em regiões de interesse.

O método de segmentação de Watershed foi escolhido pois não havia sido implementado neste laboratório e poderia ser uma boa solução para os problemas de segmentação encontrados nos

processamento de microcalcificações. Entretanto foi preciso adaptar o código C referido em [Parker 94] para linguagem **Delphi**, para depois implementar o código adaptado para Delphi e testa-lo em regiões já previamente selecionadas.

A linguagem **Delphi** foi escolhida pois apresenta maior facilidade para manipular imagens em relação à outras linguagens. Isto permite que o tempo hábil seja utilizado para implementação do algoritmo em si, ao invés de perdermos grande parte do tempo "alocando" memória para podermos utilizar as imagens.

Para testar este algoritmo, duas maneiras foram utilizadas: aplicação à imagens simuladas, criadas no **paint brush** do **Windows 95** e também aplicação em regiões de interesse, ou seja, pequenos trechos de mamografias.

Foi portanto verificado o desempenho deste algoritmo para separar duas regiões distintas, neste caso específico duas microcalcificações, que tenham sido aglutinadas por algum algoritmo de segmentação.

1.2 Métodos sobre a manipulação das imagens

O Delphi proporciona uma fácil manipulação de imagens de **bitmaps**. Tudo que é necessário é adicionar um componente **Timage.**

Através dos métodos **LoadFromFile** e **SaveToFile** pode-se ler e salvar imagens. Uma vez utilizado **LoadFromFile** para abrir uma imagem no componente **Timage**, através de um simples "loop for", é possível enviar seu conteúdo, para uma matriz, para em seguida trabalhar-se com ela.

Para isso foi utilizado o método **GetRvalue(image1.Canvas.Pixels[i,j]),** ele nos permite acessar os elementos da imagem, através dos índices i e j. Uma vez processada, esta matriz, para enviar seu conteúdo para a tela utilizamos o seguinte procedimento :

Image2.Canvas.Pixels[i,j] := RGB (ValorPixel, ValorPixel, ValorPixel);

Image2 é um objeto **Timage** onde deve-se mostrar a imagem processada. Assim, e utilizando a propriedade **image.canvas.pixels[i,j]** para receber o valor de pixel a armazenar na posição [i,j] da imagem, é possível mostrar o resultado da matriz processada pelo método de watershed.

1.3Listagem do Programa.

