UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
CURSO DE MESTRADO EM ENGENHARIA ELÉTRICA

FILTROS ESPACIAIS: IMPLEMENTAÇÃO, ESTUDO COMPARATIVO E APLICAÇÃO EM SENSORIAMENTO REMOTO

MARCELO ALVES DE BARROS

CAMPINA GRANDE
JULHO - 1990

MARCELO ALVES DE BARROS

FILTROS ESPACIAIS: IMPLEMENTAÇÃO, ESTUDO COMPARATIVO E APLICAÇÃO EM SENSORIAMENTO REMOTO

Dissertação apresentada ao Curso de MESTRADO EM ENGENHARIA ELÉTRICA, da Universidade Federal da Paraíba, em cumprimento às exigências para obtenção do Grau de Mestre.

ÁREA DE CONCENTRAÇÃO: PROCESSAMENTO DA INFORMAÇÃO

Arnaldo de Albuquerque Araújo, DSc.
(Orientador)

CAMPINA GRANDE
JULHO - 1990

B277f Barros, Marcelo Alves de.
Filtros espaciais : implementação, estudo comparativo e aplicação em sensoriamento remoto / Marcelo Alves de Barros. - Campina Grande, 1990.
156 f.

Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, 1990.
"Orientação : Prof. Dr. Arnaldo de Albuquerque Araújo".
Referências.

1. Processamento Digital de Imagens. 2. Filtros Espaciais - Software. 3. Técnicas de Suavização Espacial. 4. Dissertação - Engenharia Elétrica. I. Araújo, Arnaldo de Albuquerque. II. Universidade Federal da Paraíba - Campina Grande (PB). III. Título

CDU 621.3:004.932(043)

FILTROS ESPACIAIS: IMPLEMENTAÇÃO, ESTUDO COMPARATIVO
E APLICAÇÃO EM SENSORIAMENTO REMOTO

MARCELO ALVES DE BARROS

DISSERTAÇÃO APROVADA EM 10.07.90

ARNALDO DE ALBUQUERQUE ARAÚJO, D.Sc., UFMG
Orientador

JOÃO MARQUES DE CARVALHO, Ph.D, UFPB
Componente da Banca

PAULO ROBERTO CAMPOS DE ARAÚJO, Mestre, UFPB
Componente da Banca

ASCENDINO FLÁVIO DIAS E SILVA, Dr.Ing., UFPE
Componente da Banca

CAMPINA GRANDE - PB
JULHO - 1990

AGRADECIMENTOS

Ao Professor Arnaldo de Albuquerque Araújo, pela orientação séria, criteriosa e amiga.

Ao CNPq, pelo suporte financeiro através da concessão de bolsa de estudos.

Aos meus amigos do IAPAS, pela compreensão e incentivo durante os estudos.

Aos amigos e colegas do Laboratório Associado de Sensoriamento Remoto (LASR), por proporcionarem a infra-estrutura de hardware e um ambiente de trabalho agradável e de cooperação mútua.

Ao Professor e amigo Wilson Guerreiro Pinheiro, pelas sugestões e revisão ortográfica do trabalho.

A todas as pessoas que indiretamente contribuiram para a realização deste trabalho.

A meus pais e irmãos, pelas preocupações, sacrifícios e contínua motivação.

À Sheila, pela força motivadora e constante colaboração.

A **Deus**, pelo dom de todos os recursos empregados neste trabalho.

À minha Mãe,

SUMÁRIO

Pág.

Pág.

LISTA DE FIGURAS

Figura Pág.

Figura Pág.

Figura Pág.

Figura Pág.

Figura Pág.

Figura

Pág.

LISTA DE TABELAS

Tabela Pág.

LISTA DE ABREVIATURAS

Abreviação

ASIC	- "Application Specific Integrated Circuit"
CIRC	- imagem círculo
CIRCRD	- imagem círculo com ruído
CLOSE	- imagem de teste rosto de mulher
DESV	- desvio padrão (rotina)
DISCUVI	- display de imagens
DP	- desvio padrão
EMQ	- erro médio quadrático
ERQUAD	- erro médio quadrático (rotina)
EXSHARP	- transformada de aguçamento extremo
FACETAS	- suavização baseada no modelo de facetas
HBDET	- detetor de bordas híbrido
HIST	- histograma
JM	- Jefrey-Matusita
KIRSCH	- operador de Kirsch
KVIZ3	- média dos k vizinhos mais próximos 3x3
KVIZ5	- média dos k vizinhos mais próximos 5x5
LAPLAC1	- operador laplaciano 1
LAPLAC2	- operador laplaciano 2
LAPLAC3	- operador laplaciano 3
MAXVER	- máxima verossimilhança
MEDIA3	- filtro da média 3x3
MEDIA5	- filtro da média 5x5
MEDIANA3	- filtro da mediana 3x3
MEDIANA5	- filtro da mediana 5x5
MEDNADP	- filtro da mediana adaptativo
NUBRD	- imagem círculo nublada com ruído
ORDAP	- filtro da ordem adaptativo

Abreviação

PDI	- processamento digital de imagens
PERF	- perfil de varredura
pixel	- "picture element"
PREWITTDF	- operador diferencial de Prewitt
PREWITTDR	- operador direcional de Prewitt
RANK3	- filtro da ordem 3x3
RANK5	- filtro da ordem 5x5
RDGAUSS	- gerador de ruído gaussiano
RDIMP	- gerador de ruído impulsivo
RLIN	- realçador de linhas finas
ROBERTS	- operador de Roberts
ROBINSON3	- operador de Robinson 3 níveis
ROBINSON5	- operador de Robinson 5 níveis
SGRAD	- suavização baseada no inverso do gradiente
SIGMA	- filtro sigma
SIGMAPOL	- filtro sigma polarizado
SIGMADAP	- filtro sigma adaptativo
SITIM	- sistema de tratamento de imagens
SMDA	- suavização com vizinhança selecionada por média das diferenças absolutas
SOBEL	- operador de Sobel
SSDA	- suavização com vizinhança selecionada por soma das diferenças absolutas
SSLOG	- suavização logarítmica
SSVAR	- suavização com vizinhança selecionada por variância
TM	- "Thematic Mapper"
UVIDISC	- armazenamento de imagens em disco
WRS	- "World Reference System"

RESUMO

Os filtros espaciais têm sido cada vez mais utilizados em tarefas de realce e remoção de ruído em imagens digitais. Este trabalho descreve a implementação de um pacote modular de software para filtragem espacial de imagens e um estudo, com atualização bibliográfica, das técnicas de uso corrente em processamento digital de imagens. Um estudo comparativo entre técnicas de suavização consagradas na literatura e técnicas recentemente desenvolvidas e publicadas foi relizado. Uma aplicação de métodos baseados no critério de vizinhança seletiva e adaptativos no pré-processamento de imagens multiespectrais é apresentada.

Os algoritmos correntes na literatura foram estudados e implementados em linguagem C. Foi criada uma ferramenta de processamento digital de imagens, denominada FILTRIX e dedicada à família de equipamentos SITIM - Sistema de Tratamento de Imagens, baseado em microcomputador, de fabricação nacional e bastante difundida nas instituições de pesquisa. O pacote inclui filtros passa-baixas, passa-altas e ferramentas estatísticas e gráficas para avaliação do desempenho dos algoritmos nas diversas aplicações.

Com o rápido desenvolvimento das técnicas de processamento digital de imagens e a grande expansão das áreas de aplicação, métodos de filtragem adaptativos e eficientes têm sido propostos. Foi realizado um estudo comparativo entre várias técnicas de suavização, com o objetivo de medir seu desempenho em operações de remoção de ruído, preservação de

bordas tipo impulso, remoção de ruído impulsivo, preservação de detalhes tênues e imunidade à distorção de formas. Foram consideradas técnicas consagradas na literatura e técnicas recentemente desenvolvidas e publicadas. Para avaliação do desempenho, foram empregados medidas do desvio padrão em área homogênea, erro médio quadrático, perfis de varredura e interpretação visual. Os resultados mostraram a grande eficiência do filtro Sigma e dos algoritmos baseados no critério de vizinhança seletiva.

Outros experimentos utilizando imagens sintéticas e de fotos digitalizadas foram realizados para ilustrar os efeitos de homogeneização produzidos pelos filtros passa-baixas e a sensibilidade dos métodos de deteção de bordas na geração de imagens-gradiente.

Neste trabalho, também é apresentado um método para geração de atributos espaciais em imagens multiespectrais, através do emprego de técnicas não-lineares de filtragem espacial. Resultados preliminares são mostrados. O filtro da ordem adaptativo e a suavização com vizinhança selecionada por soma das diferenças absolutas foram aplicados a uma imagem obtida pelo satélite LANDSAT TM/5 de coordenadas WRS 223.76. O algoritmo de seleção de atributos baseado nos critérios de máximas distâncias JM média e mínima foi utilizado para se obter uma composição com 3 canais, mais discriminante para as classes de interesse. A imagem foi classificada através do método da máxima verossimilhança. Como resultado, foi obtido um aumento da precisão na classificação das composições contendo bandas pré-processadas em relação à classificação com a composição original.

ABSTRACT

Spatial-domain noise smoothing techniques have been widely used in image enhancement tasks. This work discribes a software tool for spatial image filtering, its implementation and applications. Edge-preserving smoothing methods are studied and have their performance evaluation illustrated. An application to multiespectral image preprocessing is presented.

Current algorithms were studied and implemented in the C language. A digital image processing tool named FILTRIX was performed. It is dedicated to a microcomputer-based image processing system, named SITIM (Sistema de Tratamento de Imagens). The software package includes low-pass and high-pass filters and statistical and graphical tools to evaluate the algorithms performance.

A comparative study of traditional and adaptive edge-preserving smoothing techniques was carried out. The algorithms capacity for noise-removing with edges and fine details preservation was evaluated. Image signal standard deviation in a homogeneous area, mean square error beetween the original and filtered image and scanline profiles are used to measure the algorithms efficiency. Results demonstrate a high performance of the adaptive and selective neighbourhood-based techniques. Other tests illustrate low-pass and high-pass filters effects in synthetic and natural images.

In this work, an approach to generate spatial attributes from multiespectral images, using non-linear filtering techniques is also presented. Adaptive-order statistical filtering and sum of absolute differences values smoothing were applied to a LANDSAT TM/5 image with WRS 223.76 coordinates. The feature extraction algorithm based on the maximum JM mean and maximum JM minimum distances criteria is used to obtain a three-channel plus discriminant composite for studied classes. The image was classified by the maximum likelihood method. In the preprocessed multispectral composite classification process a performance mean increasing and an abstention and confusion means decreasing related to original composite classification have been obtained. Initial results have demonstrated that spatial domain edge-preserving smoothing techniques contribute to an accuracy increasing in multispectral image classification tasks.

1 INTRODUÇÃO

A visão é considerada o mais poderoso dos sentidos. Sob a luz da teoria da informação o nosso sistema visual dispõe de uma altíssima capacidade de canal. Um fluxo de dados equivalente a vários milhões de bits por segundo (Mbits/s) percorre normalmente o nervo ótico humano, responsável pela comunicação entre os olhos (sistema sensor natural) e o cérebro (sistema processador natural) [1].

Devido à grande importância do sistema visual humano o processamento digital de imagens (PDI) é atualmente uma das mais promissoras áreas do processamento da informação, tendo motivado a criação de novos processos e tecnologias e suscitado um grande número de aplicações.

Com o surgimento dos computadores digitais de alta velocidade, esta área de conhecimento tem crescido rapidamente e tomado dimensões que abrangem aplicações científicas, industriais, militares, de segurança, de lazer e artísticas.

O desenvolvimento de máquinas com arquiteturas dedicadas e de sistemas sensores eficientes, baseados em fenômenos óticos, sônicos, eletromagnéticos e radioativos, permite, hoje, a implementação de sofisticadas técnicas de processamento digital de imagens.

Entre outras aplicações de PDI, destacam-se controle

de qualidade, inspeção de peças, automação e visão de robôs, análise térmica, classificação de solos e vegetação, análise de recursos naturais, classificação de células, compressão de dados, vídeo-texto e vídeo-fone, visão noturna, detecção de alvos e rastreamento, monitoração de tráfego, análise de impressões digitais, reconhecimento automático de pessoas, monitoração e segurança de ambientes públicos. Na medicina, técnicas de reconstrução e processamento são utilizadas em tomografia computadorizada, medicina nuclear, ressonância nuclear magnética, radiografia digital, ultrassonografia e diagnóstico semi-automático. Também são realizadas colorizações de filmes em preto e branco e transformações em imagens de TV, bem como codificação e compressão de imagens para tarefas de transmissão e armazenamento.

Junto com o desenvolvimento das máquinas, algoritmos de PDI têm-se multiplicado e o grande número de pesquisadores envolvidos tem gerado uma grande produção literária sobre o assunto. Vários livros textos são disponíveis [2]-[27], fazendo uma abordagem dos fundamentos e aplicações de processamento digital de imagens. Nos últimos dez anos, uma grande quantidade de artigos de revisão e edições especiais foi publicada enfocando os vários aspectos da área.

Uma imagem monocromática pode ser expressa através de uma função f(x,y), onde f representa o nível de cinza ou brilho no ponto de coordenadas (x,y). A imagem digital é obtida por meio de um processo, ilustrado na Figura 1.1, que envolve duas etapas: amostragem e quantização [27]. A amostragem consiste em discretizar as coordenadas da imagem, enquanto a quantização discretiza e limita a faixa de luminância presente na imagem original. O resultado é um arranjo bidimensional I(m,n), onde I representa uma medi-

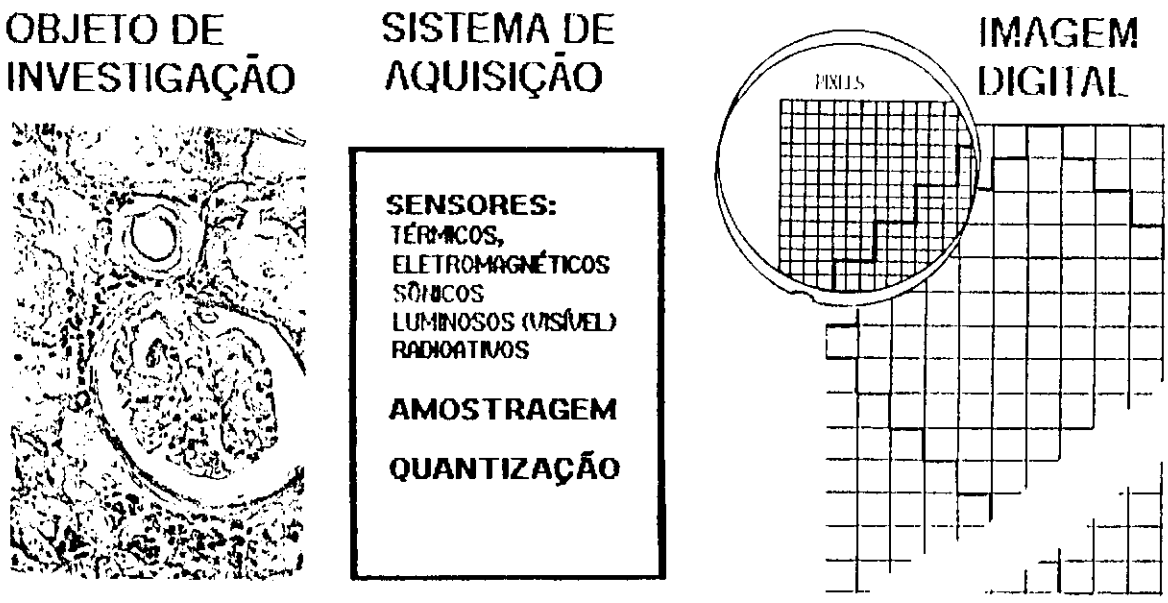

Figura 1.1 Processo de digitalização de imagens.

da discreta do brilho do ponto situado na posição (m,n) do plano. Cada elemento do arranjo é chamado pixel (abreviatura do inglês "picture element"). Dessa forma, a imagem, expressa como um conjunto bem definido de pontos no espaço euclidiano, pode ser submetida a várias manipulações matemáticas com a ajuda de um computador.

Muito comumente, as imagens sofrem algum processo de degradação que pode ser pontual, espacial ou uma combinação de ambos. Degradações pontuais incluem os ruídos aditivo e multiplicativo e distorcem os níveis de cinza dos pixels. As degradações espaciais produzem nublamento ou borramento das imagens, tornando difusos os limites entre regiões e detalhes. Estes ruídos podem ser provocados por dispersão de luz, defeitos nos sensores e, muito freqüentemente, por imperfeições em canais de transmissão [3],[5]-[7],[28]-[31].

Restauração e realce estão geralmente direcionadas a imagens degradadas. A restauração tem como objetivo princi-

pal obter, a partir de um conhecimento a priori da degradação envolvida, uma imagem tão próxima quanto possível da imagem original em sua condição antes da degradação [5]-[7],[32]-[35].

As técnicas de realce, por sua vez, procuram, com base em características do sistema visual humano, destacar algumas informações presentes em imagens degradadas ou não [2],[5]-[7],[21],[31]. Diferem das técnicas de restauração porque, nesta busca, podem até distorcer a imagem para obter o realce de algumas características mais relevantes para o estudo em questão.

Restauração tem como objetivo principal obter, a partir de um conhecimento a priori da degradação envolvida, uma imagem tão próxima quanto possível da imagem original em sua condição antes da degradação [5]-[7],[32]-[36].

Realce de imagens está geralmente relacionado com expansão de contraste, suavizacão, realce de bordas e pseudocoloração. A expansão de contraste consiste em manipular a distribuição de níveis de cinza existentes na imagem visando utilizar o máximo da faixa de luminosidade disponível no sistema de processamento [5]-[7],[37]-[39]. Isto é feito normalmente através de transformações no histograma da imagem.

Suavização busca uma homogeneização das regiões das imagens, alterando níveis de cinza descorrelacionados com a vizinhança e que podem representar um ponto ruidoso. Realce de bordas procura aumentar a definição dos limites entre as regiões, transformando variações gradativas de níveis de cinza em mudanças tipo degrau. Com isso, são corrigidos problemas de ausência de foco e nublamento, tor-

nando possível o reconhecimento de formas inicialmente confusas. Pseudocoloração baseia-se no fato de que o olho humano é capaz de perceber melhor as diferenças de cores que diferenças pequenas de intensidade luminosa. Assim, níveis de cinza são representados por cores distintas, através de tabelas de transformação, de modo que análises subjetivas podem ter resultados mais precisos.

No domínio da freqüência, suavização e realce de bordas são obtidos através de filtros passa-baixas e filtros passa-altas, respectivamente. No domínio espacial, as técnicas respectivas são integração e diferenciação, também conhecidas como filtros espaciais.

Os filtros espaciais baseiam-se na convolução de máscaras sobre a imagem. Cada pixel é transformado levando em conta alguma correlação existente entre os pixels pertencentes a uma pequena vizinhança. Estas técnicas geralmente incorporam características do ruído, o conhecimento a priori sobre as bordas existentes e propriedades do sistema visual humano, para obter realce.

1.1 Motivação do Trabalho

Esta dissertação faz parte das atividades do Laboratório de Processamento de Sinais, Imagens e Computação Gráfica do Departamento de Engenharia Elétrica da Universidade Federal da Paraíba e do Laboratório Associado de Sensoriamento Remoto do Centro de Ciências e Tecnologia da Universidade Federal da Paraíba. Ela aborda aspectos teóricos e prá-

ticos envolvidos com as técnicas de Filtragem Espacial. O trabalho foi motivado principalmente pela importância do pré-processamento de imagens através destas técnicas e pela necessidade de se obter uma ferramenta de "software" que possibilite a aplicação e o estudo de filtros no domínio espacial, em ambientes baseados em microcomputadores.

A existência de um equipamento de fabricação nacional dedicado a tarefas de processamento de imagens, contribuiu para que o pacote fosse para ele direcionado, ampliando seus recursos e aumentando seu espaço de aplicações. Este equipamento é o Sistema de Tratamento de Imagens, SITIM, já bastante difundido em universidades e instituições de pesquisa.

Os estudos e implementações realizados fazem parte de uma continuação e atualização do trabalho realizado por Araújo [40]. Em seu trabalho, Araújo desenvolveu o algoritmo de suavização com vizinhança selecionada por soma das diferenças absolutas (SSDA) e apresentou um estudo comparativo dos desempenhos de SSDA e várias técnicas de suavização, em diversas condições de ruído e em diferentes tipos de imagens. O surgimento de novos algoritmos tornou relevante a realização de um novo estudo comparativo com o objetivo de avaliar o seu desempenho em tarefas de remoção de ruído com preservação de bordas [41].

O prosseguimento da pesquisa mostrou que filtros espaciais são cada vez mais utilizados no pré-processamento de imagens. O levantamento bibliográfico sobre o assunto levou à seleção, dentre muitas, de 26 técnicas correntemente utilizadas, algumas recentemente desenvolvidas e publicadas. A literatura abordada mostrou também que há uma busca do desenvolvimento de programas aplicativos dedicados a pequenas estações de trabalho, baseadas em microcomputadores, com

fins didáticos e/ou de projetos que envolvam processamento digital de imagens.

Com esta preocupação, foi proposto e desenvolvido, em linguagem C, um pacote de "software" denominado FILTRIX [42],[43], que inclui algoritmos dedicados à suavização, à remoção de ruído com preservação de bordas e à detecção de fronteiras. A fim de permitir ao usuário medir o desempenho dos filtros espaciais disponíveis em FILTRIX ou em outro sistema de processamento, foram implementados alguns métodos estatísticos e gráficos.

A classificação de imagens multiespectrais de recursos naturais, juntamente com técnicas de segmentação, podem tornar possível o reconhecimento de diversas formações geológicas, hídricas e vegetais, além de suas estruturas predominantes. Imagens multiespectrais são compostas por várias imagens de uma mesma área, obtidas pelos sensores de um satélite em diferentes faixas do espectro eletromagnético. Nelas, os objetos ou regiões apresentam-se mais ou menos nítidos em função de sua reflectância em certa faixa do espectro. Técnicas de pré-processamento são geralmente empregadas para transformar cada banda espectral ou atributo e melhorar a definição dos elementos presentes na imagem. A aplicação de filtros espaciais em imagens multiespectrais pode realçar alguma correlação existente entre pixels vizinhos e aumentar a precisão de um processo de classificação pela criação de atributos adicionais para cada ponto. O trabalho realizado por Mascarenhas e Dutra [44] comprovou esta teoria, a menos do fato de que o índice de classificação correta foi menor nas fronteiras entre regiões. Naquele trabalho, foi utilizado o filtro da média.

Alguns filtros espaciais não lineares estudados no

presente trabalho, em particular a suavização com vizinhança selecionada por soma das diferenças absolutas e o filtro da ordem adaptativo, demonstraram ter um bom desempenho quando se deseja um efeito de homogeneização de regiões com preservação de bordas. Este fato não ocorre com o filtro da média e motivou a realização de experimentos envolvendo a classificação de imagens multiespectrais pré-processadas por estes dois algoritmos [45]. Os resultados motivaram o estudo dos efeitos da utilização de outras técnicas adaptativas de filtragem espacial em aplicações de sensoriamento remoto [46],[47].

1.2 Organização do Trabalho

Este trabalho trata das técnicas de filtragem no domínio espacial e compreende a realização de um estudo comparativo, a implementação de uma biblioteca de algoritmos de suavização e de detecção de bordas e a aplicação de filtros espaciais no pré-processamento de imagens multiespectrais em tarefas de classificação.

Nos Capítulos 2 e 3, abordam-se as técnicas de suavização e detecção de bordas correntes na literatura. Uma revisão bibliográfica é realizada e os algoritmos implementados e estudados são descritos.

No Capítulo 4, descreve-se a biblioteca de algoritmos como uma ferramenta de processamento de imagens, seus recursos e o "hardware" envolvido na implementação.

Uma avaliação do desempenho de técnicas consagradas na literatura e técnicas recentemente desenvolvidas e publicadas é apresentada no Capítulo 5. Nele, é descrita a realização de um estudo comparativo que mostra o desempenho dos algoritmos quanto à capacidade de realce, remoção de ruído com preservação de bordas e detecção de fronteiras.

No Capítulo 6, descreve-se a aplicação de filtros espaciais em tarefas de classificação de imagens multiespectrais. O efeito da suavização com preservação de bordas produzido pelo algoritmo de suavização com vizinhança selecionada por soma das diferenças absolutas (SSDA) [48] e o filtro da ordem adaptativo (ORDAP) [49], são estudados. Imagens pré-processadas por estas técnicas são classificadas pelo método da Máxima Verossimilhança (MAXVER) [50].

Finalmente, no Capítulo 7, conclusões e sugestões para aprimoramento são apresentadas.

2 TÉCNICAS DE SUAVIZAÇÃO ESPACIAL

Este capítulo apresenta as técnicas de suavização correntes na literatura através de uma revisão bibliográfica, e uma descrição mais detalhada dos métodos implementados e estudados.

2.1. Suavização Espacial

Técnicas de suavização têm como objetivos principais a remoção de ruído e a uniformização dos níveis de cinza dos pixels nas regiões presentes na imagem. No domínio espacial, as técnicas atuam diretamente nos pixels da imagem, através da convolução de uma máscara, que se desloca sobre toda a imagem, como mostra a Figura 2.1, efetuando operações lineares ou não-lineares baseadas em informação estatística localizada. O nível de cinza do pixel central da janela de imagem definida pela máscara é substituído por um valor que é funçãoo do método empregado e dos níveis de cinza da vizinhança formada pelos pixels desta janela.

Um procedimento básico consiste no cálculo do valor médio dos níveis de cinza dos pixels envolvidos pela máscara, ponderados pelos pesos da mesma. Este é o caso do filtro da média [5]-[7],[52],[53], onde os pesos são fixos. O

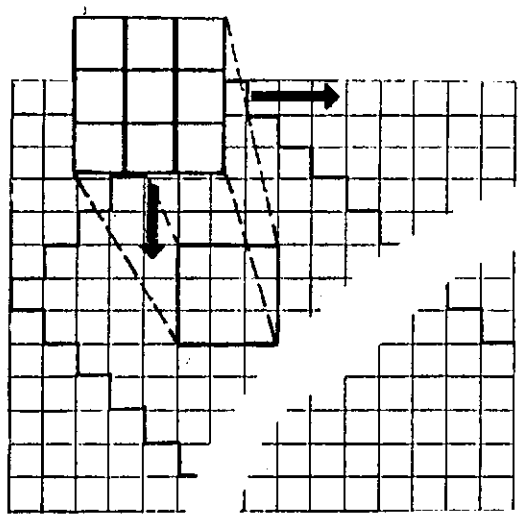

Figura 2.1 Processo de convolução (filtragem espacial)

simples cálculo da média apesar de reduzir consideravelmente o ruído em regiões homogêneas, produz o nublamento e a conseqüente perda de informações nos limites entre as regiões. Considerando este problema, vários autores apresentaram outras máscaras que utilizam coeficientes de pesos variáveis e restrições para o cálculo da média, tais como: média com pesos especiais para bordas e linhas; média com os K vizinhos, cujos níveis de cinza são os mais próximos daquele do pixel central; média com os cinco vizinhos consecutivos numa janela 3x3; média com pesos, onde o peso dado a um pixel vizinho depende de quão próximo é seu nível de cinza do nível de cinza do pixel central; média com aqueles vizinhos que satisfazem a certos requisitos relacionados com o histograma da janela; média utilizando rotulação ("labeling") probabilística e média com pesos que decrescem radialmente a partir do pixel central [54]-[60].

Operadores gaussianos otimizados foram calculados por Davies [61] para pequenas vizinhancas (3x3 e 5x5). Nestes operadores, os pesos decrescem radialmente a partir do pixel central da máscara seguindo uma distribuição gaussiana. O método visa, com essa aproximação, aumentar o efeito de isotropia no cálculo da média ponderada.

Ainda com o objetivo de evitar a supressão de informação de bordas durante a substituição da informação ruidosa, outros métodos de natureza seletiva foram desenvolvidos por Bednar e Watt [62], Pomalaza-Raez e McGillem [63], Lee e Kassam [64] e Restrepo e Bovik [65]. Eles introduziram uma classe de algoritmos chamados "trimmed-mean filters", que combinam a capacidade de preservação de bordas do filtro da mediana com a capacidade de supressão de ruído do filtro da média, através da exclusão de uma parte dos elementos da máscara no cálculo da média final. Lee [66]-[68] usou médias e variâncias locais para determinar os pesos para o cálculo do valor médio. O filtro sigma [69],[70], também apresentado por Lee, é baseado na probabilidade sigma da distribuição gaussiana e suaviza a imagem calculando a média com apenas aqueles elementos, cujas intensidades estão dentro da faixa sigma do pixel central.

O método conhecido como "box-filtering" [71] calcula a média com os pixels que estão dentro de uma faixa fixa de intensidade. Prewitt [52] e Graham [54] sugeriram algoritmos similares e Wang et alii [72] utilizaram uma máscara onde os coeficientes de peso são os inversos dos gradientes entre o pixel central e seus vizinhos. Song e Pearlman [73] apresentaram os filtros adaptativos da média, mediana e mínimo erro médio quadrático, baseados em uma máscara de tamanho variável e dependente da vizinhança de cada pixel a ser processado.

Os filtros da ordem ("rank") [74],[75] operam na imagem atribuindo ao pixel central de uma máscara de dimensão LxL o k-ésimo valor de nível de cinza dentre aqueles dos L^2 elementos da máscara, quando organizados em ordem crescente. Casos particulares e bem conhecidos são os filtros min (k=1), max ($k=L^2$) [76] e o filtro da mediana ($k=(L^2+1)/2$) [7],[77]-[79]. Uma transformada de aguçamento extremo que utiliza uma combinação dos filtros min e max foi apresentada por Lester et alii [80].

O filtro da mediana tem a importante característica de remover o ruído, conservando razoavelmente intactas as bordas presentes na imagem. Este fato tem levado a sua utilização em uma grande quantidade de aplicações, em especial no processamento de imagens médicas, e motivado muitos pesquisadores a desenvolverem variações otimizadas [81-91]. Com o objetivo de incrementar a capacidade de preservação de detalhes tênues, Nieminen et alii [92] apresentaram uma nova classe de filtros recursivos híbridos com multiníveis, que se baseia no cálculo da mediana entre o pixel central da janela e as respostas de vários subfiltros. Os subfiltros podem ser, por sua vez, baseados em outras técnicas lineares ou não-lineares e são aplicados em subregiões (vizinhanças) do interior da janela. Eles se comportam como sensores dos atributos de cada vizinhança e sua escolha apropriada torna a preservação dos detalhes bastante independente de sua orientação. Heinonen e Neuvo [93] sugeriram os filtros da mediana híbridos baseados em princípio similar, mas que utilizam apenas subfiltros lineares. Loupas et alii [80] propuseram uma técnica adaptativa que utiliza medidas locais de variância, no processo de seleção do elemento mediano de uma vizinhança.

Um método empregado por Hom Lee e Fam [49], denomina-

do filtro da ordem adaptativo, apresenta uma natureza seletiva com relação às respostas dos subfiltros. Este método baseia-se no fato de que, dada uma janela unidimensional com N níveis de cinza presentes, a diferença entre o valor médio e o mediano pode indicar quão distante o nível de cinza do elemento central está dos dois níveis extremos presentes na janela.

Assim como em outras tarefas de processamento de imagens [94],[95], também em filtragem espacial uma classe de métodos que incorpora características do sistema visual humano tem-se desenvolvido [35],[96]-[101]. Trussel [96] aprimorou um algoritmo que utiliza critérios subjetivos, introduzido por Anderson e Netravali [97], levando em conta o fato de o olho humano tolerar melhor o ruído em regiões onde o sinal é de grande atividade, suavizando, assim, apenas as áreas de baixa atividade. Chanda et alii [35] consideram que, em um campo discreto de luminância, o sistema visual humano é mais sensível à relação $\Delta I/I$ que à variação ΔI, onde I representa uma dada intensidade no campo. Este comportamento logarítmico norteia a formulação do seu algoritmo de suavização. Jung e Kim [101] introduziram modificações no filtro sigma apresentado por Lee [69],[70]. Estas modificações levam em conta a incapacidade do olho humano de perceber ruídos de pequena amplitude e que podem ser confundidos com informações de texturas detalhadas ou de bordas discretas.

A maioria destas máscaras é definida por constantes preespecificadas, obtidas por tentativa e erro, ou por algum conhecimento a priori das propriedades do ruído e da imagem. Uma busca tem sido feita no sentido de se garantir que o valor atribuído ao nível do pixel processado decorra da análise da máxima correlação entre o seu nível

antes do processamento e aqueles predominantes na vizinhança. O estudo desta correlação aumenta a precisão na inserção do pixel central em uma região à qual ele realmente pertença. Dessa preocupação, surgiu uma classe de métodos que usa o princípio de vizinhanças seletivas. Tomita e Tsuji [102] apresentaram uma técnica de suavização que associa, a cada ponto da imagem, o nível de cinza da vizinhança mais homogênea entre cinco vizinhanças retangulares. Nagao e Matsuyama [103] aprimoraram essa idéia propondo o uso de máscaras rotativas em forma de barra e o cálculo da variância em cada vizinhança como critério de avaliação da homogeneidade. Araújo [48] apresentou um método que utiliza nove máscaras 3x3 superpostas em uma janela 5x5. Neste método, o índice de homogeneidade usado é a soma das diferenças absolutas entre o nível de cinza do pixel central e aqueles dos pixels pertencentes a cada máscara 3x3.

Morfologia matemática, introduzida por Matheron [104] e Serra [105], também tem sido utilizada para filtragem espacial com preservação de bordas. Stevenson e Arce [106], e Maragos [107] sugeriram os filtros morfológicos que consideram um sinal de imagem como um conjunto no espaço euclidiano com estruturas geométricas que podem ser quantitativamente descritas. Estas estruturas são modificadas por operações não-lineares como erosão e dilatação, produzindo homogeneização e separação de regiões [108-113].

Teoria de conjuntos nebulosos ("fuzzy sets") foi sugerida por Pal e King [114] em aplicações de realce de imagens. Imagens tridimensionais têm sido processadas por métodos de suavização bidimensionais [115],[116].

Ruído do tipo impulsivo com distribuição aleatória tem características especiais e, muitas vezes, é tratado por técnicas específicas. Abdelmalek [117] apresentou um procedimento para remoção deste tipo de ruído, que trata diferentemente imagens ruidosas e predominantemente claras daquelas ruidosas e predominantemente escuras. Lin e Willson [87] introduziram o filtro da mediana com janelas de tamanho adaptativo que apresentam alto desempenho na eliminação deste tipo de ruído. Kundu et alii [118] e Mohwinkel e Kurz [119] também apresentaram métodos para remoção de ruído impulsivo. Rabanni [120] propôs um algoritmo baseado no critério da máxima probabilidade a posteriori para remoção de ruído com distribuição modelada pela de Poisson.

2.2 Filtros Passa-baixas Implementados

Os métodos foram selecionados entre aqueles correntemente utilizados em processamento digital de imagens e divulgados na literatura. Algoritmos recentemente desenvolvidos também foram incluídos. Algumas das técnicas selecionadas já foram estudadas em [40] e são descritas neste trabalho para permitirem uma avaliação das suas características principais nos contextos do estudo comparativo realizado (ver Capítulo 5) e da ferramenta de software implementada (ver Capítulo 4). Estas características envolvem complexidade, facilidade de implementação e nível de adaptabilidade às características da imagem. Por questão de simplificação, os algoritmos serão referidos pelas suas abreviaturas abaixo relacionadas. Os seguintes métodos foram implementados:

a. filtro da média 3x3	MEDIA3
b. filtro da média 5x5	MEDIA5
c. filtro da mediana 3x3	MEDIANA3
d. filtro da mediana 5x5	MEDIANA5
e. filtros da ordem 3x3	RANK3
f. filtros da ordem 5x5	RANK5
g. transformada de agucamento extremo	EXSHARP
h. filtro da ordem adaptativo	ORDAP
i. filtro sigma	SIGMA
j. filtro sigma polarizado	SIGMAPOL
k. filtro sigma adaptativo	SIGMADAP
l. filtro da mediana adaptativo	MEDNADP
m. média dos k vizinhos mais próximos 3x3	KVIZ3
n. média dos k vizinhos mais próximos 5x5	KVIZ5
o. suavização baseada no modelo de facetas	FACETAS
p. suavização com vizinhança selecionada por soma das diferenças absolutas	SSDA
q. suavização logaritmica	SSLOG
r. suavização baseada no inverso do gradiente	SGRAD
s. suavização com vizinhança selecionada por média das diferenças absolutas	SMDA
t. suavização com vizinhança selecionada por variância	SSVAR

2.2.1 Filtro da Média

O ruído numa imagem normalmente aparece como erro aditivo, aleatório e descorrelacionado, podendo ser eliminado, substituindo-se os pixels afetados por uma média local para reduzir variações de nível de cinza. Considerando uma janela WxW da imagem, cujos elementos possuem níveis

de cinza p(i), onde i=1,2,...,W^2 , a média m pode ser calculada por:

$$m = 1/W^2 \sum_{i=1}^{W^2} p(i)$$

O filtro da média realiza esta operação através da convolução da máscara da Figura 2.2. [5]-[7]. A máscara é normalizada para a unidade, para evitar que o processo de suavização introduza uma polarização de intensidade na imagem filtrada. Tomando como exemplo a janela 3x3, mostrada na Figura 2.3.a, o pixel a ser processado (ponto central P) teria seu nível de cinza 70 substituído por 21, valor médio dos níveis de cinza nesta janela. Pode ser observado que, pelo seu nível de cinza, P provavelmente representa um ponto ruidoso.

$$M = 1/9 \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Figura 2.2 Máscara utilizada pelo filtro da média (3x3).

$$W = \begin{matrix} 5 & 25 & 30 \\ 5 & 70 & 10 \\ 4 & 30 & 10 \end{matrix} \qquad W = \begin{matrix} 11 & 10 & 30 & 31 & 32 \\ 10 & 11 & 31 & 31 & 32 \\ 9 & 9 & 80 & 32 & 32 \\ 10 & 10 & 29 & 30 & 32 \\ 12 & 11 & 29 & 31 & 31 \end{matrix}$$

(a) (b)

Figura 2.3 Janelas de imagem (a) 3x3 e (b) 5x5.

2.2.2 Filtros da Ordem

Os filtros da ordem [74],[75] baseiam-se na ordenação, segundo a magnitude, dos W^2 níveis de cinza presentes em uma janela WxW. Em um filtro de ordem K, o pixel central da janela é substituído pelo K-ésimo elemento do arranjo ordenado. O algoritmo pode ser descrito como segue:

Sejam $p(1),p(2),\ldots,P(W^2)$ os níveis de cinza dos W^2 elementos de uma janela WxW da imagem I(m,n), e $z(1)$, $z(2),\ldots,$ $z(W^2)$, os mesmos níveis de cinza arranjados em ordem crescente. A função de I(m,n), definida por

$$R_K[I(m,n)] = z(K),$$

é o filtro de ordem K.

Os casos particulares K=1 e $K=W^2$, conhecidos como os operadores MIN e MAX [76], respectivamente, produzem os efeitos de compressão e expansão na imagem. Para $K=(W^2+1)/2$, obtém-se o bem conhecido filtro da mediana [7],[81]-[91]. Assim, têm-se:

- operador MIN = R_1
- operador MAX = $R_{\langle W2 \rangle}$
- filtro da MEDIANA = $R_{\langle W2+1 \rangle/2}$

Experiências mostram que os filtros de ordem maior que o filtro da mediana expandem regiões claras da imagem proporcionalmente a sua ordem. Do mesmo modo, os filtros de ordem menor que o filtro da mediana expandem regiões escuras e pequenas.

Um algoritmo denominado transformada de aguçamento extremo, apresentado por Lester et alii [80], utiliza os casos extremos dos filtros da ordem. Para uma janela WxW, o ponto central terá seu nível de cinza p substituído pelo valor extremo, presente na janela (máximo ou mínimo), mais próximo do valor de p. Caso p seja o valor médio entre os extremos, não haverá alteração.

Para a janela 3x3 da Figura 2.3.a, têm-se:

operador máximo (MAX) = 70
operador mínimo (MIN) = 5
operador extremo (OPEX) = 70

2.2.3 Filtro da Mediana

O filtro da mediana [7], [81]-[91] tem sido utilizado no processamento de sinais unidimensionais e bidimensionais. O método é não-linear e caracteriza-se principalmente pela sua capacidade de remover o ruído, preservando mudanças do comportamento do sinal processado. Para uma imagem, essas mudanças representam variações de nível de cinza, que determinam as fronteiras das formas ou objetos existentes. Sua eficiência tem motivado o desenvolvimento de diversas variações otimizadas [85]-[90] e de métodos distintos de implementação [81].

Quando aplicado em duas dimensões, o filtro da mediana consiste em substituir cada pixel da imagem pelo pixel com o valor cinza mediano dentro da janela WxW, cujos elementos foram arranjados pela ordem crescente de

seus valores de cinza. Considerando a janela 3x3, mostrada na Figura 2.3.a, o pixel central teria seu nível de cinza (70) substituído pelo nível de cinza mediano 10 (quinto elemento da seqüência arranjada em ordem crescente: (4,5,5,10,10,25,30,30,70) dos elementos da janela). Resultados apresentados por Heinonen [93] indicam que o filtro da mediana é mais eficiente na redução de ruído impulsivo.

2.2.4 Filtro da Média com os K Vizinhos Mais Próximos

Neste método, o nível de cinza p do ponto central de uma janela WxW é substituído pelo valor médio dos níveis de cinza dos K vizinhos, cujos valores mais se aproximam de p [57]. Dessa forma, é obtida a redução do ruído com preservação das bordas e características lineares. Suavização com poucos vizinhos (K menor) causará melhor preservação de detalhes, porém menos redução do ruído, enquanto que suavização com mais vizinhos trará maior redução de ruído, mas introduzirá alguma distorção nas bordas. Valores de K usados em [97] são 2, 4, 6 e 8. Para a janela 3X3 da Figura 3.2.a e para K=3, por exemplo, os níveis de cinza utilizados no cálculo da média seriam: 30,30,25 e 70.

2.2.5 Suavização Controlada por Gradiente

Este método, proposto por Wang et alii [72], leva em consideração o fato de as variações de níveis de cinza de uma pequena vizinhança, contida inteiramente numa região,

serem menores do que no caso em que ela estivesse numa área de fronteira. Este algoritmo utiliza uma máscara (Fig. 2.4), cujos pesos são os inversos normalizados dos gradientes entre o pixel central e seus vizinhos em uma janela 3x3 (Fig. 2.5).

$$M = \begin{matrix} cp(1) & cp(2) & cp(3) \\ cp(4) & cp(5) & cp(6) \\ cp(7) & cp(8) & cp(9) \end{matrix}$$

Figura 2.4 Máscara com pesos controlados por gradiente.

$$W = \begin{matrix} p(1) & p(2) & p(3) \\ p(4) & p(5) & p(6) \\ p(7) & p(8) & p(9) \end{matrix}$$

Figura 2.5 Os oito vizinhos imediatos de um ponto P=p(5) central numa janela 3x3.

O gradiente inverso absoluto, GI, é definido por:

$$GI(i) = 1/|p(i)-p(5)|$$

onde $i = 1,2,...,9$ e $i \neq 5$.

Para p(i)=p(5), o gradiente é 0 e GI(i) é definido como sendo 2. Assim, o valor de GI(i) varia entre [2,0) e GI(i) é muito menor numa borda que dentro de uma região. A máscara 3x3, mostrada na Figura 2.5, tem seus coeficientes de pesos, cp(i), calculados da seguinte maneira:

$$cp(i) = 1/2 \left[\sum GI(i) \right]^{-1} GI(i)$$

onde $i = 1,2,...,9$, $i \neq 5$ e $cp(5) = 1/2$.

O fator 1/2 é utilizado para evitar uma polarização na escala de cinza.

2.2.6 Suavização com Vizinhança Selecionada por Variância

Este algoritmo pertence a uma classe de técnicas que selecionam vizinhanças do pixel a ser processado, utilizando algum critério de homogeneidade para o cálculo da média. Com isso, procura-se evitar que as variações de níveis de cinza na vizinhança de pixels que pertencem a uma borda sejam interpretados como informação ruidosa. Apenas os pixels com alto nível de correlação são considerados no cálculo da média e isto proporciona uma suavização com preservação de bordas.

O método, proposto por Tomita e Tsuji [102], substitui o nível de cinza de cada ponto da imagem pela média dos elementos da vizinhança mais homogênea entre cinco vizinhanças 3x3, dentro de uma janela 5x5 (Fig. 2.6). Para determinar a vizinhança mais homogênea, um operador gradiente é aplicado às cinco pequenas vizinhanças retangulares.

Um aprimoramento desta técnica foi sugerido por Nagao e Matsuyama [103], que utilizaram nove vizinhanças em forma de barras alongadas ao redor do ponto P, em vez das cinco vizinhanças retangulares. A Figura 2.7 mostra a discretização deste modelo: quatro vizinhanças hexagonais,

quatro vizinhanças pentagonais e uma vizinhança retangular 3x3, centrada em P. Neste caso, a vizinhança mais homogênea é determinada pelo cálculo da variância dos níveis de cinza dentro de cada vizinhança. O nível de cinza de P é substituído pelo nível de cinza médio dos elementos da vizinhança com menor variância.

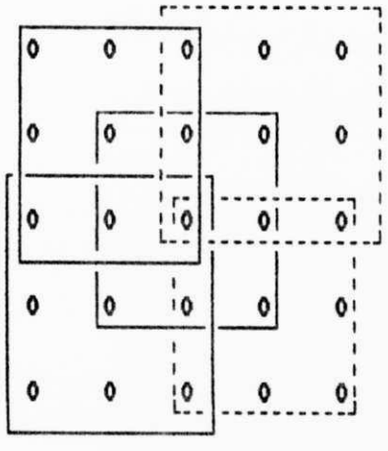

Figura 2.6 Vizinhanças usadas por Tomita e Tsuji

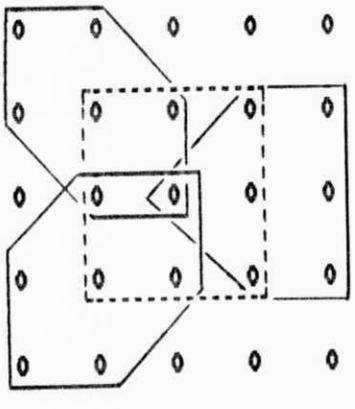

Figura 2.7 Vizinhanças usadas por Nagao e Matsuyama

2.2.7 Suavização com Vizinhança Selecionada por Soma de Diferenças Absolutas

Araújo [48],[121] sugeriu um método de suavização que utiliza nove vizinhanças 3x3 superpostas numa janela 5x5, como mostra a Figura 2.8. O algoritmo calcula um índice de homogeneidade para cada uma das nove vizinhanças, determinado pela soma das diferenças absolutas entre o nível de cinza do ponto central P da janela 5x5 e os níveis de cinza dos elementos de cada vizinhança, ou seja:

$$SDA(k) = \sum |p(i)-\hat{p}| \qquad k = 1,2,\ldots,9$$

onde p(i) é o i-ésimo nível de cinza da k-ésima vizinhança e $\hat{p}$ é o nível de cinza de P.

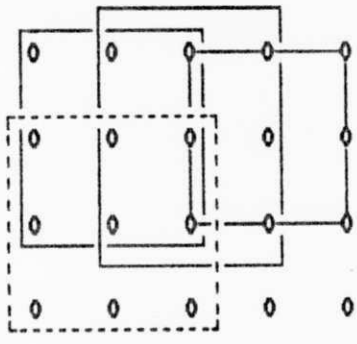

Figura 2.8 Vizinhanças usadas por Araújo

O pixel central é suavizado por uma vizinhança à qual pertence, conseguindo-se, a um custo computacional relativamente baixo, redução de ruído com preservação de bordas. Para o exemplo da Figura 2.3.b, $\hat{p}$ seria substituído

por 37, nível de cinza médio da vizinhança 3 (última da direita e acima). Uma variação deste método, o algoritmo de suavização com vizinhança selecionada pela média de diferenças absolutas (SMDA), foi apresentada em [40]. Nele, são utilizadas as vizinhanças da Figura 2.7 e o índice de homogeneidade MDA é o valor médio das diferenças absolutas entre $\hat{p}$ e os níveis de cinza dos elementos das vizinhanças. O índice MDA é dado por:

$$MDA(k) = 1/r \sum |p(i)-\hat{p}| \qquad k = 1,2,\ldots,r \qquad i = 1,2,\ldots,r$$

onde r é o número de pixels em cada vizinhança, p(i) e $\hat{p}$ são os níveis de cinza definidos anteriormente e k indica a vizinhança considerada.