Abaixo apresentamos a listagem do programa implementado.

```
unit Unit1;
{*********************************************************************
Adaptação e implementação do Algoritmo de Watershed descrito no livro
Pratical Computer Vision using C, de J.R Parker, nas páginas 305 até 308.


Este programa aplica o algoritimo de Watershed numa imagem de bit maps
(ou imagens com extensão bmp), com o objetivo de segmentar esta imagem,
e separar regiões aglutinadas.
Em [] o algoritmo está descrito em linguagem C. Optamos no entanto pelo
Delphi pela facilidade de manipulação de imagens, além do fato do aluno
estar mais familiarizado com a linguagem Pascal.
*********************************************************************}


interface

uses
```

```
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, ExtCtrls, StdCtrls;

type
  TMatImagem = array [0..500,0..500] of Longint;
  {o tipo TMatImagem é uma matriz de dimensão de 500x500
  entretanto a dimensão da matriz tem a dimensão da imagem}
  Tvetor = array [0..255] of Longint;
  {O tipo Tvetor é um vetor de dimensão 500 utilizado no procesimento Lut}

  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    Image1: TImage;
    File1: TMenuItem;
    Load1: TMenuItem;
    Save1: TMenuItem;
    Exit1: TMenuItem;
    Algoritmo1: TMenuItem;
    Watershed1: TMenuItem;
    Image2: TImage;
    Config1: TMenuItem;
    val1: TMenuItem;
    val2: TMenuItem;
    procedure Exit1Click(Sender: TObject);
    procedure Load1Click(Sender: TObject);
    procedure Save1Click(Sender: TObject);
    procedure Watershed1Click(Sender: TObject);

  private
  {Private declarations }

  {variáveis globais utilizadas em todos os procedimentos}

    A,B:TMatImagem;
    lut:TVetor;
    nr,nc: Longint;

  Procedure waters(var val:longint);
  {*************************************************************************
  procedimento de watershed, este segmenta regiões, e tenta desconectar
  regiões convexas.
  *************************************************************************}
  Procedure lutint;
  {*************************************************************************
  Este procedimento inicializa um vetor que é utilizado na Look Up Table,
  *************************************************************************}
  Procedure copy;
  { ************************************************************************
  Este procedimento copia uma matriz A para uma outra matriz B
  *************************************************************************}
  Procedure remarca(var v1:Longint; var v2 :Longint);
  {*************************************************************************
  Este procedimento remarca com valor V2qualquer elemento de uma matriz
  A caso este elemento seja igual ao valor v1
  *************************************************************************}
  Function  dt(var val: Longint): Longint;
  { ************************************************************************
  Esta função calcula a transformada da distância de uma Imagem, representada
  como uma matriz.Os pixels com valores val são trocados  pela sua distância
  ao fundo da imagem.
  *************************************************************************}
  Function  wslev(var lev:Longint):Longint;
```

```
{*********************************************************************
função que testa a condição de conectar ou não as regiões que está
sendo crescidas.
*********************************************************************}

public
   { Public declarations }
end;


var
  Form1: TForm1;

implementation

{$R *.DFM}

{Inicio do código do procedimento de Watershed}
{*********************************************************************}

Procedure Tform1.waters(var val:longint);
var

i,j,k,n:integer;
max:Longint;

begin
lutint;
max:= dt(val);
  for i:=0 to nr-1 do
  for j:=0 to nc-1 do
  if A[i,j] = max then A[i,j]:= 255;
  max:= max-1;
  copy ;
  while max > 0 do
   begin
    while n <> 0  do
     begin
      n:=wslev(max);
     end;
    max:= max-1;
   end;
  for i:=0 to nr-1 do
  for j:=0 to nc-1 do
   begin
     if (A[i,j]<>0)and(A[i,j]<>255) then A[i,j]:= 0;
   end;
end;
{*********************************************************************}
{fim do algoritmo de watershed}


{inicio do procedimento Lut}