Para a janela 5x5 da Figura 2.3.b, $\hat{p}$ seria substituído por 37, nível de cinza médio da vizinhança 3 (última da direita e acima).

2.2.8 Suavização Baseada no Modelo de Facetas

Este método baseia-se na suposição de que a imagem ideal pode ser representada por um modelo de facetas [122],[123]. A imagem é dividida em pequenas regiões conectadas, ou facetas, representadas por blocos de dimensão LxL. Cada um destes blocos é adaptado a uma função polinomial por uma aproximação mínima quadrática. Uma faceta modelada por uma função polinomial de grau zero corresponde ao modelo plano de facetas (modelo constante por partes). A adaptação por polinômios de grau maior resulta em facetas com

inclinação, facetas quadráticas e superfícies curvas discretizadas. Neste algoritmo, um plano com inclinação é adaptado a cada um dos blocos com L^2 pixels e o nível de cinza do pixel em processamento é substituído por um nível de cinza mais bem adaptado àquele pixel. Para L=3, estes blocos resultam em nove vizinhanças 3x3, superpostas numa janela 5x5 da imagem, como na Figura 2.8. Através de um procedimento mínimo quadrático, usado para determinar os parâmetros do plano inclinado, obtém-se um conjunto de máscaras lineares para adaptação da posição de cada pixel numa vizinhança 3x3 (Fig. 2.9).

$$M_1 = \begin{matrix} 8 & 5 & 2 \\ 5 & 2 & 1 \\ 2 & 1 & 4 \end{matrix} \qquad M_2 = \begin{matrix} 5 & 5 & 5 \\ 2 & 2 & 2 \\ 1 & 1 & 1 \end{matrix} \qquad M_3 = \begin{matrix} 2 & 5 & 8 \\ 1 & 2 & 5 \\ 4 & 1 & 2 \end{matrix}$$

$$M_4 = \begin{matrix} 5 & 2 & 1 \\ 5 & 2 & 1 \\ 5 & 2 & 1 \end{matrix} \qquad M_5 = \begin{matrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{matrix} \qquad M_6 = \begin{matrix} 1 & 2 & 5 \\ 1 & 2 & 5 \\ 1 & 2 & 5 \end{matrix}$$

$$M_7 = \begin{matrix} 2 & 1 & 4 \\ 5 & 2 & 1 \\ 8 & 5 & 2 \end{matrix} \qquad M_8 = \begin{matrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 5 & 5 & 5 \end{matrix} \qquad M_9 = \begin{matrix} 4 & 1 & 2 \\ 1 & 2 & 5 \\ 2 & 5 & 8 \end{matrix}$$

Figura 2.9 Máscaras de filtragem utilizadas em [123]

O algoritmo pode ser descrito como segue:

Seja B(k) o resultado normalizado (dividido por 18) da aplicação da k-ésima máscara de filtragem à k-ésima vizinhança 3x3 da janela 5x5. Calcula-se, para cada vizinhança, um erro $E^2(k)$, dado por:

$$E^2(k) = \sum (B(k)-p(i))^2 \qquad \begin{matrix} k = 1,2,\ldots,9 \\ i = 1,2,\ldots,9 \end{matrix}$$

onde p(i) é o i-ésimo nível de cinza dentro da k-ésima vizinhança.

O B(k) da vizinhança de menor erro é atribuído ao nível de cinza do pixel central da janela em processamento. Para a janela de imagem da Figura 3.2.b o pixel central teria seu nível de cinza substituído por 44, B(k) da vizinhança de menor erro (correspondente à máscara M_4).

2.2.9 Filtro Sigma

Em imagens, o ruído tem geralmente uma distribuição gaussiana. O filtro sigma [69],[70] é motivado pela probabilidade sigma da distribuição gaussiana. O pixel central tem seu nível de cinza substituído pela média calculada com apenas aqueles elementos que têm seus níveis de cinza dentro de uma faixa sigma de intensidade, fixada pelo pixel central. Dessa forma, o filtro passa a ter caráter seletivo e preserva melhor pontos pertencentes às bordas ou pequenos detalhes.

A probabilidade dos dois-sigmas é definida como sendo a probabilidade de uma variável aleatória estar dentro da faixa de dois desvios padrões do seu valor médio. Neste método, este valor médio é definido como sendo o nível de cinza p do pixel central de uma vizinhança de tamanho WxW.

Para a remoção de ruído aditivo com valor médio zero e desvio padrão D, o algoritmo pode ser descrito como segue:

a) estabelece-se a faixa dos dois-sigmas de intensidade (p-T, p+T), onde T = 2D;
b) somam-se todos os pixels da janela WxW que têm seus níveis de cinza dentro da faixa de intensidade estabelecida;
c) calcula-se a média, dividindo a soma pelo número de pixels somados;
d) atribui-se esta média a p.

Foi observado que ruído do tipo impulso, representado pelo conjunto de um ou dois pixels não seria suavizado. Para contornar esse problema, foi proposto calcular ao invés da média dos dois-sigmas, a média com os vizinhos imediatos do pixel central, se M, o número de pixels dentro da faixa de intensidade dos dois-sigmas, for menor que um certo valor K preestabelecido. Assim, o passo (d) é substituído por:

d) p(i) = média dos dois-sigmas, se M $>$ K
   p(i) = média dos vizinhos imediatos, se M $<$ K

O valor de K deve ser escolhido criteriosamente e levando em conta o tamanho da janela empregada na convolução. Para uma janela 7x7, K deve ser menor que 4 e para uma janela 5x5, K deveria ser menor que 3.

Para imagens com características de ruído desconhecidas, a faixa de intensidade T pode ser definida por uma medida aproximada do desvio padrão do ruído numa área homogênea. A aplicação consecutiva, com redução do desvio padrão

a cada iteração, proporciona a eliminação controlada do ruído. Para a janela exemplo da Figura 2.3.b, considerando-se o desvio padrão D=20,0 e K=2, ter-se-ia a faixa (40,120), e o pixel central teria seu valor, 80, substituído por 29, valor médio dos seus vizinhos imediatos.

O filtro sigma polarizado tem a propriedade de aguçar bordas do tipo rampa e de aumentar o contraste de detalhes tênues. A polarização consiste no cálculo da média considerando os pixels de duas vizinhanças distintas. Inicialmente é calculada a média com os pixels cujos níveis de cinza estão na faixa superior de intensidade (p,p+T), e em seguida com os pixels cujos níveis de cinza estão na faixa inferior de intensidade (p,p-T) [70]. As diferenças absolutas entre p e as médias superior e inferior são calculadas. O pixel central tem seu nível de cinza substituído pela média mais próxima de p.

2.2.10 Filtro da Ordem Adaptativo

Esta é uma técnica não linear e híbrida, pois utiliza uma combinação das respostas de dois subfiltros que são aplicados seqüencialmente às linhas e às colunas da imagem [49]. O método é semelhante ao filtro da ordem e os subfiltros são os da média e da mediana, aplicados em janelas unidimensionais de tamanho 2N+1. A convolução é realizada inicialmente sobre todas as linhas da imagem, utilizando uma janela horizontal e, em seguida, sobre todas as colunas, utilizando uma janela vertical.

Considerem-se X_1, X_m, ..., X_{mN+1} os níveis de cinza dos pixels pertencentes à janela, centrada no pixel candidato P. A média, m, e a mediana, Md, dos 2N+1 elementos são calculadas. Os valores de m e Md são comparados e o nível de cinza $\hat{p}$ de P é substituído de acordo com a expressão :

$$\hat{p} = \begin{cases} X_{(N+1-J)} & \text{se } m > Md \\ X_{(N+1+J)} & \text{caso contrário} \end{cases}$$

onde $X_{(i)}$ é o i-ésimo menor nível de cinza dentro da janela ($1 < i < 2N+1$) e J é um inteiro entre 1 e N, inclusive.

Se J=N, o filtro seleciona o mínimo ou o máximo valor dentro da janela. Para J=0, o método transforma-se no bem conhecido filtro da mediana. A comparação combina a capacidade de suavização do filtro da média com a capacidade de realce de bordas do filtro da mediana. O parâmetro J é selecionado pelo usuário e determina se o efeito de aguçamento de bordas será maior ou menor durante a remoção do ruído. Nos testes de remoção de ruído e no pré-processamento de imagens multiespectrais (vide Capítulo 6), foram utilizados os valores N=2 (janelas unidimensionais vertical e horizontal de tamanho 5) e J=1. Em [49], os autores usaram os valores N=3 e J=1.

O algoritmo leva em conta que, para um vetor ordenado V, com 2N+1 elementos (níveis de cinza da janela unidimensional arranjados em ordem crescente) o i-ésimo elemento $V_{(i)}$ é sempre menor ou igual à média dos elementos se $i<N$. Se $i>N$, o i-ésimo elemento será maior ou igual à média. Este fato é ilustrado na Figura 2.10. As duas curvas representam as respostas das operações média e mediana em uma convolução com janelas unidimensionais sobre uma linha da imagem. A comparação realizada pelo algoritmo combina a capacidade de

suavização do filtro da média com a capacidade de realce de bordas do filtro da mediana.

Para a janela de imagem da Figura 2.3.b e com os parâmetros utilizados nos testes (N=2 e J=1), o nível de cinza do pixel central (80) teria seu valor substituído por 31.

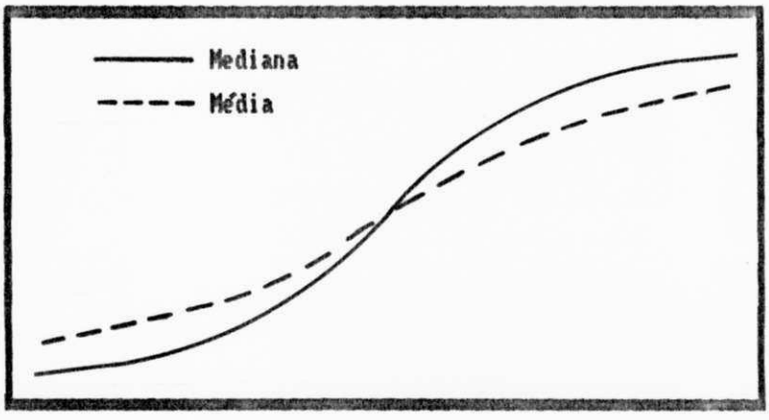

Figura 2.10 Comparação entre os efeitos das operações média e mediana

2.2.11 Suavização Logarítmica

Esta técnica leva em conta o fato de que o olho humano é mais sensível à relação $\Delta I/I$ que à diferença entre intensidades ΔI [35]. O método considera uma janela WxW centrada no pixel candidato P, cujo nível de cinza é p, dividida em 8 vizinhanças triangulares, como na Figura 2.10. Calcula-se a média, m_k, e a variância, V_k, de cada vizinhança. O nível de cinza de P é modificado, de modo a ter uma componente da média de cada vizinhança inversamente proporcional à respectiva variância.

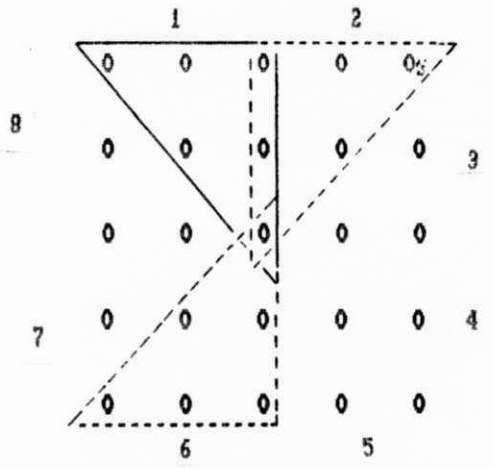

Figura 2.11 Vizinhanças para suavização logarítmica

O procedimento básico pode ser descrito como segue:

a. para cada vizinhança, a média, m_k, e a variância, V_k, são calculadas;

b. um coeficiente de peso, adaptativo, A_k, é definido como:

$$A_k = \min V / V_k$$

c. o nível de cinza, $\hat{p}$, do pixel candidato é substituído por:

$$\hat{p} = \sum (A_k^r \; m_k) / \sum A_k^r$$

onde r é um segundo fator de peso, constante, inteiro, positivo, definido pelo usuário.

Para r = 0, o nível de cinza $\hat{p}$ é dado por

$$\hat{p} = \sum m_k / 8 \ ,$$

que é equivalente à média simples de peso unitário. Se o valor de r cresce, os efeitos da média da região com menor homogeneidade decrescem. Se r tende a infinito, a técnica aproxima-se da de Nagao e Matsuyama, que atribui ao pixel candidato a média da região com menor variância. O valor 16 para r proporciona resultados satisfatórios [35] quanto à suavização com preservação de bordas e foi utilizado nos testes. Para a janela de imagem da Figura 2.3.b, o nível de cinza do pixel central (80) seria substituído por 40. Este valor é a média ponderada entre as médias de todas as vizinhanças triangulares, com peso máximo $A_{mmx} = 1.0$ para a janela 3 (a de menor variância).

2.2.12 Filtro Sigma Adaptativo

O filtro sigma convencional apresenta dois aspectos negativos. Como a variância do sinal de imagem em regiões planas e grandes é muito diferente da variância em regiões de fronteira, o fato de a faixa dos dois-sigmas ser fixa possibilita que alguns pontos ruidosos não sejam suavizados, principalmente nessas regiões planas e largas. Um outro problema é que a suavização é executada em toda a imagem, mesmo nas regiões onde o ruído é de baixa atividade e até imperceptível, provocando a distorção de texturas refinadas.

O filtro sigma adaptativo [101] tem o procedimento básico semelhante ao do sigma convencional, mas contorna o

primeiro problema utilizando uma faixa sigma S(i,j)=[p(i,j)-T,p(i,j)+T], que é função da variância do sinal em cada ponto:

$$T = a.V(i,j)$$

onde a é uma constante e V(i,j), é a variância calculada para uma janela WxW centrada no pixel candidato, P(i,j). Para garantir a inclusão de características do ruído e de informações de correlação espacial, a variância V(i,j) é calculada como segue:

$$V(i,j) = w_1.V(i,j)+w_2.V(i-1,j)+w_3.V(i-2,j)$$

onde V(i-1,j) e V(i-2,j) são as variâncias calculadas para as duas últimas janelas consideradas no processo de convolução; w_1, w_2 e w_3 são constantes de peso e $w_1=1$. No processamento dos W primeiros pontos da imagem não são consideradas variâncias anteriores e w=1.

O segundo problema é resolvido utilizando-se o processamento condicional. O filtro só é aplicado se a diferença entre o nível de cinza do pixel central e a média, m(i,j), da janela WxW for menor que o desvio padrão, D(i,j), na janela:

$$\hat{p}(i,j) = \begin{cases} \text{sigma } p(i,j), & \text{se } m(i,j)-p(i,j) > D(i,j) \\ p(i,j), & \text{em caso contrário} \end{cases}$$

onde $\hat{p}(i,j)$ é o novo nível de cinza de P, sigma p(i,j) é a média calculada com os elementos da janela cujos níveis de cinza estão dentro da faixa sigma, S(i,j); m(i,j) e D(i,j) são, respectivamente, a média e o desvio padrão calculados

considerando todos os elementos da janela. Nos testes realizados neste trabalho, foram usados os valores W=3, a=2 e w_1=1/3, propostos pelos autores em [101]. Considerando a janela exemplo 3x3 da Figura 2.3.a, como sendo a primeira da imagem e a=2, o pixel central teria seu nível de cinza, 70, substituído por 21. Este valor é a média de todos os elementos da janela, incluídos na faixa sigma em função do alto valor da variância (V(i,j)=358).

2.2.13 Filtro da Mediana Adaptativo

Este método, baseado nos filtros da mediana e da mediana ponderada, introduz o uso de pesos adaptativos no processo de ordenação [88]. A mediana ponderada de uma seqüência (X_i) é definida como a mediana simples de uma seqüência estendida , formada pelos elementos X_i, ordenados, só que repetidos w_i vezes, onde w_i são coeficientes de peso. Se w_1=4, w_2=2 e w_3=1, por exemplo, a mediana ponderada, Md_p, da seqüência (X_1,X_2,X_3), é dada por:

$$Md_p = \text{mediana}\,(X_1,X_1,X_1,X_1,X_2,X_2,X_3) = X_1$$

O filtro da mediana adaptativo considera a relação entre a variância, V, e a média, m, de uma janela WxW, para adaptar os pesos que serão usados na ponderação do processo de ordenação. Os pesos w(i,j) são dados pela expressão:

$$w(i,j) = [w((W+1)/2,(W+1)/2) - cdV/m]$$

onde c é uma constante, m e V são, respectivamente, a média e a variância calculados para os elementos da janela WxW, d

é a distância do ponto (i,j) para o centro da janela e w((W+1)/2,(W+1)/2) é um peso de referência, definido pelo usuário, para o pixel central da janela. O operador [x] considera o valor inteiro mais próximo de x, se x é positivo, ou zero, se x é negativo. Os valores de c, d, V, e m definem o compromisso entre as capacidades de remoção de ruído e de preservação de bordas. Em [88], foram usados os valores c=20, w((W+1)/2,(W+1)/2)=99 e W=9. Neste trabalho, foram utilizadas janelas 5x5 (W=5), e escolhidos experimentalmente os valores c=5 e w((W+1)/2,(W+1)/2)=20. O pixel central da janela de imagem da Figura 2.3.b teria seu nível de cinza substituído por 31, valor mediano (105º elemento) da seqüência de 209 elementos ordenada ponderadamente.

2.3 Conclusão

No decorrer deste Capítulo pôde-se observar que técnicas de suavização têm-se desenvolvido muito rapidamente nos últimos dez anos.

É possível notar pela descrição dos processos envolvidos nos filtros estudados algumas tendências quanto à eficiência e aspectos de implementação. A medida que as técnicas tornam-se adaptativas, levando em conta cada vez mais características da imagem e buscando reduzir o ruído sem provocar distorções, tornam-se mais complexas e comprometem a eficiência computacional. Alguns algoritmos que utilizam o critério de vizinhança seletiva, por exemplo, requerem um tempo de processamento relativamente grande, se implementados em microcomputadores de 16 bits.

A quantidade e a variedade de técnicas abordadas possibilitam uma razoável flexibilidade para o usuário em aplicações de realce e pré-processamento.

3 TÉCNICAS DE DETEÇÃO DE BORDAS

As técnicas de deteção de bordas são apresentadas neste Capítulo e uma revisão bibliográfica é realizada. Os métodos selecionados para estudo e implementação são descritos.

3.1 Deteção de Bordas

Uma borda é uma mudança ou descontinuidade local na luminosidade de uma imagem [7],[21]. Pode ser uma borda de luminosidade ("luminance edge"); borda de textura ("texture edge"), descontinuidade em regiões com certos padrões de homogeneidade; e borda de cores ("tristimulus" ou "color edge"), descontinuidade em imagens coloridas. Nesta seção, será dado enfoque às bordas de luminosidade.

A importância da deteção de bordas decorre do fato de, no processo de análise visual, as funções neurológicas envolvidas terem uma resposta máxima às variações dos níveis de cinza, que caracterizam a presença de uma fronteira na imagem [121].

Entre as técnicas de deteção de bordas encontram-se os métodos de realce/limiar, de adaptação de bordas e híbridos.

Os métodos de realce/limiar consistem na aplicação de operadores gradientes ou máscaras OP(m,n) a uma imagem I(m,n), gerando um conjunto de funções gradientes

$$GR(m,n) = I(m,n) * OP(m,n)$$

onde * representa uma convolução espacial. Uma imagem gradiente realçada AR(m,n) pode ser obtida a partir de uma combinação linear ou não-linear das funções gradientes. A imagem AR(m,n) pode ser representada como uma imagem do tipo gradiente analógico, se as magnitudes dos gradientes forem mostradas como níveis de cinza. Um mapa binário de bordas MP(m,n) pode ser obtido através da aplicação de um limiar L ("thresholding") à imagem gradiente,

$$MP(m,n) = 1 \quad , \text{ se } AR(m,n) > L;$$
$$MP(m,n) = 0 \quad , \text{ caso contrário.}$$

onde um valor superior ao limiar indica a existência de bordas.

Em função da precisão com que detectam a direção das bordas os operadores gradiente podem ser classificados como diferenciais ou direcionais.

Os operadores diferenciais caracterizam-se por realizarem uma diferenciação discreta da imagem para produzir a imagem gradiente, gerando pouca ou nenhuma informação sobre a direção das bordas detectadas. Roberts [7] apresentou um método simples de diferenciação bidimensional baseado nas diferenças cruzadas em janelas 2x2.

Sobel [4] e Prewitt [24] sugeriram o cálculo do gradiente, nas direções horizontal e vertical, através da

combinação de duas máscaras. Estas máscaras aproximam as derivadas parciais nas direções ortogonais x e y.

Kanopoulos et alii [125] propuseram um aprimoramento do operador de Sobel, incluindo duas máscaras para cálculo do gradiente nas direções diagonais. Neste método, a magnitude final do gradiente é definida como sendo a máxima dentre as quatro máscaras utilizadas. A direção é definida como sendo a da máscara de maior resposta.

Os operadores laplacianos [7] não têm resposta polarizada em direções ideais e são geralmente utilizados quando não existe a preocupação com problemas de orientação na imagem.

A busca de uma aproximação discreta de bordas ideais em várias direções caracteriza um operador direcional. Alguns exemplos são as máscaras direcionais ("compass") introduzidas por Prewitt [52] e Kirsch [126], e as máscaras simples de três níveis e cinco níveis apresentadas por Robinson [127].

Estes métodos aplicam as oito máscaras a cada vizinhança, onde cada máscara aproxima a orientação de bordas ideais a ângulos de 45 graus. A magnitude do gradiente é obtida a partir da resposta mais forte e a direção é dada pela orientação da máscara que a gerou. Em [128], Overington e Greenway apresentaram um método que aumenta a precisão na determinação dessas direções reduzindo os níveis de quantização dos ângulos calculados.

Métodos de adaptação de bordas consistem na adaptação de um modelo de borda ideal, uma função rampa ou um

determinado padrão de impulso bidimensional, em algumas regiões da imagem. Nestas técnicas, são sempre empregados alguns procedimentos para medir o nível de aproximação do processo de adaptação, geralmente a partir de uma indicação de um erro médio. De acordo com a grandeza deste erro, assume-se em um dado local da imagem, a existência de uma borda com os mesmos parâmetros do modelo ideal utilizado. Haralick [122],[123],[129],[130], apresentou métodos que se baseiam em modelo de facetas para imagem. Nevatia [131], Abramatic [132], Hummel [133], Morgenthaler [134], Shipman et alii [135] e Hueckel [136] sugeriram algoritmos baseados na adaptação de características da imagem para deteção de bordas.

Detetores de bordas híbridos, que se baseiam na combinação de vários operadores-gradiente diferentes aplicados em uma mesma máscara, também foram apresentados por Nuevo et alii [82]. As saídas dos operadores são submetidas a um processo seletivo baseado no filtro da mediana. Pitas e Venetsanopoulos [137] sugeriram algoritmos que utilizam a resposta de filtros não-lineares, como o filtro da ordem e média não-linear.

Vandick [138], de Souza [139], Chen et alii [140], Mascarenhas e Prado [141], e Yakimosky [142] apresentaram métodos que utilizam modelos estatísticos para os objetos, o fundo e o ruído, juntamente com procedimentos estimativos para realçar bordas e definir os contornos dos objetos.

Para deteção de bordas de textura, Rosenfeld e seus colaboradores sugeriram algoritmos que utilizam operadores locais sensíveis a algumas propriedades texturais [146]-[147]. Nestes algoritmos, são considerados os possíveis

padrões de homogeneidade presentes na imagem. Em [148], Garibotto utiliza uma função de auto-covariância associada a um fator de ponderação para garantir uma boa resposta em regiões de baixa atividade que contenham bastante informação de bordas. Kashyap e Eom [149],[150] propuseram um algoritmo baseado em um modelo de alta correlação, que permite a caracterização de texturas e sua diferenciação, utilizando um número pequeno de parâmetros. Outros operadores locais, com diferentes critérios para deteção de bordas, foram estudados por Thompson [151] e Davis e Mitiche [152],[153].

Um método para deteção de bordas que considera a resposta do sistema visual humano foi sugerido por Chanda et alii [35]. Neste algoritmo, é utilizado um limiar variante no espaço e dependente de uma função logarítmica do comportamento dos níveis de cinza da vizinhança. Clark [154],[155] e Ritcher e Ullman [156] também utilizaram princípios fisiológicos e psicofísicos do sistema visual humano para eliminar bordas espúrias geradas pela maioria dos detetores convencionais, aumentando, assim, a precisão das linhas de fronteiras encontradas.

Devido ao comportamento irregular das descontinuidades de luminância nas imagens, muito comumente, pontos vizinhos são classificados como pontos de fronteira. Isto faz com que os contornos definidos nos mapas de bordas se tornem largos. Algoritmos desenvolvidos com o propósito de melhorar a definição (tornar mais finas) das bordas geradas em processos de deteção foram propostos por Rosenfeld [157],[158], Eberlein [159], Kasvand [160], Weiss [161] e Robinson [127].

Alguns autores têm empregado modelos aleatórios para representar um pixel e classificá-lo como sendo de uma re-

gião de fronteira ou não. Pitas [162] propôs o uso de modelos markovianos. Eichel e Delp [163] e Zhou et alii [164] utilizaram modelos auto-regressivos causais e derivadas de segunda ordem para gerar campos de contorno. Outros procedimentos desenvolvidos neste sentido foram apresentados por Perkins [165], Pavlidis [166], Nevatia e Babu [167], Ikonomopoulos [168], Favre e Keller [169], Gil et alii [170] e Lacroix [171].

Trabalhos com caráter de revisão sobre técnicas de deteção de bordas foram apresentados por Pitas e Venetsanopoulos [137], [172], Lacroix [173], Torre e Poggio [174], Lyvers e Mitchell [175] e Araújo [40].

3.2 Filtros Passa-altas Implementados

Foram implementados os seguintes filtros:

operadores diferenciais:

de Roberts	ROBERTS
de Sobel	SOBEL
de Prewitt	PREWITTDF
laplacianos (03)	LAPLAC1
	LAPLAC2
	LAPLAC3
realçador de linhas finas	RLIN

operadores direcionais:

de Prewitt	PREWITTDR

de Kirsch	KIRSCH
de Robinson 3 níveis	ROBINSON3
de Robinson 5 níveis	ROBINSON5

técnicas híbridas:

detetor de bordas híbrido	HBDET

3.2.1 Operadores Diferenciais

Estes operadores, geralmente, são de fácil implementação. Consistem de máscaras com pesos definidos, de forma a produzirem a diferenciação discreta em cada ponto da imagem.

A Figura 3.1 apresenta as máscaras correspondentes ao operador de Roberts que, combinadas, geram um ponto de gradiente a partir das diferenças cruzadas em um arranjo 2x2 [7].

$$M1 = \begin{matrix} 1 & 0 \\ 0 & -1 \end{matrix} \qquad M2 = \begin{matrix} 0 & 1 \\ -1 & 0 \end{matrix}$$

Figura 3.1 Máscaras do Operador de Roberts.

O operador de Roberts pode ser representado pela equação:

$$GR(m,n) = ([I(m,n)-I(m+1,n+1)]^2+[I(m,n+1)-I(m+1,n)]^2)^{1/2}$$

Outras aproximações são:

$$GR(m,n) = |I(m,n)-I(m+1,n+1)| + |I(m,n+1)-I(m+1,n)|$$

e

$$GR(m,n) = máx(|I(m,n)-I(m+1,n+1)| , |I(m,n+1)-I(m+1,n)|)$$

As máscaras sugeridas por Sobel [24] e Prewitt [7] são mostradas na Figura 3.2. Estas máscaras fazem uma aproximação das derivadas parciais nas direções vertical e horizontal. Combinações lineares e não-lineares da magnitude das saídas das duas máscaras ortogonais geram o gradiente da imagem em cada ponto. Uma operação tangente inversa entre as respostas das máscaras determina a direção da borda detetada.

$$Mx = \begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix} \qquad Mx = \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$$

$$My = \begin{matrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{matrix} \qquad My = \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix}$$

(a) (b)

Figura 3.2 Máscaras dos operadores. (a) Sobel e (b) Prewitt.

Os operadores laplacianos caracterizam-se pelo efeito de isotropia. Não há polarização da resposta em função da direção da borda. As máscaras correspondentes às variações implementadas são mostradas na Figura 3.13.

$$L1 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad L2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad L3 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Figura 3.3 Operadores laplacianos

Em muitas aplicações, linhas e detalhes tênues presentes na imagem podem não ser aproveitados em tarefas de segmentação e classificação. O realçador de linhas finas, proposto por Lester et alii [80], é uma variação de um operador laplaciano que atribui ao pixel central de uma janela 3x3 o seu nível de cinza original p adicionado à diferença entre p e a média dos oito vizinhos imediatos:

$$\hat{p} = p+(p- \sum p(i)/8)$$

onde $i = 1,2,..,9$ e $i \neq 5$.

Nos pontos pertencentes a detalhes tênues e linhas finas, a resposta do gradiente é máxima e o nível de cinza final é proporcionalmente alterado. Se a resposta do gradiente é negativa, a vizinhança é clara em relação ao nível de cinza do pixel central e, depois de transformado, este pixel tornar-se-á ainda mais escuro, produzindo realce. Se a resposta do gradiente for positiva, ocorrerá o contrário.

3.2.2 Operadores Direcionais

Os operadores diferenciais são métodos que caracterizam-se por gerarem respostas de gradiente otimizadas em função da direção da borda encontrada. Na Figura 3.4 são apresentadas as máscaras correspondentes aos operadores direcionais de Kirsch, Prewitt e Robinson de 3 e 5 níveis. Pode-se observar que cada operador consiste de oito máscaras

Direção Borda	Prewitt	Kirsch	Robinson 3-níveis	Robinson 5-níveis
Norte	1 1 1 1 -2 1 -1 -1 -1	5 5 5 -3 0 -3 -3 -3 -3	1 1 1 0 0 0 -1 -1 -1	1 2 1 0 0 0 -1 -2 -1
Noroeste	1 1 1 1 -2 -1 1 -1 -1	5 5 -3 5 0 -3 -3 -3 -3	1 1 0 1 0 -1 0 -1 -1	2 1 0 1 0 -1 0 -1 -2
Oeste	1 1 -1 1 -2 -1 1 1 -1	5 -3 -3 5 0 -3 5 -3 -3	1 0 -1 1 0 -1 1 0 -1	1 0 -1 2 0 -2 1 0 -1
Sudoeste	1 -1 -1 1 -2 -1 1 1 1	-3 -3 -3 5 0 -3 5 5 -3	0 -1 -1 1 0 -1 1 1 0	0 -1 -2 1 0 -1 2 1 0
Sul	-1 -1 -1 1 -2 1 1 1 1	-3 -3 -3 -3 0 -3 5 5 5	-1 -1 -1 0 0 0 1 1 1	-1 -2 -1 0 0 0 1 2 1
Sudoeste	-1 -1 1 -1 -2 1 1 1 1	-3 -3 -3 -3 0 5 -3 5 5	-1 -1 0 -1 0 1 0 1 1	-2 -1 0 -1 0 1 0 1 2
Leste	-1 1 1 -1 -2 1 -1 1 1	-3 -3 5 -3 0 5 -3 -3 5	-1 0 1 -1 0 1 -1 0 1	-1 0 1 -2 0 2 -1 0 1
Nordeste	1 1 1 -1 -2 1 -1 -1 -1	-3 5 5 -3 0 5 -3 -3 -3	0 1 1 -1 0 1 -1 -1 0	0 1 2 -1 0 1 -2 -1 0

Figura 3.4 Máscaras direcionais

cujas respostas são polarizadas em oito direções ideais (direções cardeais). Em cada ponto, as 8 máscaras são aplicadas e a resposta máxima é selecionada para determinar o gradiente e a direção da borda encontrada.

3.2.3 Técnicas Híbridas

O detetor de bordas híbrido introduzido por Neuvo et alii [82] tem a estrutura mostrada na Figura 3.5. H1, H2 e H3 são subfiltros lineares ou não, e Med é o filtro da ordem 2 ou mediana para vetores de 3 elementos. As máscaras correspondentes aos subfiltros são apresentadas na Figura 3.6.

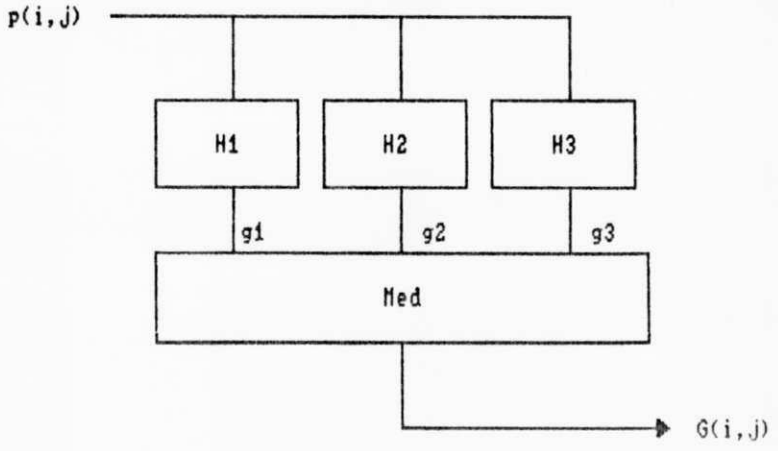

Figura 3.5 Estrutura do detetor de bordas híbrido

A resposta do gradiente no ponto (i,j) é dada por:

$$G(i,j) = \text{Med}\,[g_1(i,j),\ g_2(i,j),\ g_3(i,j)]$$

onde g_k é a resposta do k-ésimo subfiltro.

A resposta em magnitude de cada subfiltro é dada por:

$$g_k(i,j)^2 = g_{kx}(i,j)^2 + g_{ky}(i,j)^2$$

onde $g_{kx}(i,j)$ e $g_{ky}(i,j)$ são as respostas das máscaras polarizadas nas direções x e y, respectivamente, para cada subfiltro.

A imposição da resposta nula para o subfiltro H3 ($g_3(i,j)=0$), faz com que a operação mediana selecione a menor resposta entre g_1 e g_2. Isto aumenta a capacidade do operador híbrido de atenuar o ruído enquanto deteta bordas, em comparação com os detetores linares.

$$H1x = \begin{matrix} 1/2 & 1/2 & -1 \\ 1/2 & 1/2 & -1 \\ 1/2 & 1/2 & -1 \end{matrix} \qquad H2x = \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix} \qquad H3x = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

$$H1y = \begin{matrix} 1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 \\ -1 & -1 & -1 \end{matrix} \qquad H2y = \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix} \qquad H3y = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

Figura 3.6 Máscaras do detetor de bordas híbrido

3.3 Conclusão

As técnicas de deteção de bordas em geral são muito mais simples do que as de suavização espacial e, conseqüentemente mais rápidas. A revisão bibliográfica mostra a tendência dos métodos de incorporarem

características adaptativas.

Além dos filtros mencionados nos Capítulos 2 e 3, foram implementadas rotinas para manipulação de arquivos de imagem na unidade de visualização e em disco, rotinas para geração de ruído gaussiano aditivo e impulsivo, e rotinas para medição do efeito do ruído e dos filtros nas imagens processadas. Estas rotinas são:

display de imagens	DISCUVI
armazenamento de imagens em disco	UVIDISC
ruído gaussiano	RDGAUSS
ruído impulsivo	RDIMP
erro médio quadrático	ERQUAD
desvio padrão	DESV
perfil de varredura	PERF
histograma	HIST

4. O PACOTE DE SOFTWARE

Este capítulo apresenta o pacote de "software" FILTRIX e descreve suas funções, características especiais, interface homem-máquina, aspectos envolvidos com a implementação e o "hardware" empregado.

4.1 As Tendências

Tarefas de processamento de imagens normalmente envolvem a manipulação de uma quantidade muito grande de informação e, algumas vezes, exige uma arquitetura dedicada. Máquinas sofisticadas têm sido desenvolvidas para o processamento paralelo e em tempo real [176]-[207] com "softwares" dedicados. Com o advento da microeletrônica, uma alternativa muito procurada tem sido o desenvolvimento de circuitos integrados de aplicação específica [125],[208]-[216] para processamento de imagens. Esta, infelizmente, ainda é uma opção viável apenas para poucos países detentores da tecnologia VLSI. Assim, ambos, "hardware" e "software", tornam-se caros e de difícil acesso. Trabalhos recentes têm mostrado a busca de uma maior flexibilidade dos pacotes de "software", para torná-los capazes de operar em um número cada vez maior de configurações diferentes [217]-[232]. Os sistemas baseados em microprocessadores surgem, neste contexto, como uma alternativa econômica

para a criação de uma grande quantidade de estações de tratamento de imagens em universidades e empresas, aumentando o espaço para o desenvolvimento de novas técnicas e produtos nesta área do conhecimento. "Softwares" concebidos para este tipo de equipamento podem ter bom nível de transportabilidade [224],[227],[230],[233],[234]. O pacote de "software" FILTRIX foi desenvolvido em um ambiente baseado em microcomputador e consistte de uma biblioteca de algoritmos de filtragem espacial e de métodos estatísticos e gráficos destinados à análise das imagens processadas.

4.2 O "Hardware" Utilizado

O trabalho foi desenvolvido no Sistema de Tratamento de Imagens SITIM-110 (Sistema de Tratamento de Imagens) [235], do Laboratório Regional de Sensoriamento Remoto de Campina Grande, LASR-CG. Este equipamento faz parte de uma família de equipamentos de fabicação nacional, dedicados ao processamento digital de imagens.

O SITIM-110 é mostrado no diagrama de blocos da Figura 4.1 e possui as seguintes características:

a. memória de imagens 4 x 512 x 512 x 8 bits;
b. palete de cores de 256 cores;
c. monitor de imagem de média resolução (512x512);
d. microprocessador de 16 bits;
e. sistema operacional ANALIX e CPU Intel 8086;
f. coprocessador aritmético 8087, 256 Kbytes de memória, unidade de disco flexível, monitor colorido, teclado, impressora matricial, disco rí-

gido de 10 Mbytes, unidade de fita para armazenamento em massa e plotadora.

A unidade de visualização é composta por 3 canais que permitem a visualização simultânea de até 3 imagens de 512x512 pontos e um canal gráfico com a mesma resolução. O sistema dispõe ainda de um pacote, chamado SITIM, com funções básicas de tratamento de imagens e dedicado à classificação de imagens de satélite.

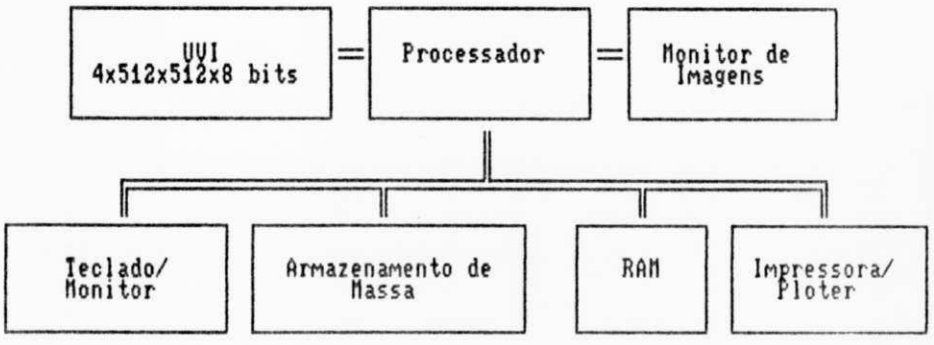

Figura 4.1. Diagrama de blocos do SITIM-110

4.3 A Biblioteca de Filtros Espaciais

A biblioteca foi implementada em linguagem C por ser esta uma linguagem versátil e apropriada para algoritmos de processamento digital de imagens [236]-[241]. O pacote foi concebido de forma a ser utilizado em duas aplicações básicas de processamento de imagens:

a. tarefas de realce e remoção de ruído;
b. pré-processamento em tarefas de classificação de imagens.

Da maneira como foi configurada, dispõe de 5 funções principais:

a. suavização espacial;
b. detecção de bordas;
c. estatísticas e medições;
d. manipulação de arquivos de imagem;
e. geração de ruído.

As funções de suavização espacial e detecção de bordas abrigam os programas correspondentes às técnicas mencionadas nas Seções 2.2 e 3.2.

A função estatísticas e medições compreende os programas **DESV** (desvio padrão), **ERQUAD** (erro médio quadrático), **PERF** (perfil de linha) e **HIST** (histograma). A função de manipulação de arquivos de imagens é composta pelas rotinas de salvamento (que transferem uma imagem da unidade de visualização para um arquivo em disco), **UVIDISC**, e de display de imagens (que carrega na tela uma imagem armazenada em disco), **DISCUVI**. No Apêndice A, são apresentados os códigos em linguagem C dos programas que compõem o pacote.

Para a interface homem-máquina, foi buscada, tanto quanto possível, uma padronização com o sistema SITIM [242]. Uma função é chamada pela seleção de sub-áreas ativas da tela. Os menus ou níveis de funções são apresentados na forma de janelas. Um menu principal em português (Fig. 4.2), apresenta as 5 funções básicas. Menus secundários (Figs. 4.3, 4.4 e 4.5), apresentam os filtros e utili-

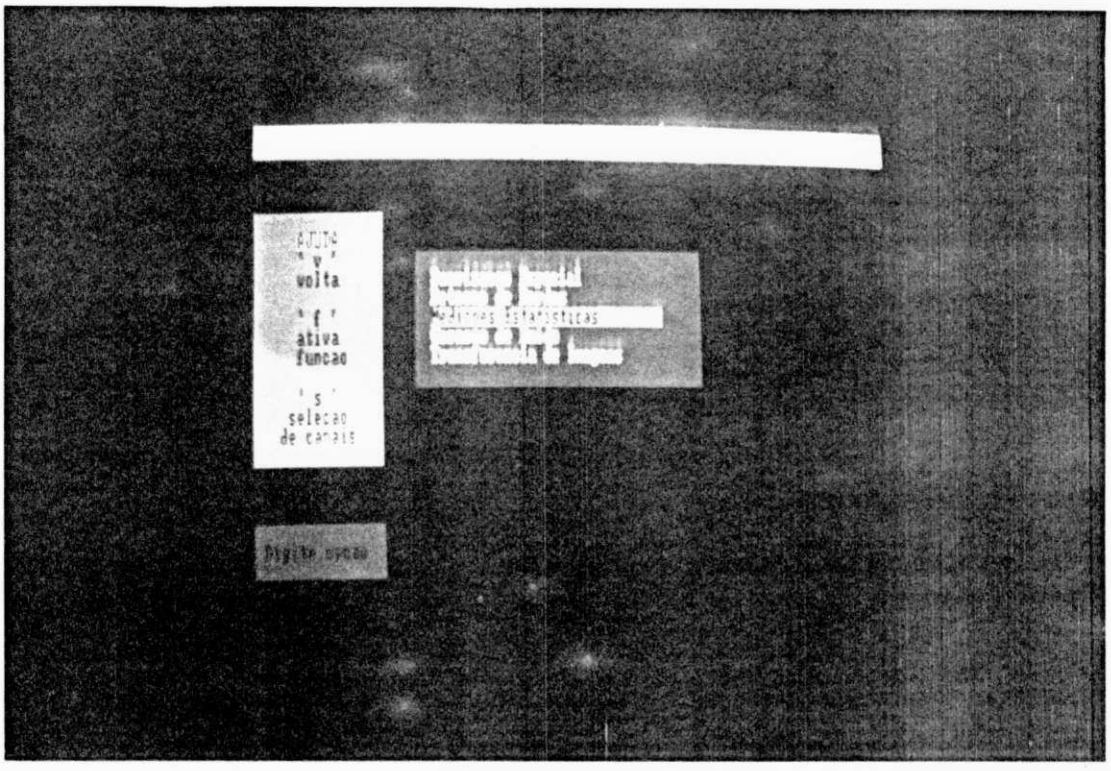

Figura 4.2 Menu Principal

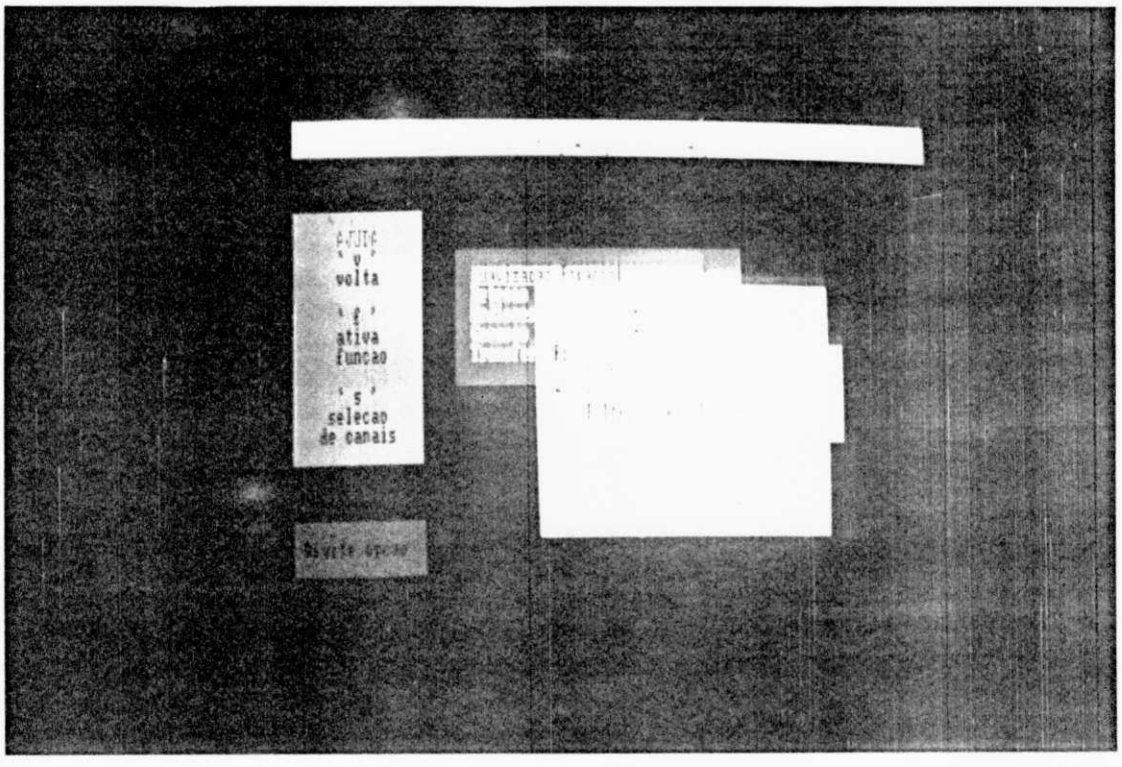

Figura 4.3 Menu de filtros e utilitários
(suavização espacial)

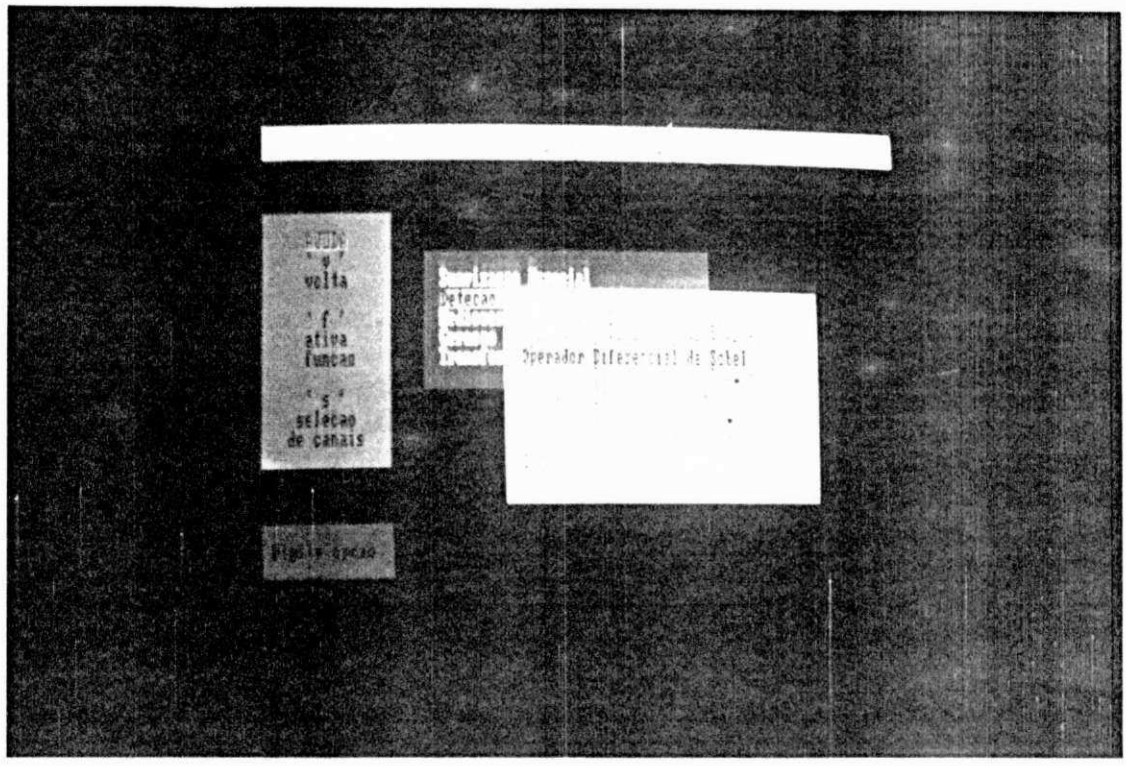

Figura 4.4 Menu de filtros e utilitários (detecção de bordas)

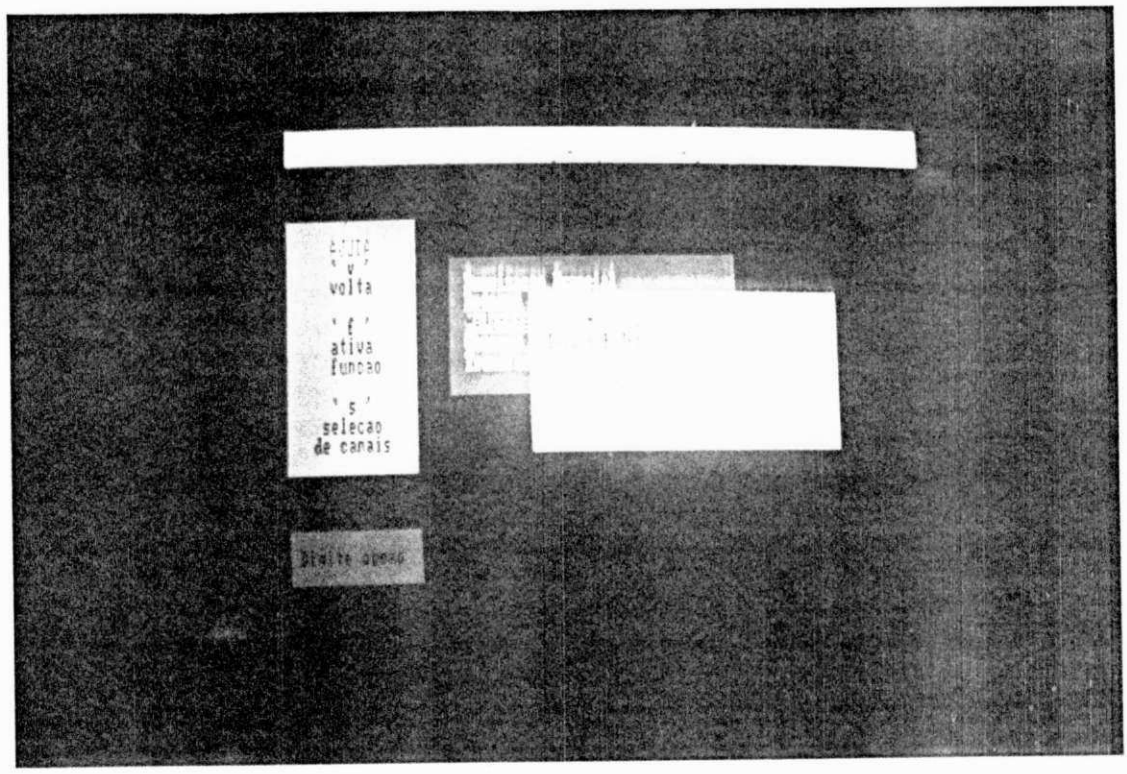

Figura 4.5 Menu de filtros e utilitários (estatísticas)

tários disponíveis . No último nível de janela, é realizada a passagem dos parâmetros dos algoritmos. Em todos os níveis uma área de "help" situada no canto superior esquerdo do vídeo orienta a seleção de funções e a passagem de parâmetros (Fig. 4.6).

Em algumas aplicações, há o interesse em processar apenas uma pequena parte da imagem. As implementações propostas para os filtros e os utilitários de FILTRIX prevêem a capacidade de processamento regional seletivo que permite ao usuário, através de um cursor retangular (Fig. 4.7), selecionar uma janela de qualquer tamanho e em qualquer posição na imagem, para processamento. Em alguns casos, este recurso garante os resultados da análise da imagem com menor tempo de processamento.

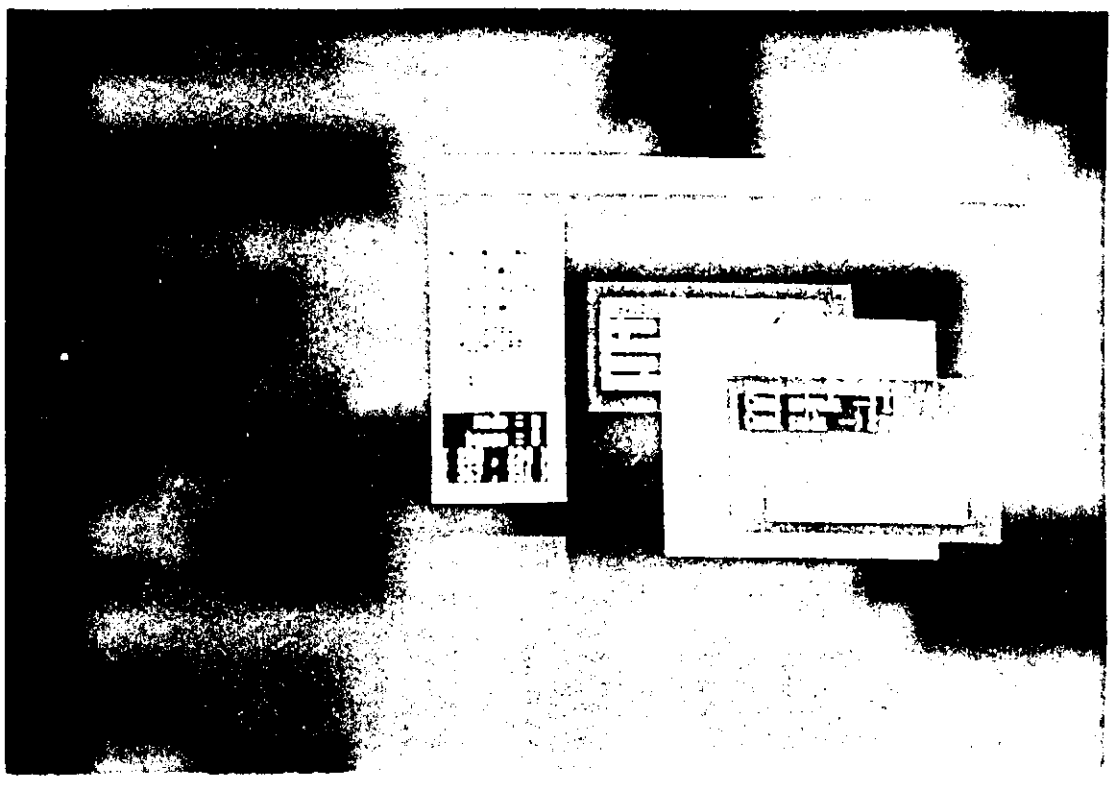

Figura 4.6 Passagem de parâmetros

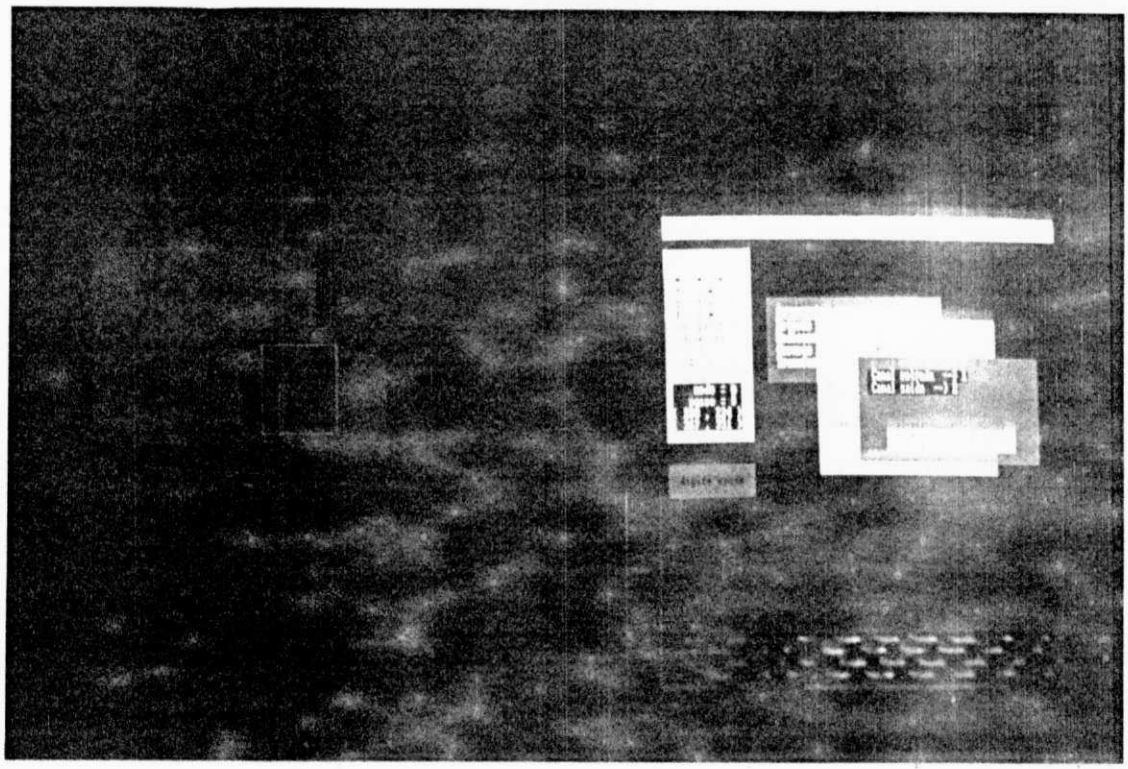

Figura 4.7 Seleção de janelas de imagem

As coordenadas dos pixels das imagens são representadas de acordo com o sistema de referência da Figura 4.8. O limite da resolução da tela em 512 pontos não impede que imagens maiores possam ser processadas. Nestes casos, a imagem pode ser dividida em quadros e cada quadro

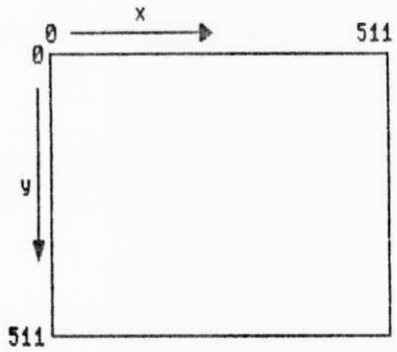

Figura 4.8 Sistema de coordenadas das imagens

pode ser carregado e processado seqüencialmente. Este procedimento é apresentado por Tamura e Mori [223] como uma alternativa para o tratamento de imagens de satélite que, em geral, contêm mais de um Mbyte de informação.

A reconstrução da imagem resultante através da concatenação dos quadros processados não apresenta falhas de continuidade, devido à capacidade dos programas de incluírem informação excedente, como será descrito adiante.

O modo de processamento utilizado é o modo de processamento em tela ou monitorizado. A imagem é carregada em um dos canais da unidade de visualização de imagens e o resultado da filtragem é carregado em um outro canal selecionado pelo usuário. Isto permite a visualização simultânea da imagem original e de até duas imagens resultantes de várias etapas de processamento, possibilitando comparações após a filtragem e durante a filtragem.

O pacote está adaptado para o SITIM-110 e sistema operacional ANALIX. Neste ambiente, sua portabilidade está basicamente condicionada à configuração mínima do equipamento utilizado. O sistema deve dispor pelo menos de um microprocessador de 16 bits e uma unidade de visualização. Atualmente, encontra-se em fase final de desenvolvimento uma versão do FILTRIX para operação em ambiente DOS, dedicada ao SITIM-150.

4.4 Metodologia de Acesso aos Dados

No processamento de uma imagem digital, a meto-

dologia de acesso aos pixels para as operações, em geral, depende do tipo da máquina utilizada. Máquinas de arquitetura paralela permitem até o acesso simultâneo de todos os pontos da imagem. Computadores convencionais de grande porte, com memória virtual, podem comportar várias imagens de 512x512 pontos e "softwares" de propósito geral manipulam diretamente imagens inteiras na forma de arranjos bidimensionais. Em se tratando de "softwares" de aplicação específica e que serão utilizados em microcomputadores, é interessante que esta metodologia seja otimizada.

Isto pode ser feito, se for levada em consideração alguma particularidade da técnica implementada. No caso das técnicas de filtragem espacial estudadas neste trabalho, o procedimento básico consiste na convolução de uma máscara de tamanho definido, LxL, sobre todos os pontos de uma imagem de tamanho NxM. Assim, foi proposto e utilizado o seguinte método:

1. as primeiras L linhas da imagem são lidas na UVI e armazenadas em L vetores unidimensionais de tamanho M, formando um "buffer" de dimensão LxM;

2. a máscara é convoluída sobre o "buffer", transformando os pixels centrais de cada vizinhança LxL até gerar uma linha processada;

3. a linha processada é escrita na janela e canal da UVI selecionados pelo usuário;

4. o conteúdo do "buffer" é deslocado, transferindo-se o conteúdo do vetor i para o vetor i-1 :

$$vetor_{i-1}(k) = vetor_{i}(k)$$

onde : i = 2, ... L

k = 1, ... M

5. uma nova linha é lida na UVI e armazenada no vetor i = L;

6. volta-se ao passo 2.

Desta forma, apenas L linhas da imagem sob processamento ocupam permanentemente a memória, que, nos microcomputadores comuns no mercado, é geralmente muito pequena (inferior a 640 Kbytes) para comportar, além de programas e utilitários ativos, o arquivo de imagem.

Como mostra a Figura 4.9, a imagem resultante passa a ter dimensões N-L+1 x M-L+1. Para garantir o processamento regional seletivo sem falhas na continuidade da

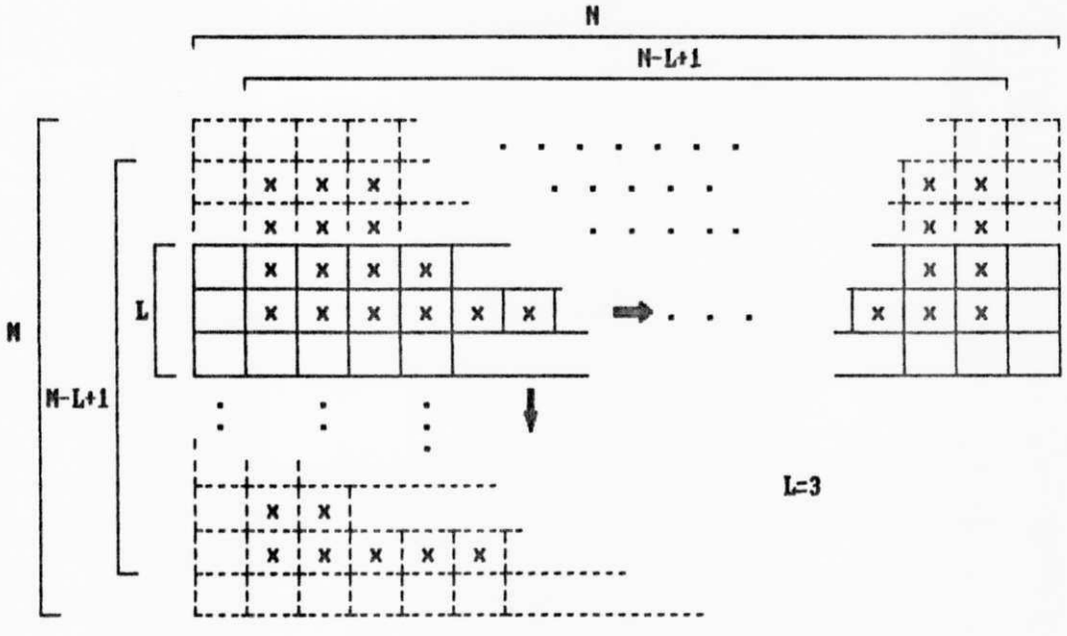

Figura 4.9 Utilização de pequenos "buffers"

imagem reconstruída, os algoritmos utilizam no processamento a inclusão de informação excedente. Trata-se da leitura das (L-1)/2 linhas imediatamente acima e abaixo da imagem a ser processada, bem como das (L-1)/2 colunas imediatamente à esquerda e à direita (Figura 4.10). Deste modo, todas as linhas e colunas da imagem NxM resultante são processadas com informação da imagem original completa.

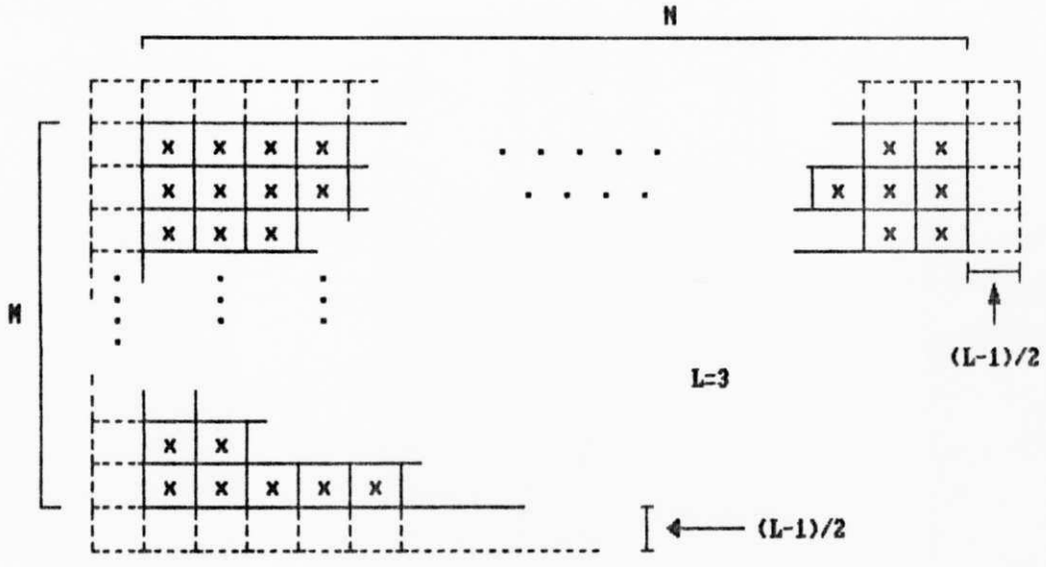

Figura 4.10 Inclusão de informação excedente

4.5 Conclusão

A biblioteca está adaptada para o SITIM-110, Sistema Operacional Analix e configuração mínima de um microcomputador de 16 bits e uma unidade de visualização de imagens. Uma versão do FILTRIX para operação em ambiente DOS e dedicada ao SITIM-150 [242] está sendo implementada.

O método proposto para acesso aos pixels da imagem a ser processada apresenta-se como uma solução econômica para o uso da memória em microcomputadores de pequeno porte.

A inclusão de informação excedente no processo de leitura proporciona o processamento particionado da imagem sem perdas de informação e sem falhas de continuidade da imagem resultante.

Um outro pacote para filtragem espacial de imagens, denominado PCFILT, está sendo desenvolvido [243]. Este "software", foi projetado para instalação em qualquer microcomputador compatível com o padrão IBM-XT e executa todo o processamento em disco rígido, dispensando a unidade de visualização ou qualquer arquitetura adicional para processamento de imagens. Neste caso, as operações são realizadas com uma monitoração simulada no monitor do computador. As imagens processadas são armazenadas em disco e podem ser transportadas para um outro equipamento dotado de unidade de visualização. Como resultado das experiências de implementação de FILTRIX e de um estudo das limitações de resolução espacial e luminosa, bem como dos métodos correntes na literatura, foi concebida uma ferramenta de "software" dedicada à impressão de imagens multiespectrais em dispositivos matriciais [244].

5. ESTUDO COMPARATIVO

Neste capítulo, são descritos vários testes destinados à medição do desempenho dos filtros estudados [41]. São utilizadas ferramentas estatísticas e gráficas, correntemente utilizadas em processamento digital de imagens, para avaliação dos efeitos provocados pelos algoritmos sobre o sinal de imagem com ruído ou sem ruído. Todos os experimentos foram realizados utilizando recursos disponíveis em FILTRIX.

5.1 Descrição dos Testes

O estudo comparativo aqui realizado consiste de uma continuação e atualização de trabalho realizado por Araújo [40], através da inclusão nos testes de técnicas adaptativas, recentemente desenvolvidas. Além disso, é propósito ainda desta etapa do trabalho ilustrar o potencial da ferramenta de software implementada, FILTRIX, através de uma avaliação da eficiência das técnicas disponíveis na mesma. No caso dos filtros passa-baixas, o objetivo foi a avaliação do seu desempenho quanto às capacidades de remoção de ruído, preservação de bordas, tipo impulso e tipo degrau, imunidade de distorção de formas e aguçamento de bordas tipo rampa. Para análise do comportamento dos filtros passa-altas, foi considerada a sensibilidade dos algoritmos na geração de

imagens-gradiente.

Para o estudo das técnicas de suavização, foram geradas três imagens sintéticas. A primeira, denominada CIRC, consiste de um círculo de raio 50, com nível de cinza constante e igual a 150, inserido em um quadro de 128x128 pixels, com nível de cinza igual a 50. Esta imagem foi considerada apropriada para os testes, por apresentar um mesmo padrão de bordas em todas as direções, e também por ter sido usada no trabalho de Araújo [40]. A segunda imagem, denominada CIRCRD, foi gerada a partir da imagem CIRC, aplicando-se-lhe ruído aditivo, com distribuição gaussiana, com média zero e desvio padrão 20,0. Finalmente, a terceira imagem, NUBRD, foi criada, nublando-se CIRC através de duas iterações do filtro da média 5x5, e aplicando-se-lhe ruído gaussiano com média zero e desvio padrão 10,0. Estes tipos de imagens-teste têm sido utilizados por diversos autores [40],[92].

A análise dos filtros passa-baixas compreendeu dois testes. O primeiro consistiu na aplicação sucessiva dos algoritmos à imagem CIRCRD por cinco vezes. O objetivo foi observar sua atuação na primeira aplicação e os efeitos produzidos pela iteração. Nesta fase, foram testados os filtros MEDIA 3x3, MEDIANA 3x3, KVIZ 3x3 com K=6, FACETAS 5x5, SSDA 5x5, SSVAR 5x5, SIGMA 5x5 com K=2 e desvios padrões iguais a 20,0, 10,0, 5,0, 2,5 e 1,25 (desvios padrões reduzidos à metade a cada iteração), ORDAP 5x5 com J=1, SIGMADP 3x3 com a=2 e w_1=1/3 e MEDNADP 5x5 com c=5 e w((W+1)/2,(W+1)/2)=20,0. Estes algoritmos representam uma amostra considerável das técnicas disponíveis em FILTRIX e dos métodos correntemente utilizados em processamento digital de imagens. Como critérios de avaliação foram empregadas medidas do desvio padrão em área homogênea e do erro médio

quadrático e perfis de varredura. Estes critérios foram utilizados por Araújo [40] e aqui repetidos com a finalidade de permitir a comparações entre os resultados obtidos por Araújo e neste trabalho. Puderam ser analizados os seguintes parâmetros: remoção de ruído, preservação de bordas e imunidade à distorção de formas. A Tabela 5.1 apresenta os valores do erro médio quadrático, calculados para as 5 iterações. O desvio padrão foi calculado para uma área homogênea de tamanho 20x20, situada no canto superior esquerdo das imagens. Os valores obtidos para as 5 iterações são mostrados na Tabela 5.2. Os perfis de varredura da linha 40, das imagens CIRC, CIRCRD e das imagens referentes às iterações 0 e 4 são apresentados nas Figuras 5.1 e 5.2.

O segundo teste consistiu na aplicação dos filtros uma única vez à imagem NUBRD. Foram estudados os filtros usados no primeiro teste e mais os filtros MEDIA 5x5, MEDIANA 5x5, KVIZ 5x5 com K=12, RANK 3x3 com K=6, RANK 5x5 com K=10, EXSHARP, SMDA e SIGMAPOL. Foram utilizados a análise visual e perfis de varredura. Os parâmetros avaliados foram aguçamento de bordas tipo rampa, remoção de ruído e preservação de bordas. As imagens CIRC, NUBRD e filtradas são mostradas nas Figuras 5.3 a 5.6. Os perfis de varredura da linha 48 são mostrados nas Figuras 5.7 a 5.13.

5.2 Critérios de Avaliação

Grande parte das atividades que envolvem processamento digital de imagens tem, como resultado final, a interpretação visual de uma cena. Assim, a análise visual de uma imagem é, em muitas situações, instrumento apropriado para

avaliação de sua qualidade. Além deste, foram utilizados outros métodos, quantitativos, tais como erro médio quadrático, desvio padrão em área homogênea e a representação de perfis de varredura.

O erro médio quadrático (EMQ), é definido pela expressão:

$$EMQ = \sum (I(m,n) - IT(m,n))^2 / N$$

onde I(m,n) é a imagem original, IT(m,n) é a imagem I(m,n) transformada pela atuação do ruído ou de um filtro, e N é o número de pontos das imagens consideradas (128x128, no caso das imagens de teste). Esta medida fornece um índice de fidelidade da imagem transformada em relação à imagem original.

O desvio padrão (DP), calculado em uma área homogênea, permite a medição da capacidade dos filtros de reduzir o ruído. Este parâmetro é dado por:

$$DP = [(\sum p^2)/n - ((\sum p)/n)^2]^{1/2}$$

onde p é o nível de cinza de cada pixel e n é o número de pixels da área considerada.

O perfil de varredura ou perfil de linha é a representação no plano, dos níveis de cinza presentes em uma determinada linha da imagem, em função de sua posição nesta linha. Esta representação proporciona a visualização dos efeitos provocados pelo ruído e pela filtragem na imagem.

5.3 Resultados Obtidos

Os resultados do estudo foram apresentados através de tabelas para o erro médio quadrático (Tab. 5.1) e desvio padrão (Tab. 5.2), fotografias (Figs. 5.3 a 5.13) e perfis de varredura (Figs. 5.1 e 5.2).

5.3.1 Remoção de Ruído e Preservação de Bordas

Como pode ser observado a partir dos dados obtidos para o desvio padrão, alguns filtros reduziram consideravelmente o ruído já na primeira iteração. Este foi o caso de FACETAS e MEDIA3. FACETAS destacou-se, nessa etapa, por proporcionar um alto nível de fidelidade em relação a outros métodos (redução do erro médio quadrático de 85,0%), conforme pode ser verificado na Tabela 1. Os perfis de varredura da linha 40, para a iteração 0 (Fig. 5.1), mostram um melhor desempenho dos métodos baseados no critério de vizinhança seletiva, quanto à preservação de bordas.

As iterações mostraram as tendências e algumas características peculiares dos algoritmos. O filtro MEDIA3, por exemplo, apesar de reduzir consideravelmente o ruído, provocou graves distorções nas regiões de fronteira (vide Figs. 5.1.c e 5.2.c) e, conseqüentemente, um acréscimo no erro médio quadrático a partir da quarta iteração. Destacaram-se, nesta fase, dos testes os algoritmos SIGMA e SSDA que, de acordo com as medidas do desvio padrão (Tab. 5.2) reduziram o ruído em 88,1% e 81,0%, respectivamente. Os perfis de linha para a iteração 4, mostrados na Figura 5.2, confirmam

	ERRO MÉDIO QUADRÁTICO			CIRC EMQ= 0.0 CIRCRD EMQ= 391,00	
FILTROS	IT 0	IT 1	IT 2	IT 3	IT 4
MEDIA3	82,09	73,61	79,85	86,06	93,96
MEDIANA3	70,21	50,06	41,11	35,36	31,38
KUIE	81,96	54,08	45,49	40,91	39,71
SSVAR	71,17	51,02	47,12	46,30	46,09
SSDA	71,70	42,98	38,01	33,16	30,11
FACETAS	59,01	44,20	41,05	40,91	40,79
SIGMA	73,41	29,06	17,12	13,99	12,98
ORDAP	72,12	60,92	56,17	54,08	50,99
MEDNADP	70,19	52,97	40,11	36,14	34,01
SIGMADP	73,05	57,02	49,93	40,49	35,08

Tabela 5.1 Erro Médio Quadrático

	DESVIO PADRÃO			Região Janela 20x20 CIRC DP= 0.00 CIRCRD DP= 20.06	
FILTROS	IT 0	IT 1	IT 2	IT 3	IT 4
MEDIA3	7,09	5,01	3,89	3,45	3,11
MEDIANA3	8,12	5,92	5,04	4,40	4,09
KUIE	8,98	6,90	5,77	5,01	4,33
SSVAR	7,90	5,69	5,07	5,06	5,06
SSDA	7,89	5,60	4,69	4,11	3,81
FACETAS	7,19	5,92	5,73	5,59	5,58
SIGMA	8,01	4,94	3,01	2,72	2,39
ORDAP	9,21	7,91	7,09	6,90	6,43
MEDNADP	7,99	6,10	5,11	4,59	4,52
SIGMADP	8,25	6,61	5,70	5,09	4,79

Tabela 5.2 Desvio Padrão em área homogênea

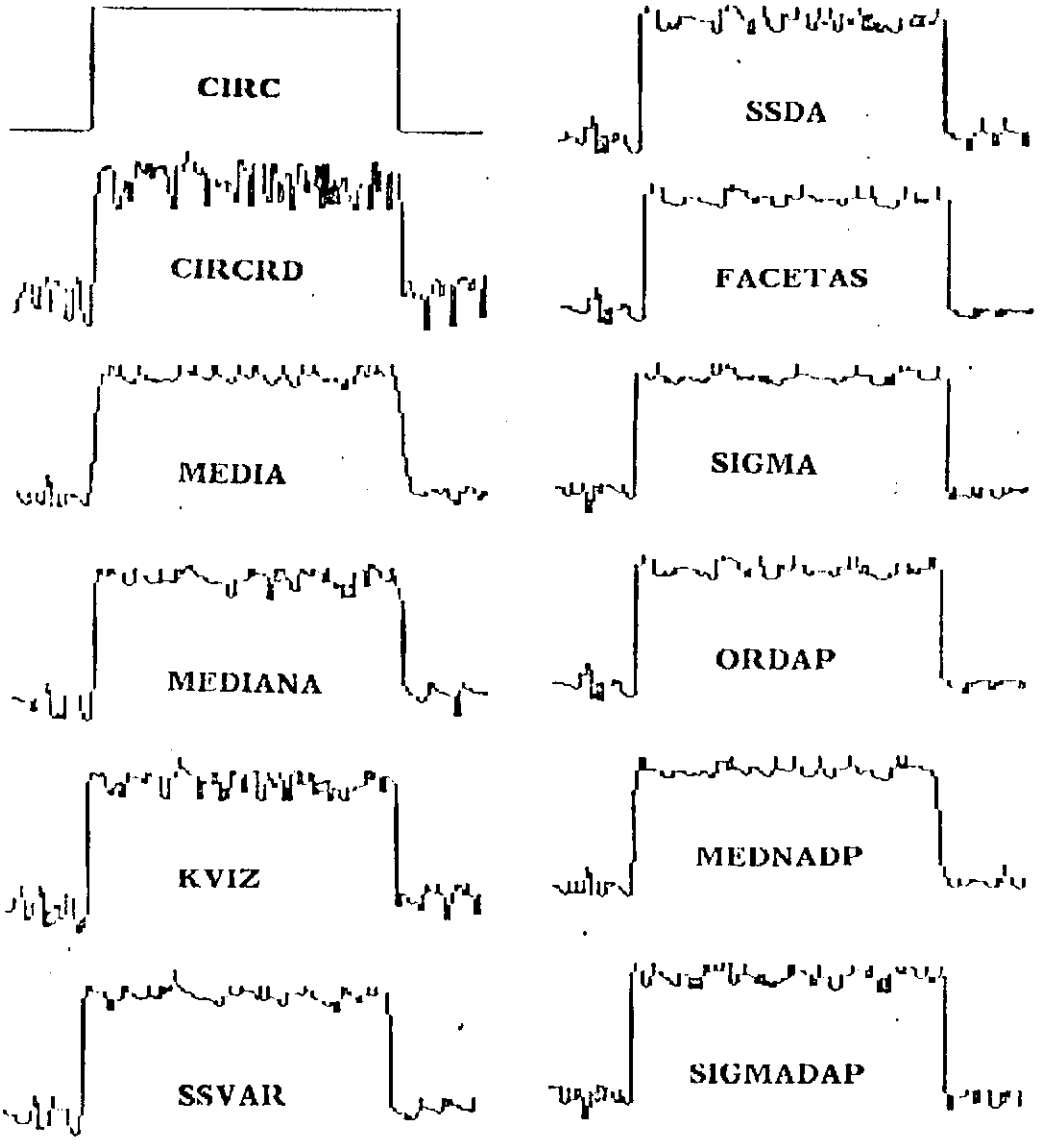

Figura 5.1 Perfis da linha 40 da imagem CIRC (a) sem ruído, (b) com ruído gaussiano (0,20.0), e filtrada (iteração 0) pelos algoritmos (c) MEDIA3, (d) MEDIANA, (e) KVIZ3, com k=6, (f) SSVAR, (g) SSDA, (h) FACETAS, (i) SIGMA, (j) ORDAP com J=1, (k) MEDNADP 5x5 com c=5 e peso central=20 e (l) SIGMADP com K=3, a=3 e w_i=1/3.k

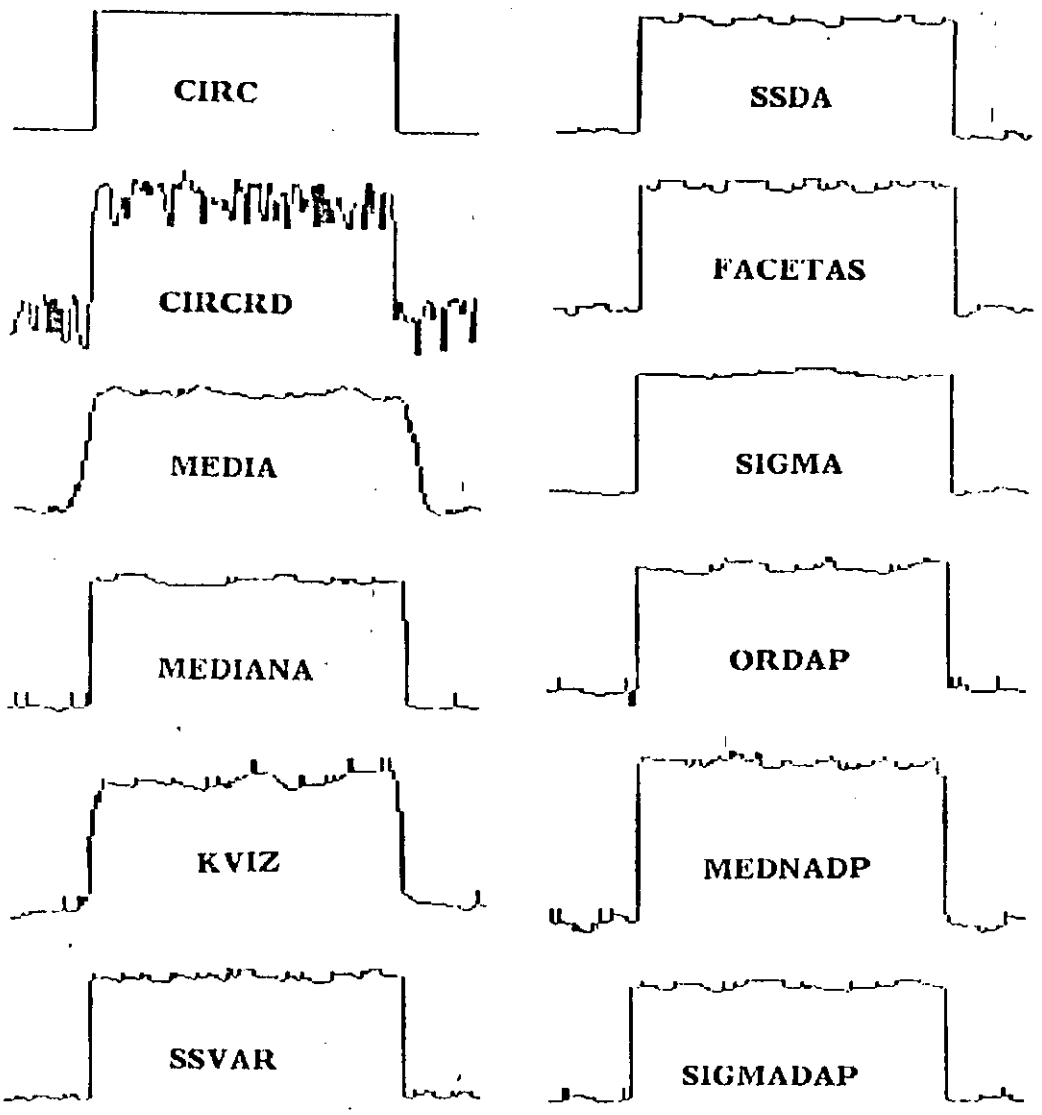

Figura 5.2 Perfis da linha 40 da imagem CIRC (a) sem ruído, (b) com ruído gaussiano (0,20.0), e filtrada (iteração 4) pelos algoritmos (c) MEDIA3, (d) MEDIANA, (e) KVIZ3, com k=6, (f) SSVAR, (g) SSDA, (h) FACETAS, (i) SIGMA, (j) ORDAP com J=1, (k) MEDNADP 5x5 com c=5 e peso central=20 e (l) SIGMADP com K=3, a=3 e w_i=1/3.

a eficiência destes dois métodos quanto à capacidade de preservar fronteiras. Entre as técnicas adaptativas, nesta etapa do experimento, SIGMADP apresentou bons resultados e o algoritmo ORDAP, apesar de não remover com eficiência o ruído, manteve a integridade das bordas.

5.3.2 Aguçamento de Bordas

Os resultados do segundo teste, envolvendo a imagem NUBRD, são apresentados através de fotografias, nas Figuras 5.3 a 5.6, e dos perfis de varredura da linha 48, nas Figuras 5.7 a 5.13. Pode ser observado que os filtros MEDIA3 e MEDIA5 (Figuras 5.3.c e 5.3.d) aumentaram ainda mais o efeito de nublar. RANK3, RANK5, MEDIANA3, MEDIANA5, KVIZ3, KVIZ5, SSLOG e SSGRAD, apesar de proporcionarem redução no ruído, não aguçaram as bordas. Destacaram-se ORDAP, SIGMAPOL, SSDA, SSVAR e FACETAS, os quais realçaram as bordas, aproximando-as mais das bordas tipo impulso existentes antes da degradação.

5.3.3 Imunidade à Distorção

Uma característica ideal para um filtro espacial é a capacidade de preservar detalhes, linhas finas e pequenos objetos presentes na imagem. Os filtros MEDIA3, MEDIA5, KVIZ3 e KVIZ5 distorcem e até eliminam linhas finas devido ao efeito de nublar. Este efeito é mais grave quando se usam tamanhos maiores de janela ou máscara. Alguns algoritmos

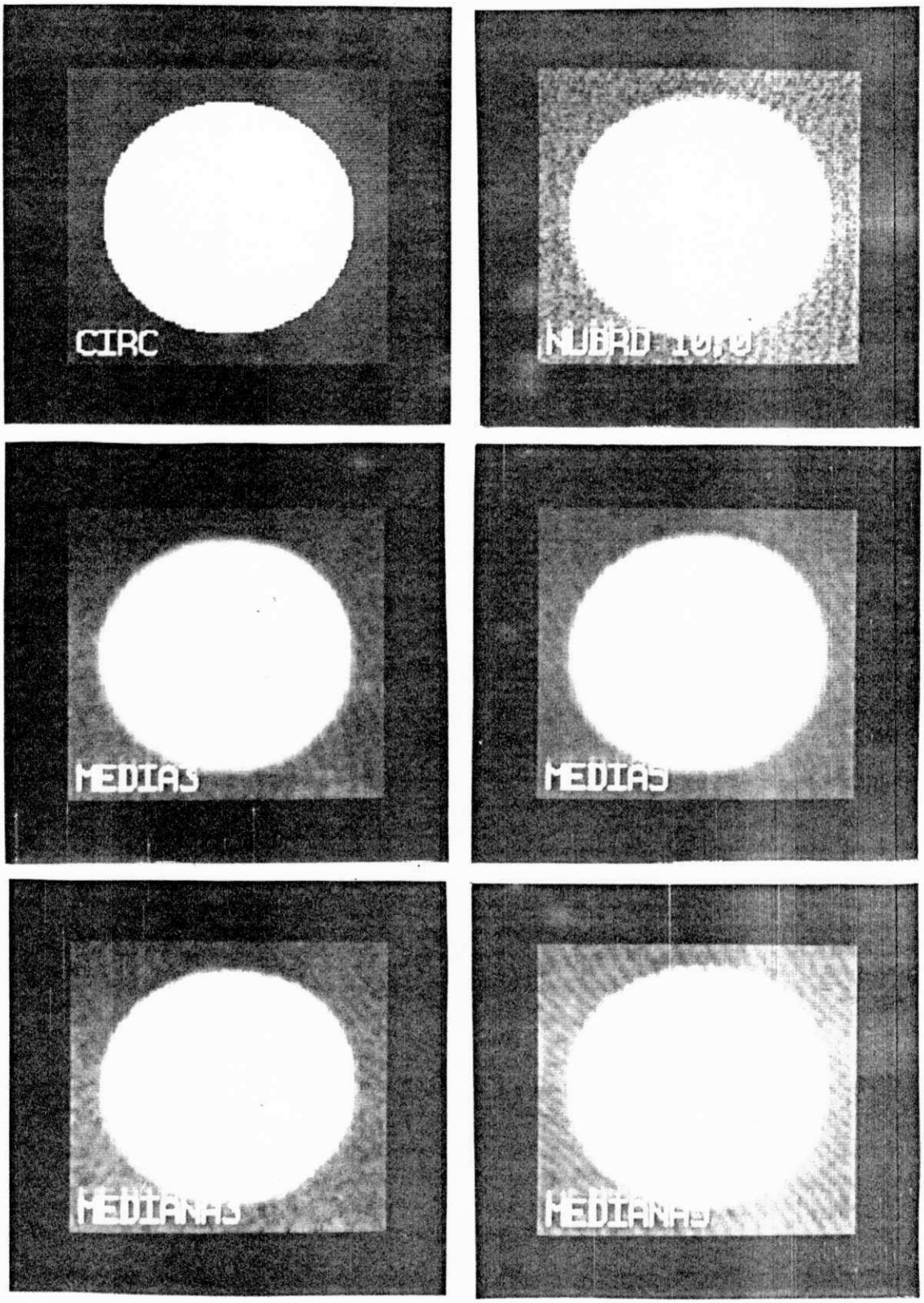

Figura 5.3 Imagem CIRC (a) sem ruido, (b) com ruído gaussiano (0,10.0) e nublada (NUBRD) e filtrada (it. 0) pelos filtros (c) MEDIA3, (d) MEDIA5, (e) MEDIANA3 e (f) MEDIANA5.

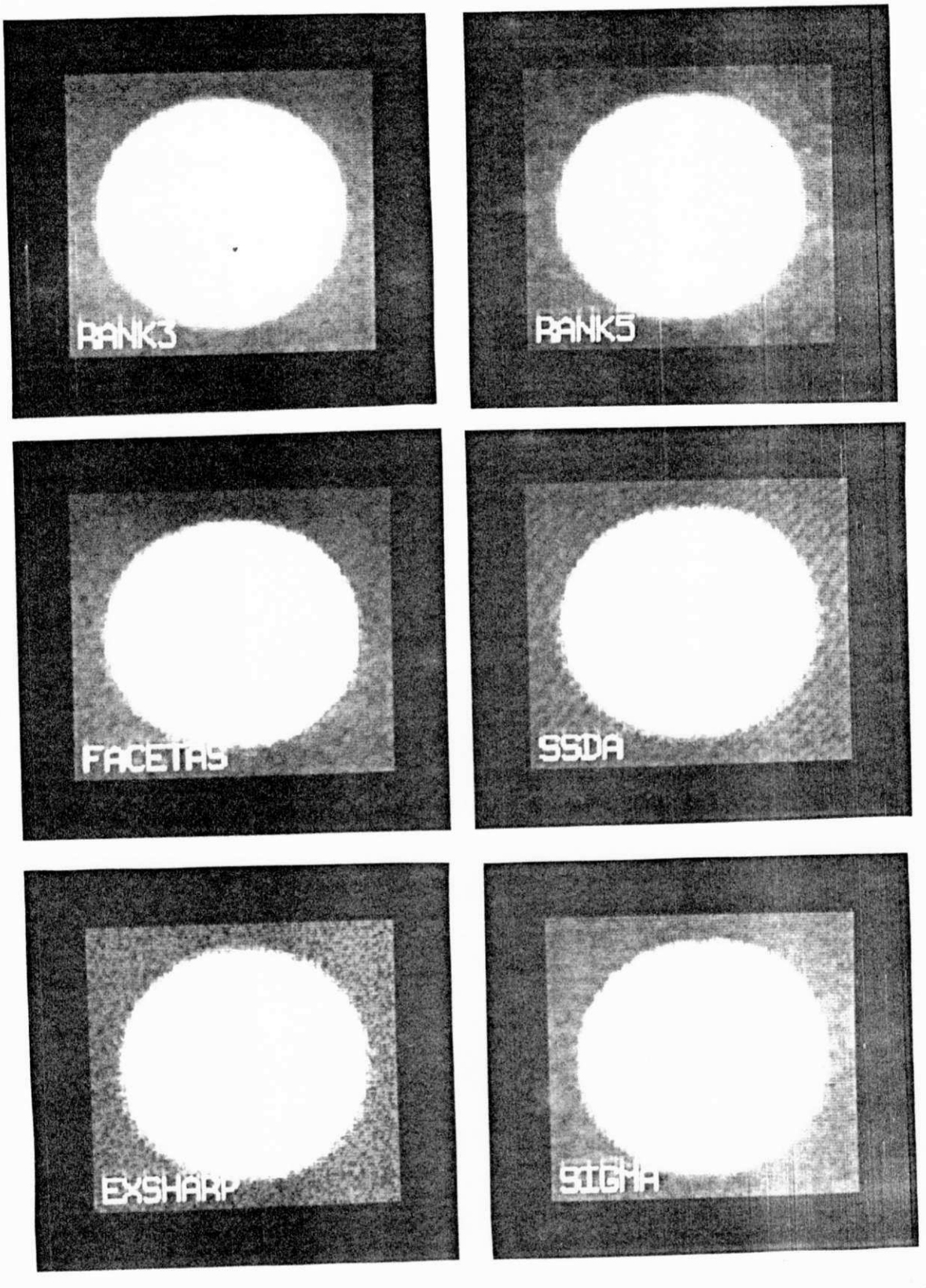

Figura 5.4 Imagem NUBRD filtrada (it.0) pelos filtros (a) RANK3, (b) RANK5, (c) FACETAS, (d) SSDA, (e) EXSHARP e (f) SIGMA.