{*********************************************************************}
procedure TForm1.lutint;
begin
{0-40}
lut[0]:=0;lut[1]:=1;lut[2]:=1;lut[3]:=1;lut[4]:=1;lut[5]:=0;lut[6]:=1;
lut[7]:=1;lut[8]:=1;lut[9]:=1;lut[10]:=1;lut[11]:=1;lut[12]:=0;lut[13]:=0;
lut[14]:=1;lut[15]:=1;lut[16]:=1;lut[17]:=0;lut[18]:=1;lut[19]:=1;lut[20]:=1;
lut[21]:=0;lut[22]:=1;lut[23]:=1;lut[24]:=0;lut[25]:=0;lut[26]:=1;lut[27]:=1;
lut[28]:=0; lut[29]:=0;lut[30]:=1;lut[31]:=1;lut[32]:=1;lut[33]:=0;lut[34]:=0;
lut[35]:=0;lut[36]:=0;lut[37]:=0;lut[38]:=0;lut[39]:=0;lut[40]:=1;
{41-80}
```

```
lut[41]:=1;lut[42]:=1;lut[43]:=1;lut[44]:=0;lut[45]:=0;lut[46]:=1;
lut[47]:=1;lut[48]:=0;lut[49]:=0;lut[50]:=0;lut[51]:=0;lut[52]:=0;
lut[53]:=0;lut[54]:=0;lut[55]:=0;lut[56]:=0;lut[57]:=0;lut[58]:=1;
lut[59]:=1;lut[60]:=0;lut[61]:=0;lut[62]:=1;lut[63]:=1;lut[64]:=1;
lut[65]:=0;lut[66]:=0;lut[67]:=0;lut[68]:=0;lut[69]:=0;lut[70]:=0;
lut[71]:=0;lut[72]:=1;lut[73]:=1;lut[74]:=0;lut[75]:=1;lut[76]:=0;
lut[77]:=0;lut[78]:=1;lut[79]:=1;lut[80]:=1;
{81-120}
lut[81]:=0;lut[82]:=1;lut[83]:=1;lut[84]:=1;lut[85]:=0;lut[86]:=1;
lut[87]:=1;lut[88]:=1;lut[89]:=1;lut[90]:=1;lut[91]:=1;lut[92]:=1;
lut[93]:=1;lut[94]:=1;lut[95]:=1;lut[96]:=1;lut[97]:=0;lut[98]:=0;
lut[99]:=0;lut[100]:=0;lut[101]:=0;lut[102]:=0;lut[103]:=1;lut[104]:=1;
lut[105]:=1;lut[106]:=1;lut[107]:=1;lut[108]:=0;lut[109]:=0;lut[110]:=1;
lut[111]:=1;lut[112]:=1;lut[113]:=0;lut[114]:=1;lut[115]:=1;lut[116]:=1;
lut[117]:=0;lut[118]:=1;lut[119]:=1;lut[120]:=1;
{121-200}
lut[121]:=1;lut[122]:=1;lut[123]:=1;lut[124]:=1;lut[125]:=1;lut[126]:=1;
lut[127]:=1;lut[128]:=1;lut[129]:=0;lut[130]:=0;lut[131]:=0;lut[132]:=0;
lut[133]:=0;lut[134]:=0;lut[135]:=0;lut[136]:=0;lut[137]:=0;lut[138]:=0;
lut[139]:=0;lut[140]:=0;lut[141]:=0;lut[142]:=0;lut[143]:=0;lut[144]:=1;
lut[145]:=0;lut[146]:=1;lut[147]:=1;lut[148]:=1;lut[149]:=0;lut[150]:=1;
lut[151]:=1;lut[152]:=0;lut[153]:=0;lut[154]:=1;lut[155]:=1;lut[156]:=0;
lut[157]:=0;lut[158]:=1;lut[159]:=1;lut[160]:=0;lut[161]:=0;lut[162]:=0;
lut[163]:=0;lut[164]:=0;lut[165]:=0;lut[166]:=0;lut[167]:=0;lut[168]:=0;
lut[169]:=0;lut[170]:=0;lut[171]:=0;lut[172]:=0;lut[173]:=0;lut[174]:=0;
lut[175]:=0;lut[176]:=0;lut[177]:=0;lut[178]:=0;lut[179]:=0;lut[180]:=0;
lut[181]:=1;lut[182]:=1;lut[183]:=0;lut[184]:=0;lut[185]:=1;lut[186]:=1;
lut[187]:=1;lut[188]:=0;lut[189]:=0;lut[190]:=0;lut[191]:=1;lut[192]:=1;
lut[193]:=0;lut[194]:=0;lut[195]:=0;lut[196]:=0;lut[197]:=0;lut[198]:=0;
lut[199]:=0;lut[200]:=1;
{201-255}
lut[201]:=1;lut[202]:=1;lut[203]:=1;lut[204]:=0;lut[205]:=0;lut[206]:=1;
lut[207]:=1;lut[208]:=1;lut[209]:=0;lut[210]:=1;lut[211]:=1;lut[212]:=1;
lut[213]:=0;lut[214]:=1;lut[215]:=1;lut[216]:=1;lut[217]:=1;lut[218]:=1;
lut[219]:=1;lut[220]:=1;lut[221]:=1;lut[222]:=1;lut[223]:=1;lut[224]:=1;
lut[225]:=0;lut[226]:=0;lut[227]:=0;lut[228]:=0;lut[229]:=0;lut[230]:=0;
lut[231]:=0;lut[232]:=1;lut[233]:=1;lut[234]:=1;lut[235]:=1;lut[236]:=0;
lut[237]:=0;lut[238]:=1;lut[239]:=1;lut[240]:=1;lut[241]:=0;lut[242]:=1;
lut[243]:=1;lut[244]:=1;lut[245]:=0;lut[246]:=1;lut[247]:=1;lut[248]:=1;
lut[249]:=1;lut[250]:=1;lut[251]:=1;lut[252]:=1;lut[253]:=1;lut[254]:=1;
lut[255]:=1;
end;
{*************************************************************************}
{fim do procedimento lut}



{Inicio do procedimento copy}
{*************************************************************************}
procedure Tform1.copy;
var
i,j:integer;
begin
 for i:=0 to nr-1 do
 for j:=0 to nc-1 do
  begin
   B[i,j]:= A[i,j];
  end;
end;
{*************************************************************************}
{fim do procedimento copy}
```

```
{inicio do procedimento remarca}
{*************************************************************************}
procedure tform1.remarca(var v1:Longint; var v2 :Longint);
var
i,j: Longint;
begin
for i:=0 to nr-1 do
for j:=0 to nc-1 do
 if (A[i,j]=V1) then A[i,j]:= v2;
end;
{*************************************************************************}
{fim do procedimento de remarcar}



{funcao que calcula a transformada da distancia}
{*************************************************************************}
function tform1.dt(var val: Longint): Longint;
     var
      i,j,k,n,m,ii,jj,v1,v2:Longint;
       begin
         v1:=val;
         v2:=254;
        remarca(v1,v2);
        v1:=255;
        v2:=0;
        remarca(v1,v2);
        n:=0;
        m:=1;
        while m<>0 do
         begin
          m:= 0;
           for i:= 0 to nr-1 do
            for j:= 0 to nc-1 do
             for ii:= -1 to 1 do
              begin
               if A[i,j]<>254
                then continue;
               if ((i + ii)<0)or((i + ii)>=nr)then continue;
               for jj:= -1 to 1 do
                begin
                 if (ii<>0)and(jj<>0) then continue;
                 if ((j+jj)<0)or((j+jj)>=nc) then continue;
                 if A[i+ii,j+jj]= n then
                  begin
                   A[i,j]:= n+1;
                   m:=1;
                  end;
                end;
              end;
             n:=n+1;
         end;
         dt:= n-1;
       end;


{*************************************************************************}
{fim da função que calcula a transformada da distância}



{Inicio da funçao Wlev}
```

```
{*************************************************************************}
function tform1.wslev(var lev:Longint):Longint;
var
ind,i,j,k,n,m: Longint;
mask : array [0..20,0..20] of Longint;
begin
mask[2,2]:= 01;
mask[2,1]:= 02;
mask[1,2]:= 04;
mask[1,2]:= 010;
mask[1,0]:= 020;
mask[0,2]:= 040;
mask[0,1]:= 0100;
mask[0,0]:= 0200;
k:=0; ind:=0;
for i:=1 to (nr-2) do
for j:=1 to (nc-2) do
 begin
  if A[i,j] <> lev then continue;
  ind:= 0;
  for n:=-1 to 1 do
   begin
    for m:=-1 to 1 do
     begin
      if (n=0)and(m=0)then continue;
      if  A[i+n,j+m] = 255 then
      ind := 1;
     end;
   end;

 if ind=0 then continue;
 ind:=0;

 for n:=-1 to 1 do
  for m:=-1 to 1 do
   begin
    if (n=0)and(m=0) then continue;
    if B[i+n,j+m]= 255 then ind:=ind + mask[n+1,m+1];
   end;

  if lut[ind]<>0 then
   begin
    B[i,j]:= 255;
    k:=1;
   end;
 end;

if k<>0 then copy;
wslev:=k;
end;
{*************************************************************************}
{fim do procedimento de wlev}



{Procedimentos ligados aos objetos visuais que compõe
o programa}
{*************************************************************************}

procedure TForm1.Exit1Click(Sender: TObject);
begin
close;
end;
```

```
{carrega uma imagem para o objeto image1}
procedure TForm1.Load1Click(Sender: TObject);
begin
with OpenDialog1 Do
    if execute Then
      Begin
      with Image1.picture do
       begin
       LoadFromfile(Filename);
       end;
       Caption := ExtractFilename(Filename);
      end;
end;

{salva uma imagem ligada ao objeto image2}
procedure TForm1.Save1Click(Sender: TObject);
begin
with Savedialog1 do
    if execute then
      begin
          with image2.picture do
          begin
           Savetofile(filename);
          end;
      end;
end;

{fim dos procedimentos ligados a parte visual}
{************************************************************************}



{inicio do subprograma que chama o procedimento de watershed
além de transformar as imagens , arquivos bmp em matrizes}
{************************************************************************}
procedure TForm1.Watershed1Click(Sender: TObject);

var
valorpixel,i,j,val: Longint;

begin{inicio do subprograma principal}

{Podemos acessar a dimensão da matriz através das propriedades
abaixo, obtendo assim sua dimensão heightxwidht}
{Deste modo trabalhamos com matrizes, facilitando
tanto o cálculo quanto o entendimento.
Uma vez lida a imagem podemos facilmente
armazenar o seu conteúdo numa matriz para podermos trabalhar
com esta}

 nr:= image1.picture.bitmap.height;
 nc:= image1.picture.bitmap.width;

 {copia imagem para uma matriz A}
  for i:=0 to nr-1 do
   for j:=0 to nc-1 do
    begin
     A[i,j]:=GetRvalue(image1.Canvas.Pixels[i,j]);
    end;
    {fim do procedimento que copia imagem para
    a matriz A}

 val:=10;
```

```
Screen.Cursor:= crHourGlass;
waters(val);{chama procediemto de watershed para
ser aplicado sobre a matriz }

for i:=0 to nr-1 do
 for j:=0 to nc-1 do
   begin
    valorpixel:= A[i,j];
    Image2.Canvas.Pixels[i,j] := RGB (ValorPixel, ValorPixel, ValorPixel);
   end; {imagem final é mostrada}

 Screen.Cursor:= crDefault;


end;
{fim do subprograma que le, aplica o watershed e mostra a imagem}
{************************************************************************}


{fim do programa principal}

{************************************************************************}
end.
```