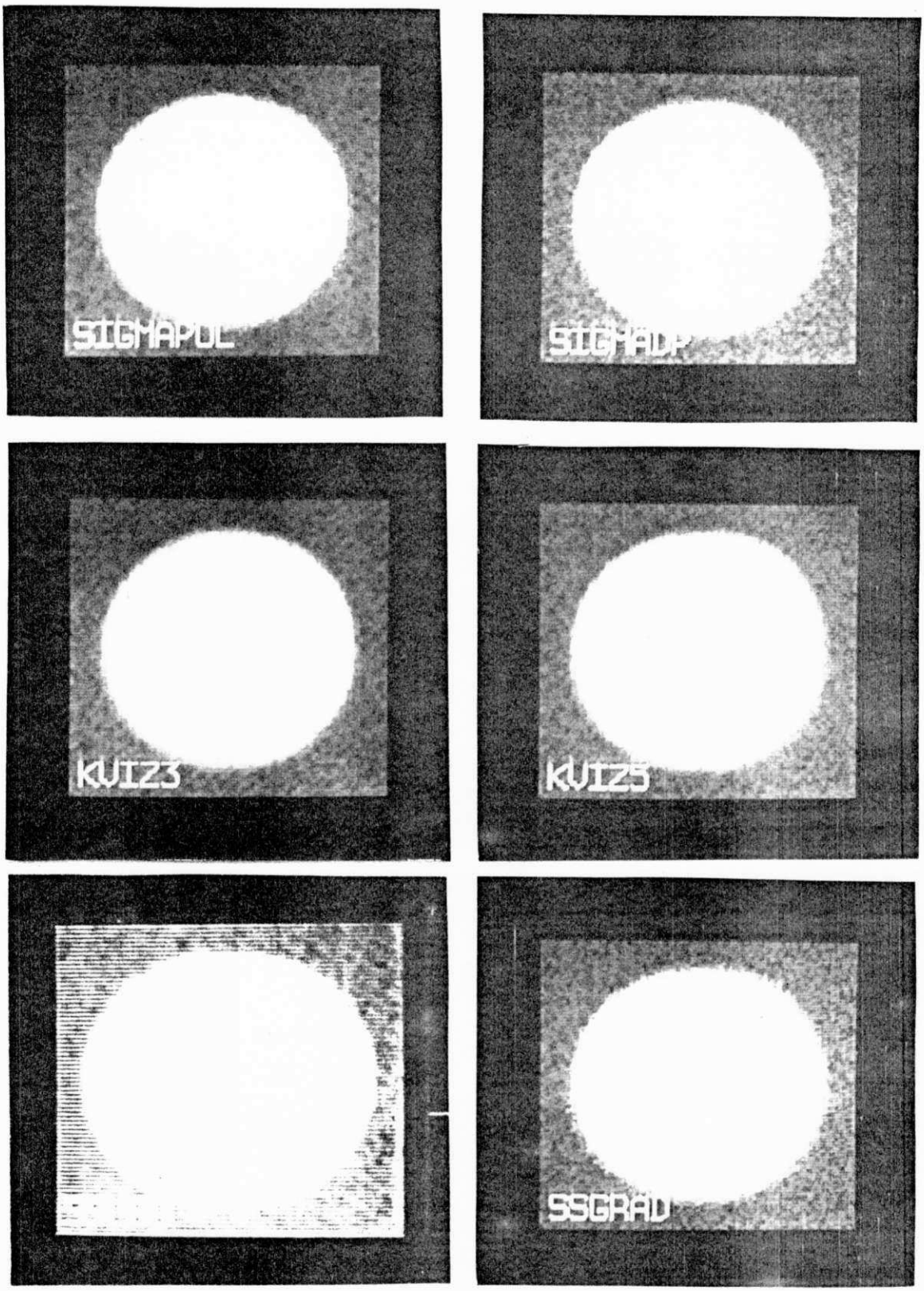

Figura 5.5 Imagem NUBRD filtrada (it. 0) pelos filtros (a) SIGMAPOL, (b) SIGMADAP, (c) KVIZ3, (d) KVIZ5, (e) SSLOG (m=16) e (f) SGRAD.

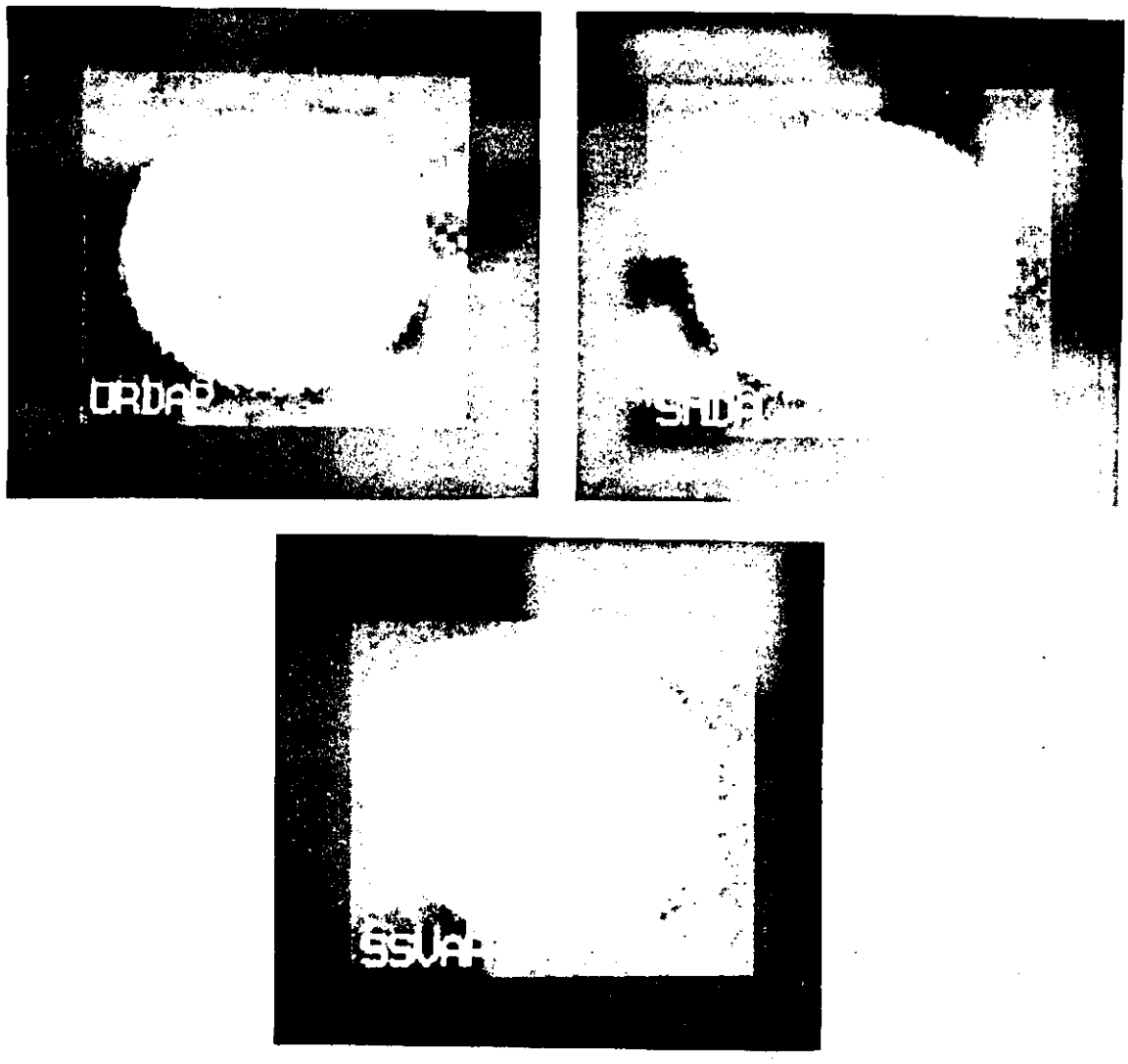

Figura 5.6 Imagem NUBRD filtrada (it. 0) pelos filtros (a) ORDAP (N=2, J=1), (b) SMDA e (c) SSVAR.

dispõem de parâmetros que determinam o compromisso entre os efeitos de suavização (redução do ruído) e preservação de bordas ou detalhes. O filtro SIGMA preserva detalhes, desde que a diferença de intensidade entre eles e o fundo seja maior que a faixa dos dois-sigmas. As técnicas adaptativas têm seus parâmetros de compromisso variáveis em função de informações locais, avaliadas a cada passo da convolução.

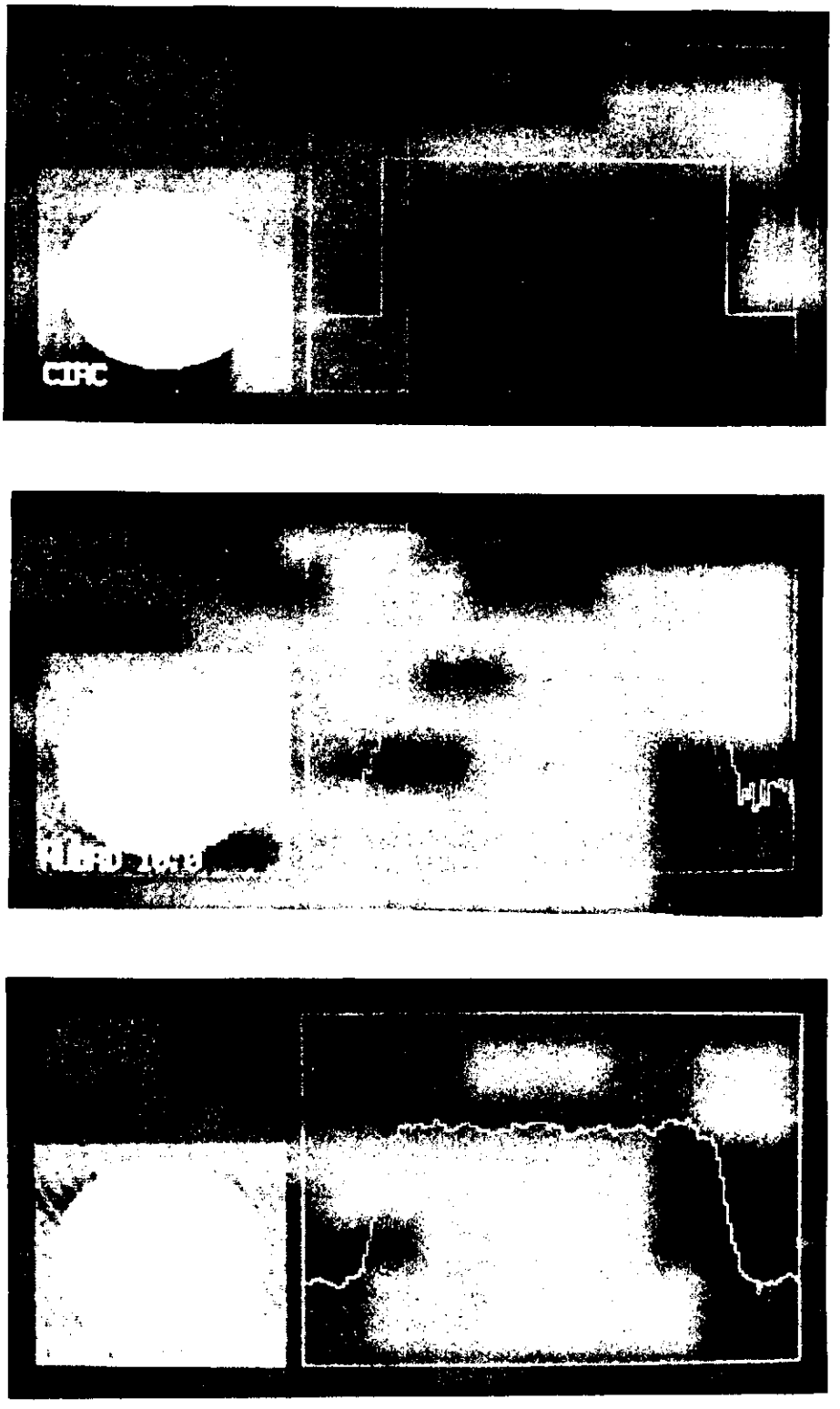

Figura 5.7 Perfil da linha 48 da imagem CIRC (a) sem ruído, (b) com ruído gaussiano (0,10.0) e nublada (NUBRD) e (c) filtrada (iteração 0) pelo filtro MEDIA3.

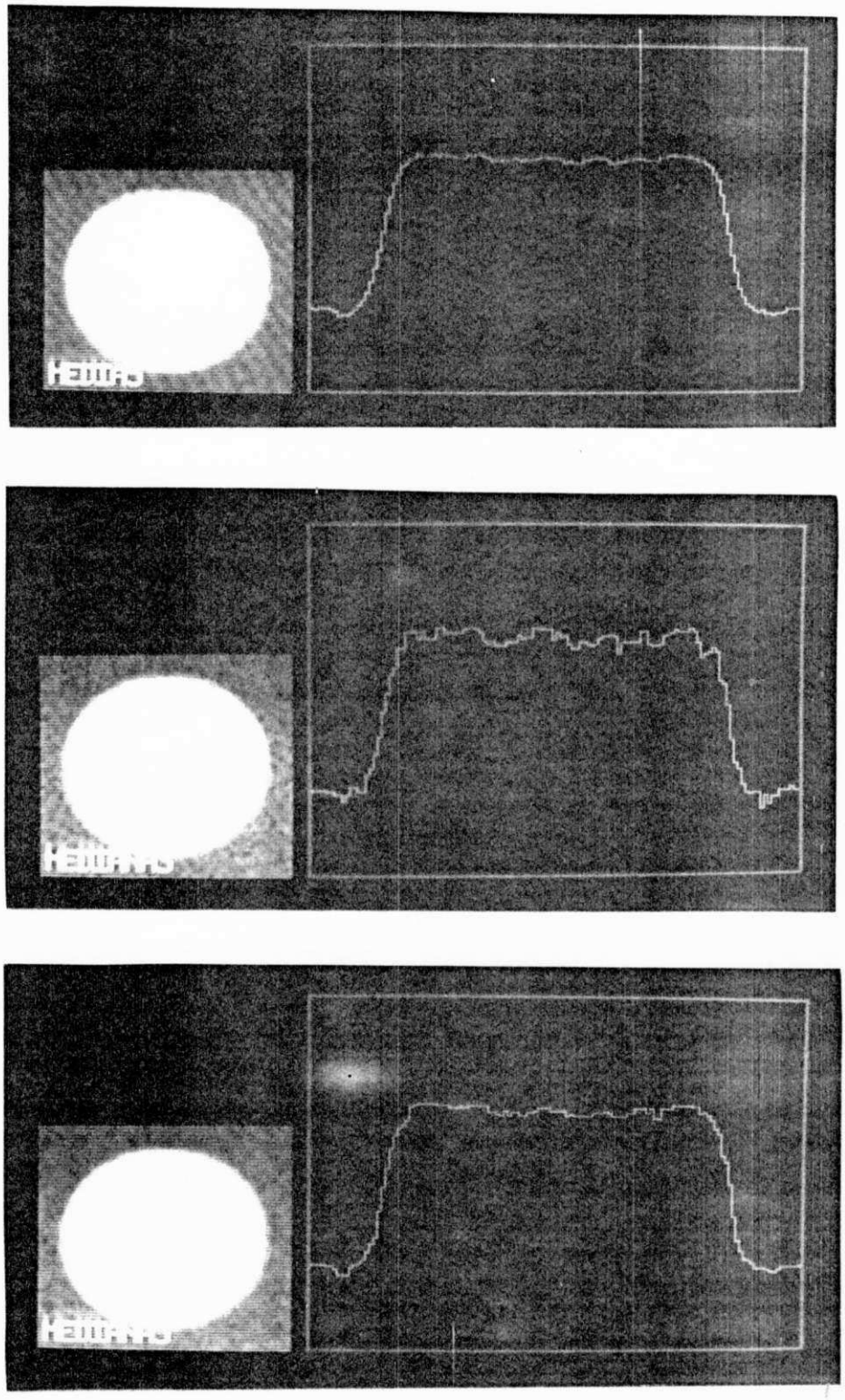

Figura 5.8 Perfil da linha 48 da imagem NUBRD filtrada (it. 0) pelos filtros (a) MEDIA5, (b) MEDIANA3 e (c) MEDIANA5.

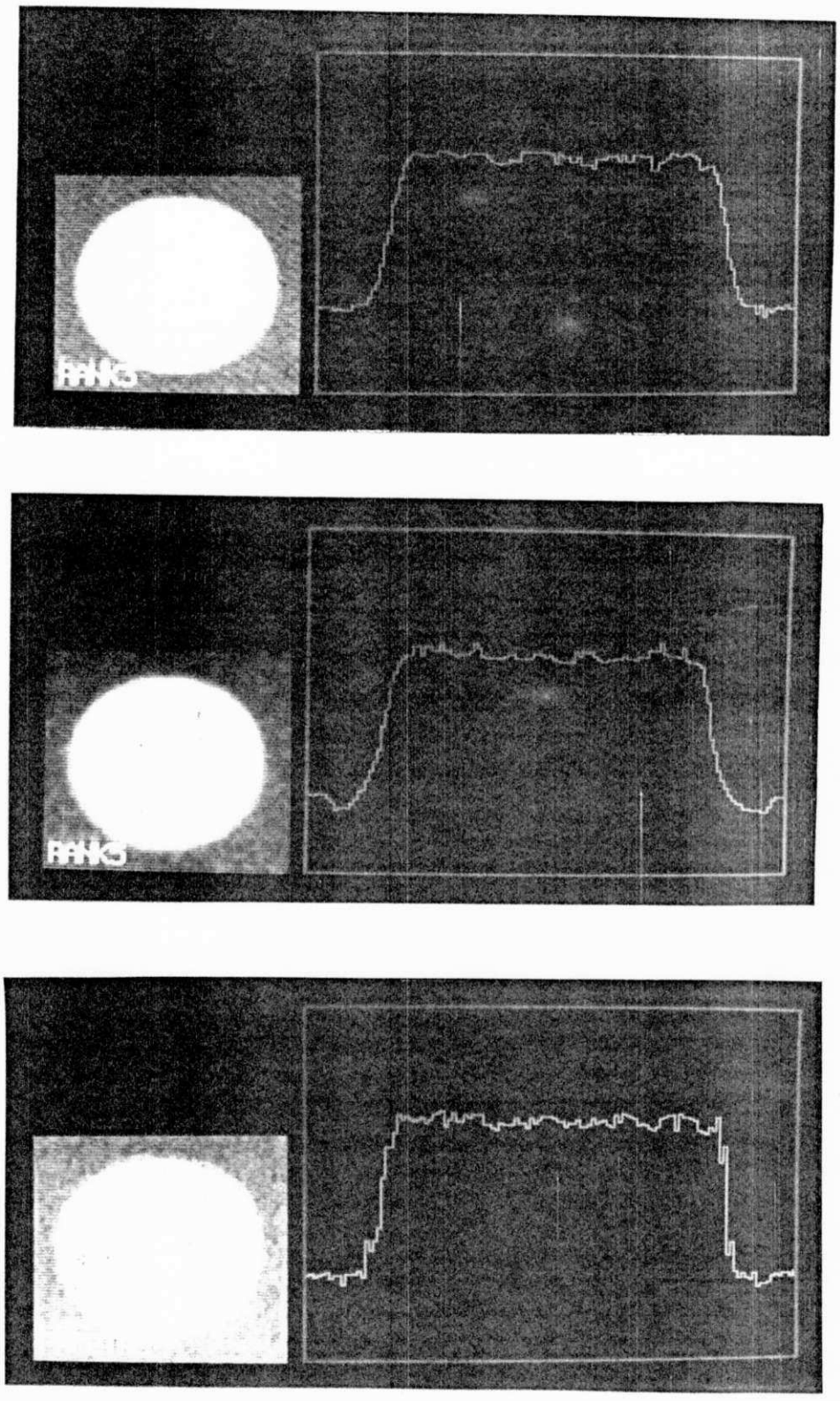

Figura 5.9 Perfil da linha 48 da imagem NUBRD filtrada (it. 0) pelos filtros (a) RANK3, (b) RANK5 e (c) FACETAS.

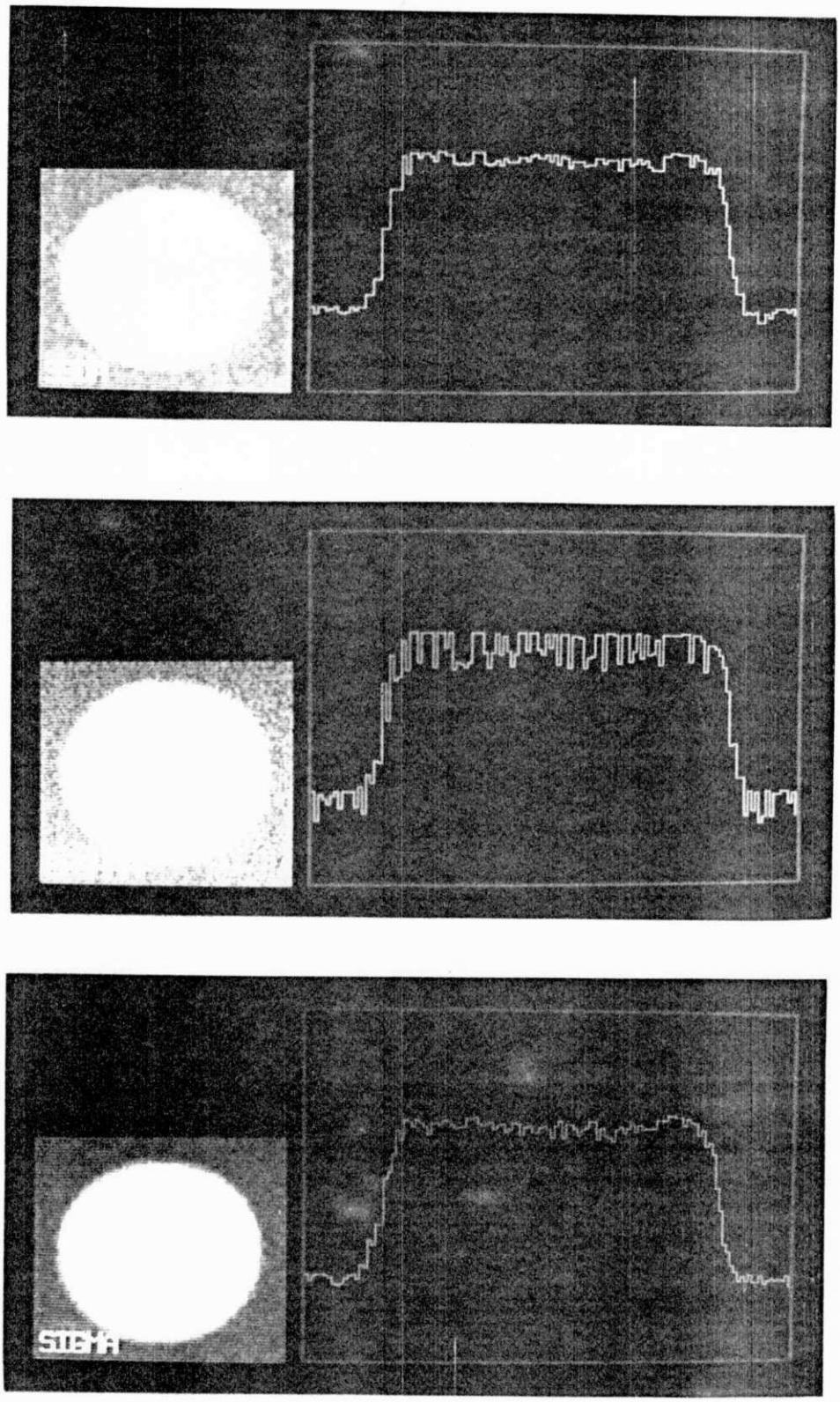

Figura 5.10 Perfil da linha 48 da imagem NUBRD filtrada (it. 0) pelos filtros (a) SSDA, (b) EXSHARP e (c) SIGMA.

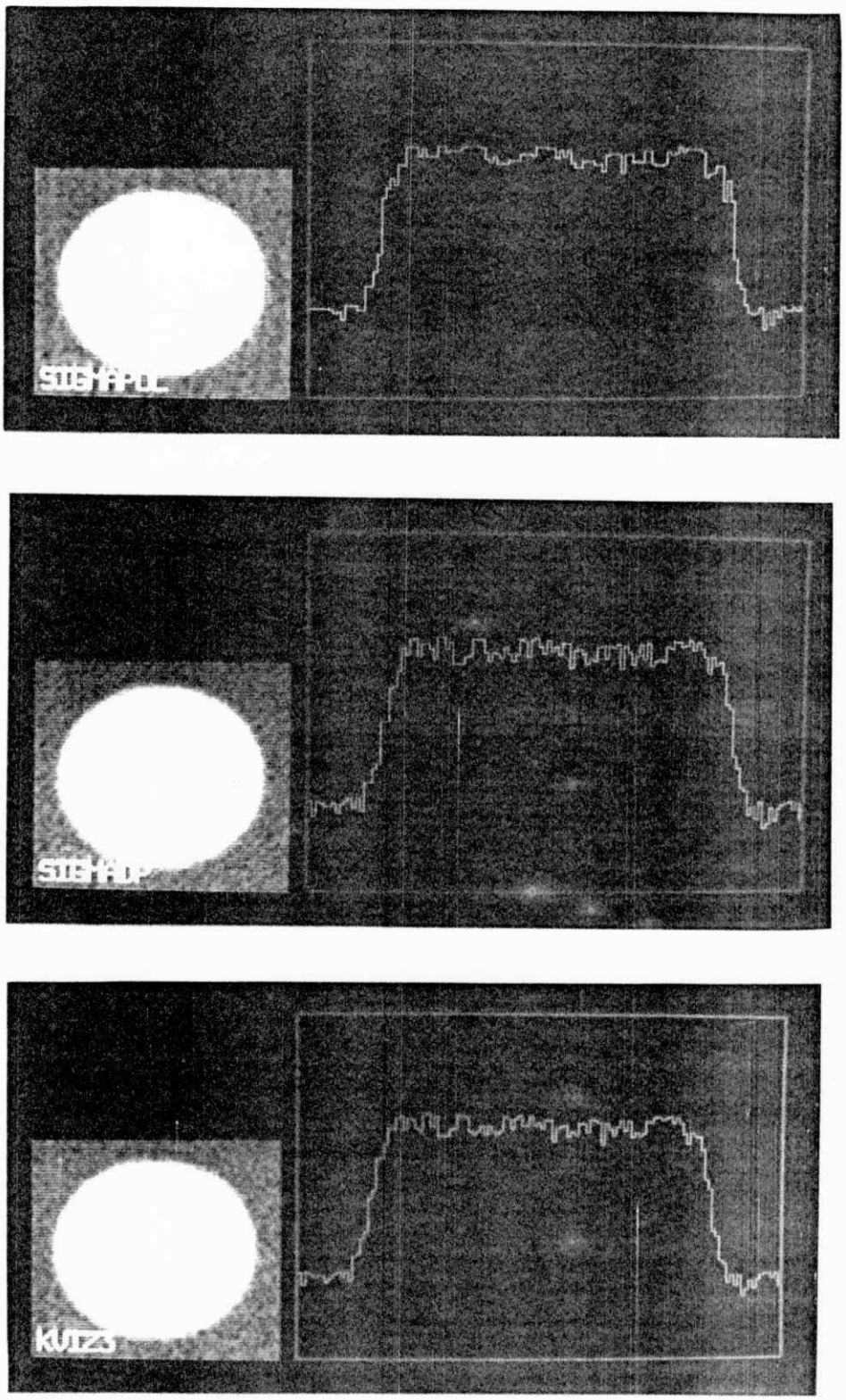

Figura 5.11 Perfil da linha 48 da imagem NUBRD filtrada (it. 0) pelos filtros (a) SIGMAPOL, (b) SIGMADAP (K=3, a=2 e w_i=1/3) e (c) KVIZ3.

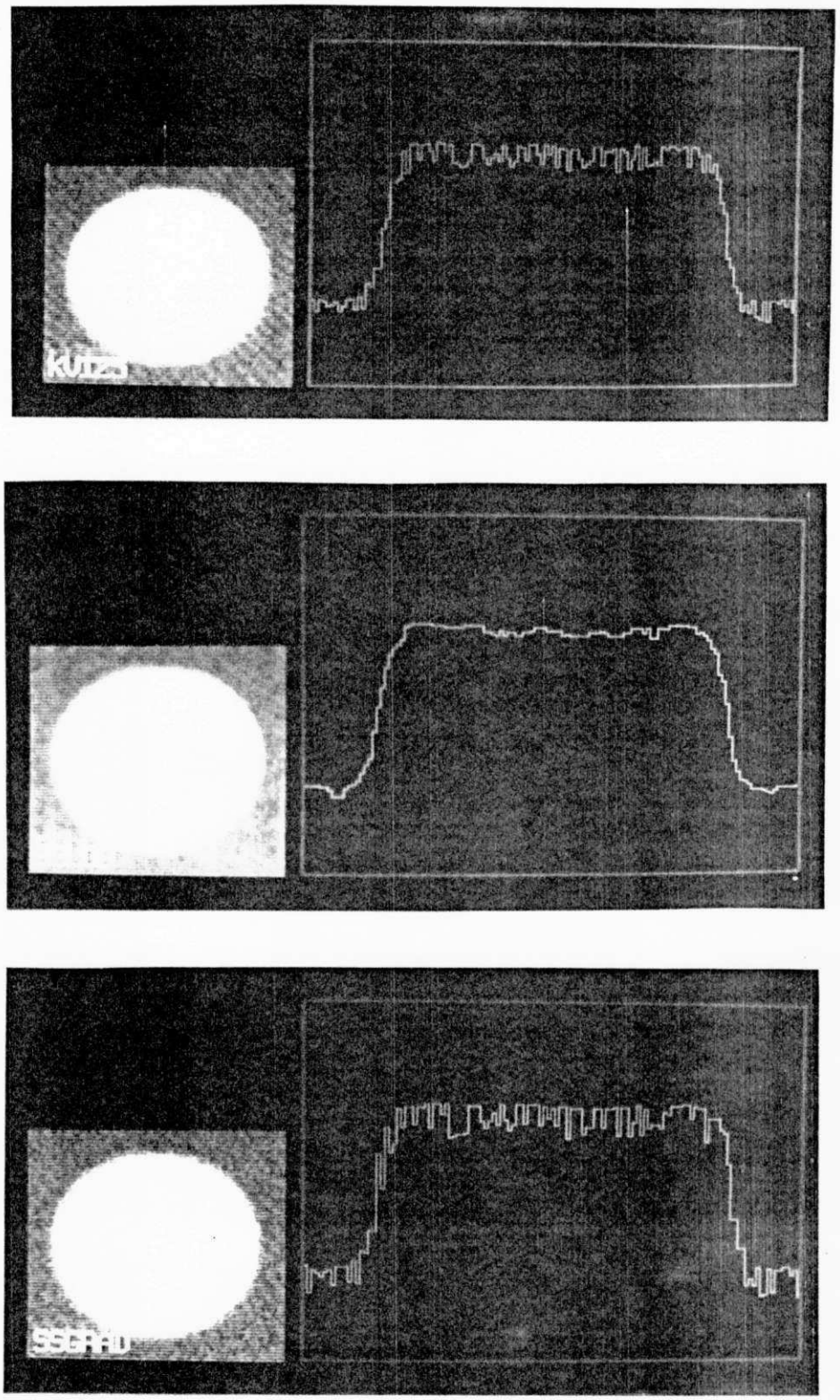

Figura 5.12 Perfil da linha 48 da imagem NUBRD filtrada (it. 0) pelos filtros (a) KVIZ5, (b) SSLOG (m=16) e (c) SGRAD.

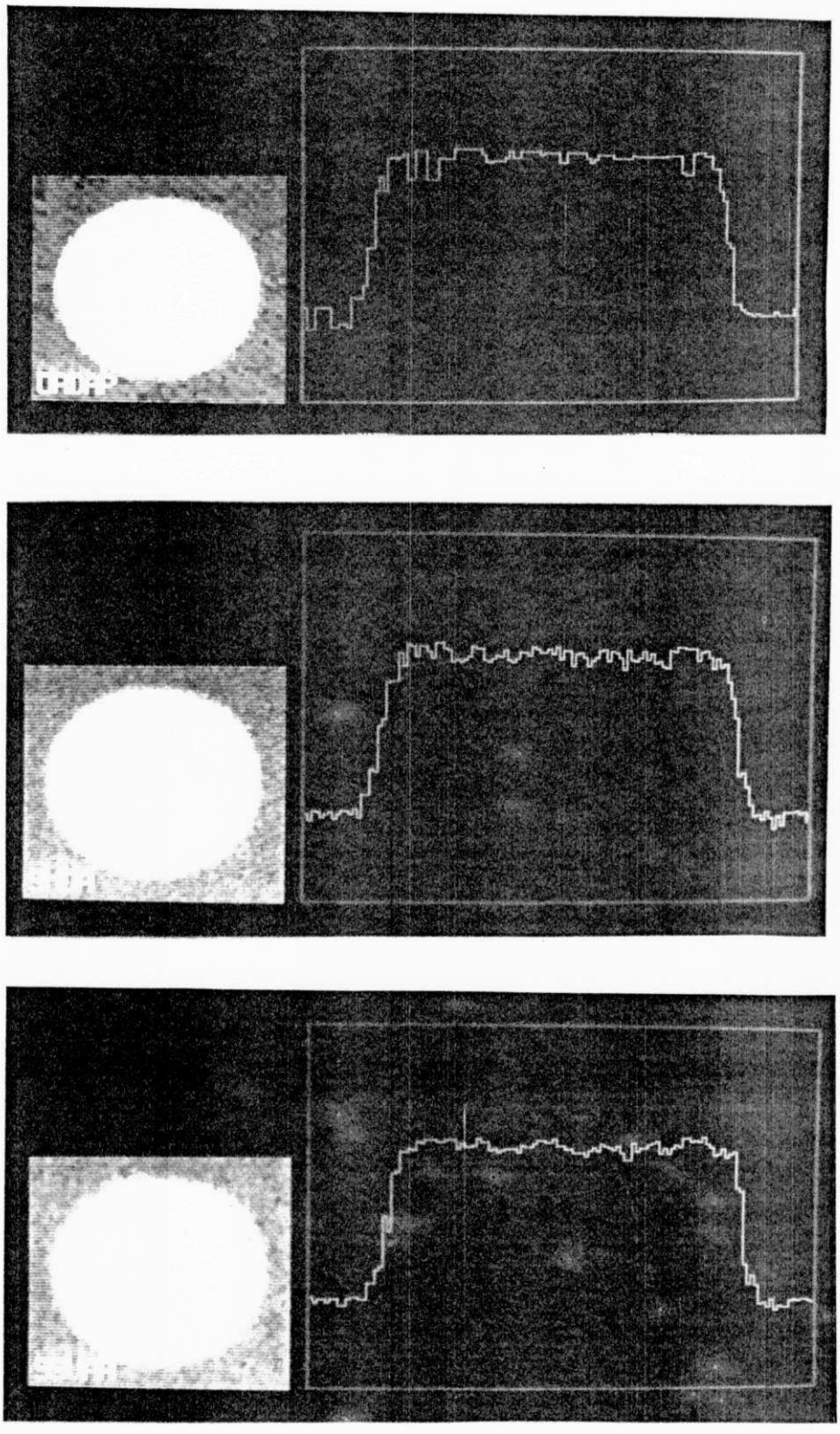

Figura 5.13 Perfil da linha 48 da imagem NUBRD filtrada (it. 0) pelos filtros (a) ORDAP (N=2, J=1), (b) SMDA e (c) SSVAR.

Este fato as torna mais sensíveis à presença de detalhes, fazendo-as até dispensar a operação de suavização em alguns pontos. Os métodos baseados no critério de vizinhança seletiva também produzem distorções por considerarem subgrupos de 7, 8 ou 9 pixels no cálculo da média. O algoritmo EXSHARP produziu um aguçamento das bordas, mas provocou distorções nas regiões homogêneas e não reduziu consideravelmente o ruído.

5.4 Detecção de Bordas

Para os filtros passa-altas, foi utilizada uma imagem natural, denominada CLOSE, que consiste de um rosto de mulher, obtida através de uma câmera de vídeo e digitalizada. Neste teste, pôde ser observada a resposta em magnitude dos detetores de borda estudados na Seção 3.2, através das imagens-gradiente geradas. Estas imagens demonstram uma das principais características dos algoritmos que é a sua sensibilidade aos sinais de alta freqüência. Na imagem-gradiente, podem ser observados os efeitos de saturação, a largura das bordas, a imunidade ao ruído, o comportamento do operador em regiões homogêneas e a definição final dos contornos esperados.

Os filtros passa-altas foram aplicados uma única vez à imagem CLOSE sem ruído (Fig. 5.14.a). As Figuras 5.14.b a 5.16.b mostram as imagens-gradiente geradas pelos operadores diferenciais de Roberts, de Sobel, de Prewitt, laplacianos (3, 4 e 5) e , pelos operadores direcionais de Prewitt, de Kirsh, de Robinson 3 níveis e de Robinson 5 níveis, e pelo detetor de bordas híbrido, respectivamente.

Alguns métodos apresentaram uma hipersensibilidade aos sinais de alta freqüência que caracterizam possivelmente as bordas. Este foi o caso dos operadores SOBEL, PREWITTDF, PREWITTDR, ROBINSON3 E ROBINSON5 (Figs. 5.14.c, 5.15.d, 5.14.d, 5.15.f e 5.16.a). Este fato produz uma saturação nas áreas onde existem muitos detalhes, como pode ser visto nas regiões correspondentes aos olhos, lábios e partes dos cabelos na imagem CLOSE filtrada por estes algoritmos.

Os operadores laplacianos (LAPLAC3, LAPLAC4 e LAPLAC5), devido ao fato de não possuírem polarização quanto à direção das bordas, geram muitos pontos de borda em regiões relativamente homogêneas, os quais dificilmentte são conectáveis para formar um contorno válido (Figuras 5.15.a, 5.15.b e 5.15.c).

Os operadores de Robinson (ROBINSON3 e ROBINSON5) produzem bordas bastante nítidas em função do critério de seleção da máxima resposta dentre as oito disponíveis em cada ponto.

O operador de Roberts (ROBERTS) apresenta duas características importantes. A primeira é função das dimensões de suas máscaras. Usando janelas 2x2, ele permite detectar bordas com largura de apenas um pixel. Esta é uma resposta ideal para um algoritmo de detecção de contornos. A segunda característica é a própria simplicidade do método e, conseqüentemente, sua eficiência computacional. Isto torna-o, considerando as implementações em linguagem C estudadas e a aplicação na imagem CLOSE, 9 vezes mais rápido que o operador de Sobel e 20 vezes mais rápido que o detetor de bordas híbrido. Como pode ser observado na Figura 5.14.b, os pontos de borda detectados aparecem conectáveis, definindo

os contornos esperados com bastante exatidão. Nas regiões com muitos detalhes, como aquelas referentes aos olhos, lábios e parte superior dos cabelos na imagem CLOSE, não ocorre saturação. Não há manifestação de respostas de grande magnitude nas regiões homogêneas, o que indica uma razoável imunidade ao ruído.

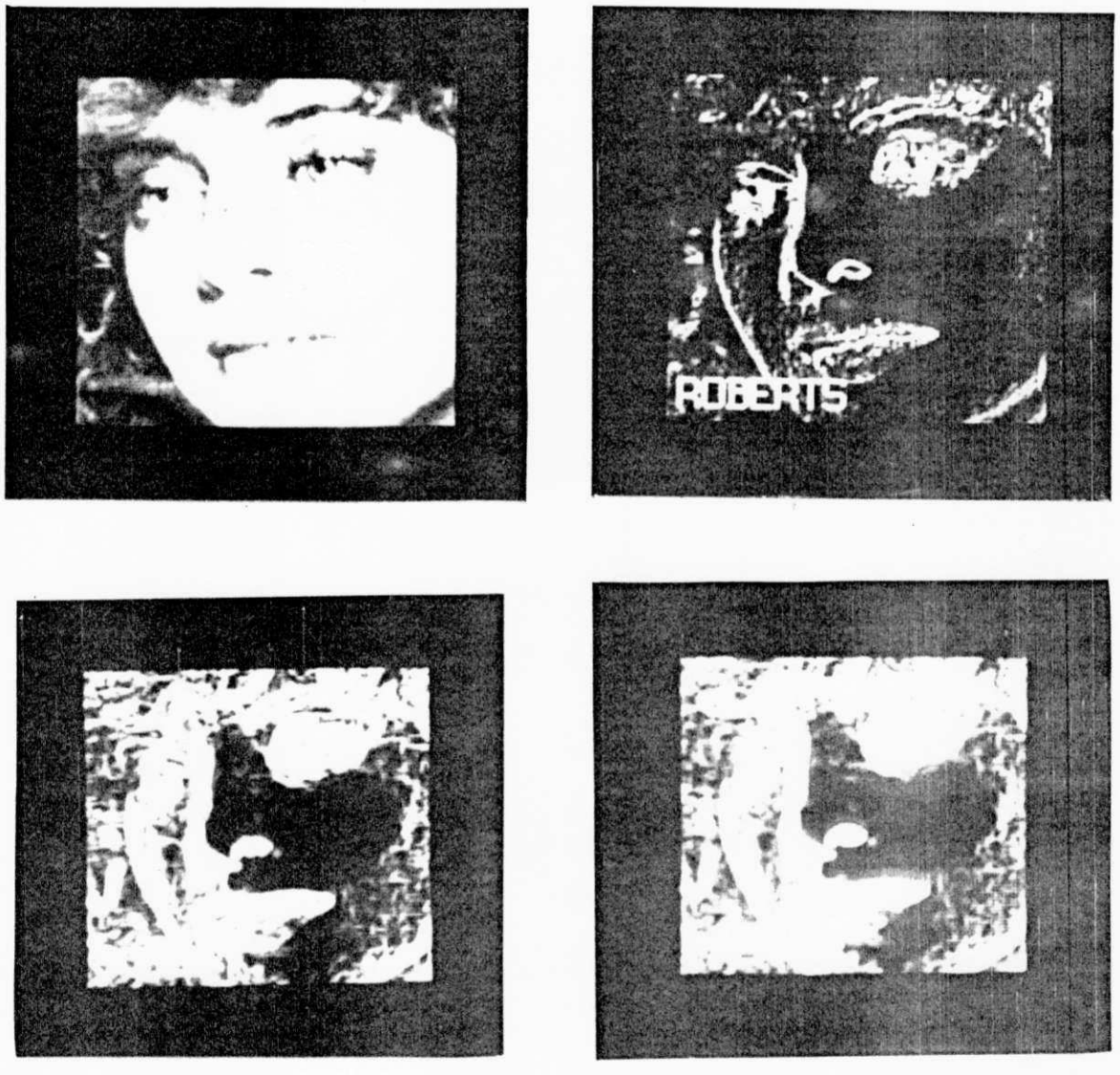

Figura 5.14 (a) Imagem CLOSE, e imagens CLOSE gradiente geradas pelos filtros (b) ROBERTS, (c) SOBEL e (d) PREWITTDR.

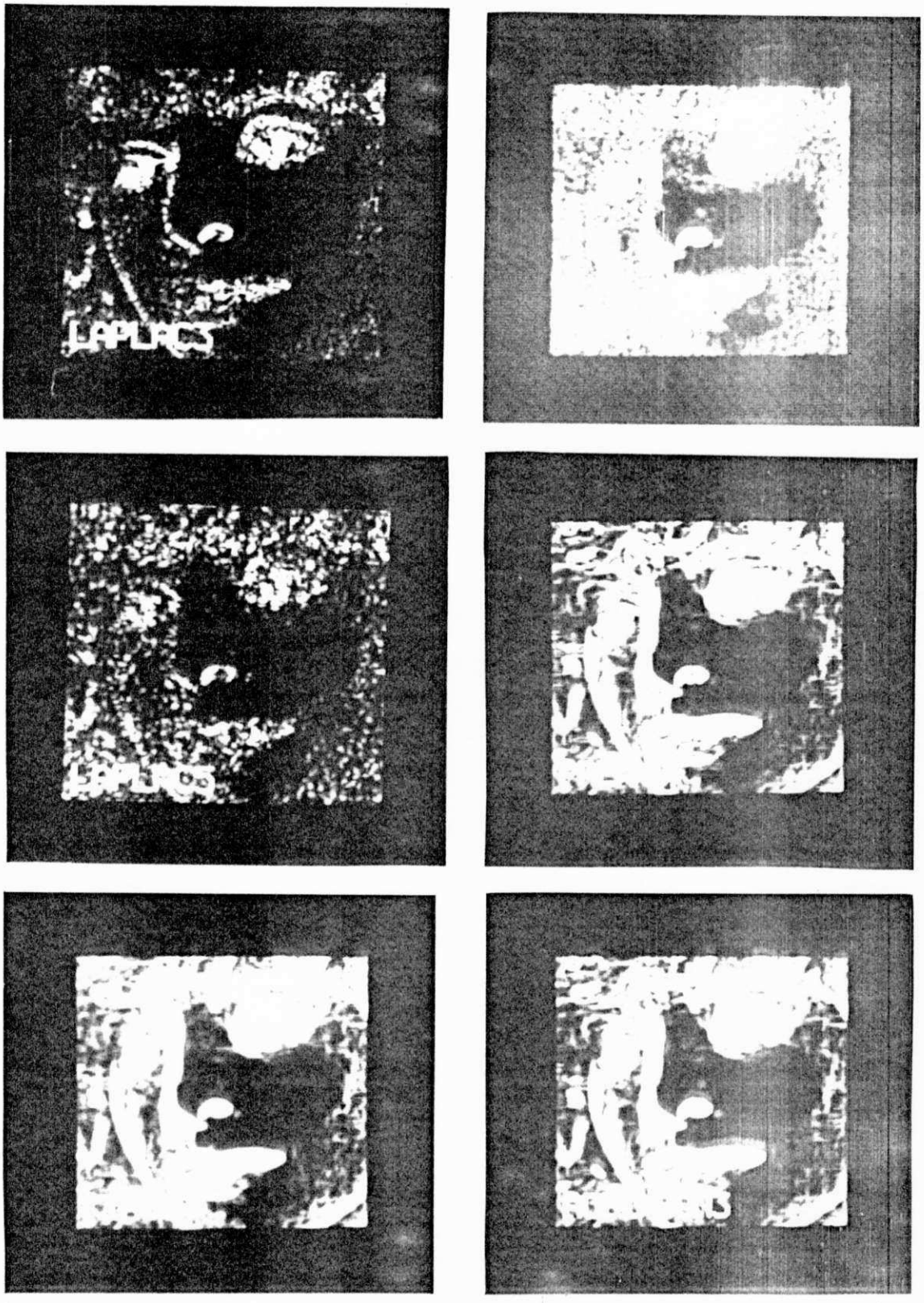

Figura 5.15 Imagens CLOSE gradiente geradas pelos filtros (a) LAPLAC3, (b) LAPLAC4, (c) LAPLAC5, (d) PREWITTDF, (e) KIRSCH e (f) ROBINSON3.

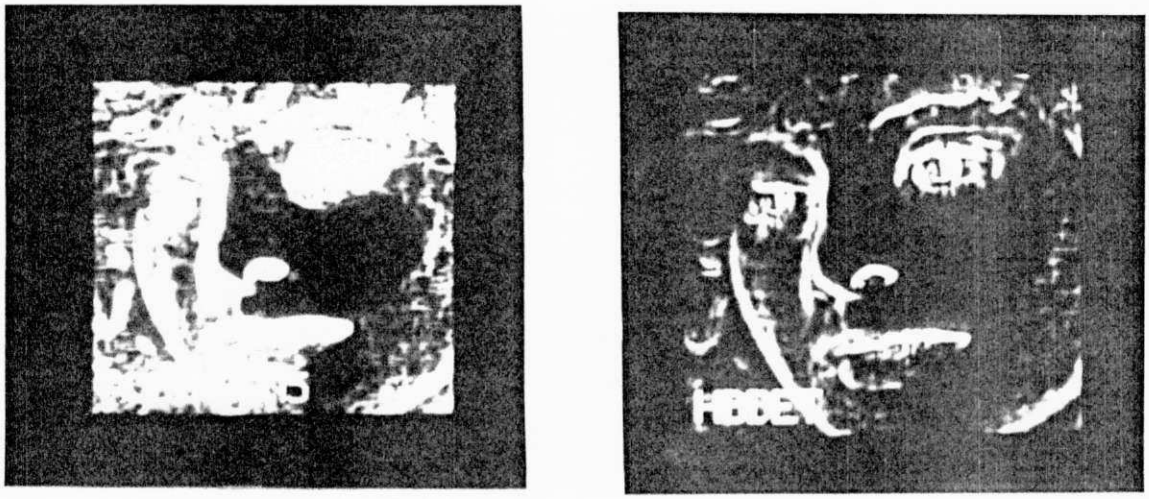

Figura 5.16 Imagens CLOSE gradiente geradas pelos filtros (a) ROBINSON5 e (b) HBDET.

O detetor de bordas híbrido também apresentou boa sensibilidade, sem produzir saturações. Isto se deve ao fato de sua resposta incluir propriedades não lineares da operação mediana e dos subfiltros empregados.

Os operadores ROBERTS, ROBINSON3, ROBINSON5 e HBDET forneceram as melhores respostas para tarefas de aplicação de limiar, análise visual ou pré-processamento de imagens.

5.5 Conclusão

Os filtros baseados no critério de vizinhança seletiva (SIGMA, FACETAS, SSDA, SSVAR, SSLOG) destacaram-se pela

sua capacidade de remover o ruído, com preservação de bordas. Os fitros sigma, da mediana e da ordem adaptativos, recentemente desenvolvidos, também demonstraram um bom desempenho. Alguns algoritmos baseados no critério de vizinhança seletiva mostraram-se muito eficientes, porém implicam em uma grande carga computacional. Isto os torna mais lentos quando implementados em arquiteturas baseadas em microcomputador e aplicados a imagens grandes.

As técnicas de detecção de bordas implementadas demonstraram boa sensibilidade às variações do sinal da imagem. Desempenho especial foi mostrado pelos operadores de Robinson e pelo detetor de bordas híbrido. O operador de Roberts, apesar de sua simplicidade, apresentou uma grande eficiência na geração de imagens-gradiente, em relação a outros métodos mais complexos. Este resultado motivou uma proposta de implementação de um circuito integrado de aplicação específica ("ASIC") para detecção de bordas em tempo real [245]-[247]. O "chip" utiliza tecnologia CMOS, segue a filosofia de projeto STANDARD CELL e será incluído no 6º Programa Multiusuário Brasileiro (6º PMU).

O desempenho das técnicas de realce, em particular dos filtros espaciais, é função do tipo de ruído existente, do tipo de imagem utilizada e da aplicação envolvida. Faz-se necessário, por parte do usuário, uma avaliação desses aspectos para que seja utilizado o filtro mais adequado.

6. APLICAÇÃO DE FILTROS ESPACIAIS EM CLASSIFICAÇÃO DE IMAGENS MULTIESPECTRAIS

Análise de imagens tem sido largamente empregada em aplicações de Sensoriamento Remoto, visando obter resultados cada vez mais precisos no estudo dos recursos naturais. Vários procedimentos utilizando filtros espaciais têm sido propostos para tarefas que envolvem remoção de ruído, detecção de bordas e classificação de padrões em imagens multiespectrais [248]-[251]. Este capítulo descreve a utilização dos filtros SSDA e ORDAP, estudados na Seção 3.1, no pré-processamento de imagens geradas pelo satélite LANDSAT TM/5, e uma análise de seus efeitos no processo de classificação [41],[45].

6.1 Classificação de Imagens Multiespectrais

Uma imagem multiespectral é, na verdade, uma composição de várias imagens ou planos de imagem, de uma mesma cena natural, obtidos em várias faixas do espectro de freqüência. Os elementos presentes na cena têm reflectâncias distintas para cada faixa ou banda espectral. A energia refletida pelos elementos da cena é captada pelos transdutores de um satélite, digitalizada espacialmente e em intensidade, formando uma matriz de pontos ou plano de imagem. Cada ponto é

representado por um vetor de características cujos elementos representam os atributos da imagem naquela pequena área correspondente à resolução do transdutor utilizado. Um atributo é uma propriedade da imagem que pode ser medida e compreende características como nível de cinza, bordas, textura ou mesmo resultados de manipulação da imagem, como histogramas de freqüência espacial. O atributo mais usado em tarefas de classificação de padrões é o nível de cinza, que representa a resposta espectral dos corpos presentes na cena [252]. Assim, usualmente, cada plano de imagem é considerado um atributo. Seus níveis de cinza constituem, em cada ponto, um elemento do vetor de características do ponto respectivo na composição multiespectral. Estes planos são também comumente chamados de bandas ou canais espectrais.

As tarefas mais comuns de classificação consistem, basicamente, em atribuir um objeto a uma classe dentre diversas classes existentes. Geralmente, os métodos desenvolvidos são baseados em uma formulação estatística e tentam se aproximar do processo de percepção dos seres vivos. Na maioria dos casos, em uma imagem multiespectral, cada ponto é classificado individualmente, levando-se em conta as informações contidas nas bandas ou atributos disponíveis. Os sistemas de classificação mais comuns não levam em conta a informação de correlação espacial entre pontos vizinhos em cada banda da imagem multiespectral, uma vez que cada pixel é classificado individualmente [21],[50].

6.2 Extração de Atributos

Normalmente, é necessário efetuar um processo de

redução de dimensionalidade (seleção de atributos) do espaço de informações disponível, antes da fase de classificação, para reduzir o esforço computacional envolvido. Esta não é uma tarefa fácil, pois é necessário que os atributos escolhidos sejam os melhores para caracterizar as classes de interesse.

Uma maneira de se extrair atributos espaciais é através do pré-processamento das bandas originais com filtros espaciais e, em seguida, submeter o novo espaço obtido a um processo de seleção de atributos. Este novo espaço conterá, além de informações sobre a correlação espectral, informações sobre a correlação espacial realçadas pela atuação do filtro.

6.2.1 Extração de Atributos Espaciais

Uma vez que o comportamento em freqüência de uma imagem expressa a correlação espacial entre pixels, a aplicação de filtros espaciais em imagens multiespectrais pode realçar algumas das relações existentes e gerar novos atributos contendo informações que melhor caracterizam as classes de interesse [44]. A classificação a partir desse novo espaço, mesmo que seja executada ponto a pontoo, levará em conta alguma correlação espacial.

No caso dos filtros espaciais passa-baixas, isto se deve ao fato de estes produzirem, sobre uma imagem qualquer, um efeito de homogeneização que, controlado, pode ser interpretado como uma pré-classificação dos pixels altamente correlacionados, ou seja, daqueles que provavelmente perten-

cem a uma mesma região ou objeto na imagem.

No interior das regiões, este efeito é desejado e produz uma melhor distinção de classes em uma composição multiespectral. No entanto, a suavização nos limites entre regiões ou fronteiras deve ser evitada, pois pode provocar a classificação indevida de pontos intermediários como pertencentes a qualquer uma das regiões adjacentes [21]. A técnica de suavização, portanto, deve ter capacidade de preservação de bordas, a fim de produzir homogeneização, se possível, sem afetar as fronteiras entre regiões.

Neste trabalho, foram utilizados a técnica de suavização com vizinhança selecionada pela soma das diferenças absolutas (SSDA) [48],[183] e o filtro da ordem adaptativo (ORDAP) [49] para gerar atributos espaciais, a partir das bandas 3, 4 e 5 de uma imagem gerada pelo satélite LANSAT TM/5 (LANDSAT Thematic Mapper 5). Estes filtros, estudados na Seção 3.1, apresentam bom desempenho em operações de suavização com preservação de bordas.

6.3 O Algoritmo de Classificação por Máxima Verossimilhança

O método de classificação por máxima verossimilhança (MAXVER), disponível no SITIM, é do tipo estatístico ou paramétrico. Nele, é suposto que o comportamento das classes envolvidas pode ser descrito por funções de densidade de probabilidade gaussianas.

A classificação é realizada ponto a ponto, sem consi-

derar a correlação existente entre pixels vizinhos. Alguns algoritmos, chamados de métodos de decisão composta, incluem parâmetros texturais e espaciais, porém têm eficiência computacional muito menor. O método de decisão por máxima verossimilhança pode ser descrito resumidamente como segue [50]:

Suponha-se que o ponto a ser classificado tem densidade de probabilidade condicional p[x/W(j)], onde x é o vetor de características e W(j) representa a j-ésima classe. O ponto x deve, então, ser atribuído à classe, cuja probabilidade P[W(j)/x] seja máxima, ou seja:

$$P[W(j)/x] = \max [\ p[x/W(j)].P[W(j)])/\ p[x]\ ]$$

onde p[x/W(j)] e p[x] são funções de densidade de probabilidade, e P[W(j)/x] e P[W(j)] são as probabilidades da classe W(j) condicionada e não condicionada aos valores das características, respectivamente.

Para encontrar os valores máximos de P[W(j)/x], é necessário conhecer os parâmetros média e matriz de covariância da distribuição normal das classes nas características. Estes parâmetros são calculados em uma fase de treinamento, que consiste da seleção e análise de pequenas àreas (amostras) da imagem.

A seleção de atributos é baseada em funções que indicam a distância normalizada entre funções de densidade de probabilidade. Quanto maior a distância, menor a probabilidade de erro envolvida entre as classes. A medida de separabilidade estatística é a distância de Jeffrey-Matusita (JM).

6.4 Procedimento

Para a realização do experimento, foi utilizada uma composição colorida das bandas 5, 4 e 3 de uma imagem gerada pelo LANDSAT TM/5, com coordenadas WRS 223.76 (World Reference System). A imagem cobre uma área com uso intensivo e bem distribuído da terra, às margens do rio Ivaí, situada próximo à cidade de Maringá (noroeste do Paraná).

O procedimento consistiu das seguintes etapas:

1. Geração de 5 atributos adicionais, através da filtragem das bandas originais 5, 4 e 3, totalizando 8 canais. Este número de atributos adicionais foi determinado pela capacidade máxima do método de classificação disponível no sistema SITIM-110.

2. Seleção de 3 atributos ou canais, pelos critérios da máxima distância de Jeffreys-Matusita (JM) média e da máxima distância JM mínima.

3. Obtenção das matrizes de classificação para a área em estudo.

4. Classificação, através do método da máxima verossimilhança, da composição escolhida pela seleção de atributos.

Mascarenhas e Dutra [44] realizaram experimento semelhante, utilizando o filtro da média (máscaras 5x5). Naquele trabalho, foi constatado que a percentagem de classificação correta diminuiu nas áreas próximas às fronteiras entre diferentes regiões. Isto decorre do efeito de nublamento

provocado pelo filtro da média nas bordas presentes na imagem. As técnicas não-lineares empregadas neste trabalho têm a propriedade de preservar as bordas existentes.

Da área total da imagem estudada foram selecionadas duas áreas de estudo e criadas duas novas imagens diferentes, denominadas SMAR1 (área de estudo 1) e SMAR2 (área de estudo 2), mostradas em suas composições multiespectrais originais (bandas 5, 4 e 3) nas Figuras 6.1 e 6.2, respectivamente.

Os elementos naturais correspondentes às classes de interesse presentes nas áreas de estudo 1 e 2 são:

soja (classe 1) - Amarelo médio e claro. Representa a área predominante. As variações de tom indicam as diferentes fases da cultura.

cana de açúcar fase 1 (classe 2) - Verde claro. É uma cana caracterizada pela baixa altitude e pequena densidade da folhagem.

cana de açúcar fase 2 (classe 3) - Verde escuro. Cana adulta, de maior altura e mais densa.

água (classe 4) - Azul. Lâmina do rio Ivaí que corta as vegetacões cultivada e nativa.

milho (classe 5) - Carmin. Cultura em diferentes fases de desenvolvimento.

solo nu (classe 6) - Lilás médio e forte. Área praticamente isenta de vegetação.

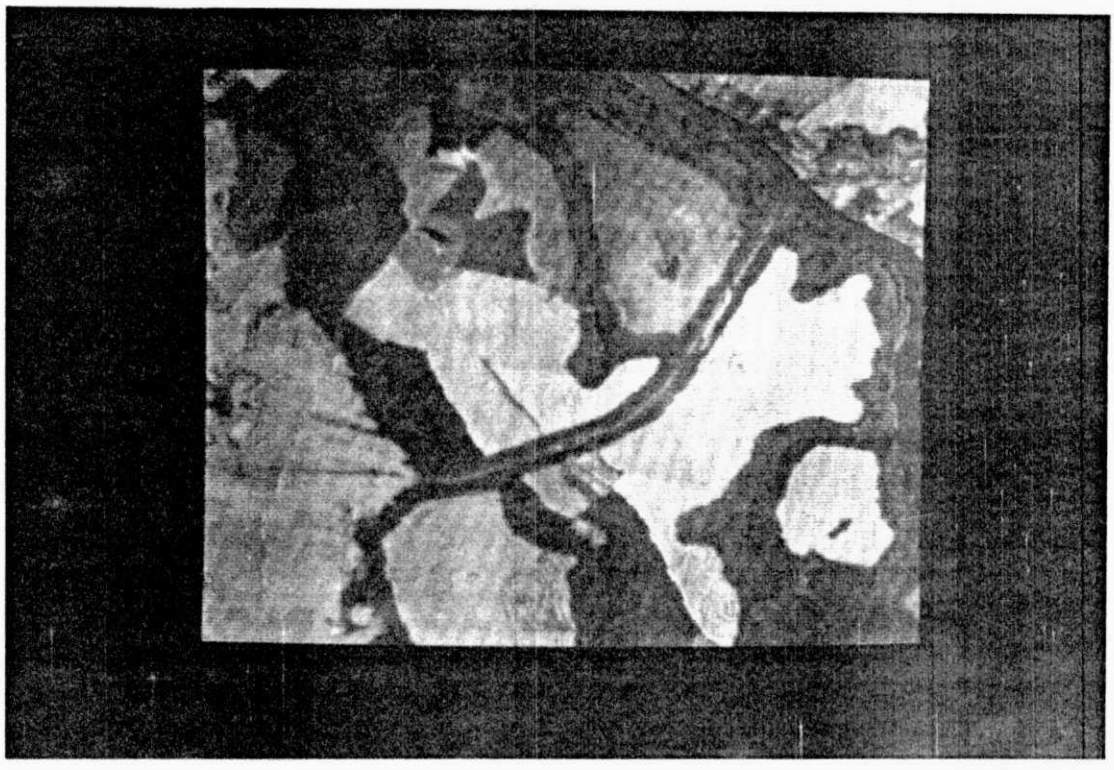

Figura 6.1 Área de estudo 1 (SMAR1). Composição colorida (bandas 5, 4 e 3 do LANDSAT TM/5)

Figura 6.2 Área de estudo 2 (SMAR2). Composição colorida (bandas 5, 4 e 3 do LANDSAT TM/5)

mata (classe 7) - Marron esverdeado. Grande área de vegetação nativa, densa, de média altitude e diversificada em espécie.

6.5 Resultados

Para a área de estudo 1, foram utilizados os seguintes canais:

- canais 5, 4 e 3 (canais 5, 4 e 3 originais do LANDSAT);
- canais i, j e k (canais 5, 4 e 3 originais e pré-processados pelo algoritmo de suavização com vizinhança selecionada por soma de diferenças absolutas, iteração 0);
- canais l e m (canais 5 e 4 originais pré-processados pelo filtro da ordem adaptativo, iteração 0).

As Figuras 6.3 a 6.10 apresentam os atributos 5, 4, 3, i, j, k, l e m, respectivamente, para a área de estudo 1.

Foi utilizado o algoritmo de seleção de atributos, baseado nos critérios da máxima distância JM média e máxima distância JM mínima, para obter um subconjunto com três canais. Esta quantidade foi determinada pela limitação do sistema de processamento de imagens usado (SITIM-110). Os resultados são mostrados na Tabela 6.1 e os canais escolhidos foram l, m e k, correspondentes aos canais 5 e 4 do LANSAT, pré-processados por ORDAP, e ao canal 3, filtrado por SSDA.

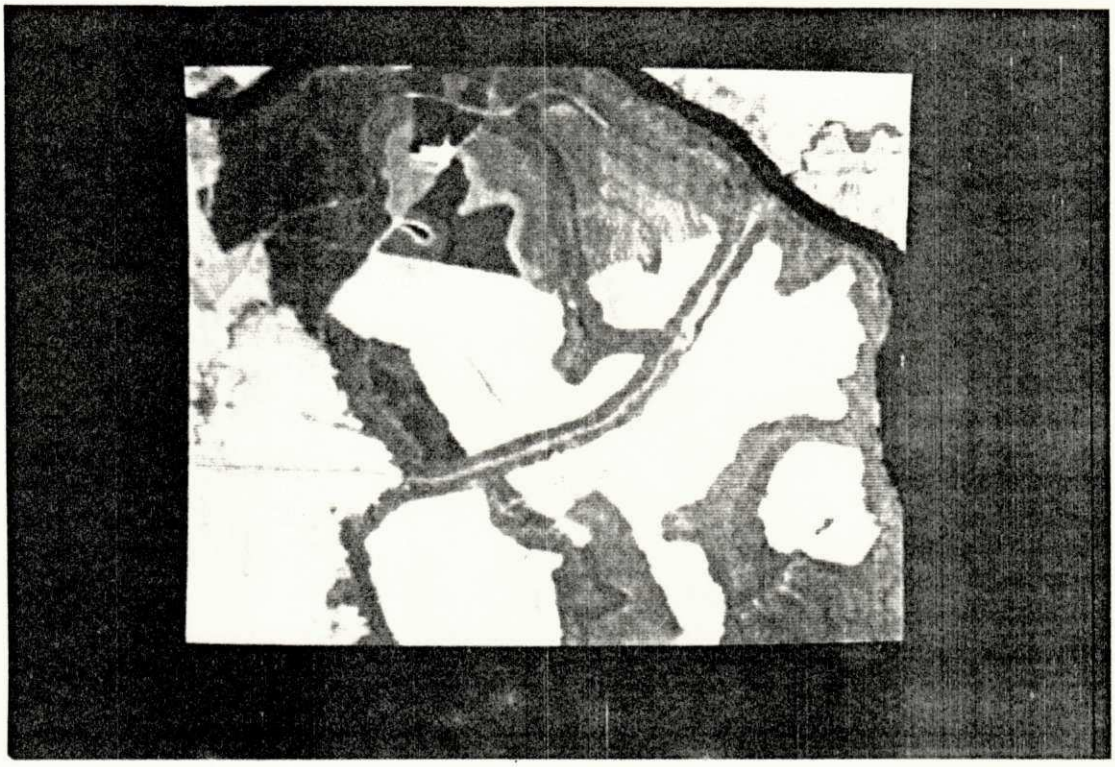

Figura 6.3 Imagem SMAR1 (banda 5 do LANDSAT TM/5)

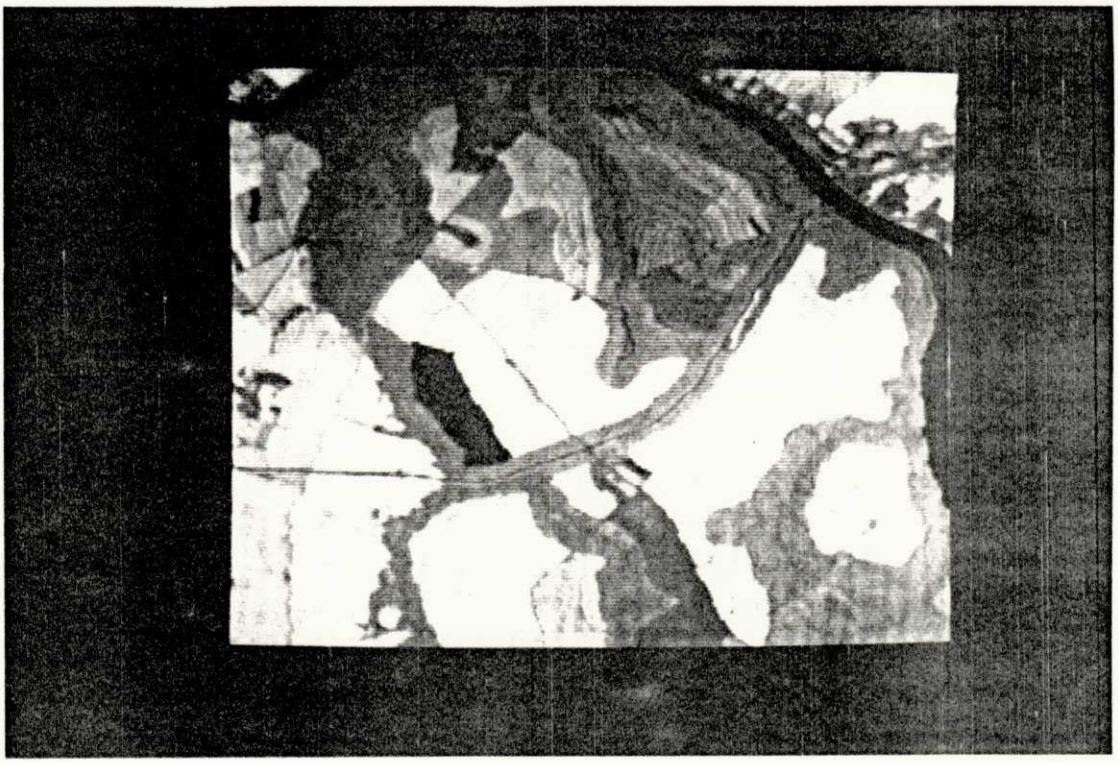

Figura 6.4 Imagem SMAR2 (banda 4 do LANDSAT TM/5)

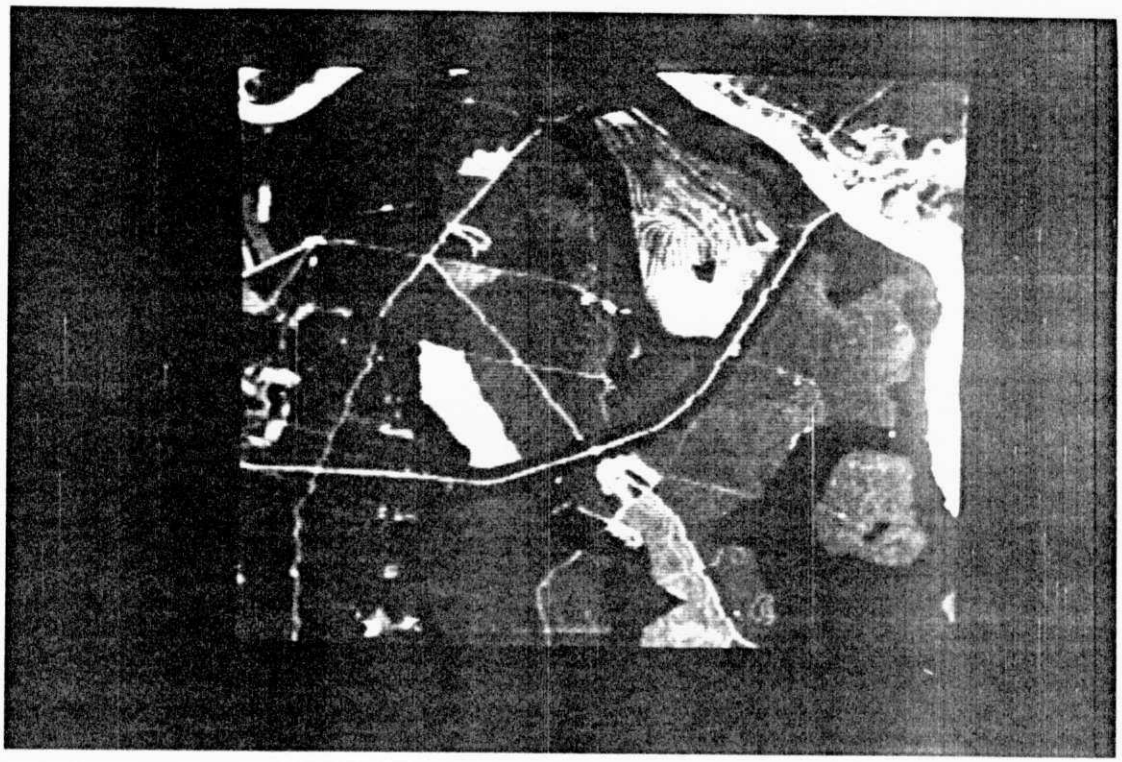

Figura 6.5 Imagem SMAR1 (banda 3 do LANDSAT TM/5)

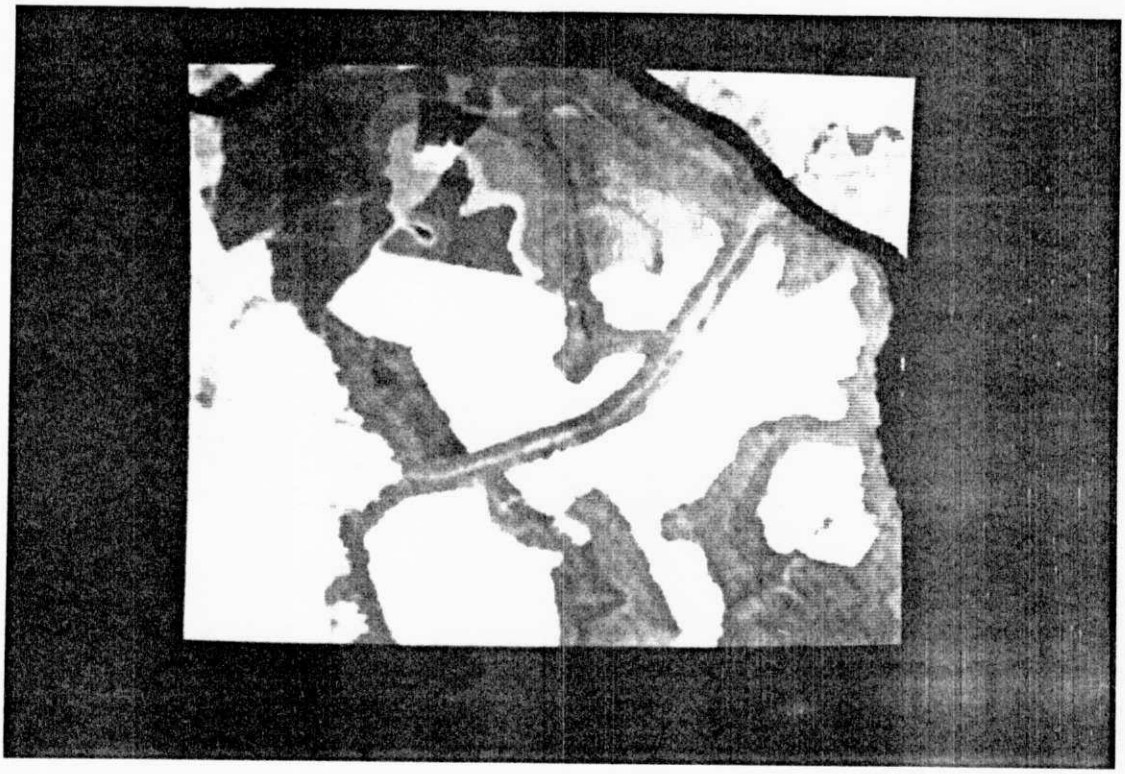

Figura 6.6 Imagem SMAR1 (banda 5 do LANDSAT TM/5 filtrada por SSDA)

Figura 6.7 Imagem SMAR1 (banda 4 do LANDSAT TM/5 filtrada por SSDA)

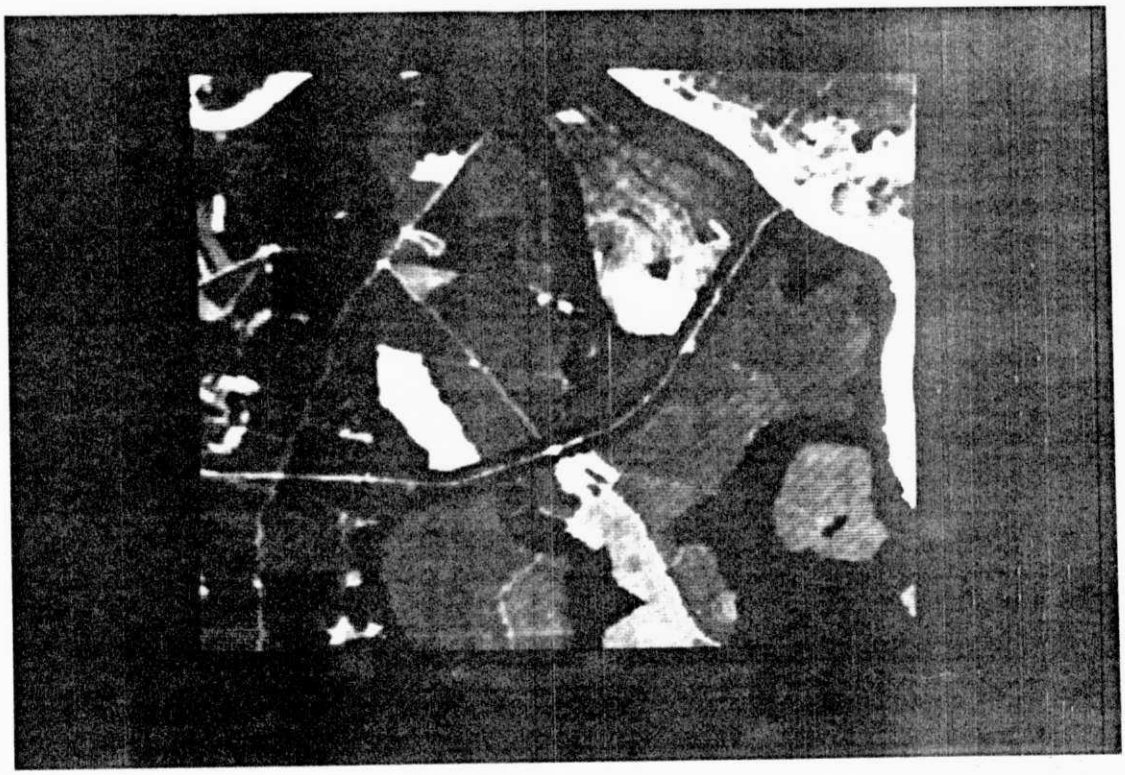

Figura 6.8 Imagem SMAR1 (banda 3 do LANDSAT TM/5 filtrada por SSDA)

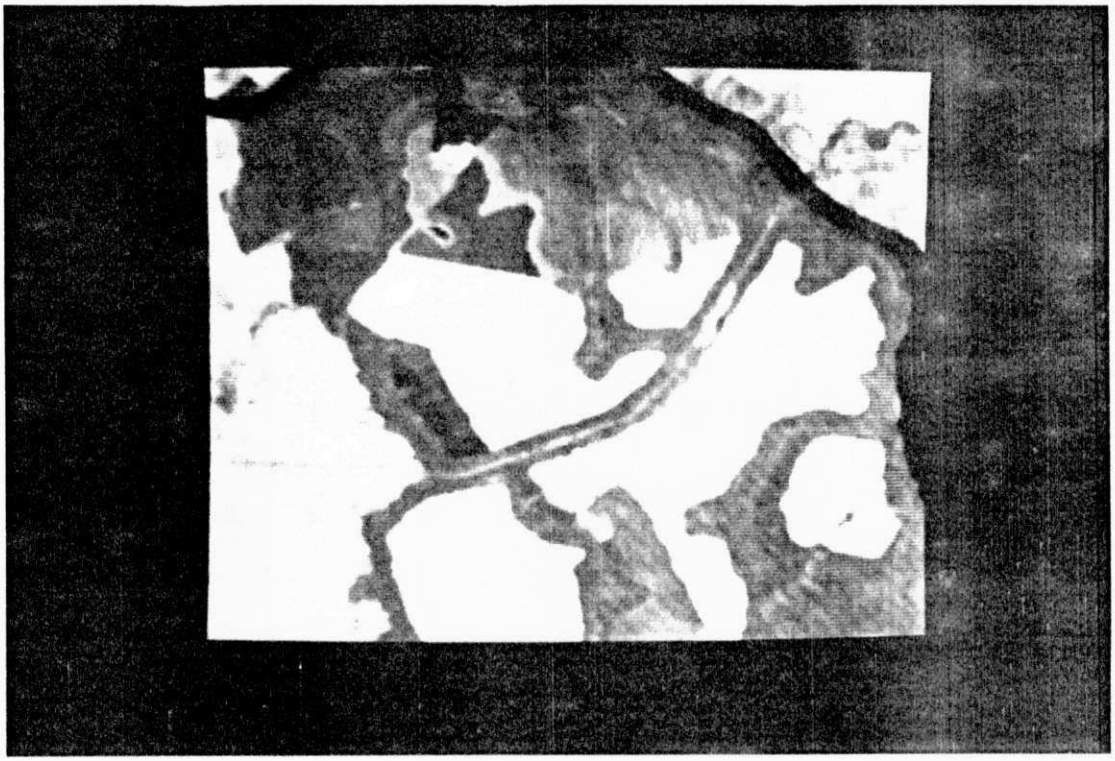

Figura 6.9 Imagem SMAR1 (banda 5 do LANDSAT TM/5 filtrada por ORDAP)

Figura 6.10 Imagem SMAR1 (banda 4 do LANDSAT TM/5 filtrada por ORDAP)

Em seguida, foram obtidas as matrizes de classificação das áreas de treinamento para as composições multiespectrais 543 e lmk (Tabelas 6.2 e 6.3). Áreas de treinamento são áreas de identificação conhecida e que foram utilizadas na fase de treinamento.

Foi utilizado um limiar de 6,20, tornando a classificação mais rigorosa no sentido de rejeitar os pontos cujas características não coincidam perfeitamente com as das classes de interesse.

As matrizes de classificação fornecem uma medida percentual da eficiência de todo o processo de classificação. Elas indicam os erros cometidos, ao classificarem-se pontos de identidade conhecida, e permitem estimar os erros envolvidos na classificação das àreas não utilizadas para a extração de parâmetros (fase de treinamento). Para ilustrar este fato, consulte-se, por exemplo, a matriz da composição 543 (Tabela 6.2). Pode ser observado que 0,6 % da informação correspondente à soja (classe 6) foi classificada como sendo cana na fase 1 (classe 2). Na composição lmk (vide labela 6.3), este erro caiu para 0,4 %. Comparando-se as matrizes, pode ser notado que a classificação da composição lmk apresenta menores índices de erro para todas as classes.

As colunas identificadas nas matrizes pelo rótulo nclas ("não classe") indicam os percentuais de pontos classificados como não pertencentes a nenhuma classe de interesse. Estes valores representam o índice de abstenção da classificação.

A Tabela 6.4 mostra os índices médios relativos às matrizes de classificação, fornecidos pelo MAXVER. O desempenho médio indica o percentual de classificação correta,

calculado com base nos percentuais de todas as classes e ponderado pelo número de pontos das áreas de treinamento. A abstenção média indica o percentual de pontos não classificados e a confusão média representa os erros cometidos na classificação. Estes valores comprovam um significativo acréscimo na precisão da classificação para a composição com as bandas filtradas l, m e k.

Finalmente, foi procedida a classificação das composições original (bandas 5, 4 e 3) e pré-processada (bandas l, m e k). As imagens resultantes são mostradas nas figuras 6.11 e 6.12, respectivamente. A Tabela 6.5 apresenta os resultados da classificação em unidades de área para cada classe.

A propriedade de aguçamento de bordas dos filtros empregados contribuiu para uma melhor discriminação dos

ÁREA DE ESTUDO 1

	Composição	Distância JM Média	Distância JM Mínima	Classes com mínima distância na composição
Composições que maximizam a distância JM Média	lmk	1,414208	1,414121	7 e 2
	ljk	1,414206	1,414114	7 e 2
	ijk	1,414196	1,413912	7 e 2
Composições que maximizam a distância JM Mínima	lmk	1,414206	1,414121	7 e 2
	imk	1,414191	1,414114	7 e 2
	ijk	1,414196	1,413912	7 e 2

Tabela 6.1 Seleção de atributos para imagem SMAR1 pelos critérios de máximas distâncias JM média e mínima.

%		ÁREA DE ESTUDO 1	COMPOSIÇÃO 543		LIMIAR 6,20			
CLASSES		nclas	1	2	3	4	5	6	7
1	agua	4,3	95,7	-	-	-	-	-	-
2	cana 1	10,8	-	89,2	-	-	-	-	-
3	cana 2	10,7	-	-	89,3	-	-	-	-
4	milho	6,3	-	-	-	93,8	-	-	-
5	solo nu	10,6	-	-	-	-	89,4	-	-
6	soja	5,3	-	0,6	0,5	-	0,9	93,0	0,1
7	mata	2,5	-	0,1	-	0,4	0,1	2,0	94,9
INDICES MEDIOS Desempenho: 93,19 Abstenção: 5,13 Confusão: 5,68									

Tabela 6.2 Matriz de classificação para a imagem SMAR1 com composição original 543 (método MAXVER)

%		ÁREA DE ESTUDO 1	COMPOSIÇÃO lmk		LIMIAR 6,20			
CLASSES		nclas	1	2	3	4	5	6	7
1	agua	3,2	96,8	-	-	-	-	-	-
2	cana 1	8,0	-	92,0	-	-	-	-	-
3	cana 2	9,5	-	-	90,5	-	-	-	-
4	milho	6,3	-	-	-	93,8	-	-	-
5	solo nu	6,6	-	-	-	-	93,4	-	-
6	soja	4,1	-	0,4	0,1	-	0,7	94,7	0,1
7	mata	1,6	-	0,1	-	0,4	-	2,0	95,8
INDICES MEDIOS Desempenho: 94,70 Abstenção: 3,89 Confusão: 1,41									

Tabela 6.3 Matriz de classificação para a imagem SMAR1 com composição lmk (método MAXVER)

	ÁREA DE ESTUDO 1	
	COMPOSIÇÃO 543	COMPOSIÇÃO km1
Desempenho médio (%)	93.19	94.70
Abstenção média (%)	5.13	3.89
Confusão média (%)	5.68	1.41

Tabela 6.4 Indices médios da classificação de SMAR1 (Desempenho, Abstenção e Confusão)

	ÁREA DE ESTUDO 1 (área em km^2)	
CLASSES	COMPOSIÇÃO 543	COMPOSIÇÃO 1mk
agua	1,3	1,3
cana 1	1,5	1,0
cana 2	0,6	0,5
milho	0,7	0,7
solo nu	1,1	0,9
soja	22,6	22,0
mata	17,2	16,5
nao classe	14,0	16,2

Tabela 6.5 Resultados da classificação de SMAR1 em unidades de área.

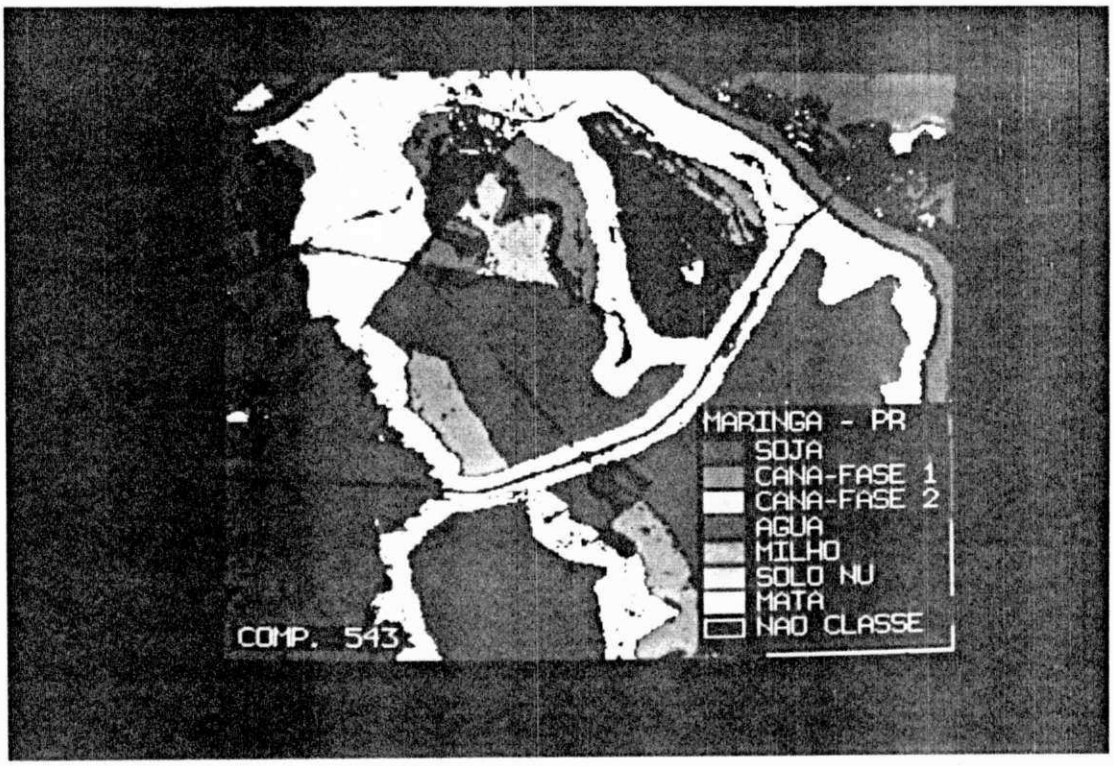

Figura 6.11 Imagem SMAR1 composição 543 classificada

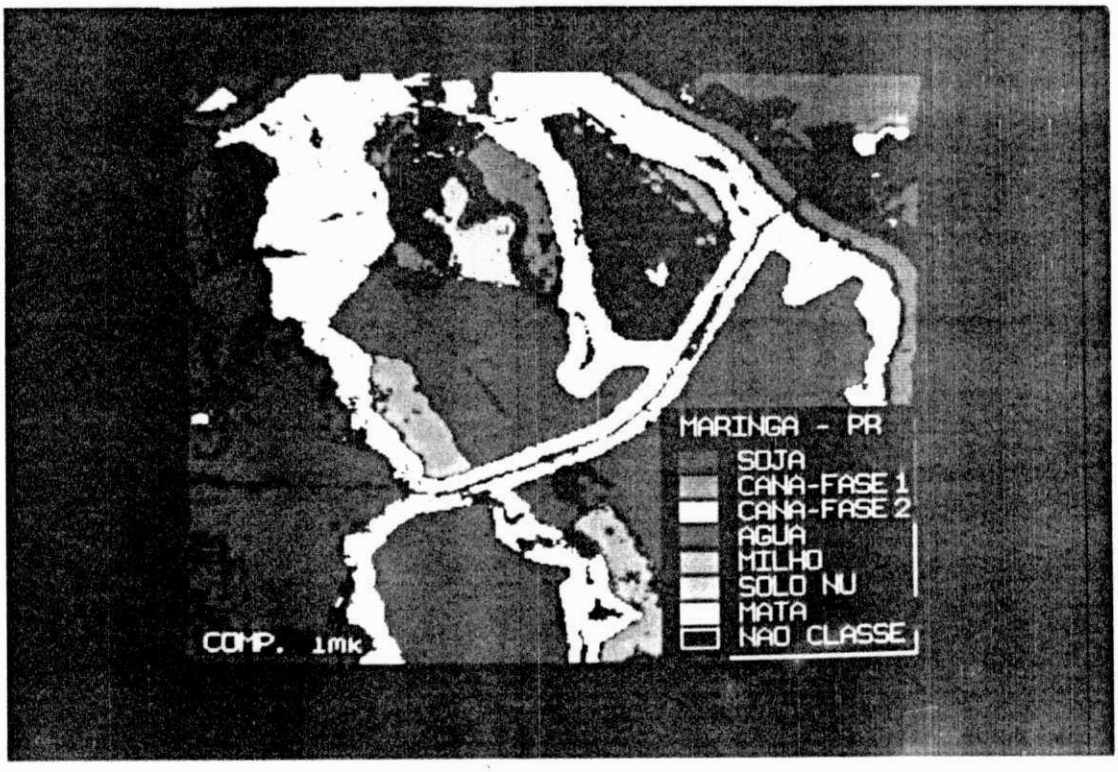

Figura 6.12 Imagem SMAR1 composição 1mk classificada

pontos próximos às fronteiras. Isto é mostrado pelo estreitamento das regiões de fronteira na classificação da composição lmk.

Os pontos situados exatamente nas fronteiras não foram classificados. Este é um resultado positivo, pois tais pontos pertencem a regiões de transição de textura, que normalmente representam formações naturais (solo, vegetação, etc.) também de transição, as quais não pertencem a nenhuma das regiões vizinhas. Em outras palavras, os pontos de borda são preferencialmente não classificados, ao invés de serem classificados erroneamente. Esta abstenção demonstra que a classificação se tornou mais discriminatória e, conseqüentemente, mais precisa.

O experimento foi repetido sob as mesmas condições para a área de estudo 2 (imagem SMAR2).

Foram utilizados os seguintes canais:

- canais 5, 4 e 3 (canais 5, 4 e 3 originais do LANDSAT);
- canais i, j e k (canais 5, 4 e 3 originais e pré-processados pelo algoritmo de suavização com vizinhança selecionada por soma de diferenças absolutas, iteração 0);
- canais l e m (canais 5 e 4 originais pré-processados pelo filtro da ordem adaptativo, N=2, J=1, iteração 0).

As Figuras 6.13 a 6.20 apresentam os atributos 5, 4, 3, i, j, k, l e m, respectivamente, para a área de estudo 2. A seleção de atributos elegeu, através dos mesmos critérios a composição lmk como sendo a mais discriminatória

(vide Tabela 6.6).

As novas matrizes de classificação (Tabelas 6.7 e 6.8) e os índices médios (Tabela 6.9) obtidos, mostram, como no primeiro experimento, um aumento da precisão da classificação para a composição lmk, em relação à composição original.

As imagens classificadas com base nas composições original e filtradas (lmk) são mostradas nas figuras 6.21 e 6.22 respectivamente. A Tabela 6.10 apresenta o resultado da classificação em unidades de área.