2 Resultados.

A imagem seguinte é uma região de interesse extraída de uma mamografia, digitalizada e selecionada previamente.

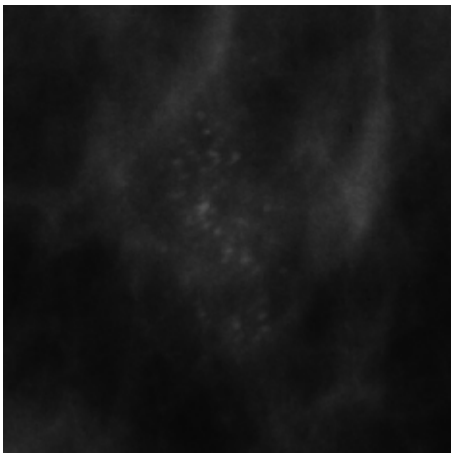

Figura 21- Região de interesse contendo microcalcificações

Após aplicar o algoritmo de Watershed , é obtida a imagem contida na figura 4:

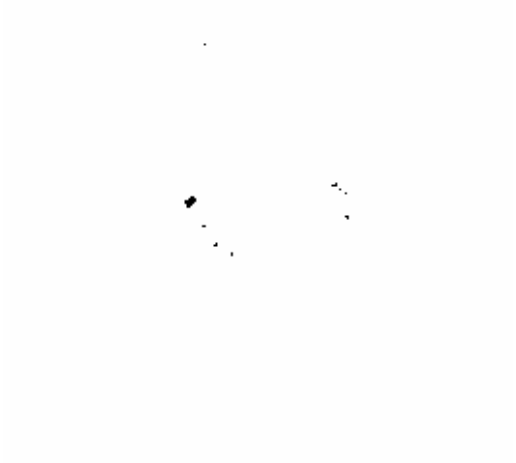

Figura 22-Resultado da segmentação "watershed"

Pode-se observar que as microcalcificações são destacadas, após a aplicação deste algoritmo, inclusive com a eliminação de parte do ruído de fundo da imagem.

Utilizando outra região de interesse de uma mamografia pré- selecionada , mostrada abaixo:

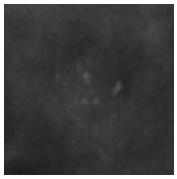

Figura 23- Região de interesse contendo as microcalcificações.

Após a aplicação do algoritmo é obtida a imagem mostrada na figura 6:

Figura 24- Resultado da segmentação "watershed"

As microcalcificações são destacadas, no entanto, bastante ruído de fundo também foi selecionado.