Figura 6.13 Imagem SMAR2 (banda 5 do LANDSAT TM/5)

Figura 6.14 Imagem SMAR2 (banda 4 do LANDSAT TM/5)

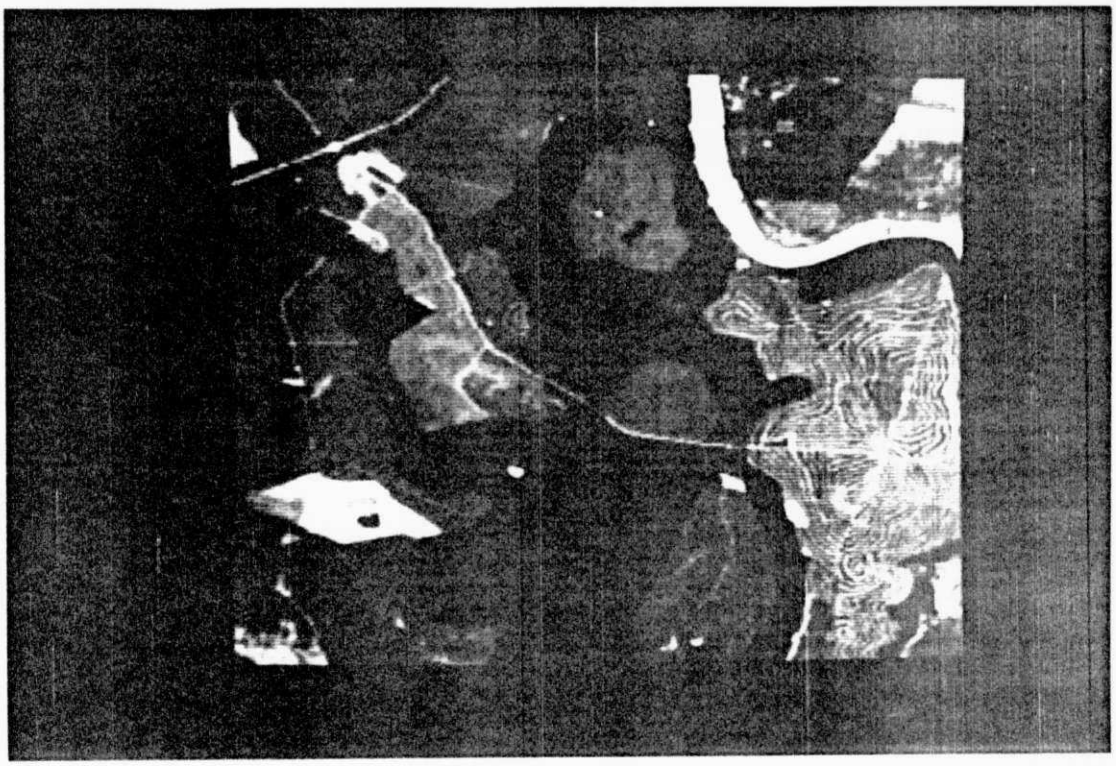

Figura 6.15 Imagem SMAR2 (banda 3 do LANDSAT TM/5)

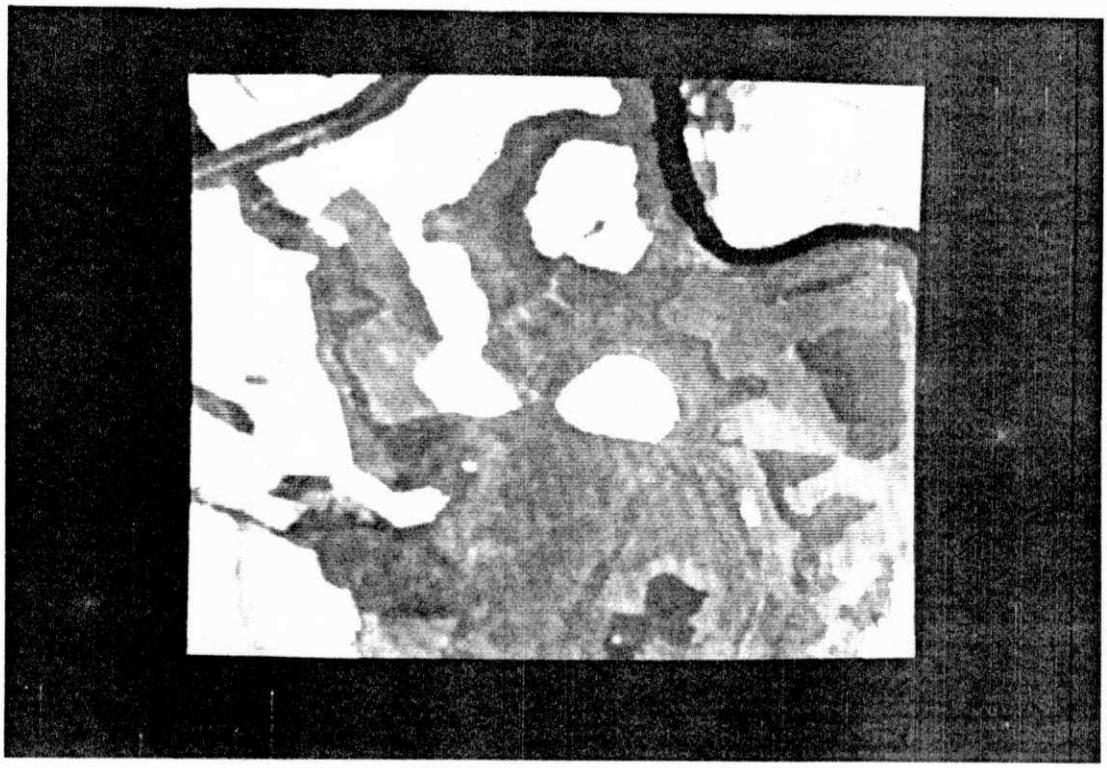

Figura 6.16 Imagem SMAR2 (banda 5 do LANDSAT TM/5 filtrada por SSDA)

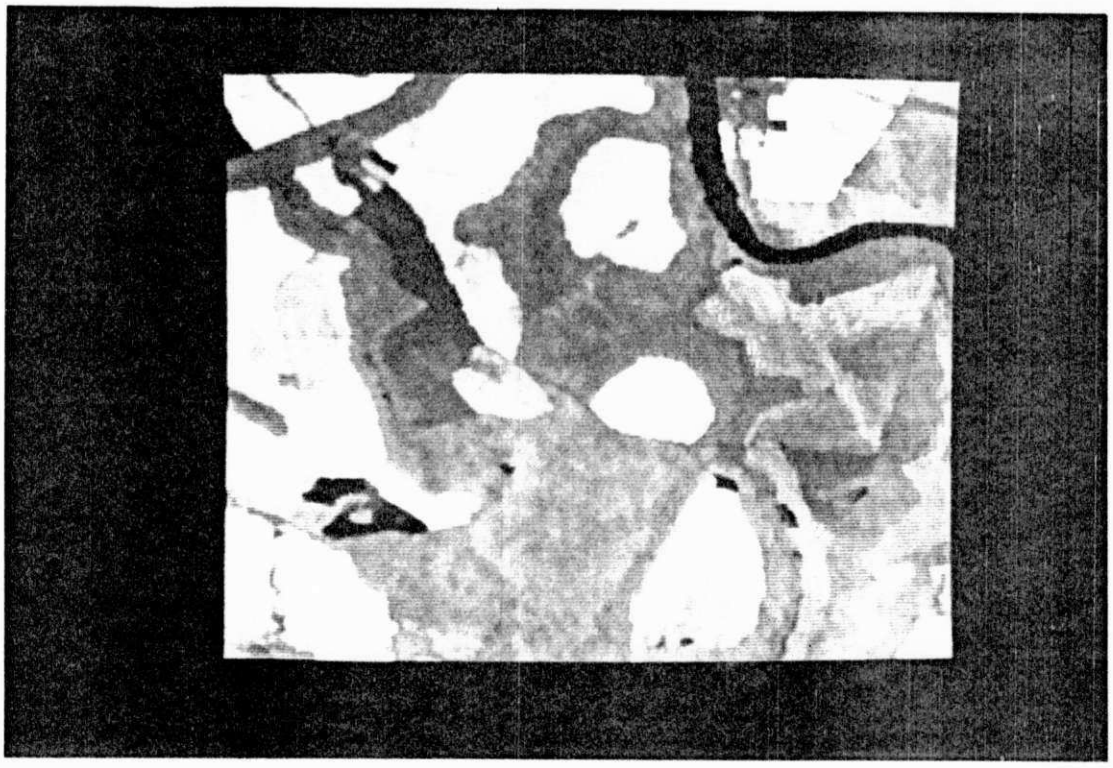

Figura 6.17 Imagem SMAR2 (banda 4 do LANDSAT TM/5 filtrada por SSDA)

Figura 6.18 Imagem SMAR2 (banda 3 do LANDSAT TM/5 filtrada por SSDA)

Figura 6.19 Imagem SMAR2 (banda 5 do LANDSAT TM/5 filtrada por ORDAP, N=2, J=1)

Figura 6.20 Imagem SMAR2 (banda 4 do LANDSAT TM/5 filtrada por ORDAP, N=2, J=1)

AREA DE ESTUDO 2

	Composição	Distância JM Média	Distância JM Mínima	Classes com mínima distância na composição
Composições que maximizam a distância JM Média	lmk	1,106913	1,106882	7 e 2
	ljk	1,106912	1,106869	7 e 2
	imk	1,106901	1,106871	7 e 2
Composições que maximizam a distância JM Mínima	lmk	1,106913	1,106882	7 e 2
	imk	1,106901	1,106871	7 e 2
	ijk	1,106912	1,106869	7 e 2

Tabela 6.6 Seleção de atributos para imagem SMAR2 pelos critérios de máximas distâncias JM média e mínima.

%		ÁREA DE ESTUDO 2	COMPOSIÇÃO 543		LIMIAR 6,20			
CLASSES		nclas	1	2	3	4	5	6	7
1	agua	4,8	95,2	-	-	-	-	-	-
2	cana 1	9,6	-	90,4	-	-	-	-	-
3	cana 2	9,4	-	0,5	90,1	-	-	-	-
4	milho	5,6	-	-	-	94,4	-	-	-
5	solo nu	4,1	-	-	-	-	90,6	5,4	-
6	soja	5,1	-	-	-	-	0,1	94,2	0,6
7	mata	10,4	-	-	-	-	-	-	89,6
INDICES MÉDIOS Desempenho: 92,24 Abstenção: 6,61 Confusão: 1,15									

Tabela 6.7 Matriz de classificação para a imagem SMAR2 com composição original 543 (método MAXVER)

%		ÁREA DE ESTUDO 2	COMPOSIÇÃO lmk		LIMIAR 6,20			
CLASSES		nclas	1	2	3	4	5	6	7
1	agua	4,8	95,2	-	-	-	-	-	-
2	cana 1	8,8	-	91,2	-	-	-	-	-
3	cana 2	10,3	-	-	89,3	-	-	-	-
4	milho	5,0	-	-	-	95,0	-	-	-
5	solo nu	3,4	-	-	-	-	91,2	5,4	-
6	soja	4,2	-	-	-	-	0,1	95,6	0,1
7	mata	9,8	-	-	-	-	-	-	90,2
INDICES MÉDIOS Desempenho: 93,21 Abstenção: 5,90 Confusão: 0,89									

Tabela 6.8 Matriz de classificação para a imagem SMAR2 com composição lmk (método MAXVER)

	ÁREA DE ESTUDO 2	
	COMPOSIÇÃO 543	COMPOSIÇÃO lmk
Desempenho médio (%)	**92.24**	**93.21**
Abstenção média (%)	**6.61**	**5.90**
Confusão média (%)	**1.15**	**0.89**

Tabela 6.9 Índices médios da classificação de SMAR2 (Desempenho, Abstenção e Confusão)

	ÁREA DE ESTUDO 2 (área em km^2)	
CLASSES	COMPOSIÇÃO 543	COMPOSIÇÃO lmk
agua	**0,8**	**0,8**
cana 1	**2,1**	**1,4**
cana 2	**1,2**	**0,6**
milho	**0,4**	**0,4**
solo nu	**3,6**	**2,6**
soja	**14,6**	**14,6**
mata	**17,4**	**14,1**
nao classe	**19,0**	**24,4**

Tabela 6.10 Resultados da classificação de SMAR2 em unidades de área.

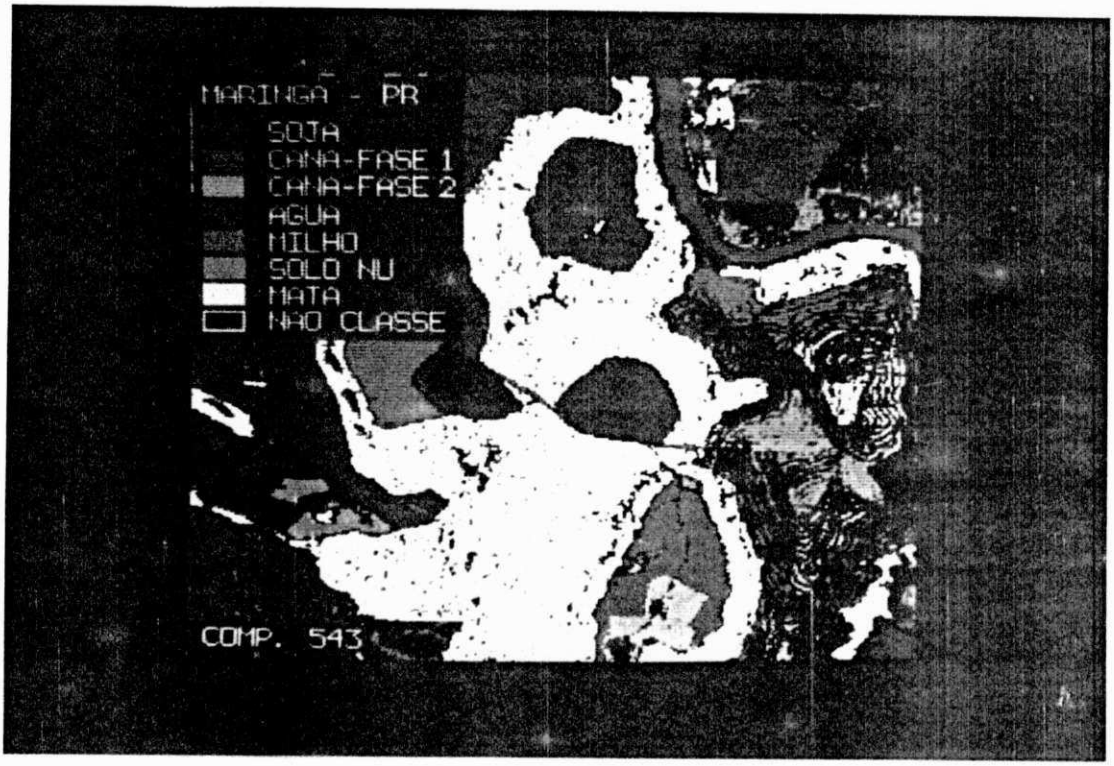

Figura 6.21 Imagem SMAR2 composição 543 classificada

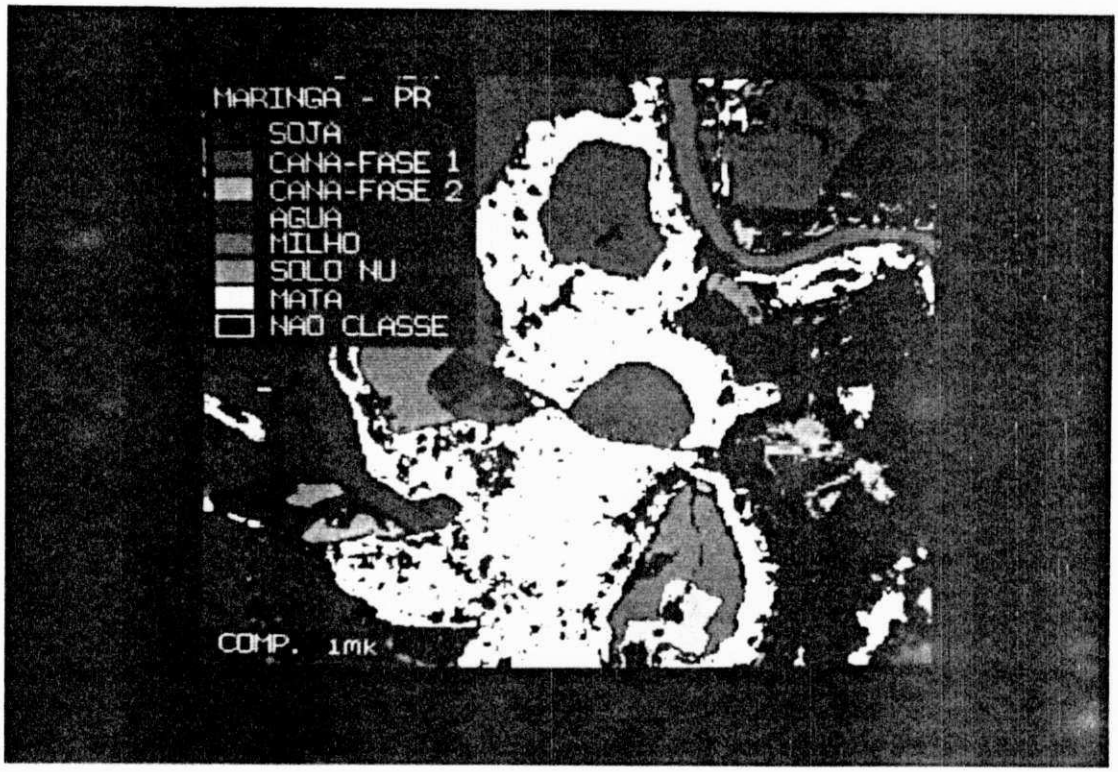

Figura 6.22 Imagem SMAR2 composição 1mk classificada

6.6 Conclusão

A aplicação da técnica de suavização com vizinhança selecionada por soma das diferenças absolutas e do filtro da ordem adaptativo em imagens geradas pelo satélite LANDSAT TM/5 produziu um aumento na precisão da classificação pelo método da máxima verossimilhança. A suavização com preservação de bordas aumentou, nos experimentos realizados, o poder de discriminação do método MAXVER. Os resultados iniciais indicam que o pré-processamento de imagens multiespectrais, utilizando técnicas de suavização que preservem bordas, pode contribuir para um aumento de precisão em tarefas de classificações de padrões. Este experimento motivou a realização de outros estudos envolvendo a aplicação de técnicas adaptativas de filtragem espacial, em aplicações de sensoriamento remoto [46],[47].

7. CONCLUSÃO

Esta dissertação trata das técnicas de filtragem de imagens no domínio espacial. O trabalho teve, como metas principais, estudar as técnicas de filtragem espacial correntes na literatura, gerar uma fonte bibliográfica atualizada, através de consultas à literatura disponível, implementar um pacote de "software" dedicado à filtragem espacial e adaptado para ambiente baseado em microcomputador, avaliar o desempenho de alguns filtros espaciais passa-baixas e passa-altas, e aplicar algumas técnicas baseadas no critério de vizinhança seletiva e filtros da ordem adaptativos no pré-processamento de imagens multiespectrais, em tarefas de classificação.

As atividades desenvolvidas fazem parte de uma continuação e atualização do trabalho realizado por Araújo [40] nesta área de Processamento Digital de Imagens.

7.1 Visão Geral do Trabalho

Técnicas de filtragem espacial têm sido cada vez mais utilizadas no pré-processamento de imagens. Novos métodos são desenvolvidos rapidamente, envolvendo campos da ciência, como matemática, estatística e neurofisiologia. O desenvolvimento do processamento digital de imagens e, em

particular, das técnicas de filtragem no domínio espacial, foi abordado em seus vários aspectos no capítulo 1.

A remoção de ruído em imagens digitais é uma tarefa fundamental e normalmente exige a consideração de propriedades inerentes do ruído, de características da imagem processada e do tipo de aplicação envolvida. No caso de deteção de bordas, vários critérios semelhantes também devem ser levados em conta. Em termos de desempenho, as técnicas implementadas em FILTRIX apresentam comportamentos distintos. Os algoritmos mostram-se mais ou menos eficientes em aspectos como: eliminação de ruído impulsivo, eliminação de ruído gaussiano aditivo, suavização com preservação de bordas, aguçamento de bordas, deteção de bordas, geração de imagens-gradiente, etc. Nos capítulo 2 e 3, foram estudadas as técnicas lineares e não lineares implementadas e suas características principais.

O rápido crescimento do processamento digital de imagens tem motivado o desenvolvimento de pequenas estações de trabalho baseadas em microcomputadores, mais acessíveis a universidades e instituições de pesquisa. Acompanhando esta tendência, foi proposto e desenvolvido, em linguagem C, o pacote FILTRIX, contendo filtros espaciais passa-baixas e passa-altas, além de métodos estatísticos e gráficos destinados ao estudo dos efeitos dos algoritmos. Os aspectos envolvidos com a implementação, configuração, técnicas de manipulação de dados, compatibilidade e interface homem-máquina foram apresentados no capítulo 4.

A avaliação do desempenho de filtros espaciais pode ser realizada através de métodos estatísticos, gráficos, ou de análise visual. Um estudo comparativo envolvendo técnicas tradicionals e técnicas recentemente desenvolvidas e publi-

cadas foi realizado. Um procedimento simples baseado na aplicação dos algoritmos em uma imagem padrão com ruído foi utilizado. Perfis de uma linha da imagem modelo e medidas do desvio padrão em área homogênea e do erro médio quadrático foram empregados para verificar os diferentes efeitos da suavização produzidos pelas técnicas aplicadas. Estes resultados foram mostrados no capítulo 5.

A sensibilidade de alguns métodos de detecção de bordas em tarefas de geração de imagens-gradiente foi avaliada.

O efeito de homogeneização com preservação de bordas produzido por algumas técnicas não lineares foi utilizado com o objetivo de aumentar a precisão em tarefas de classificação de imagens multiespectrais. Os experimentos demonstrando os resultados positivos deste pré-processamento com os algoritmos de suavização com vizinhança selecionada por soma das diferenças absolutas e o filtro da ordem adaptativo foram apresentados também no capítulo 6.

Durante o desenvolvimento deste trabalho, vários estudos e experimentos foram realizados, envolvendo a aplicação de filtros espaciais [41]-[43],[45]-[47],[243]-[247].

7.2 Considerações Finais

A revisão bibliográfica realizada mostra o desenvolvimento das técnicas de filtragem espacial e fornece uma relação considerável de referências, abordando fundamentos, implementações e aplicações, para estudos mais aprofundados

sobre a área.

Seguindo a preocupação atual de concepção de pequenas estações de trabalho para processamento digital de imagens, baseadas em microprocessadores, foi proposto e implementado em linguagem C um pacote de "software" dedicado à filtragem espacial de imagens, denominado FILTRIX. Este "software" inclui, além dos filtros passa-baixas e passa-altas, algoritmos para avaliação de seu desempenho e apresenta-se como uma ferramenta eficiente para aplicações didáticas e de projeto, envolvendo o tratamento de imagens digitais.

Algumas técnicas não lineares disponíveis em FILTRIX proporcionam uma homogeneização de regiões com preservação de bordas. Este efeito produziu um sensível aumento de precisão na classificação de imagens multiespectrais pré-processadas pelos algoritmos SSDA e filtro da ordem adaptativo. O método de classificação por máxima verossimilhança apresentou maior índice de desempenho médio e menores índices de abstenção e de confusão média, em relação à classificação da composição multiespectral original. O experimento mostra também que outras técnicas de suavização capazes de preservar ou aguçar bordas, podem ser utilizadas com o mesmo objetivo.

FILTRIX pode ser utilizado como ferramenta de pré-processamento em diversas tarefas de processamento digital de imagens. Sua aplicação na análise de imagens geradas na medicina, por exemplo, pode contribuir sensivelmente para a obtenção de resultados mais precisos em tarefas que envolvam o reconhecimeneto de padrões e a interpretação visual.

Considera-se alcançado o objetivo de dar continuidade e atualizar o trabalho realizado por Araújo [40].

Durante a realização deste trabalho, foram encontrados problemas de várias espécies e realizadas tarefas envolvendo várias áreas do conhecimento. Este processo teve uma contribuição significativa para a formação científica do Autor.

7.3 Sugestões para Trabalho Futuro

Com o rápido desenvolvimento das técnicas de filtragem espacial, é importante que sejam implementados novos algoritmos a fim de atualizar FILTRIX e aumentar seu espaço de aplicações.

Ainda com o objetivo de aprimorar o pacote podem ser implementados métodos para análise do comportamento dos filtros no domínio da frequência.

Para dar continuidade a este trabalho, sugerem-se ainda os seguintes pontos:

- Adaptação do "software" FILTRIX ao ambiente MS-DOS para sua utilização na versão SITIM-150;

- Realização de estudo comparativo do desempenho dos filtros no domínio da freqüência;

- Realização de experimentos utilizando filtros espaciais no pré-processamento de imagens em tarefas de segmentação;

- Realização de experimentos utilizando o filtro sigma

no pré-processamento de imagens multiespectrais em tarefas de classificação. Esta técnica de suavização produz uma homogeneização controlada pelo valor estimado do desvio padrão em uma pequena vizinhança;

- Utilização dos recursos de FILTRIX no pré-processamento de imagens geradas na medicina.

REFERÊNCIAS BIBLIOGRÁFICAS

1 - B. Girod. The digital image processing revolution. Siemens Review, 56, 1989, 33-36.

2 - B. S. Lipkin and A. Rosenfeld. **Picture Processing and Psychopictorics**. Academic Press, 1970

3 - T. S. Huang, Ed. **Picture Processing and Digital Filtering**. Spring-Verlag, 1975.

4 - R. O. Duda and P. E. Hart. **Pattern Classification and Scene Analysis**. John Wiley & Sons, 1973.

5 - A. Rosenfeld and A. C. Kak. **Digital Picture Processing**. Academic Press, 1982.

6 - R. C. Gonzalez and P. Wintz. **Digital Image Processing**. Addison-Wesley, 1987.

7 - W. K. Pratt. **Digital Image Processing**. John Wiley & Sons, 1978.

8 - L. E. Larsen and J. H. Jacobi. **Medical Applications of Microwave Imaging**. IEEE Press, 1985.

9 - R. Chellappa and A. Sawchuk. **Digital Image Processing and Analysis: Volume 2 - Digital Image Analysis: Tutorial**. IEEE Computer Soc. Press, 1986.

10 - R. Chellappa and A. A. Sawchuk. **Digital Image Processing and Analysis: Volume 1 - Digital Image Processing**. IEEE Computer Soc. Press, 1985.

11 - H. Lee and G. Wade. **Imaging Technology**. IEEE Press, 1985.

12 - H. Lee and G. Wade. **Modern Acoustical Imaging**. IEEE Press, 1986.

13 - A. C. Kak and M. Slaney. **Principles of Computerized Tomographic Imaging**. IEEE Press, 1988.

14 - J. K. Aggarwal, R. O. Duda and A. Rosenfeld. **Computer Methods in Image Analysis**. IEEE Press, 1977.

15 - R. Bemstein. **Digital Image Processing for Remote Sensing**. IEEE Press, 1978.

16 - K. S. Fu and T. L. Kunii. **Picture Engineering**. IEEE Press, 1982.

17 - T. S. Huang. **Two-Dimensional Digital Signal Processing I - Linear Filters**. IEEE Press, 1981.

18 - T. S. Huang. **Two-Dimensional Digital Signal Processing II - Transform and Median Filters**. IEEE Press, 1981.

19 - L. Bolk and Z. Kulpa. **Digital Image Processing Systems**. IEEE Press, 1981.

20 - L. P. Yaroslavsky. **Digital Picture Processing. An Introduction**. IEEE Press, 1985.

21 - J. A. Richards. **Remote Sensing Digital Image Analysis. An Introduction**. IEEE Press, 1986.

22 - J. S. Lim. **Two-Dimensional Signal and Image Processing**. Prentice Hall, 1989.

23 - H. C. Andrews and B. R. Hunt. **Digital Image Restoration**. Prentice Hall, 1977.

24 - K. R. Castleman. **Digital Image Processing**. Prentice Hall, 1979.

25 - S. Haykin. **Array Signal Processing**. Prentice Hall, 1985.

26 - G. S. Kino. **Acoustic Waves: Devices, Imaging and Analog Signal Processing**. Prentice Hall, 1987.

27 - N. D. A. Mascarenhas e F. R. D. Velasco. **Processamento digital de imagens**. Editorial KAPELUSZ S.A., 1989.

28 - L. E. Ravich. 1989 Trends in image processing technology. Laser Focus Word, 25, 1989, 159-186.

29 - I. Hirschberg. 1989 Trends in electronic imaging. Laser Focus Word, 25, 1989, 150-156.

30 - J. Macmichael. Optical RAMs for low-cost monochromatic imaging. New Electronics, 20, 1987, 28-29.

31 - V. S. Frost, J. A. Stiles, K. S. Shanmugan, and J. C. Holtzman. A model for radar images and its

application to adaptive digital filtering of multiplicative noise. IEEE Trans. on Pattern Analysis and Machine Intelligence, 4, 1982, 157-166.

32 - R. E. Crippen. A simple spatial filtering routine for the cosmetic removal of scan-line noise from LANDSAT-TM P-tape imagery. Photogrammetric Engineering and Remote Sensing, 55, 1989, 327-333.

33 - T. J. Uhl. Linear and nonlinear restoration methods in comparison. Proceedings of the Third European Signal Processing Conference, II, 1986, 739-742.

34 - P. H. Westerink, J. Biemond and P. H. L. de Bruin. Digital color image restoration. Proceedings of the Third European Signal Processing Conference, II, 1986, 761-764.

35 - B. Chanda, B. B. Chanduri, and D. D. Majunder. Some algorithms for image enhancement incorporating human visual response. Pattern Recognition, 17, 1984, 423-428.

36 - R. L. Lagendijk, J. Biemond and D. E. Boekee. Iterative image restoration with ringing reduction. Proceedings of the Third European Signal Processing Conference, II, 1986, 769-772.

37 - T. M. Lillisand and R. W. Kieser, **Remote Sensing and Image Interpretation**. John Wiley & Sons Inc., 1987.

38 - T. Bestul and L. S. Davis. On computing complete histograms of images in LOG(n) steps using hypercubes. IEEE Trans. on Pattern Analysis and

Machine Intelligence, 11, 1989, 212-214.

39 - E. Alparslan and F. Ince. Image enhancement by local histogram stretching. IEEE Trans. on Systems, Man, and Cybernetics, 11, 1981, 376-385.

40 - A. de A. Araújo. Filtros Espaciais: Estudo Comparativo e Aplicação em Segmentação e Classificação de Imagens. Tese de Doutorado em Ciências, Universidade Federal da Paraiba, 1987.

41 - A. de A. Araújo, M. A. de Barros e J. E. R. de Queiroz. Sum of Absolute Difference Values Smoothing: Comparison to New Algorithms and Application to Remote Sensing. Trabalho aceito para a 5th. European Signal Processing Conference, Barcelona, Espanha, Setembro, 1990.

42 - A. de A. Araújo e M. A. de Barros. FILTRIX: Um pacote de software para filtragem espacial de imagens. Anais do 1º. Simpósio Brasileiro de Processamento de Imagens e Computação Gráfica, 1988, 258-260.

43 - M. A. de Barros. Novos Recursos para Filtragem Espacial em Ambiente SITIM-110. INPE-PDI Noticias, 1, 1988, pp. 5.

44 - L. V. Dutra e N. D. A. Mascarenhas. Extração de atributos espaciais em imagens multiespectrais. Anais do II Simpósio Brasileiro de Sensoriamento Remoto, 1982, 539-546.

45 - M. A. de Barros, J. E. R. de Queiroz e A. de A. Araújo. Aplicação de Filtros Espaciais no Pré-

processamento de Imagens Multiespectrais para Tarefas de Classificação. Anais do 4º Simpósio Latino-Americano de Sensoriamento Remoto, San Carlos de Bariloche, Argentina, Novembro, Vol. II, 1989, 1039-1048.

46 - M. A. de Barros e J. E. R. de Queiroz. O Filtro da Mediana: Aplicações em Imagens Multiespectrais. Trabalho aceito para XIX Simpósio Brasileiro de Engenharia Agrícola, Piracicaba, Julho, 1990.

47 - M. A. de Barros, J. E. R. de Queiroz e A. de A. Araújo. Aplicação de Técnicas Adaptativas de Filtragem Espacial no Pré-processamento de Imagens Multiespectrais em Tarefas de Classificação. Relatório Técnico do Laboratório Associado de Sensoriamento Remoto, Universidade Federal da Paraíba - CCT, LASR-CG 03RT-004/90, Junho, 1990.

48 - A. de A. Araújo. Sum of absolute grey level differences: an edge-preserving smoothing approach. Electronics Letters, 21, 1985, 1219-1220.

49 - Y. H. Lee and A. T. Fam. An edge gradient enhancing adaptive order statistic filter. IEEE Trans. on Acoustics, Speech, and Signal Processing, 35, 1987, 680-695.

50 - F. R. D. Velasco, L. O. C. Prado, e R. C. M. Souza. Sistema MAXVER - Manual do Usuário. INPE, 1978.

51 - D. C. C. Wang, A. H. Vagnucci, and C. C. LI. Digital image enhancement: a survey. Computer Graphics and Image Processing, 24, 1983, 363-381.

52 - J. M. Prewitt. Object enhancement and extraction. in T. S. Huang Ed., Picture Processing and Digital Filtering. Springer-Verlag, 1975, 75-149.

53 - L. S. Davis, A. Rosenfeld and J. S. Weszka. Region extraction by averaging and thresholding. IEEE Trans. on Systems, Man, and Cybernetics, 5, 1975, 383-388.

54 - R. E. Graham. Snow removal a noise-striping process for picture signal. IEE Trans. on Information Theory 8, 1976, 129-144.

55 - D. W. Brown. Digital computer analysis and display of the radionuclide scan. Journal of Nuclear Medicine, 7, 1976, 740-753.

56 - A. Lev, S. W. Zucker and A. Rosenfeld. Iterative enhancement of noisy images. IEEE Trans. on Systems, Man, and Cybernetics, 7, 1977, 435-442.

57 - L. S. Davis and A. Rosenfeld. Noise cleaning by iterated averaging. IEEE Trans. on Systems, Man, and Cybernetics, 8, 1978, 705-710.

58 - A. Scher, F. R. D. Velasco and A. Rosenfeld. Some new image smoothing techniques. IEEE Trans. on Systems, Man, and Cybernetics, 10, 1980, 153-158.

59 - K. A. Narayana and A. Rosenfeld. Image smoothing by local use of global information. IEEE Trans. on Systems, Man, and Cybernetics, 11, 1981, 826-83.

60 - L. Kitchen, M. Pietikainen, A. Rosenfeld, and C.Y.

Wang. Multispectral image smoothing guided by global distribution of pixel values. IEEE Trans. on Systems, Man, and Cybernetics, 13, 1983, 626-631.

61 - E. R. Davies. Design of optimal gaussian operators in small neighbourhoods. Image and Vision Computing 5, 1987, 199-205.

62 - J. B. Bednar and T. L. Watt. Alpha-trimmed means and their relationship to median filters. IEEE Trans. on Acoustics, Speech, and Signal Processing, 32, 1984, 145-153.

63 - C. A. P. Raez and C. D. McGillem. An adaptive, nonlinear edge-preserving filter. IEEE Trans. on Acoustics, Speech, and Signal Processing, 32, 1984, 571-576.

64 - Y. H. Lee and S. A. Kassam. Generalized median filtering and related nonlinear filtering techniques. IEEE Trans. on Acoustics, Speech, and Signal Processing, 33, 1985, 672-683.

65 - A. Restrepo and A. C. Bovik. Adaptive trimmed filters for image restoration. IEEE Trans. on Acoustics, Speech and Signal Processing, 36, 1988, 1326-1337.

66 - J. S. Lee. Digital image enhancement and noise filtering by use of local statistics. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2, 1980, 165-168.

67 - J. S Lee. Refined filtering of image noise using local statistics. Computer Graphics and Image Processing, 15, 1981,380-389.

68 - J. S. Lee. Speckle analysis and smoothing of synthetic aperture radar images. Computer Graphics and Image Processing, 17, 1981, 24-32.

69 - J. S. Lee. Digital image smoothing and the sigma filter. Computer Graphics and Image Processing, 24, 1983, 255-269.

70 - J. S. Lee. A Simple speckle smoothing algorithm for synthetic aperture radar images. IEEE Trans. on Systems, Man, and Cybernetics, 13, 1983, 85-89.

71 - M. J. McDonnel. Box-filtering techniques. Computer Graphics and Image Processing, 17, 1981, 65-10.

72 - D. C. C. Wang, A. H. Vagnucci and C. C. Li. Gradient inverse smoothing scheme and the evaluation of its performance. Computer Graphics and Image Processing 15, 1981, 167-181.

73 - W. J. Song and W. A. Pearlman. Edge-preserving noise filtering based on adaptive windowing. IEEE Trans. on Circuits and Systems, 35, 1988, 1048-1055.

74 - V. Kim and L. Yaroslavskii. Rank algorithms for picture processing. Computer Vision, Graphics, and Image Processing, 35, 1986, 234-258.

75 - G. Heygster. Rank filters in digital image processing. Computer Graphics and Image Processing,

19, 1982, 148-164.

76 - Y. Nakagawa and A. Rosenfeld. A note on the use of local min and max operations in digital picture processing. IEEE Trans. on Systems, Man, and Cybernetics, 8, 1978, 632-635.

77 - B. Justusson. Noise reduction by median filtering. Proceedings of the International Conference on Pattern Recognition, 1978, 502-504.

78 - G. J. Yang and T. S. Huang. The effect of median filtering on edge location estimation. Computer Graphics and Image Processing, 15, 1981, 224-245.

79 - P. E. Danielsson. Getting the median faster. Computer Graphics and Image Processing 17, 1981, 71-78.

80 - J. M. Lester, J. F. Brenner and W. D. Selles. Local transforms for biomedical image analysis. Computer Graphics and Image Processing, 13, 1980, 17-30.

81 - M. O. Ahmad and D. Sundararajan. A fast algorithm for two-dimensional median filtering. IEEE Trans. on Circuits and Systems, 34, 1987, 1364-1373.

82 - Y. Neuvo, P. Heinonen and I. Defee. Linear-median hybrid edge detectors. IEEE Trans. on Circuits and Systems, 34, 1987, 1337-1343.

83 - A. C. Bovik. Streaking in median filtered images. IEEE Trans. on Acoustics, Speech, and Signal Processing, 35, 1987, 493-503.

84 - R. Bernstein. Adaptive Nonlinear filters for simultaneous removal of different kinds of noise in images. IEEE Trans. on Circuits and Systems, 34, 1987, 1275-1291.

85 - T. A. Nodes and N. C. Gallagher. Median filters: some modifications and their properties. IEEE Trans. on Acoustics, Speech, and Signal Processing, 30, 1982, 739-746.

86 - N. C. Gallagher Jr. and G. L. Wise. A theoretical analysis of the properties of median filters. IEEE Trans. on Acoustics, Speech, and Signal Processing, 29, 1981, 1136-1141.

87 - H. M. Lin and A. N. Willson Jr. Median filters with adaptive length. IEEE Trans. on Circuits and Systems, 35, 1988, 675-690.

88 - T. Loupas, W. N. Mcdicken and P. L. Allan. An adaptive weighted median filter for speckle supression in medical ultrasonic images. IEEE Trans. on Circuits and Systems, 36, 1989, 129-135.

89 - C. Rey and R. K. Ward. A parametrized family of nonlinear image smoothing filters. IEEE Trans. on Acoustics, Speech, and Signal Processing, 37, 1989, 1458-1462.

90 - G. R. Arce and R. E. Foster. Detail-preserving ranked-order based filters for image processing. IEEE Trans. on Acoustics, Speech, and Signal Processing, 37, 1989, 83-98.

91 - A. Kundu and W. Wu. Double-window Hodges-Lehman (D) filter and hybrid D-median filter for robust image smoothing. IEEE Trans. on Acoustics, Speech, and Signal Processing, 37, 1989, 1293-1298.

92 - A. Nieminen, P. Heinonen and Y. Neuvo. A new class of detail-preserving filters for image processing. IEEE Trans. on Pattern Analysis and Machine Intelligence, 9, 1987, 74-90.

93 - P. Heinonen and Y. Neuvo. FIR-Median hybrid filters. IEEE Trans. on Acoustics, Speech, and Signal Processing, 35, 1987, 832-838.

94 - M. Kunt, A. Ikonomopoulos and M. Kocher. Second-generation image-coding techniques. Proceedings of the IEEE, 73, 1985, 549-574.

95 - M. Kunt, M. Benard and R. Leonardi. Recents results in high-compression image coding. IEEE Trans. on Circuits and Systems, 34, 1987, 1306-1336.

96 - H. J. Trussel. A fast algorithm for noise smoothing based on a subjective criterion. IEEE Trans. on Systems, Man, and Cybernetics, 7, 1977, 677-678.

97 - G. L. Anderson and A. N. Netravali. Image restoration based on a subjective criterion. IEEE Trans. on Systems, Man, and Cybernetics, 6, 1976, 845-853.

98 - E. Peli. Adaptive enhancement based on a visual model. Optical Engineering, 26, 1987, 655-660.

99 - D. P. Lulich and K. A. Stevens. Differential

Contributions of circular and elongated spatial filters to the Café Wall Illusion. Biological Cybernetics, 61, 1989, 427-435.

100 - J. D. Fahnestock and R. A. Schowengerdt. Spatially variant contrast enhancement using local range modification. Optical Engineering, 22, 1983, 378-381.

101 - S. H. Jung and N. C. Kim. Adaptive image restoration of sigma filter using local statistics and human visual characteristics. Eletronics Letters, 24, 1988, 201-233.

102 - F. Tomita and S. Tsuji. Extraction of multiple regions by smoothing in selected neighbourhoods. IEEE Trans. on Systems, Man, and Cybernetics, 7, 1977, 107-109.

103 - M. Nagao and T. Matsuyama. Edge preserving smoothing. Computer Graphics and Image Processing, 9, 1979, 394-407.

104 - G. Matheron. Random Sets and Integral Geometry. Wiley, 1975.

105 - J. Serra. Image analysis and mathematical morphology. Academic, 1982.

106 - R. L. Stevenson and G. R. Arce. Morphological filters: statistics and further syntactic properties. IEEE Trans. on Circuits and Systems, 34, 1987, 1292-1303.

107 - P. Maragos. Morphology-based multidimensional signal processing. Proceedings of the Conf. on Information Sciences and Systems, 1987.

108 - F. Y. Shih and O. Mitchell. Threshold decomposition of gray-scale morphology into binary morphology. IEEE Trans. on Pattern Analysis and Machine Intelligence, 11, 1989, 31-42.

109 - J. Serra. Introduction to mathematical morphology. Computer Vision, Graphics, and Image Processing, 35, 1986, 283-305.

110 - S. R. Sternberg. Gray-scale morphology. Computer Vision, Graphics, and Image Processing, 35, 1986, 333-355.

111 - R. M. Haralick, S. R. Sternberg and X. Zhuang. Image analysis using mathematical morphology. IEEE Trans. on Pattern Analysis and Machine Intelligence, 9, 1987, 532-550.

112 - F. Y. Shih and O. R. Mitchell. Automated fast recognition and location of arbitrary shaped objects by image morphology. Proceedings of the Computer Vision and Pattern Recognition Conf., 1988, 774-779.

113 - F. Y. Shih and O. R. Mitchell. Industrial parts recognition and inspection by image morphology. Proceedings of the IEEE Int. Conf. Robotics Automat., 1988, 273-276.

114 - S. K. Pal and R. A. King. Image enhancement using

smoothing with fuzzy sets. IEEE Trans. on Systems, Man, and Cybernetics, 11, 1981, 494-501.

115 - H. K. Huang. Progress in image processing technology related to radiological sciences: a five years review. Computer Methods Prog. Biomed., 25, 1987, 103-111.

116 - S. L. Hurt and A. Rosenfeld. Noise reduction in three-dimensional digital images. Pattern Recognition, 17, 1984, 407-421.

117 - N. N. Abdelmalek. Noise filtering in digital images and approximation theory. Pattern Recognition, 29, 1986, 417-424.

118 - A. Kandu, S. K. Mitra and P. P. Vaidynathan. Application of two-dimensional generalized mean filtering for removal of impulse noises from images. IEEE Trans. on Acoustics, Speech, and Signal Processing, 32, 1984, 600-609.

119 - C. Mohwinkel and L. Kurz. Computer picture processing and enhancement by localized operations. Computer Graphics and Image Processing, 5, 1976, 401-424.

120 - M. Rabanni. Bayesian filtering of poisson noise using local statistics. IEEE Trans. on Acoustics, Speech and Signal Processing, 36, 1988, 933-937.

121 - A. de A. Araújo. Sum of absolute difference values smoothing: evaluation and application. Proceedings of the European Signal Processing Conference, 1986,

773-776.

122 - R. M. Haralick. Edge and region analysis for digital image data. Computer Graphics and Image Processing, 12, 1980, 60-73.

123 - R. M. Haralick. A facet model for image data. Computer Graphics and Image Processing, 15, 1981, 634-647.

124 - A. F. Korn. Toward a symbolic representation of intensity changes in images. IEEE Trans. on Pattern Analysis and Machine Intelligence, 10, 1988, 610-125.

125 - N. Kanopoulos, N. Vasanthavada and R. L. Baker. Design of an image edge detection filter using the Sobel operator. IEEE Journal of Solid-State Circuits, 23, 1988, 358-367.

126 - R. Kirsch. Computer determination of the constituent structure of biological images. Computers and Biomedical Research, 4, 1971, 315-328.

127 - G. S. Robinson. Edge detection by compass gradient masks. Computer Graphics and Image Processing, 6, 1977, 492-501.

128 - I. Overington and P. Grennway. Pratical first difference edge detection with subpixel accuracy. Image and Vision Computing, 5, 1987, 217-224.

129 - R. M. Haralick. Digital step edges from zero crossing of second directional derivatives. IEEE Trans.

on Pattern Analysis and Machine Intelligence, 6, 1984, 58-68.

130 - R. M. Haralick. Ridges and valleys on digital image. Computer Graphics and Image Processing, 22, 1983, 28-38.

131 - R. Nevatia. Evaluation of simplified Hueckel edge line detector. Computer Graphics and Image Processing, 6, 1977, 582-588.

132 - J. F. Abramatic. Why the simplest Hueckel edge detector is a Roberts operator. Computer Graphics and Image Processing, 17, 1981, 79-83.

133 - R Hummel. Feature detection using basis functions. Computer Graphics and Image Processing, 9, 1979, 40-55.

134 - D. G. Morgenthaler. A new hybrid edge detector. Computer Graphics and Image Processing, 16, 1981, 166-176.

135 - A. L. Shipman, R. R. Bitmead and G. H. Allen. Diffuse edge fitting and following: a location-adaptive approach. IEEE Trans. on Pattern Analysis and Machine Inteligence, 6, 1984, 96-102.

136 - M. H. Hueckel. A local visual operator which recognizes edges and lines. Journal of the ACM, 20, 1973, 634-647.

137 - I. Pitas and A. N. Venetsanopoulos. Edge detectors based on nonlinear filters. IEEE Trans. on Pattern

Analysis and Machine Intelligence, 8, 1986, 538-550.

138 - D. Vandick, F. Vandenplas, W. Coene and H. Zendenbergen. Robust statistical methods in image processing. Scanning Microscopy, SUPPL-2, 1989, 185-191.

139 - P. de Souza. Edge detection using sliding statistical tests. Computer Vision, Graphics and Image Processing, 23, 1983, 1-14.

140 - J. S. Chen and G. Medioni. Detection, localization and estimation of edges. IEEE Trans. on Pattern Anallysis and Machine Intelligence, 11, 1989, 191-198.

141 - N. D. A. Mascarenhas and L. O. C. Prado. Edge detection in images: a hypotesis testing aproach. Proceedings of International Conference on Pattern Recognition, 1978, 707-709.

142 - Y. Yakimosky. Boundary and object detection in real world images. Journal of the ACM 23, 1976, 599-618.

143 - D. Jeulyn. Mathematical morphology and material image analysis. Scanning Microscopy, SUPPL-2, 1989, 165-185.

144 - F. Sloboda. Smooth and sharp laplacian operators. Computer Artificial Intelligence, 4, 1985, 153-162.

145 - L. J. Van Vliet, I. T. Young and G. L. Beckers. A nonlinear laplace operator as edge detection in noisy images. Computer Vision, Graphics and Image

Processing, 45, 1989, 167-195.

146 - S. W. Zucker, A. Rosenfeld and L. S. Davis. Picture segmentation by texture discrimination. IEEE Trans. on Computers, 24, 1975, 1228-1233.

147 - D. P. Panda and A. Rosenfeld. Image segmentation by pixel classification in (gray level, edge value) space. IEEE Trans. on Computers, 27, 1978, 875-879.

148 - G. Garibotto. A New edge-detection scheme based on local correlation function. Proceedings of the Third European Signal Processing Conference, II, 1986, 925-927.

149 - R. L. Kashyap and K. Eom. Texture boundary detection using long correlation model. Proceedings of the Int. Geoscience and Remote Sensing Symp. Amherst, MA, 1985.

150 - R. L. Kashyap and K. Eom. Texture boundary detection based on the long correlation model. IEEE Trans. on Pattern Analysis and Machine Intelligence, 11, 1989, 58-67.

151 - W. B. Thompson. Textural boundary analysis. IEEE Trans. on Computers, 26, 1977, 272-274.

152 - L. S. Davis and A. Mitiche. Edge detection in textures. Computer Graphics and Image Processing, 12, 1980, 25-39.

153 - L. S. Davis and A. Mitiche. Edge detection in textures - maxima selection. Computers Graphics and

Image Processing,16, 1981, 158-165.

154 - J. J. Clark. Singularities of contrast functions in scale space. Proceedings of the First Int. Conf. Comput. Vision, 1987, 491-495.

155 - J. J. Clark. Authenticating edges produced by zero-crossing algorithms. IEEE Trans. on Pattern Analysis and Machine Intelligence,11, 1989, 43-55.

156 - J. Ritcher and S. Ullman. Non-linearities in cortical simple cells and the possible detection of zero crossings. Biologie Cybernetic, 53, 1986, 195-202.

157 - R. Stefanelli and A. Rosenfeld. Some parallel thinning algorithm for digital pictures. Journal of the ACM, 18, 1971, 255-264.

158 - T. H. Hong, C. R. Dyer and A. Rosenfeld. Texture primitive extraction using an edge-based approach. IEEE Trans. on Systems, Man and Cybernetics, 10, 1980, 659-675.

159 - R. B. Eberlein. An iterative gradient edge detection algorithm. Computer Graphics and Image Processing, 5, 1976, 245-253.

160 - T. Kasvand. Iterative edge detection. Computer Graphics and Image Processing, 4, 1975, 279-286.

161 - I. Weiss. Line Fitting in a noyse image. IEEE Trans. on Pattern Analysis and Machine Intelligence, 11, 1989, 325-329.

162 - I. Pitas. Markovian models for image labeling and edge detection. Signal Processing, **15**, 1988, 365-374.

163 - P. Eichel and E. Delp. Sequential edge detection in correlated random fields. Proceedings of the Conference on Computer Vision and Pattern Recognition, 1985, 14-21.

164 - Y. T. Zhou, V. Vienkateswar and R. Chellappa. Edge detection and linear Feature extraction using a 2-D random field model. IEEE Trans. on Pattern Analysis and Machine Intelligence, **11**, 1989, 84-94.

165 - W. A. Perkins. Area segmentation of images using edge description. Computer Graphics and Image Processing, **13**, 1982, 179-195.

166 - T. Pavlidis. A thinning algorithm for discrete binary images. IEEE Trans. on Pattern Analysis and Machine Intelligence, **2**, 1980, 8-15.

167 - R. Nevatia and K. R. Babu. Linear feature extraction and description. Computer Graphics and Image Processing, **13**, 1980, 257-269.

168 - A. Ikonomopoulos. An approach to edge detection based on the direction of edge elements. Computer Graphics and Image Processing, **19**, 1982, 179-195.

169 - A. Favre and H. Keller. Parallel syntatic thinning by recoding of binary pictures. Computer Graphics and Image Processing, **23**, 1983, 99-112.

170 - B. Gil, A. Mitiche and J. K. Aggarwal. Experiments in combining intensity and range edge maps. Computer Graphics and Image Processing, 21, 1983, 395-411.

171 - V. Lacroix. A three-module strategy for edge detection. IEEE Trans. on Pattern Analysis and Mschine Intelligence, 11, 1989, 803-810.

172 - I. Pitas and A. N. Venetsanopoulos. Nonlinear order statistic filters for image filtering and edge detection. Signal Processing, 10, 1986, 395-413.

173 - V. Lacroix. Edge detection : A tutorial review. Proceedings of International Conference on Acoustical, Speech and Signal Processing, 1982, 1172-1175.

174 - V. Torre and T. A. Poggio. On edge detection. IEEE Trans. on Pattern Analysis and Machine Intelligence, 8, 1986, 187-193.

175 - E. P. Lyvers and O. R. Mitchell. Precision edge contrast and orientation estimation: comparision of various methods. IEEE Trans. on Pattern Analysis and Machine Intelligence, 10, 1988, 927-937.

176 - R. C. Cook. Soft and hardware for imaging. Photonics Spectra, 23, 1989, 96-97.

177 - Special issue on computer architecture for pattern analysis and image database management. IEEE Trans. on Computers, 31, 1982.

178 - S. T. Wilson. Machine vision turns to morphology.

Photonics Spectra, 23, 1989, 104-107.

179 - Special issue on robotics and automation. Computer, 15, 1982.

180 - M. A. Kraaijveld, P. P. Jonker, R. Nouta and R. P. W. Duin. The VLSI realization of a binary-image processor. Proceedings of the Third European Signal Processing Conference, II, 1986, 1231-1234.

181 - Special issue on computer architectures for image processing. Computer, 16, 1983.

182 - Special issue on applications for array processors. Computer, 16, 1983.

183 - M. Duft. Parallel processors for digital image processing. in P. Stuck, Ed. Advances in Digital Image Processing: Theory, Application and Implementation. Plenum Press, 1979, 265-276.

184 - V. Di Gesu. An overview of pyramid machines for image processing. Information Sciences, 47, 1989, 17-35.

185 - A. Peled. A low-cost image processing facility employing a hardware realization of high-speed signal processors. in P. Stuck, Ed. Advances in Digital Image Processing: Theory, Application and Implementation. Plenum Press, 1979, 301-323.

186 - K. R. Sloan Jr. and C. M. Brown. Color map techniques. Computer Graphics and Image Processing, 10, 1979, 297-311.

187 - C. Reader and L. Hubble. Trends in image display systems. Proceedings of the IEEE 69, 1981, 606-614.

188 - B. Marangelli and N. Mirizzi. Pseudocolor encoder for analog and digital refresh memories. Computers Graphics and Image Processing, 17, 1981, 60-64.

189 - D. H. Bass. Using the video lookup table for reflectivity calculations: specific techniques and graphic results. Computer Graphics and Image Processing, 17, 1981, 249-261.

190 - S. M. Pizer. Intensity mappings to linearize display devices. Computer Graphics and Image Processing, 17, 1981, 262-268.

191 - M. Manohar and H. K. Kampriyan. Connected component labeling of binary images on a mesh connected massively parallel processor. Computer Vision, Graphics and Image Processor, 45, 1989, 133-150.

192 - P. F. Leonard. Pipeline architectures for real-time machine vision. Proceedings of the Computer Architectures for Pattern Analysis and Image Database Managment Conference, 1986, 502-505.

193 - R. M. Haralick. A reconfigurable systolic network for computer vision. Proceedings of the Computer Architectures for Pattern Analysis and Image Database Managment Conference, 1986, 507-515.

194 - W. K. Pratt. A pipeline architecture for image processing and analysis. Proceedings of the

Computer Architectures for Pattern Analysis and Image Database Managment Conference, 1986, 507-515.

195 - A. Fiat and A. Shamir. Polymorphic arrays: a novel VLSI layout for systolic computers. Journal of Computer and Systems Sciences, 33, 1986, 47-65.

196 - L. O. Hertzberger and A. Choudry. Abstract data types and multiprocessor architecture for image understanding. Proceedings of the Pattern Recognition in Practice II Conference, 1986, 125-133.

197 - H. T. Kung and J. A. Webb. Mapping image processing operations onto a linear systolic machine. Distributed Computing, 1, 1986, 246-257.

198 - L. Uhr. Parallel architectures for image processing, computer vision and pattern perception. Handbook of Pattern Recognition and Image Processing, 1986, 437-469.

199 - T. Y. Young and P. S. Liu. VLSI array architecture for pattern analysis and image processing. Handbook of Pattern Recognition and Image Processing, 1986, 471-496.

200 - I. Khan. Implementation of conditional processing and pyramids with a general pipelined pixel processor. Proceedings of the Conference on Computer Vision and Pattern Recognition, 1986, 288-292.

201 - D. V. Ramanamurthy, N. J. Dimopoulos, K. F. Li, R. V.

Patel and A. J. Al-Khalili. Parallel algorithms for low level vision on the homogeneous multi-processor. Proceedings of the Conference on Computer Vision and Pattern Recognition, 1986, 421-423.

202 - O. H. Ibarra, S. M. Kim and M. A. Palis. Designing systolic algorithms using sequencial machines. IEEE Trans. on Computers, 35, 1986, 531-542.

203 - T. Fukushima, Y. Kobayashi, S. Miura and K. Asada. Architecture of an image signal processor. Proceedings of the International Conference on Pattern Recognition, 1986, 38-48.

204 - G. Qiang and Z. R. Li. A knowledge base architecture for computer vision. Proceedings of the International Conference on Pattern Recognition, 1986, 859-861.

205 - B. Lindskog and P. E. Danielsson. PICAP3: A parallel processor tuned for 3D image operations. Proceedings of the International Conference on Pattern Recognition, 1986, 1248-1250.

206 - P. A. Ruetz and R. W. Brodersen. Architectures and design techniques for real-time image processing IC's. Journal of Solid-State Circuits, 22, 1987, 233-250.

207 - K. N. Ngan, A. A. Kassim and H. S. Singh. Parallel image processing system based on the TMS32010 digital signal processor. IEE Proceedings, 134, 1987, 119-124.

208 - L. Curran. Chip set speeds color image processing. Electronic Design, 37, 1989, 147-159.

209 - F. Jutand, A. Artieri, G. Concordel and N. Demassieux. VLSI architectures for image filtering. Proceedings of the SPIE, 1988.

210 - A. Ruetz and W. Brodersen. Architectures and design techniques for real-time image processing ICs. IEEE Journal of Solid State Circuits, 22, 1988, 233-250.

211 - N. Demasssieux, F. Jutand, C. Joanblank and M. Bernard. VLSI architectures for real time image convolution with large symetrical kernels. Proceedings of the ICASSP, 1988.

212 - R. M. Lea. SCAPE: A single-chip array processing element for signal and image processing. IEE Proceedings, 133, 1986, 145-151.

213 - M. O'Donnell. Applications of VLSI circuits to medical imaging. Proceedings of the IEEE, 76, 1988, 1106-1114.

214 - M. Sugah, A. Kanuma, K. Suzuki, and M. Kubo. VLSI processor for image processing. Proceedings of the IEEE, 75, 1987, 1160-1165.

215 - D. Bursky. CMOS four-chip set processes images at 20 MHz data rates. Eletronic Design, 28, 1987, 39-44.

216 - C. Lee, F. V. M. Catthoor and H. J. De Man. An efficient ASIC architeture for real-time edge

detection. IEEE Trans. on Circuits and Systems, 36, 1989, 1350-1359.

217 - A. M. Janjua. Image processing looks to the future. Photonics Spectra, 23, 1989, 97-103.

218 - K. D. Alehrs. Microcomputer based digital image processing system developed to count and size laser-generated small particle images. Optical Engineering, 24, 1985, 1060-1065.

219 - S. B. Seshadri. Desisgn of a medical image management system: a practical cost-effective approach. Computer Methods Prog in Biomedicine, 25, 1987.

220 - K. Preston Jr. Image processing software: a survey. in "Progress in Pattern Recognition" (L. N. Kanal and A. Rosenfeld Eds.) Vol. 1, 1981, 123-148.

221 - H. Tamura, S. Sakane, F. Tomita, N. Yokoya, M. Kaneko, and K. Sakane. Design and implementation of SPIDER - a transportable image processing software package. Computer Vision, Graphics and Image Processing, 23, 1982, 273-294.

222 - S. Krusemark and R. M. Haralick. Archieving port-ability in image processing software. Proceedings of the IEEE Conference on Pattern Recognition and Image Processing 1982, 451-457.

223 - H. Tamura and S. Mori. A data managment system for manipulating large images. Proceedings of the IEEE Workshop on Picture Data Description and Managment,

1977, 45-54.

224 - S. C. Ahearn, R. W. Kiefer, A. Rambe and M. A. Raimadoya. IBM-PC/XT Microcompuer-based digital image processing system for remote sensing education in developing countries. Proceedings of the 50th Annual Meeting of the American Society of Photogrammetry, 1984, 253-262.

225 - J. R. Eyton. A hibrid classification instructional package. Photogrammetric Engineering and Remote Sensing, 49, 1983, 1175-1181.

226 - S. Hsu. Experimental automated image processing and mapping system with Z-80 based microcomputer. Technical Papers of the ACSM-ASP Annual Convention 1982, 129-132.

227 - J. R. Jensen and P. H. Openheimer. Image processing education: A microcomputer based alternative. Proceedings of the 50th Annual Meeting of the American Society of Photogrammetry, 1984, 439-446.

228 - R. W. Kiefer and F. J. Gunther. Digital image processing using the APPLE II microcomputer. Photogrammetric Engineering and Remote Sensing, 49, 1983, 1167-1174.

229 - F. Scarpace, R. Kiefer, S. Ahearn and P. Weiler. Digital image processing using the IBM PC. Proceedings of the ASP-ACSM Fall Convention, 1984, 511-517.

230 - D. C. Walklet. The personal image processor: the

micro solution for a major problem. Proceedings of the ASP-ACSM Fall Convention, 1984, 518-525.

231 - B. Lay. MORPHOLOG: An image processing software package. Proceedings of the Computer Architecture for Pattern Analysis and Image Database Management Conference, 1986, 463-468.

232 - J. M. Chassery and G. Bourrel. An image package software: IPS - Design and abilities. Proceedings of the International Conference on Pattern Recognition, 1986, 913-915.

233 - PCIPS: Personal Computer Image Processing System. User's Guide. IBM Personal Computer Software, 1986.

234 - N. J. Leite. PICTOREA: Um sistema didático para processamento digital de imagens. Dissertação de Mestrado em Engenharia Elétrica, Universidade Federal da Paraíba, 1989.

235 - SITIM: Sistema de Tratamento de Imagens. Manual do Usuário. Instituto de Pesquisas Espaciais, 1984.

236 - G. E. Sobelman and D. E. Knekelberg. C Avançado - Tecnologia e Aplicações. Editora Campos, 1989.

237 - K. Jamsa. The C Library. McGraw-Hill Inc., 1988.

238 - H. Schildt. Turbo C - The Pocket Reference. McGraw-Hill Inc., 1989.

239 - F. Cabral. A Linguagem C e o PC BIOS. Editora Campos, 1988.

240 - H. Schildt. The Complete Reference Book. McGraw-Hill, 1989.

241 - H. Schildt. C Avançado - Guia do Usuário. McGraw-Hill, 1987.

242 - SITIM 150: Sistema de Tratamento de Imagens. Manual do Usuário. Instituto de Pesquisas Espaciais, 1988.

243 - M. A. de Barros e A. de A. Araújo. PCFILT: Um Pacote de Software para Filtragem Espacial de Imagens em Microcomputadores. Relatório Técnico do Laboratório Associado de Sensoriamento Remoto, Universidade Federal da Paraíba - CCT, LASR-CG 03RT-005/90, Maio, 1990.

244 - J. E. R. de Queiroz e M. A. de Barros. Impressão de Imagens Multiespectrais por Dispositivos Matriciais. Trabalho aceito para o 19º Congresso Brasileiro de Engenharia Agrícola, Piracicaba, Brasil, Julho, 1990

245 - L. A. de Barros, M. A. de Barros, O. Giordano e W. F. Giozza. O operador de Roberts: Uma implementação usando tecnologia CMOS para detecção de bordas em tempo real. Anais do 3º Simpósio Brasileiro Computação Gráfica e Processamento de Imagens, Junho, 1990, 258-268.

246 - L. A. de Barros. Uma implementação utilizando tecnologia CMOS para detecção de bordas em tempo real com o operador de Roberts. Dissertação de Mestrado em Engenharia Elétrica, Universidade Federal da Paraíba, 1990.

247 - L. A. de Barros, M. A. de Barros, O. Giordano e W. F. Giozza. Uma implementação utilizando tecnologia CMOS para detecção de bordas em tempo real com o operador de Roberts. Trabalho aceito para o 5º Simpósio Brasileiro de Microeletrônica, Julho, 1990.

248 - L. V. Dutra e N. D. A. Mascarenhas. Some experimentals with spatial feature extraction methods in multiespectral classification. International Journal of Remote Sensing, 5, 1984, 303-313.

249 - J. W. Merchant. Using spatial logic in classification of LANDSAT TM data. In Proc. PECORA9: Spatial Information Technologies for Remote Sensing Today and Tomorrow. Silver Spring, MD:IEEE Computer Press, 1984, 378-375.

250 - A. A. Vassiliou, M. Boulianne and J. A. R. Blais. On the application of averaging median filters in remote sensing. IEEE Trans. on Geoscience and Remote Sensing, 26, 1988, 832-838.

251 - S. C. Ahearn. Combining laplacian images of different spatial frequencies scales: implications for remote sending image analysis. IEEE Trans. on Geoscience and Remote Sensing, 26, 1988, 826-831.

252 - F. A. Mitsuo, L. V. Dutra e C. L. Mendes. Comparação entre os métodos de entropia e da distancia de Jeffrey-Matusita em problemas de seleção de atributos. Anais do II Simpósio Brasileiro de Sensoriamento Remoto. 1982, 621-267.