Foi editado uma imagem de **bit maps** de modo a simular uma região contendo objetos de formas semelhantes, que poderiam estar aglutinados. O algoritmo implementado deveria separar estas regiões.

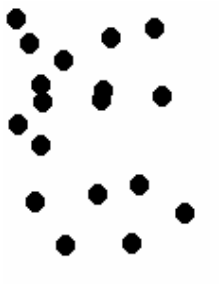

Figura 25- Imagem de Bit maps

Ao aplicar o algoritmo na imagem acima, foi obtido o seguinte resultado, apresentado na figura 8.

Figura 26- Resultado do Algoritmo de Watershed sobre a imagem da figura 7.

Pode-se verificar na figura 8 que, apesar de perder a informações sobre a sua forma, os objetos sobrepostos foram separados. Deste modo se o objetivo fosse contar o número de objetos na imagem, o procedimento de Watershed seria de grande auxilio.

Foi editado uma outra imagem de bit maps (desta vez utilizando uma forma diferente para os objetos a serem separados) apresentada na figura baixo:

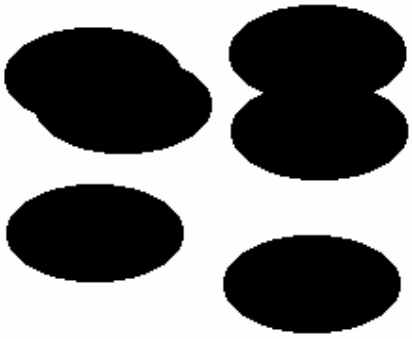

Figura 27- Imagem de Bit maps

O resultado da aplicação do algoritmo de Watershed esta apresentado na figura 10.

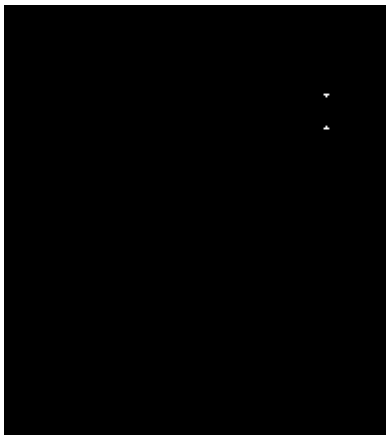

Figura 28- Imagem resultante da aplicação do Watershed sobre a imagem da figura 9.

Na figura 10 observam-se dois pares de objetos sobrepostos. Na imagem processada apenas um destes pares foi separado. O outro par, e outros objetos não conectados, foram perdidos . Entretanto o par que pode ser mais facilmente considerado como sendo dois objetos conectados, foi separado. O outro par não separado, foi interpretado como sendo uma estrutura única. Pode-se perceber que a forma do objeto influi na atuação do algoritmo. Objetos como discos são reconhecidos enquanto que objetos de forma cilíndrica não o são, se sua forma é estreita.

III Conclusões.

Quando aplicado em regiões de interesse o algoritmo apresenta uma boa eficiência, no processo de segmentar as microcalcificações. Entretanto o teste em mamogramas completos deveria ser tentado futuramente. O algoritmo quando usado para separar objetos convexos conectados, apresenta uma eficiência variável como mostrado nas figuras 8 e 10 já que nesta última, houve perda do sinal original.

Entretanto, este algoritmo foi desenvolvido para segmentar estruturas de dimensões maiores do que as microcalcificações. Estas apresentam um tamanho bastante reduzido, o que compromete certamente, a eficiência do algoritmo. Faz-se necessário um aperfeiçoamento deste algoritmo futuramente, de modo a adaptar seus procedimentos às características das microcalcificações.

Além disso, é preciso verificar a atuação do Watershed em conjunto com outras técnicas de segmentação, abrindo um campo novo de possibilidades.

Quanto às técnicas de digitalização, ainda é preciso buscar os parâmetros ótimos para este processo Mais ainda: devemos definir qual critério é melhor para digitalzarmos uma mamografia, o visual ou àqueles parâmetros que possibilitam um melhor resultado no processamento. Concluímos no entanto, que a melhor "ferramenta" para a digitalização, é utilizar os auto ajuste que acompanha o sistema do digitalizador. Isto é claro levando em conta o critério visual.

As técnicas de pré processamento (filtros) auxiliam sobremaneira na inspeção visual das microcalcificações em mamogramas digitais. Entretanto para efeito de processamento elas não são eficientes. Quanto às técnicas de segmentação elas requerem ainda microcalcificações bem específicas, para conseguir resultados expressivos.

IV Referências bibliográficas.

[Chan 87]

CHAN H.P; DOI K; GALLOTRA S; VYBORNY CJ; JOKICH P.M ;**Image feature analysis and computer- aided diagnosis on digital radiology**. Medical Physics vol.14, Jul/Ago 1987.

[Chan 88]

CHAN H.P; DOI K; VYBORNY C.J; LAM K.L; SCHIMDT R.A; **Computer aided detection of microcalcifications in mammograms: methodology and preliminary clinical studys.** Investigative Radiology. V.23, n.9, p 664-671. Set 1988.

[Davies 90]

DAVIES D.H; DANCE D.R;**Automatic computer detection of clustered calcifications in digital mammograms**. Physics and medical biology, v.15, n.8, p 1111-1118, 1990.

[Dengler 93]

DENGLER J; BEHREN S; DESAGA; J.F; **Segmentation of microcalcifications in mammograms**. IEEE trans.on medical imaging. v 12, n. 4, Dez 1993.

[Fam 88]

FAM B.W; OLSONS H; WINTWR P.F; SCHOLZ F.J**;Algoritm for the detecttion of the fine clustered calcifications on the film mammomagrams.** Radiology,v. 169., n.2, p 33-337, Nov/1988.

[Gonzales 87]

GONZALES, R.C; WINTZ P; Digital **Image Processing. Addison-** Wesley Publishing Company, United States Of America, 1987.

[Mascarenhas 89]

MASCARENHAS, N.D.A; VELASCO F.R.D; **Processo Digital de Imagens**. Editora Kapelusz S.A, Buenos Aires- Argentina, 1989.

[Myler 1993]

MYLER H.R; WEEKS A R; **Computer Imaging Recipes in C**; P T R Prentice Hall 1993.

[Ricardo 96]

VILLELA, R.L; **Otimização da segmentação de microcalcificações em imagens mamográficas;** Dissertação de mestrado; São Carlos 1996.

[Russ 95]

RUSS J.C; **The Image Processing Handbook**; CRC Press, Inc 1995.

[Shen 93]

SHEN L; RANGAYYAN R.M; DESAUTELS J.E.L; **Detection and classification of mammographic calcifications.** Interrnational Journal of Pattern Recognition and Artificial Intelligence, v.7(6), p.1403-1416, 1993.

[Shen 94]

SHEN L; RANGAYYAN R.M; DESAUTELS J.E.L. **Application of shape analysis to mammographic calcifications**. IEEE transactions on Medical Imaging, v 13(2), p 263-274, 1994.

[Villela 96]

VILLELA, R.L;SLAETS, A F. F. ; P.M.A; SCHIABEL, H; FLORIAN, R.V; FERRARI, R.J; **Estudo Comparativo das Técnicas de Segmentação de Microcalcificações em Imagens Mamográficas;**

[Woods 94]

WOODS K.S; SOLKA J.L; PRIEBE, C.E; KEGELMEYER **W.P;Comparative evaluation of pattern recognition techniques for detectition of microcalcifications in Mammograms;** State of Art in Digital Mammographic Analysis, Word Scientific- New Jersey, p. 841-852; 1994.

[Carroll 88]

Carrol, D.W; Programação em Turbo Pascal. Makron Books do Brasil Editora Ltda, 1988.

[Holzner 95]

Holzner, S; C Programação. Editora Campus 1993.

[Mueller 96]

Mueller, Jonh; Guia para o Delphi 2. Makron Books do Brasil Editora Ltda, 1997.

[Parker 94]

Parker, J.R; Pratical Computer Vision Using C. Jonh Wiley & Sons, Inc 1994.

[Ruben 95]

Rubenking, N.J; Programação em Delphi para Leigos. Berkley Brasil Editora, 1995.