A P Ê N D I C E A

CÓDIGOS EM LINGUAGEM C DOS ALGORITMOS E UTILITÁRIOS IMPLEMENTADOS

F I L T R I X

PROGRAMA PRINCIPAL RESPONSAVEL P
INTERFACE COM O USUARIO E PELA
CHAMADA DAS ROTINAS DOS ALGORITMOS
E UTILITARIOS

```
#include (dia.h)
#include (uai.h)
#include (dev.h)
#include (stdio.h)
#include (atr.h)
#include (math.h)
struct  dev_entry   buffer;
main()
{
int o,flag3,flag2,flag1,k1,j1,ja,k2,j2,j2a,canal,i,k,j,p;
char aux[512], b[512], c,l,li;
unsigned  save;
flag1=0;
i=0;
j=8;
k=0;
get_attr(stdout,&buffer);
save=buffer.tty_options;
buffer.tty_options &= ~EDIT;.
buffer.tty_options &= ~ECHO;
set_attr(stdout,&buffer);

escabre();
msg0();
for(;;){
  l= getchar();
  if(l == 'v'){
    printf("\n\n");
    goto final;
  }
  if(l == 'c') goto saida1;
}
```

```
saidai:
limpa();
telainf();
msg0();
for(;;){
  l= getchar();
  if(l == 'v'){
    printf("\n\n");
    goto final;
  }
  if(l == 'c') goto inicio;
}

inicio:
limpa();
eschelpi();
msg0();
escretela();
j=8;
k=0;

for (;; ) {
  if (flagi=1) {
    if (k==0)
      dia_poe_texto(j,21,"Suavizacao Espacial          ",8,2);
      if (k==1) dia_poe_texto(j,21,"Detecao de Bordas           ",8,2);
      if (k==2) dia_poe_texto(j,21,"Medicoes Estatisticas       ",8,2);
      if (k==3Ê dia_poe_texto(j,21,"Geracao de Ruido            ",8,2);
      if (k==4Ê dia_poe_texto(j,21,"Transferencia de Imagens    ",8,2);
    }
    switch(l=getchar()) {
      case '8':
        flagi=0;
        j--; k--;
        if (j<8) j=12;
        if (k<0) k=4;
        o= j+1;
        if (j==12) o=8;

        if (k==0){
          dia_poe_texto(o,21,"Detecao de Bordas           ",6,1);
          dia_poe_texto(j,21,"Suavizacao Espacial         ",8,2);
        }
        if (k==1){
          dia_poe_texto(o,21,"Medicoes Estatisticas       ",6,1);
          dia_poe_texto(j,21,"Detecao de Bordas           ",8,2);
        }
        if (k==2){
          dia_poe_texto(o,21,"Geracao de Ruido            ",6,1);
          dia_poe_texto(j,21,"Medicoes Estatisticas       ",8,2);
        }
        if (k==3){
          dia_poe_texto(o,21,"Transferencia de Imagens    ",6,1);
          dia_poe_texto(j,21,"Geracao de Ruidoo            ",8,2);
        }
```

```
  if (k==4){
    dia_poe_texto(o,21,"Suavizacao Espacial          ",6,1);
    dia_poe_texto(j,21,"Transferencia de Imagens     ",8,2);
  }
  break;
case '2':
  flag1=0;
  j++; k++;
  if (j)12) j=8;
  if (k)4) k=0;
  o= j-1;
  if (j==8) o=12;

  if (k==0){
    dia_poe_texto(o,21,"Transferencia de Imagens     ",6,1);
    dia_poe_texto(j,21,"Suavizacao Espacial          ",8,2);
  }
  if (k==1){
    dia_poe_texto(o,21,"Suavizacao Espacial          ",6,1);
    dia_poe_texto(j,21,"Detecao de Bordas            ",8,2);
  }
  if (k==2){
    dia_poe_texto(o,21,"Detecao de Bordas            ",6,1);
    dia_poe_texto(j,21,"Medicooes Estatisticas        ",8,2);
  }
  if (k==3){
    dia_poe_texto(o,21,"Medicoes Estatisticas        ",6,1);
    dia_poe_texto(j,21,"Geracao de Ruido             ",8,2);
 }
  if (k==4){
    dia_poe_texto(o,21,"Geracao de Ruido             ",6,1);
    dia_poe_texto(j,21,"Transferencia de Imagens     ",8,2);
  }
  break;
case 'v':
  printf("\n\n\n\n\n\n\n\n\n");
  goto final;
  break;
case 's':
  limphelp();
  uaihelp();
  switch(li=getchar()){
    case '0':
      shell("2:/cmds/uai 0");
      break;
    case '1':
      shell("2:/cmds/uai 1");
      break;
    case '2':
      shell("2:/cmds/uai 2");
      break;
    case 'g':
      shell("2:/cmds/uai 3");
      break;
```

```
  case 'c':
    shell("2:/cmds/uai");
    break;
  case 'v':
    goto inicio;
    break;
}
break;
case 'f':
nivel2:
if (k==0) escretela1();
if (k==1) escretela2();
if (k==2) escretela3();
if (k==3) escretela4();
if (k==4) escretela5();
ki=0;  ji=10;

eschelpi();
flag2=1;
for (;;) {
 if (flag2==1){
   if (k==0)   {
     if (ki==0) dia_poe_texto(ji,31,"Filtro da Media                  ",8,2);
     if (ki==1) dia_poe_texto(ji,31,"Filtro da Mediana                ",8,2);
     if (ki==2) dia_poe_texto(ji,31,"Filtros Sigma                    ",8,2);
     if (ki==3) dia_poe_texto(ji,31,"Filtros da Ordem                 ",8,2);
     if (ki==4) dia_poe_texto(ji,31,"Media K vizinhos mais Proximos   ",8,2);
     if (ki==5) dia_poe_texto(ji,31,"Transformada Agucamento Extremo  ",8,2);
     if (ki==6) dia_poe_texto(ji,31,"Suavizacao Soma Dif Absolutas    ",8,2);
     if (ki==7) dia_poe_texto(ji,31,"Suavizacao Media Dif Absolutas   ",8,2);
     if (ki==8) dia_poe_texto(ji,31,"Suavizacao Logaritmica           ",8,2);
     if (ki==9) dia_poe_texto(ji,31,"Suavizacao Inverso do Gradiente  ",8,2);
     if (ki==10Ê dia_poe_texto(ji,31,"Suavizacao Variancia             ",8,2);
   }
 if (k==1) {
   if (ki==0) dia_poe_texto(ji,31,"Operadores Diferenciais Laplacianos ",8,2);
   if (ki==1) dia_poe_texto(ji,31,"Operador Diferencial de Roberts     ",8,2);
   if (ki==2) dia_poe_texto(ji,31,"Operador Diferencial de Sobel       ",8,2);
  if (ki==3) dia_poe_texto(ji,31,"Operador Diferencial de Prewitt     ",8,2);
   if (ki==4) dia_poe_texto(ji,31,"Operador Direcional de Kirsch       ",8,2);
   if (ki==5) dia_poe_texto(ji,31,"Operador Direcional de Prewitt      ",8,2);
   if (ki==6) dia_poe_texto(ji,31,"Operadores Direcionais de Robinson  ",8,2);
   if (ki==7) dia_poe_texto(ji,31,"Realcador de Linhas                 ",8,2);
   if (ki==8) dia_poe_texto(ji,31,"Detetor de Bordas Hibrido           ",8,2);
 }
 if (k==2) {
   if (ki==0) dia_poe_texto(ji,31,"Erro Medio Quadratico        ",8,2);
   if (ki==1) dia_poe_texto(ji,31,"Desvio Padrao                ",8,2);
   if (ki==2) dia_poe_texto(ji,31,"Perfil de Linha              ",8,2);
   if (ki==3) dia_poe_texto(ji,31,"Histograma                   ",8,2);
   if (ki==4) dia_poe_texto(ji,31,"Relacao Sinal/Ruido          ",8,2);
   if (ki==5) dia_poe_texto(ji,31,"Variacao Espacial            ",8,2);
 }
 if (k==3) {
   if (ki==0) dia_poe_texto(ji,31,"Ruido Distrib. Gaussiana     ",8,2);
   if (ki==1) dia_poe_texto(ji,31,"Ruido  Impulsivo             ",8,2);
   if (ki==2) dia_poe_texto(ji,31,"Ruido Distrib. Uniforme      ",8,2);
 }
```

```
  if (k==4) {
    if (ki==0) dia_poe_texto(ji,3i,"Disco --)  UVI                ",8,2);
    if (ki==1) dia_poe_texto(ji,3i,"UVI --)  Disco                ",8,2);
    if (ki==2) dia_poe_texto(ji,3i,"Disco --) Disco               ",8,2);
  }
}
switch(l=getchar()) {
  case '8':
    flag2=0;
    ji--; ki--;
    if(k == 3 || k == 4){
      if (ji(i0) ji=i2;
      if (ki(0) ki=2;
      ja= ji+i;
      if (ji==i2) ja=i0;
    }
    if (k==2){
      if (ji(i0) ji=i5;
      if (ki(0) ki=5;
      ja= ji+i;
      if (ji==i5) ja=i0;
    }
    if (k==i){
      if (ji(i0) ji=i8;
      if (ki(0) ki=8;
      ja= ji+i;
      if (ji==i8) ja=i0;
    }
    if (k==0){
      if (ji(i0) ji=20;
      if (ki(0) ki=i0;
      ja= ji+i;
      if (ji==20) ja=i0;
    }
    if (k==0)   {
      if (ki==0){
        dia_poe_texto(ja,3i,"Filtro da Mediana               ",8,7);
        dia_poe_texto(ji,3i,"Filtro da Media                 ",8,2);
      }
      if (ki==i){
        dia_poe_texto(ja,3i,"Filtros Sigma                   ",8,7);
        dia_poe_texto(ji,3i,"Filtro da Mediana               ",8,2);
      }
      if (ki==2){
       dia_poe_texto(ja,3i,"Filtros da Ordem                ",8,7);
        dia_poe_texto(ji,3i,"Filtros Sigma                   ",8,2);
      }
      if (ki==3){
        dia_poe_texto(ja,3i,"Media K vizinhos mais Proximos  ",8,7);
        dia_poe_texto(ji,3i,"Filtros da Ordem                ",8,2);
      }
      if (ki==4){
        dia_poe_texto(ja,3i,"Transformada Agucamento Extremo ",8,7);
        dia_poe_texto(ji,3i,"Media K vizinhos mais Proximos  ",8,2);
      }
```

```
  if (ki==5){
    dia_poe_texto(ja,3i,"Suavizacao Soma Dif Absolutas    ",8,7);
    dia_poe_texto(ji,3i,"Transformada Agucamento Extremo  ",8,2);
  }
  if (ki==6){
    dia_poe_texto(ja,3i,"Suavizacao Media Dif Absolutas   ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Soma Dif Absolutas    ",8,2);
  }
  if (ki==7){
    dia_poe_texto(ja,3i,"Suavizacao Logaritmica           ",8,7);;
    dia_poe_texto(ji,3i,"Suavizacao Media Dif Absolutas   ",8,2);
  }
  if (ki==8){
    dia_poe_texto(ja,3i,"Suavizacao Inverso do Gradiente  ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Logaritmica           ",8,2);
  }
  if (ki==9){{
    dia_poe_texto(ja,3i,"Suavizacao Variancia             ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Inverso do Gradiente  ",8,2);
  }
  if (ki==i0){
    dia_poe_texto(ja,3i,"Filtro da Media                  ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Variancia             ",8,2);
  }
}
if (k==i)   {
  if (ki==0){
    dia_poe_texto(ja,3i,"Operador Diferencial de Roberts     ",8,7);
    dia_poe_texto(ji,3i,"Operadores Diferenciais Laplacianos ",8,2);
  }
  if (ki==i){
    dia_poe_texto(ja,3i,"Operador Diferencial de Sobel       ",8,7);
    dia_poe_texto(ji,3i,"Operador Diferencial de Roberts     ",8,2);
  }
  if (ki==2){
    dia_poe_texto(ja,3i,"Operador Diferencial de Prewitt     ",8,7);
    dia_poe_texto(ji,3i,"Operador Diferencial de Sobel       ",8,2);
  }
  if (ki==3){
    dia_poe_texto(ja,3i,"Operador Direcional de Kirsch       ",8,7);
    dia_poe_texto(ji,3i,"Operador Diferencial de Prewitt     ",8,2);
  }
  if (ki==4){
    dia_poe_texto(ja,3i,"Operador Direcional de Prewitt      ",8,7);
    dia_poe_texto(ji,3i,"Operador Direcional de Kirsch       ",8,2);
  }

  if (ki==5){
    dia_poe_texto(ja,3i,"Operadoress Direcionais de Robinson  ",8,7);
    dia_poe_texto(ji,3i,"Operador Direcional de Prewitt       ",8,2);
  }
  if (ki==6){
    dia_poe_texto(ja,3i,"Realcador de Linhas                  ",8,7);
          dia_poe_texto(ji,3i,"Operadores Direcionais de Robinson  ",8,2);
  }
```

```
  if (ki==7){
    dia_poe_texto(ja,3i,"Detetor de Bordas Hibrido          ",8,7);
    dia_poe_texto(ji,3i,"Realcador de Linhas                ",8,2);
  }
  if (ki==8){
    dia_poe_texto(ja,3i,"Operadores Diferanciais Laplacianos ",8,7);
    dia_poe_texto(ji,3i,"Detetor de Bordas Hibrido          ",8,2);
  }
}
if (k==2) {
  if (ki==0){
    dia_poe_texto(ja,3i,"Desvio Padrao                ",8,7);
    dia_poe_texto(ji,3i,"Erro Medio Quadratico        ",8,2);
  }
  if (ki==1){
    dia_poe_texto(ja,3i,"Perfil de Linha              ",8,7);
    dia_poe_texto(ji,3i,"Desvio Padrao                ",8,2);
  }
  if (ki==2){
    dia_poe_texto(ja,3i,"Histograma                   ",8,7);
    dia_poe_texto(ji,3i,"Perfil de Linha              ",8,2);
  }
  if (ki==3){
    dia_poe_texto(ja,3i,"Relacao Sinal/Ruido          ",8,7);
    dia_poe_texto(ji,3i,"Histograma                   ",8,2);
  }
  if (ki==4){
    dia_poe_texto(ja,3i,"Variacao Espacial            ",8,7);
    dia_poe_texto(ji,3i,"Relacao Sinal/Ruido          ",8,2);
  }
  if (ki==5){
    dia_poe_texto(ja,3i,"Erro Medio Quadratico        ",8,7);
    dia_poe_texto(ji,3i,"Variacao Espacial            ",8,2);
  }
}
if (k==3) {
  if (ki==0){
    dia_poe_texto(ja,3i,"Ruido  Impulsivo             ",8,7);
    dia_poe_texto(ji,3i,"Ruido Distrib. Gaussiana     ",8,2);
  }
  if (ki==1){
    dia_poe_texto(ja,3i,"Ruido Distrib. Uniforme      ",8,7);
    dia_poe_texto(ji,3i,"Ruido  Impulsivo             ",8,2);
  }
  if (ki==2){
    dia_poe_texto(ja,3i,"Ruido Distrib. Gaussiana     ",8,7);
    dia_poe_texto(ji,3i,"Ruido Distrib. Uniforme      ",8,2);
  }
}
if (k==4) {
  if (ki==0){
    dia_poe_texto(ja,3i,"UVI --)  Disco               ",8,7);
    dia_poe_texto(ji,3i,"Disco --)  UVI               ",8,2);
  }
```

```
    if (ki==1){
      dia_poe_texto(ja,3i,"Disco --) Disco              ",8,7);
      dia_poe_texto(ji,3i,"UVI --)  Disco               ",8,2);
    }
    if (ki==2){
      dia_poe_texto(ja,3i,"Disco --)  UVI               ",8,7);
      dia_poe_texto(ji,3i,"Disco --) Disco              ",8,2);
    }
  }
 break;
case '2':
  flag2=0;
  ji++; ki++;
  if(k==3 || k==4){
    if (ji)i2) ji=i0;
    if (ki)2) ki=0;
    ja= ji-i;
    if (ji==i0) ja=i2;
  }
  if (k==2){
    if (ji)i5) ji=i0;
    if (ki)5) ki=0;
    ja= ji-i;
    if (ji==i0) ja=i5;
  }
  if (k==i){
    if (ji)i8) ji=i0;
    if (ki)8) ki=0;
    ja= ji-i;
    if (ji==i0) ja=i8;
  }
  if (k==0){
    if (ji)20) ji=i0;
    if (ki)i0) ki=0;
    ja= ji-i;
    if (ji==i0) ja=20;
  }
  if(k==0){
    if (ki==0){
      dia_poe_texto(ja,3i,"Suavizacao Variancia         ",8,7);
      dia_poe_texto(ji,3i,"Filtro da Media              ",8,2);
    }
    if (ki==i){
      dia_poe_texto(ja,3i,"Filtro da Media              ",8,7);
      dia_poe_texto(ji,3i,"Filtro da Mediana            ",8,2);
    }
    if (ki==2){
      dia_poe_texto(ja,3i,"Filtro da Mediana            ",8,7);
      dia_poe_texto(ji,3i,"Filtros Sigma                ",8,2);
    }
    if (ki==3){
      dia_poe_texto(ja,3i,"Filtros Sigma                ",8,7);
      dia_poe_texto(ji,3i,"Filtros da Ordem             ",8,2);
    }
```

```
  if (ki==4){
    dia_poe_texto(ja,3i,"Filtros da Ordem                ",8,7);
    dia_poe_texto(ji,3i,"Media K vizinhos mais Proximos  ",8,2);
  }
  if (ki==5){
    dia_poe_texto(ja,3i,"Media K vizinhos mais Proximos  ",8,7);
    dia_poe_texto(ji,3i,"Transformada Agucamento Extremo ",8,2);
  }
  if (ki==6){
    dia_poe_texto(ja,3i,"Transformada Agucamento Extremo ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Soma Dif Absolutas   ",8,2);
  }
  if (ki==7){
    dia_poe_texto(ja,3i,"Suavizacao Soma Dif Absolutas   ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Media Dif Absolutas  ",8,2);
  }
  if (ki==8){
    dia_poe_texto(ja,3i,"Suavizacao Media Dif Absolutas  ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Logaritmica          ",8,2);
 }
  if (ki==9){
    dia_poe_texto(ja,3i,"Suavizacao Logaritmica          ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Inverso do Gradiente ",8,2);
  }
  if (ki==10){
    dia_poe_texto(ja,3i,"Suavizacao Inverso do Gradiente ",8,7);
    dia_poe_texto(ji,3i,"Suavizacao Variancia            ",8,2);
  }
}
if (k==1)   {
  if (ki==0){
    dia_poe_texto(ja,3i,"Detetor de Bordas Hibrido          ",8,7);
    dia_poe_texto(ji,3i,"Operadores Diferenciais Laplacianos ",8,2);
  }
  if (ki==1){
    dia_poe_texto(ja,3i,"Operadores Diferenciais Laplacianos ",8,7);
    dia_poe_texto(ji,3i,"Operador Diferencial de Roberts     ",8,2);
  }
  if (ki==2){
    dia_poe_texto(ja,3i,"Operador Diferencial de Roberts     ",8,7);
    dia_poe_texto(ji,3i,"Operador Diferencial de Sobel       ",8,2);
  }
  if (ki==3){
    dia_poe_texto(ja,3i,"Operador Diferencial de Sobel       ",8,7);
    dia_poe_texto(ji,3i,"Operador Diferencial de Prewitt     ",8,2);
  }
  if (ki==4){
    dia_poe_texto(ja,3i,"Operador Diferencial de Prewitt     ",8,7);
    dia_poe_texto(ji,3i,"Operador Direcional de Kirsch       ",8,2);
  }
  if (ki==5){
    dia_poe_texto(ja,3i,"Operador Direcional de Kirsch       ",8,7);
    dia_poe_texto(ji,3i,"Operador Direcional de Prewitt      ",8,2);
  }
```

```
  if (ki==6){
    dia_poe_texto(ja,3i,"Operador Direcional de Prewitt      ",8,7);
    dia_poe_texto(ji,3i,"Operadores Direcionais de Robinson  ",8,2);
  }
  if (ki==7){
    dia_poe_texto(ja,3i,"Operadores Direcionais de Robinson  ",8,7);
    dia_poe_texto(ji,3i,"Realcador de Linhas                 ",8,2);
  }
  if (ki==8){
    dia_poe_texto(ja,3i,"Realcador de Linhas                 ",8,7);
    dia_poe_texto(ji,3i,"Detetor de Bordas Hibrido           ",8,2);
  }
}
if (k==2) {
  if (ki==0){
    dia_poe_texto(ja,3i,"Variacao Espacial             ",8,7);
    dia_poe_texto(ji,3i,"Erro Medio Quadratico         ",8,2);
  }
  if (ki==1){
    dia_poe_texto(ja,3i,"Erro Medio Quadratico         ",8,7);
    dia_poe_texto(ji,3i,"Desvio Padrao                 ",8,2);
  }
  if (ki==2){
    dia_poe_texto(ja,3i,"Desvio Padrao                 ",8,7);
    dia_poe_texto(ji,3i,"Perfil de Linha               ",8,2);
  }
  if (ki==3){
    dia_poe_texto(ja,3i,"Perfil de Linha               ",8,7);
    dia_poe_texto(ji,3i,"Histograma                    ",8,2);
 }
  if (ki==4){
    dia_poe_texto(ja,3i,"Histograma                    ",8,7);
    dia_poe_texto(ji,3i,"Relacao Sinal/Ruido           ",8,2);
  }
  if (ki==5){
    dia_poe_texto(ja,3i,"Relacao Sinal/Ruido           ",8,7);
    dia_poe_texto(ji,3i,"Variacao Espacial             ",8,2);
  }
}
if (k==3) {
  if (ki==0){
    dia_poe_texto(ja,3i,"Ruido Distrib. Uniforme       ",8,7);
    dia_poe_texto(ji,3i,"Ruido Distrib. Gaussiana      ",8,2);
  }
  if (ki==1){
    dia_poe_texto(ja,3i,"Ruido Distrib. Gaussiana      ",8,7);
    dia_poe_texto(ji,3i,"Ruido  Impulsivo              ",8,2);
  }
  if (ki==2){
    dia_poe_texto(ja,3i,"Ruido  Impulsivo              ",8,7);
    dia_poe_texto(ji,3i,"Ruido Distrib. Uniforme       ",8,2);
  }
}
if (k==4) {
  if (ki==0){
    dia_poe_texto(ja,3i,"Disco --) Disco               ",8,7);
```

```
        dia_poe_texto(j1,3i,"Disco --)  UVI              ",8,2);
      }
      if (ki==1){
        dia_poe_texto(ja,3i,"Disco --)  UVI              ",8,7);
        dia_poe_texto(j1,3i,"UVI --)  Disco              ",8,2);
      }
      if (ki==2){
        dia_poe_texto(ja,3i,"UVI --)  Disco              ",8,7);
        dia_poe_texto(j1,3i,"Disco --) Disco             ",8,2);
      }
    }
    break;
  case 'v':
    goto inicio;
    break;
  case 's':
    limphelp();
    uaihelp();
    switch(li=getchar()){
      case '0':
        shell("2:/cmds/uai 0");
        break;
      case '1':
        shell("2:/cmds/uai 1");
        break;
      case '2':
        shell("2:/cmds/uai 2");
        break;
      case 'g':
        shell("2:/cmds/uai 3");
        break;
      case 'c':
        shell("2:/cmds/uai");
        break;
      case 'v':
        goto nivel2;
        break;
   }
    break;
  case 'f':
    nivel3:
    if((k==0 && ki==2) || (k==0.&& ki==3)){
      if(k==0 && ki==2) escsigma();
      if(k==0 && ki==3) escordem();
      k2=0;   j2=13;
      flag3=1;
      for(;;){
        if(flag3==1){
          if(ki==2){
            if(k2==0) dia_poe_texto(j2,35,"Filtro Sigma Convencional ",0,2);
            if(k2==1) dia_poe_texto(j2,35,"Filtro Sigma Polarizado   ",0,2);
            if(k2==2) dia_poe_texto(j2,35,"Filtro Sigma Adaptativo   ",0,2);
          }
          if (ki==3){
            if(k2==0) dia_poe_texto(j2,35,"Filtro da Ordem masc  3x3   ",0,2);
            if(k2==1) dia_poe_texto(j2,35,"Filtro da Ordem masc  5x5   ",0,2);
```

```
    if(k2==2) dia_poe_texto(j2,35,"Filtro da Ordem Adaptativo  ",0,2);
  }
}
switch(l=getchar()){
  case '8':
    flag3=0;
    k2--; j2--;
    if (k2(0) k2=2;
    if (j2(13) j2=15;
    j2a= j2+1;
    if (j2==15) j2a=13;
    if(ki==2 && k2==0){
      dia_poe_texto(j2a,35,"Filtro Sigma Polarizado   ",1,5);
      dia_poe_texto(j2,35,"Filtro Sigma Convencional ",0,2);
    }
    if(ki==2 && k2==1){
      dia_poe_texto(j2a,35,"Filtro Sigma Adaptativo   ",1,5);
      dia_poe_texto(j2,35,"Filtro Sigma Polarizado   ",0,2);
    }
    if(ki==2 && k2==2){
      dia_poe_texto(j2a,35,"Filtro Sigma Convencional ",1,5);
      dia_poe_texto(j2,35,"Filtro Sigma Adaptativo   ",0,2);
    }
    if(ki==3 && k2==0){
      dia_poe_texto(j2a,35,"Filtro da Ordem masc  5x5   ",1,5);
      dia_poe_texto(j2,35,"Filtro da Ordem masc  3x3   ",0,2);

    if(ki==3 && k2==1){
      dia_poe_texto(j2a,35,"Filtro da Ordem Adaptativo  ",1,5);
      dia_poe_texto(j2,35,"Filto da Ordem masc  5x5   ",0,2);
    }
    if(ki==3 && k2==2){
      dia_poe_texto(j2a,35,"Filtro da Ordem masc  3x3   ",1,5);
      dia_poe_texto(j2,35,"Filtro da Ordem Adaptativo  ",0,2);
    }
    break;
  case '2':
    flag3=0;
    k2++; j2++;
    if (k2)2) k2=0;
    if (j2)15) j2=13;
    j2a= j2-1;
    if (j2==13) j2a=15;
    if(ki==2 && k2==0){
      dia_poe_texto(j2a,35,"Filtro Sigma Adaptativo   ",1,5);
     dia_poe_texto(j2,35,"Filtro Sigma Convencional ",0,2);

    if(ki==2 && k2==1){
      dia_poe_texto(j2a,35,"Filtro Sigma Convencional ",1,5);
      dia_poe_texto(j2,35,"Filtro Sigma Polarizado   ",0,2);
    }
    if(ki==2 && k2==2){
      dia_poe_texto(j2a,35,"Filtro Sigma Polarizado   ",1,5);
      dia_poe_texto(j2,35,"Filtro Sigma Adaptativo   ",0,2);
    }
```

```
        if(k1==3 && k2==0){
          dia_poe_texto(j2a,35,"Filtro da Ordem Adaptativo  ",1,5);
          dia_poe_texto(j2,35,"Filtro da Ordem masc  3x3   ",0,2);
        }
        if(k1==3 && k2==1){
          dia_poe_texto(j2a,35,"Filtro da Ordem masc  3x3   ",1,5);
          dia_poe_texto(j2,35,"Filtro da Ordem masc  5x5   ",0,2);
        }
        if(k1==3 && k2==2){
          dia_poe_texto(j2a,35,"Filtro da Ordem masc  5x5   ",1,5);
          dia_poe_texto(j2,35,"Filtro da Ordem Adaptativo  ",0,2);
        }
        break;
      case 'v':
        goto nivel2;
        break;
      case 's':
        limphelp();
        uaihelp();
        switch(li=getchar()) {
          case '0':
            shell("2:/cmds/uai 0");
            break;
          case '1':
            shell("2:/cmds/uai 1");
            break;
          case '2':
            shell("2:/cmds/uai 2");
            break;
          case 'g':
            shell("2:/cmds/uai 3");
            break;
          case 'c':
            shell("2:/cmds/uai");
            break;
          case 'v':
            goto nivel3;
            break;
        }
        break;
      case 'f':
        goto saida;
        break;
    }
  }
}
saida:
flag2=1;
buffer.tty_options=save;
set_attr(stdout,&buffer);
if (k==0) {
  if (k1==0) shell ("1:/filtlib/media");
  if (k1==1) shell ("1:/filtlib/mediana");
  if (k1==2 && k2==0) shell ("1:/filtlib/sigma");
 if (k1==2 && k2==1) shell ("1:/filtlib/sigmapol");
  if (k1==2 && k2==2) shell ("1:/filtlib/sigmadp");
```

```
            if (ki==3 && k2==0) shell ("i:/filtlib/rank3");
            if(ki==3 && k2==1) shell ("i:/filtlib/rank5");
            if (ki==3 && k2==2) shell ("i:/filtlib/ordap");
            if (ki==4) shell ("i:/filtlib/kviz");
            if (ki==5) shell ("i:/filtlib/exsharp");
            if (ki==6) shell ("i:/filtlib/ssda");
            if (ki==7) shell ("i:/filtlib/smda");
            if (ki==8) shell ("i:/filtlib/sslog");
            if (ki==9) shell ("i:/filtlib/ssgrad");
            if (ki==10) shell ("i:/filtlib/ssvar");
          }
          if (k==1) {
            if (ki==0) shell ("i:/filtlib/laplac");
            if (ki==1) shell ("i:/filtlib/roberts");
            if (ki==2) shell ("i:/filtlib/sobel");
            if (ki==3) shell ("i:/filtlib/difprewitt");
            if (ki==4) shell ("i:/filtlib/kirsch");
            if (ki==5) shell ("i:/filtlib/dirprewitt");
            if (ki==6) shell ("i:/filtlib/robinson");
            if (ki==7) shell ("i:/filtlib/realclin");
            if (ki==8) shell ("i:/filtlib/hibridet");
          }
          if (k==2) {
            if (ki==0) shell ("i:/estatlib/erquad");
            if (ki==1) shell ("i:/estatlib/desvpad");
            if (ki==2) shell ("i:/estatlib/perflin");
            if (ki==3) shell ("i:/estatlib/hist");
            if (ki==4) shell ("i:/estatlib/varesp");
          }
          if (k==3) {
            if (ki==0) shell ("i:/rdlib/rdgauss");
            if (ki==1) shell ("i:/rdlib/rdimp");
            if (ki==2) shell ("i:/rdlib/rdunif");
          }
          if (k==4) {
            if (ki==0) shell ("i:/movlib/disc_uvi");
            if (ki==1) shell ("i:/movlib/uvi_disc");
            if (ki==2) shell ("i:/movlib/disc_disc");
          }
          get_attr(stdout,&buffer);
          save=buffer.tty_options;
          buffer.tty_options &= ~EDIT;
          buffer.tty_options &= ~ECHO;
          set_attr(stdout,&buffer);
          limpa();
          flag1=1;
          goto inicio;
          break;
        }
      }
    }
  }
  final:
  buffer.tty_options=save;
  set_attr(stdout,&buffer);
}
```

```
/*
**
*Ë       FUNCOEõ Dä AUXILIO
**
**
*/
escretela()
{
  int i;
  i=1;
  dia_poe_texto(7,19,"                                    ",6,i);
  dia_poe_texto(8,19,"  Suavizacao Espacial               ",6,i);
  dia_poe_texto(9,19,"  Detecao de Bordas                 ",6,i);
  dia_poe_texto(10,19,"  Medicoes Estatisticas             ",6,i);
  dia_poe_texto(11,19,"  Geracao de Ruido                  ",6,i);
  dia_poe_texto(12,19,"  Transferencia de Imagens          ",6,i);
  dia_poe_texto(13,19,"                                    ",6,i);
}


escretela1()
{
  int i=7;
  dia_poe_texto(9,29,"                                   ",8,i);
  dia_poe_texto(10,29,"  Filtro da Media                  ",8,i);
  dia_poe_texto(11,29,"  Filtro da Mediana                ",8,i);
  dia_poe_texto(12,29,"  Filtros Sigma                    ",8,i);
  dia_poe_texto(13,29,"  Filtros da Ordem                 ",8,i);
  dia_poe_texto(14,29,"  Media K vizinhos mais Proximos   ",8,i);
  dia_poe_texto(15,29,"  Transformada Agucamento Extremo  ",8,i);
  dia_poe_texto(16,29,"  Suavizacao Soma Dif Absolutas    ",8,i);
  dia_poe_texto(17,29,"  Suavizacao Media Dif Absolutas   ",8,i);
  dia_poe_texto(18,29,"  Suavizacao Logaritmica           ",8,i);
  dia_poe_texto(19,29,"  Suavizacao Inverso do Gradiente  ",8,i);
  dia_poe_texto(20,29,"  Suavizacao Variancia             ",8,i);
  dia_poe_texto(21,29,"                                   ",8,i);
}

escretela2()
{
  int i=7;
  dia_poe_texto(9,29,"                                     ",8,i);
  dia_poe_texto(10,29,"  Operadores Diferenciais Laplacianos  ",8,i);
  dia_poe_texto(11,29,"  Operador Diferencial de Roberts      ",8,i);
  dia_poe_texto(12,29,"  Operador Diferencial de Sobel        ",8,i);
  dia_poe_texto(13,29,"  Operador Diferencial de Prewitt      ",8,i);
  dia_poe_texto(14,29,"  Operador Direcional de Kirsch        ",8,i);
  dia_poe_texto(15,29,"  Operador Direcional de Prewitt       ",8,i);
  dia_poe_texto(16,29,"  Operadores Direcionais de Robinson   ",8,i);
  dia_poe_texto(17,29,"  Realcador de Linhas                  ",8,i);
  dia_poe_texto(18,29,"  Detetor de Bordas Hibrido            ",8,i);
  dia_poe_texto(19,29,"                                       ",8,i);
}
```

```
escretela3()
{
  int i=7"
  dia_poe_texto(9,29,"                                          ",8,i);
  dia_poe_texto(10,29,"  Erro Medio Quadratico                   ",8,i);
  dia_poe_texto(11,29,"  Desvio Padrao                           ",8,i);
  dia_poe_texto(12,29,"  Perfil de Linha                         ",8,i);
  dia_poe_texto(13,29,"  Histograma                              ",8,i);
  dia_poe_texto(14,29,"  Relacao Sinal/Ruido                     ",8,i);
  dia_poe_texto(15,29,"  Variacao Espacial                       ",8,i);
  dia_poe_texto(16,29,"                                          ",8,i);
}

escretela4()
{
  int i=7;
  dia_poe_texto(9,29,"                                          ",8,i);
  dia_poe_texto(10,29,"  Ruido Distrib. Gaussiana                ",8,i);
  dia_poe_texto(11,29,"  Ruido  Impulsivo                        ",8,i);
  dia_poe_texto(12,29,"  Ruido Distrib. Uniforme                 ",8,i);
  dia_poe_texto(13,29,"                                          ",8,i);
}


escretela5()
{
  int i=7;
  dia_poe_texto(9,29,"                                          ",8,i);
  dia_poe_texto(10,29,"  Discoo --)  UVI                         ",8,i);
  dia_poe_texto(11,29,"  UVI --)  Disco                          ",8,i);
  dia_poe_texto(12,29,"  Disco --) Disco                         ",8,i);
  dia_poe_texto(13,29,"                                          ",8,i);
}


escabre()
{
  putchar(FORMFEED);
  dia_poe_texto(0,0,"                                                                ",8,7);
  dia_poe_texto(1,0,"          Filtragem  Espacial  de  Imagens                      ",1,7);
  eschelp0();
                                        /* linha 05 */
  dia_poe_texto(5,27,"     ",0,1);
  dia_poe_texto(5,37,"    ",0,1);
  dia_poe_texto(5,44,"    ",0,1);
                                       /* linha 06 */
  dia_poe_texto(6,26,"      ",0,1);
  dia_poe_texto(6,37,"    ",0,1);
  dia_poe_texto(6,44,"    ",0,1);
                                        /* linha 07 */
  dia_poe_texto(7,25,"     ",0,1);
  dia_poe_texto(7,37,"    ",0,1);
  dia_poe_texto(7,44,"    ",0,1);
```

```
                                                    /* linha 08 */
 dia_poe_texto(8,25,"    ",0,1);
 dia_poe_texto(8,37,"    ",0,1);
 dia_poe_texto(8,44,"    ",0,1);
                                                    /* linha 09 */
 dia_poe_texto(9,25,"    ",0,1);
 dia_poe_texto(9,37,"    ",0,1);
 dia_poe_texto(9,44,"    ",0,1);
 dia_poe_texto(9,76,"R ",2,0);
                                                    /* linha 10 */
 dia_poe_texto(10,25,"    ",0,1);
 dia_poe_texto(10,37,"    ",0,1);
 dia_poe_texto(10,44,"    ",0,1);
                                                    /* linha 11 */
 dia_poe_texto(11,25,"      ",0,1);
 dia_poe_texto(11,32,"    ",0,1);
 dia_poe_texto(11,37,"    ",0,1);
 dia_poe_texto(11,42,"       ",0,1);
 dia_poe_texto(11,53,"      ",0,1);
 dia_poe_texto(11,61,"    ",0,1);
 dia_poe_texto(11,66,"    ",0,1);
 dia_poe_texto(11,72,"    ",0,1)
                                                    /* linha 12 */
 dia_poe_texto(12,25,"      ",0,1);
 dia_poe_texto(12,32,"    ",0,1);
 dia_poe_texto(12,37,"    ",0,1);
 dia_poe_texto(12,42,"       ",0,1);
 dia_poe_texto(12,52,"       ",0,1);
 dia_poe_texto(12,61,"    ",0,1);
 dia_poe_texto(12,66,"    ",0,1);
 dia_poe_texto(12,72,"    ",0,1);
                                                    /* linha 13 */
 dia_poe_texto(13,25,"    ",0,1);
 dia_poe_texto(13,37,"    ",0,1);
 dia_poe_texto(13,44,"    ",0,1);
 dia_poe_texto(13,51,"    ",0,1);
 dia_poe_texto(13,67,"       ",0,1);
                                                    /* linha 14 */
 dia_poe_texto(14,25,"    ",0,1);
 dia_poe_texto(14,32,"    ",0,1);
 dia_poe_texto(14,37,"    ",0,1);
 dia_poe_texto(14,44,"    ",0,1);
 dia_poe_texto(14,51,"    ",0,1);
 dia_poe_texto(14,61,"    ",0,1);
 dia_poe_texto(14,68,"     ",0,1);
                                                    /* linha 15 */
 dia_poe_texto(15,25,"    ",0,1);
 dia_poe_texto(15,32,"    ",0,1);
 dia_poe_texto(15,37,"    ",0,1);
 dia_poe_texto(15,44,"    ",0,1);
 dia_poe_texto(15,51,"    ",0,1);
 dia_poe_texto(15,61,"    ",0,1);
 dia_poe_texto(15,68,"     ",0,1);
                                                    /* linha 16 */
 dia_poe_texto(16,25,"    ",0,1);
 dia_poe_texto(16,32,"    ",0,1);
```

```
    dia_poe_texto(16,37,"    ",0,1);
    dia_poe_texto(16,44,"    ",0,1);
    dia_poe_texto(16,51,"    ",0,1);
    dia_poe_texto(16,61,"    ",0,1);
    dia_poe_texto(16,69,"    ",0,1)
                                        /* linha 17 */
    dia_poe_texto(17,25,"    ",0,1);
    dia_poe_texto(17,32,"    ",0,1);
    dia_poe_texto(17,37,"    ",0,1);
    dia_poe_texto(17,44,"    ",0,1);
    dia_poe_texto(17,51,"    ",0,1);
    dia_poe_texto(17,61,"    ",0,1);
    dia_poe_texto(17,69,"    ",0,1);
                                        /* linha 18 */
    dia_poe_texto(18,25,"    ",0,1);
    dia_poe_texto(18,32,"    ",0,1);
    dia_poe_texto(18,37,"    ",0,1);
    dia_poe_texto(18,44,"    ",0,1);
    dia_poe_texto(18,51,"    ",0,1);
    dia_poe_texto(18,61,"    ",0,1);
    dia_poe_texto(18,68,"      ",0,1);
                                        /* linha 19 */
    dia_poe_texto(19,25,"    ",0,1);
    dia_poe_texto(19,32,"    ",0,1);
    dia_poe_texto(19,37,"    ",0,1);
    dia_poe_texto(19,44,"    ",0,1);
    dia_poe_texto(19,51,"    ",0,1);
    dia_poe_texto(19,61,"    ",0,1);
    dia_poe_texto(19,68,"      ",0,1);
                                       /* linha 20 */
    dia_poe_texto(20,25,"    ",0,1);
    dia_poe_texto(20,32,"    ",0,1);
    dia_poe_texto(20,37,"    ",0,1);
    dia_poe_texto(20,44,"    ",0,1);
    dia_poe_texto(20,51,"    ",0,1);
    dia_poe_texto(20,61,"    ",0,1);
    dia_poe_texto(20,67,"       ",0,1);
                                        /* linha 21 */
    dia_poe_texto(21,25,"    ",0,1);
    dia_poe_texto(21,32,"    ",0,1);
    dia_poe_texto(21,37,"    ",0,1);
    dia_poe_texto(21,44,"    ",0,1);
    dia_poe_texto(21,51,"    ",0,1);
    dia_poe_texto(21,61,"    ",0,1);
    dia_poe_texto(21,66,"    ",0,1);
    dia_poe_texto(21,72,"    ",0,1);
                                        /* linha 22 */
    dia_poe_texto(22,25,"    ",0,1);
    dia_poe_texto(22,32,"    ",0,1);
    dia_poe_texto(22,37,"    ",0,1);
    dia_poe_texto(22,44,"    ",0,1);
    dia_poe_texto(22,51,"    ",0,1);
    dia_poe_texto(22,61,"    ",0,1);
    dia_poe_texto(22,66,"    ",0,1);
    dia_poe_texto(22,72,"    ",0,1);
}
```

```
eschelp0()
{
  dia_poe_texto(5,0,"             ",0,2);
  dia_poe_texto(6,0,"    AJUDA    ",1,2);
  dia_poe_texto(7,0,"             ",0,2);
  dia_poe_texto(8,0,"    ` v '    ",0,2);
  dia_poe_texto(9,0,"  volta ao   ",0,2);
  dia_poe_texto(10,0,"    ANALIX   ",0,2);
  dia_poe_texto(11,0,"             ",0,2);
  dia_poe_texto(12,0,"    ` c '    ",0,2);
  dia_poe_texto(13,0,"  continua   ",0,2);
  dia_poe_texto(14,0,"             ",0,2);
}


telainf()
{
  dia_poe_texto(0,0,"                                                                ",8,7);
  dia_poe_texto(1,0,"    F I L T R I X      Filtragem Espacial de Imagens            ",1,7);
  dia_poe_texto(5,20,"                                                          ",1,7);
  dia_poe_texto(6,20,"    Este software  e' dedicado `a  Filtragem  Espacial  de ",1,7);
  dia_poe_texto(7,20," Imagens para tarefas de Realce e Preprocessamento.       ",1,7);
  dia_poe_texto(8,20,"                                                          ",1,7);
  dia_poe_texto(9,20,"    Filtrix dispoe de tecnicas de Suavizacao e de  Detecao ",1,7);
  dia_poe_texto(10,20," de Bordas, bem como de metodos estatisticos para avaliacao ",1,7);
  dia_poe_texto(11,20," do desempenho dos algoritmos em cada aplicacao.         ",1,7);
  dia_poe_texto(12,20,"                                                         ",1,7);
  dia_poe_texto(13,20,"    Maiores informacoes contactar:                       ",1,7);
  dia_poe_texto(14,20,"                                                         ",1,7);
  dia_poe_texto(15,20,"    Marcelo Alves de Barros                              ",1,7);
  dia_poe_texto(16,20,"    Depto Engenharia Eletrica - UFPb                     ",1,7);
  dia_poe_texto(17,20,"    Rua Carmem Abreu Silveira 206  (083) 322 3415        ",1,7);
  dia_poe_texto(18,20,"    Campina Grande - Paraiba - Cep 58.100                ",1,7);
  dia_poe_texto(19,20,"                                                         ",1,7);
}

eschelp1()
{
  dia_poe_texto(5,0,"             ",0,2);
  dia_poe_texto(6,0,"    AJUDA    ",1,2);
  dia_poe_texto(7,0,"    ` v '    ",0,2);
  dia_poe_texto(8,0,"    volta    ",0,2);
  dia_poe_texto(9,0,"             ",0,2);
  dia_poe_texto(10,0,"    ` f '    ",0,2);
  dia_poe_texto(11,0,"    ativa    ",0,2);
  dia_poe_texto(12,0,"    funcao   ",0,2);
  dia_poe_texto(13,0,"             ",0,2);
  dia_poe_texto(14,0,"    ` s '    ",0,2);
  dia_poe_texto(15,0,"   selecao   ",0,2);
  dia_poe_texto(16,0,"  de canais  ",0,2);
  dia_poe_texto(17,0,"             ",0,2);
}
```

```
uaihelp()
{
  dia_poe_texto(5,0,"              ",0,2);
  dia_poe_texto(6,0,"     AJUDA    ",1,2);
  dia_poe_texto(7,0,"      ' v '   ",0,2);
  dia_poe_texto(8,0,"     volta    ",0,2);
  dia_poe_texto(9,0,"              ",0,2);
  dia_poe_texto(10,0,"    CANAIS:   ",0,2);
  dia_poe_texto(11,0,"              ",0,2);
  dia_poe_texto(12,0,"   0,  1,  2  ",0,2);
  dia_poe_texto(13,0,"              ",0,2);
  dia_poe_texto(14,0," c - colorido ",0,2);
  dia_poe_texto(15,0," g - grafico  ",0,2);
  dia_poe_texto(16,0,"              ",0,2);
}

msg0()
{
  dia_poe_texto(21,0,"              ",0,1);
  dia_poe_texto(22,0," Digite opcao ",0,1);
  dia_poe_texto(23,0,"              ",0,1);
  dia_poe_texto(22,13," ",0,1);
}


limpa()
{
  int i;
  for(i=5; i(24; i++){
    dia_poe_texto(i,17,"                                ",0,0);
    dia_poe_texto(i,47,"                                ",0,0);
  }
}


limphelp()
{
  int i;
  for(i=3; i(20; i++){
    dia_poe_texto(i,0,"              ",0,0);
  }
}

escsigma()
{
  dia_poe_texto(12,33,"                                  ",1,5);
  dia_poe_texto(13,33,"  Filtro Sigma Convencional       ",1,5);
  dia_poe_texto(14,33,"  Filtro Sigma Polarizado         ",1,5);
  dia_poe_texto(15,33,"  Filtro Sigma Adaptativo         ",1,5);
  dia_poe_texto(16,33,"                                  ",1,5);
}
```

```
escordem()
{
  dia_poe_texto(12,33,"                                    ",1,5);
  dia_poe_texto(13,33,"  Filtro da Ordem masc  3x3          ",1,5);
  dia_poe_texto(14,33,"  Filtro da Ordem masc  5x5          ",1,5);
  dia_poe_texto(15,33,"  Filtro da Ordem Adaptativo         ",1,5);
  dia_poe_texto(16,33,"                                    ",1,5);
}
```

```
#include (stdio.h)
#include (uai.h)
#include (math.h)
#include (dia.h)
/*
**
**   FILTRO  DA MEDIA P/MASCARAS 3x3
**
*/
main()
{
int  j[4], jj[4];
int  i,k,b,t,ce,cs,is,smed;
unsigne char  bi[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]=(j[0]-1); jj[1]=(j[1]+2);
jj[2]=(j[2]-1); jj[3]=(j[3]+2);

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i((j[3]+2); i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    smed= bi[k]+bi[k+1]+bi[k+2];
    smed= smed+b2[k]+b2[k+1]+b2[k+2];
    bs[k]= (smed+b3[k]+b3[k+1]+b3[k+2])/9.+0.5;
  }
  is = i-2;
  uai_esc_linha(cs,is,bs,j);
  for(b=0; b(j[1]+2; b++){
    bi[b]= b2[b];
    b2[b]= b3[b];
  }
}
}
```

```
#include (stdio.h)
#include (uai.h)
#include (math.h)
#include (dia.h)
/*
**
**  FILTRO  DA MEDIA P/MASCARAS 5x5
**
*/
main()
{
float smed;
int  j[4], jj[4];
int  i,k,z,t,ce,cs,is;
unsigne char  bi[512], b2[512], b3[512];
unsigne cha  b4[512], b5[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]=(j[0]-2); jj[1]=(j[1]+4);
jj[2]=(j[2]-2); jj[3]=(j[3]+4);

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);

for(i=4; i((j[3]+4); i++){
  uai_le_linha(ce,i,b5,jj);
  for(k=0; k(j[1]; k++){
    smed= 0.;
    for(t=k; t((k+5); t++){
      smed= smed+b1[t]+b2[t]+b3[t];
      smed= smed+b4[t]+b5[t];
    }
    bs[k]= smed/25.+ 0.5;
  }
  is = i-4;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t(j[1]+4; t++){
    b1[t]= b2[t];
    b2[t]= b3[t];
    b3[t]= b4[t];
    b4[t]= b5[t];
  }
}
}
```

```
#include (dia.h)
#include (uai.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**      FILTRO  DA  MEDIANA  3 X 3
**
**
*/
int   i,k,ce,cs,q,is,t,z,w,c,a,min,indmin,indaux;
int   j[4], jj[4];
unsigned int vord[9], vet[9];
unsigne char  bi[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados((;

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= (j[0]-1); jj[1]= (j[1]+2);
jj[2]= (j[2]-1); jj[3]= (j[3]+2);

min= 0;
indmin= 0;
uai_le_linha(ce,0,bi,jj);
uai_le_linha(ce,1,b2,jj);

for(i= 4; i(j[3]+2; i++){
  uai_le_linha(ce,i,b3,jj);
  for(k= 0; k(j[1]; k++){
    vet[0]= bi[k];
    vet[1]= bi[k+1];
    vet[2]= bi[k+2];
    vet[3]= b2[k];
    vet[4]= b2[k+1];
    vet[5]= b2[k+2];
    vet[6]= b3[k];
    vet[7]= b3[k+1];
    vet[8]= b3[k+2];
    bubble(vet,9);
    bs[k]= vet[4];
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(a= 0; a((j[1]+2); a++){
    bi[a]= b2[a];
    b2[a]= b3[a];
  }
}
}
```

```
#include <dia.h>
#include <uai.h>
#include <stdio.h>
#include <math.h>
main()
{
/*
**
**      FILTRO  DA  MEDIANA  5 X 5
**
*/
int   lbi,i,k,ce,cs,q,is,t,z,w,c,a,min,indmin,indaux;
int   j[4], jj[4];
unsigned int vord[25], vet[25];
unsigne char  bi[512], b2[512], b3[512];
unsigne cha  b4[512], b5[512], bs[512];

limpahelp();
mssg();
pegdados;

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= (j[0]-2); jj[1]= (j[1]+4);
jj[2]= (j[2]-2); jj[3]= (j[3]+4);

min= 0;
indmin= 0;
uai_le_linha(ce,0,bi,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);

for(i= 4; i<j[3]+4; i++){
  uai_le_linha(ce,i,b5,jj);
  for(k= 0; k<j[1]; k++){
    vet[0]= bi[k];
    vet[1]= bi[k+1];
    vet[2]= bi[k+2];
    vet[3]= bi[k+3];
    vet[4]= bi[k+4];
    vet[5]= b2[k];
    vet[6]= b2[k+1];
    vet[7]= b2[k+2];
    vet[8]= b2[k+3];
    vet[9]= b2[k+4];
    vet[10]= b3[k];
    vet[11]= b3[k+1];
    vet[12]= b3[k+2];
    vet[13]= b3[k+3];
    vet[14]= b3[k+4];
    vet[15]= b4[k];
    vet[16]= b4[k+1];
    vet[17]= b4[k+2];
```

```
      vet[18]= b4[k+3];
      vet[19]= b4[k+4];
      vet[20]= b5[k];
      vet[21]= b5[k+1];
      vet[22]= b5[k+2];
      vet[23]= b5[k+3];
      vet[24]= b5[k+4];
      bubble(vet,25);
      bs[k]=vet[12];
    }
    is= i-4;
    uai_esc_linha(cs,is,bs,j);
    for(a= 0; a<((j[1]+4); a++){
      b1[a]= b2[a];
      b2[a]= b3[a];
      b3[a]= b4[a];
      b4[a]= b5[a];
    }
  }
  }
```

```
#include (dia.h)
#include (uai.h)
#include (stdio.h)
#include (math.h)

main()
{
/*
** FILTRO  DA  MEDIA DOS K-VIZINHOS MAIS PROXIMOS
**
*/
int i,ce,cs,ta,kk,j[4];
limpahelp();
msg();
pegdad2();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

if( ta == 3 ){
  int   k,q,is,t,z,w,l,c,a,indmin,min,pxmin,pxc;
  int  jj[4];
  unsigned int  vet[9],dif[9],pxord[9];
  unsigne in  diford[9],dift[9];
  unsigne char  b1[512], b2[512], b3[512], bs[512];
  float medk,som;

  jj[0]= (j[0]-1); jj[1]= (j[1]+2);
  jj[2]= (j[2]-1); jj[3]= (j[3]+2);
```

```
uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i((j[3]+2); i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[i]; k++){
    for(q=0; q(9; q++){
      diford[q]= 0;
      pxord[q]= 0;
      vet[q]= 0;
      dif[q]= 0;
      dift[q]= 0;
    }
    min= 0; indmin= 0; pxmin= 0;
    pxc= b2[k+1];
    vet[0]= b1[k];
    dif[0]= abs(pxc - b1[k]);
    vet[1]= b1[k+1];
    dif[1]= abs(pxc - b1[k+1]);
    vet[2]= b1[k+2];
    dif[2]= abs(pxc - b1[k+2]);
    vet[3]= b2[k];
    dif[3]= abs(pxc - b2[k]);
    vet[4]= b2[k+1];
    dif[4]= 0;
    vet[5]= b2[k+2];
    dif[5]= abs(pxc - b2[k+2]);
    vet[6]= b3[k];
    dif[6]= abs(pxc - b3[k]);
    vet[7]= b3[k+1];
    dif[7]= abs(pxc - b3[k+1]);
    vet[8]= b3[k+2];
    dif[8]= abs(pxc - b3[k+2])`
    for(z=0; z(9; z++) dift[z]= dif[z];

    for(z=8; z)=0; z--){ /* ORDENA VETOR DE PARES */
      min=  dif[z];
      indmin= z;
      pxmin= vet[z];
      for(t=0; t((z+1); t++){
        if(dif[t] (= min){
          min= dif[t];
          indmin= t;
          pxmin= vet[t];
        }
      }
      w= 8-z;
      diford[w]= min;
      pxord[w]= pxmin; /* ORDENA POR PROXIMIDADE */
      if(w == (kk-1)) c= z;
      for(c=indmin; c(z; c++) dif[c]= dif[c+1];
      for(c=indmin; c(z; c++) vet[c]= vet[c+1];
    }
```

```
      som= 0;    /* MEDIA DOS K VIZINHOS  */
      for(q=0; q<kk; q++) som= som + pxord[q];
      medk= som / kk;
      bs[k]= medk + 0.5;
    }
    is= i-2;      /* ESC LINHA SAIDA       */
    uai_esc_linha(cs,is,bs,j);
    for(a= 0; a<(j[1]+2); a++){
      bi[a]= b2[a];
      b2[a]= b3[a];
    }
  }
}

if( ta == 5 ){

  int   k,q,is,t,z,w,l,c,a,indmin,min,pxmin,pxc;
  int  jj[4];
  unsigned int  vet[25],dif[25],pxord[25];
  unsigne in  diford[25],dift[25];
  unsigne char  bi[512], b2[512], b3[512];
  unsigne cha  b4[512], b5[512], bs[512];
  float medk,som;

  jj[0]= (j[0]-2); jj[1]= (j[1]+4);
  jj[2]= (j[2]-2); jj[3]= (j[3]+4);

  uai_le_linha(ce,0,b1,jj);
  uai_le_linha(ce,1,b2,jj);
  uai_le_linha(ce,2,b3,jj);
  uai_le_linha(ce,3,b4,jj);
  for(i=4; i<(j[3]+4); i++){
    uai_le_linha(ce,i,b5,jj);
    for(k=0; k<j[1]; k++){
      min= 0; indmin= 0; pxmin= 0; pxc= 0;
      pxc= b3[k+2];
      for(z=0; z<5; z++){
        vet[z]= bi[k+z];
        dif[z]= abs(pxc - bi[k+z]);
        vet[5+z]= b2[k+z];
        dif[5+z]= abs(pxc - b2[k+z]);
        vet[10+z]= b3[k+z];
        dif[10+z]= abs(pxc - b3[k+z]);
        vet[15+z]= b4[k+z];
        dif[15+z]= abs(pxc - b4[k+z]);
        vet[20+z]= b5[k+z];
        dif[20+z]= abs(pxc - b5[k+z]);
      }
      for(z=0; z<25; z++) dift[z]= dif[z];
      for(z=24; z>=0; z--){ /* ORDENA VETOR DE PARES */
        min=  dif[z];
        indmin= z;
        pxmin= vet[z];
        for(t=0; t<(z+1); t++){
          if(dif[t] <= min){
            min= dif[t];
```

```
            indmin= t;
            pxmin= vet[t];
          }
        }
        w= 24-z;
        diford[w]= min;
        pxord[w]= pxmin;  /* ORDENA POR PROXIMIDADE */
        if(w == (kk-1)) c= z;
        for(c=indmin; c(z; c++) dif[c]= dif[c+1];
        for(c=indmin; c(z; c++) vet[c]= vet[c+1];
      }
      som= 0; /* MEDIA DOS K VIZINHOS  */
      for(q=0; q(kk; q++) som= som + pxord[q];
      medk= som / kk;
      bs[k]= medk + 0.5;
    }
    is= i-4;  /* ESC LINHA SAIDA  */
    uai_esc_linha(cs,is,bs,j);
    for(a= 0; a((j[1]+4); a++){
      b1[a]= b2[a];
      b2[a]= b3[a];
      b3[a]= b4[a];
      b4[a]= b5[a];
    }
  }
}
}
```

```
#include (stdio.h)
#include (uai.h)
#include (math.h)
#include (dia.h)
main()
{
/*
**
**              FILTRO  DA ORDEM ADAPTATIVO
**
*/

float smed;
int   u,ss,w,z,c,min,indmin;
int   ln, i,k,n,m,x,y,ce,cs,r,t,med,s,is,pj,mediana,csh,csv;
int   vet[5],vetest[5],vord[5], j[4], jj[4];
unsigne char  b1[512], b2[512], b3[512];
unsigne cha  b4[512], b5[512], bs[512], b[512];

limpahelp();
msg();
pegdados();
```

```
j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= j[0]-2; jj[1]= j[1]+4;
jj[2]= j[2]-2; jj[3]= j[3]+4;

for(i=0; i<4; i++){  /* LE E PROCESSA LINHAS INICIAIS   */
  uai_le_linha(ce,i,b,jj);
  for(k=0; k<j[1]; k++){
    for(z=0; z<5; z++){
      vord[z]= 0;
      vetest[z]= 0;
      vet[z]= 0;
    }
    vet[0]= b[k];
    vet[1]= b[k+1];
    vet[2]= b[k+2];
    vet[3]= b[k+3];
    vet[4]= b[k+4];
    for( z=0; z<5; z++) vetest[z]= vet[z];
                        /* CALCULA MEDIANA */
    for(z=4; z>=0; z--){
      min= vet[z];
      indmin= z;
      for(t=0; t<(z+1); t++){
        if(vet[t] < min){
          min= vet[t];
          ndmin= t;
        }
      }
      w= 4-z;
      vord[w]= min;
      for(c=indmin; c<z; c++) vet[c] = vet[c+1];
    }
    mediana= vord[2];
              /* CALCULA MEDIA */
    smed= 0.;
    for(t=0; t<5; t++)  smed= smed + vord[t];
    med= smed / 5 + 0.5;
             /* CALCULA CSH */
    if( med >= mediana ) csh= vord[2-pj];
    else csh= vord[2+pj];
    if(i==0 ) b1[k+2]= csh;
    if(i==1 ) b2[k+2]= csh;
    if(i==2 ) b3[k+2]= csh;
    if(i==3 ) b4[k+2]= csh;
  }
}
          /* COMECA PROCESSO TODA  A IMAGEM */
for(ln=0; ln<j[3]+4; ln++){     i= ln+4;
  uai_le_linha(ce,i,b5,jj);
  for(k=0; k<j[1]; k++){
               /* CALCULA CSH DA LINHA ATUAL  */
          /*  calcula mediana usando b5[k] b5[k+1]
                 b5[k+2] b5[k+3] b5[k+4] */
```

```
    for(z=0; z(5; z++) vord[z]= 0;
    vet[0]= b5[k]; vet[1]= b5[k+1];
    vet[2]= b5[k+2]; vet[3]= b5[k+3];
    vet[4]= b5[k+4];
    bubble(vet,5);
    mediana= vet[2];
                  /* calcula media */
    smed= 0.;
    for(t=0; t(5; t++)  smed= smed + vord[t];
    med= smed/5 + 0.5;
                      /*  SELECIONA  */
    if( med )= mediana ) csh= vord[2-pj];
    else  csh= vord[2+pj];
    b5[k+2]= csh;
  }
  for(k=0; k(j[1]; k++){
    u= k+2;
            /*  CALCULA  CSV  DOS 5 BUFFERS  */
           /* calcula mediana usando b1[u]
              b2[u] b3[u] b4[u] e b5[u] */

    for(z=0; z(5; z++) vord[z]= 0;
      vet[0]= b1[u]; vet[1]= b2[u]; vet[2]= b3[u];
      vet[3]= b4[u]; vet[4]= b5[u];
      bubble(vet,5);
      mediana= vet[2];
      smed= 0.;
      for(t=0; t(5; t++ê smed= smed + vord[t];
      med= smed / 5 + 0.5;
      if( med )= mediana ) csv= vord[2-pj];
      else  csv= vord[2+pj];
      bs[k]= csv;
    }
    is= i-4;
    uai_esc_linha(cs,is,bs,j); /* ESCREVE LINHA DE SAIDA */

    for(r=0; r(j[1]+4; r++){
      b1[r]= b2[r];
      b2[r]= b3[r];
      b3[r]= b4[r];
      b4[r]= b5[r];
    }
  }
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)

main()
{
/*
*Ë  SUAVIZACAO  BASEADA  NO  MODELO  DE  FACETAS
**
*/
int  i,k,b,t,r,z,s,eeind,is,ks,ce,cs;
int j[4], jj[4];
float fac[9],ee[9],eemin,x1,x2,x3,y1,y2,y3,w1,w2,w3;
unsigne char b1[512], b2[512], b3[512];
unsigned char b4[512], b5[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]= (j[0]-2); jj[1]= (j[1]+4);
jj[2]= (j[2]-2); jj[3]= (j[3]+4);

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);

for(i=4; i((j[3]+4); i++){
  uai_le_linha(ce,i,b5,jj);
  for(k=0; k(j[1]; k++){
    fac[0]= (8*b1[k]+5*b1[k+1]+2*b1[k+2]+5*b2[k]+2*b2[k+1]-b2[k+2]+2*b3[k]-b3[k+1]-4*b3[k+2])/18.;
    fac[1]= (5*(b1[k+1]+b1[k+2]+b1[k+3])+2*(b2[k+1]+b2[k+2]+b2[k+3])-b3[k+1]-b3[k+2]-b3[k+3])/18.;
    fac[2]= (2*b1[k+2]+5*b1[k+3]+8*b1[k+4]-b2[k+2]+2*(b2[k+3]+b3[k+4])+5*b2[k+4]-4*b3[k+2]-b3[k+3])/18.;
    fac[3]= (5*(b2[k]+b2[k]+b2[k])+2*(b2[k+1]+b3[k+1]+b4[k+1])-b2[k+2]-b3[k+2]-b4[k+2])/18.;
    fac[4]= (b2[k+1]+b3[k+1]+b4[k+1]+b2[k+2]+b3[k+2]+b4[k+2]+b2[k+3]+b3[k+3]+b4[k+3])/9.;
    fac[5]= (-b2[k+2]-b3[k+2]-b4[k+2]+2*(b2[k+3]+b3[k+3]+b4[k+3])+5*(b2[k+4]+b3[k+4]+b4[k+4]))/18.;
    fac[6]= (2*(b3[k]+b4[k+1]+b5[k+2])-b3[k+1]-4*b3[k+2]+5*b4[k]-b4[k+2]+8*b5[k]+5*b5[k+1])/18.;
    fac[7]= (-b3[k+1]-b3[k+2]-b3[k+3]+2*(b4[k+1]+b4[k+2]+b4[k+3])+5*(b5[k+1]+b5[k+2]+b5[k+3]))/18.;
    fac[8]= (2*(b3[k+4]+b4[k+3]+b5[k+2])-4*b3[k+2]-b3[k+3]-b4[k+2]+5*(b4[k+4]+b5[k+3])+8*b5[k+4])/18.;

    for(z=0; z(9; z++) ee[z]= 0.;
    for(t=k; t((k+3); t++){
      x1= (fac[0]-b1[t]); x2= (fac[0]-b2[t]); x3= (fac[0]-b3[t]);
      if (x1(0) x1= -x1;
      if (x2(0) x2= -x2;
      if (x3(0) x3= -x3;
      ee[0]= ee[0] + x1*x1 + x2*x2 + x3*x3;
```

```
  yi= (fac[3]-b2[t]); y2= (fac[3]-b3[t]); y3= (fac[3]-b4[t]);
  if (yi(0) yi= -yi;
  if (y2(0) y2= -y2;
  if (y3(0) y3= -y3;
  ee[3]= ee[3] + yi*yi + y2*y2 + y3*y3;

  wi= (fac[6]-b3[t]); w2= (fac[6]-b4[t]); w3= (fac[6]-b5[t]);
  if (wi(0) wi= -wi;
  if (w2(0) w2= -w2;
  if (w3(0) w3= -w3;
  ee[6]= ee[6] + wi*wi + w2*w2 + w3*w3;
}
for(t= (k+1); t((k+4); t++){
  xi= (fac[1]-bi[t]); x2= (fac[1]-b2[t]); x3= (fac[1]-b3[t]);
  if (xi(0) xi= -xi;
  if (x2(0) x2= -x2;
  if (x3(0) x3= -x3;
  ee[1]= ee[1] + xi*xi + x2*x2 + x3*x3;

  yi= (fac[4]-b2[t]); y2= (fac[4]-b3[t]); y3= (fac[4]-b4[t]);
  if (yi(0) yi= -yi;
  if (y2(0) y2= -y2;
  if (y3(0) y3= -y3;
  ee[4]= ee[4] + yi*yi + y2*y2 + y3*y3;

  wi= (fac[7]-b3[t]); w2= (fac[7]-b4[t]); w3= (fac[7]-b5[t]);
  if (wi(0) wi= -wi;
  if (w2(0) w2= -w2;
  if (w3(0) w3= -w3;
  ee[7]= ee[7] + wi*wi + w2*w2 + w3*w3;
}
or(t= (k+2); t( (k+5); t++){
  xi= (fac[2]-bi[t]); x2= (fac[2]-b2[t]); x3= (fac[2]-b3[t]);
  if (xi(0) xi= -xi;
  if (x2(0) x2= -x2;
  if (x3(0) x3= -x3;
  ee[2]= ee[2] + xi*xi + x2*x2 + x3*x3;

  yi= (fac[5]-b2[t]); y2= (fac[5]-b3[t]); y3= (fac[5]-b4[t]);
  if (yi(0) yi= -yi;
  if (y2(0) y2= -y2;
  if (y3(0) y3= -y3;
  ee[5]= ee[5] + yi*yi + y2*y2 + y3*y3;

  wi= (fac[8]-b3[t]); w2= (fac[8]-b4[t]); w3= (fac[8]-b5[t]);
  if (wi(0) wi= -wi;
  if (w2(0) w2= -w2;
  if (w3(0) w3= -w3;
  ee[8]= ee[8] + wi*wi + w2*w2 + w3*w3;
}
eemin = ee[0];
for(r= 0; r (9; r++){
  if(eemin ) ee[r]){
    eemin= ee[r];
```

```
            eeind = r;
          }
        }
        ks= fac[eeind]+0.5;
        if( ks(0 ) ks= 0;
        if( ks)255 ) ks= 255;
        bs[k]= ks;
      }
      is= i-4;
      uai_esc_linha(cs,is,bs,j);
      for(s= 0; s( (j[1]+4); s++){
        b1[s]= b2[s];
        b2[s]= b3[s];
        b3[s]= b4[s];
        b4[s]= b5[s];
      }
    }
    }
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
*ë   SUAVIZACAO  COM  VIZINHANCA  SELECIONADA  POR
**       MEDIA  DAS  DIFERENCAS  ABSOLUTAS
*/
int  is, jj[4], j[4];
int  z,i,k,b,t,r,s,xc,mdaind,ce,cs;
int smed[9],sda[9],mda[9],mdamin,med[9];
unsigned pc,t1,t2,t3,t4,t5;
unsigne cha bs[512]í b1[512], b2[512];
unsigned char b3[512], b4[512], b5[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= j[0]-2; jj[1]= j[1]+4;
jj[2]= j[2]-2; jj[3]= j[3]+4;  /* JAN ENTRADA */
```

```
uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);
for(k=0; k<jj[1]; k++) b4[k]= b4[k];

for(i=4; i<jj[3]; i++){ /* LOOP VARREDURA VERTICAL   */
  uai_le_linha(ce,i,b5,jj);    /* LE NOVA LINHA  */
  for(k=0; k<jj[1]; k++){      /* LOOP VARREDURA HORIZ  */
    for(z=0; z<9; z++){
        sda[z]= 0;
        smed[z]= 0;
        med[z]= 0;
    }
    pc= b3[k+2];
    for(t= k; t<(k+3); t++){
      t1=abs(pc-b1[t]); t2=abs(pc-b2[t]);
      t3=abs(pc-b3[t]); t4=abs(pc-b4[t]);
      t5=abs(pc-b5[t]);

      sda[0]= sda[0]+t1+t2+t3;
      smed[0]= smed[0]+b1[t]+b2[t]+b3[t];
      sda[1]= sda[1]+t2+t3+t4;
      smed[1]= smed[1]+b2[t]+b3[t]+b4[t];
      sda[2]= sda[2]+t3+t4+t5;
      smed[2]= smed[2]+b3[t]+b4[t]+b5[t];
    }
    for(t= (k+1); t<(k+4); t++){
      t1=abs(pc-b1[t]); t2=abs(pc-b2[t]);
      t3=abs(pc-b3[t]); t4=abs(pc-b4[t]);
      t5=abs(pc-b5[t]);

      sda[3]= sda[3]+t1+t2+t3;
      smed[3]= smed[3]+b1[t]+b2[t]+b3[t];
      sda[4]= sda[4]+t2+t3+t4;
      smed[4]= smed[4]+b2[t]+b3[t]+b4[t];
      sda[5]= sda[5]+t3+t4+t5;
      smed[5]= smed[5]+b3[t]+b4[t]+b5[t];
    }
    for(t= (k+2); t<(k+5); t++){
      t1=abs(pc-b1[t]); t2=abs(pc-b2[t]);
      t3=abs(pc-b3[t]); t4=abs(pc-b4[t]);
      t5=abs(pc-b5[t]);

      sda[6]= sda[6]+t1+t2+t3;
      smed[6]= smed[6]+b1[t]+b2[t]+b3[t];
      sda[7]= sda[7]+t2+t3+t4;
      smed[7]= smed[7]+b2[t]+b3[t]+b4[t];
      sda[8]= sda[8]+t3+t4+t5;
      smed[8]= smed[8]+b3[t]+b4[t]+b5[t];
    }
    for(z=0; z<9; z++) med[z]= smed[z]/9; /* CALCULA MEDIA */
      mda[0]= (sda[0]-abs(pc-b3[k])-abs(pc-b1[k+2]))/7.;
      med[0]= (smed[0]-b3[k]-b1[k+2])/7.;
```

```
      mda[1]= (sda[1]-abs(pc-bi[k+2])-abs(pc-b4[k+2]))/7.;
      med[1]= (smed[1]-bi[k+2]-b4[k+2])/7.;
      mda[2]= (sda[2]-abs(pc-b3[k])-abs(pc-b5[k+2]))/7.;
      med[2]= (smed[2]-b3[k]-b5[k+2])/7.;
      mda[3]= (sda[3]-abs(pc-b3[k+1])-abs(pc-b3[k+3]))/7.;
      med[3]= (smed[3]-b3[k+1]-b3[k+3])/7.;
      mda[4]= sda[4]/8.;
      med[4]= smed[4]/8.;
      mda[5]= (sda[5]-abs(pc-b3[k+1])-abs(pc-b3[k+3]))/7.;
      med[5]= (smed[5]-b3[k+1]-b3[k+3])/7.;
      mda[6]= (sda[6]-abs(pc-bi[k+2])-abs(pc-b3[k+4]))/7.;
      med[6]= (smed[6]-bi[k+2]-b3[k+4])/7.;
      mda[7]= (sda[7]-abs(pc-b2[k+2])-abs(pc-b4[k+2]))/7.;
      med[7]= (smed[7]-b2[k+2]-b4[k+2])/7.;
      mda[8]= (sda[8]-abs(pc-b3[k+4])-abs(pc-b5[k+2]))/7.;
      med[8]= (smed[8]-b3[k+4]-b5[k+2])/7.;

      mdamin= mda[0];  /* ENCONTRA MDA MINIMO */
      mdaind= 0;
      for(r=0; r<9; r++){
        if( mda[r] <= mdamin){
          mdamin= mda[r];
          mdaind = r;
        }
      }
      bs[k]= med[mdaind];
    }
    is= i-4;
    uai_esc_linha(cs,is,bs,j);
    for(s=0; s<jj[i]; s++){
      bi[s]= b2[s];
      b2[s]= b3[s];
      b3[s]= b4[s];
      b4[s]= b5[s];
    }
  }
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**      SUAVIZACAO  COM  VIZINHANCA  SELECIONADA  POR
**          SOMA  DAS  DIFERENCAS  ABSOLUTAS
**
*/
int  is, jj[4], j[4];
int  z,i,k,b,t,r,s,xc,sdaind,ce,cs;
int smed[9],sda[9],sdamin,med[9];
unsigned pc,t1,t2,t3,t4,t5;
unsigne char b1[512], b2[512], b3[512], b4[512], b5[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= j[0]-2; jj[1]= j[1]+4; jj[2]= j[2]-2; jj[3]= j[3]+4;

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);

for(i=4; i(jj[3]; i++){    /* LOOP VARREDURA VERTICAL   */
  uai_le_linha(ce,i,b5,jj);   /* LE NOVA LINHA    */
  for(k=0; k(j[1]; k++){       /* LOOP VARREDURA HORIZ  */
    for(z=0; z(9; z++){
      sda[z]= 0;
      smed[z]= 0;
      med[z]= 0;
    }
    pc= b3[k+2];     /* ABCISSA PIXEL CENTRAL     */
                     /* CALCULA SDA JANELAS 1,2,3 */
    for(t= k; t((k+3); t++){
      t1=abs(pc-b1[t]); t2=abs(pc-b2[t]);
      t3=abs(pc-b3[t]); t4=abs(pc-b4[t]);
      t5=abs(pc-b5[t]);

      sda[0]= sda[0]+t1+t2+t3;
      smed[0]= smed[0]+b1[t]+b2[t]+b3[t];
      sda[1]= sda[1]+t2+t3+t4;
      smed[1]= smed[1]+b2[t]+b3[t]+b4[t];
      sda[2]= sda[2]+t3+t4+t5;
      smed[2]= smed[2]+b3[t]+b4[t]+b5[t];
    }
```

```
                /* CALCULA SDA JANELAS 4,5,6 */
    for(t= (k+1); t<(k+4); t++){
      t1=abs(pc-b1[t]); t2=abs(pc-b2[t]);
      t3=abs(pc-b3[t]); t4=abs(pc-b4[t]);
      t5=abs(pc-b5[t]);

      sda[3]= sda[3]+t1+t2+t3;
      smed[3]= smed[3]+b1[t]+b2[t]+b3[t];
      sda[4]= sda[4]+t2+t3+t4;
      smed[4]= smed[4]+b2[t]+b3[t]+b4[t];
      sda[5]= sda[5]+t3+t4+t5;
      smed[5]= smed[5]+b3[t]+b4[t]+b5[t];
    }
                /* CALCULA SDA JANELAS 7,8,9 */
    for(t= (k+2); t<(k+5); t++){
      t1=abs(pc-b1[t]); t2=abs(pc-b2[t]);
      t3=abs(pc-b3[t]); t4=abs(pc-b4[t]);
      t5=abs(pc-b5[t]);

      sda[6]= sda[6]+t1+t2+t3;
      smed[6]= smed[6]+b1[t]+b2[t]+b3[t];
      sda[7]= sda[7]+t2+t3+t4;
      smed[7]= smed[7]+b2[t]+b3[t]+b4[t];
      sda[8]= sda[8]+t3+t4+t5;
      smed[8]= smed[8]+b3[t]+b4[t]+b5[t];
    }
                    /* CALCULA MEDIA */
    for(z=0; z<9; z++) med[z]= smed[z]/9;
      sdamin= sda[0];   /* ENCONTRA SDA MINIMO */
      sdaind= 0;
      for(r=0; r<9; r++){
        if( sda[r] <= sdamin){
          sdamin= sda[r];
          sdaind = r;
        }
      }
      bs[k]= med[sdaind];
    }
    is= i-4;
    uai_esc_linha(cs,is,bs,j);
            /* DESLOCA CONTEUDO BUFFERS */
    for(s=0; s<jj[i]; s++){
      b1[s]= b2[s];
      b2[s]= b3[s];
      b3[s]= b4[s];
      b4[s]= b5[s];
    }
  }
}
```

```
#include (dia.h)
#include (uai.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**      FILTRO  DA  ORDEM
**
**
*/

int i,ce,cs,ta,ordem,j[4];
limpahelp();
msg();
pegdad1();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

if ( ta == 3 ){
  int   k,n,m,q,is,x,y,t,z,w,c,a,min,indmin,indaux;
  int   jj[4];
  unsigned int vord[9], vet[9];
  char  b1[512], b2[512], b3[512], bs[512];

  jj[0]= (j[0]-1); jj[1]= (j[1]+2);
  jj[2]= (j[2]-1); jj[3]= (j[3]+2);

  min= 0;
  indmin= 0;
  uai_le_linha(ce,0,b1,jj);
  uai_le_linha(ce,1,b2,jj);
  for(i= 2; i( (j[3]+2); i++){
    uai_le_linha(ce,i,b3,jj);
    for(k= 0; k(j[1]; k++){
      for(t=0; t(9; t++){
        vet[t]= 0;
        vord[t]= 0;
      }
      vet[0]= b1[k];
      vet[1]= b1[k+1];
      vet[2]= b1[k+2];
      vet[3]= b2[k];
      vet[4]= b2[k+1];
      vet[5]= b2[k+2];
      vet[6]= b3[k];
      vet[7]= b3[k+1];
      vet[8]= b3[k+2];
      bubble(vet,0);

      bs[k]= vet[ordem-1];
    }
    is= i-2;
```

```
    uai_esc_linha(cs,is,bs,j);
    for(a=0; a(jj[i]; a++){
      bi[a]= b2[a];
      b2[a]= b3[a];
    }
  }
}
if ( ta == 5 ){
  int   k,q,is,t,z,w,c,a,min,indmin,indaux;
  int   jj[4];
  unsigned int vord[25], vet[25];
  unsigne char  bi[5i2], b2[5i2], b3[5i2];
  indigne char b4[5i2], b5[5i2], bs[5i2];

  jj[0]= (j[0]-2); jj[i]= (j[i]+4);
  jj[2]= (j[2]-2); jj[3]= (j[3]+4);

  min= 0;
  indmin= 0;
  uai_le_linha(ce,0,bi,jj);
  uai_le_linha(ce,i,b2,jj);
  uai_le_linha(ce,2,b3,jj);
  uai_le_linha(ce,3,b4,jj);

  for(i= 4; i(jj[3]; i++){
    uai_le_linha(ce,i,b5,jj);
    for(k= 0; k(j[i]; k++){
      for(t=0; t(25; t++ê vet[t]= 0;
      vet[0]= bi[k],
      vet[i]= bi[k+i];
      vet[2]= bi[k+2];
      vet[3]= bi[k+3];
      vet[4]= bi[k+4];
      vet[5]= b2[k];
      vet[6]= b2[k+i];
      vet[7]= b2[k+2];
      vet[8]= b2[k+3];
      vet[9]= b2[k+4];
      vet[i0]= b3[k];
      vet[ii]= b3[k+i];
      vet[i2]= b3[k+2];
      vet[i3]= b3[k+3];
      vet[i4]= b3[k+4];
      vet[i5]= b4[k];
      vet[i6]= b4[k+i];
      vet[i7]= b4[k+2];
      vet[i8]= b4[k+3];
      vet[i9]= b4[k+4];
      vet[20]= b5[k];
      vet[2i]= b5[k+i];
      vet[22]= b5[k+2];
      vet[23]= b5[k+3];
      vet[24]= b5[k+4];

      bubble(vet,25);
```

```
      bs[k]= vet[ordem-i];
    }
    is= i-4;
    uai_esc_linha(cs,is,bs,j);
    for(a=0; a(jj[i]; a++){
      bi[a]= b2[a];
      b2[a]= b3[a];
      b3[a]= b4[a];
      b4[a]= b5[a];
    }
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (math.h)
#include (stdio.h)

main()
{
/*
**
**  SUAVIZACAO  COÎ VIZINHANCA SELECIONADA
*Ë              POR VARIANCIA
**
*/
double med[9],medq[9],v[9],vmin;
int   i,k,j[4],jj[4],is,t,w,p,z,r,ce,cs,vind;
unsigned char BS[512], bi[512], b2[512];
unsigne cha b3[512], b4[512], b5[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]= (j[0]-2); jj[1]= (j[1]+4);
jj[2]= (j[2]-2); jj[3]= (j[3]+4);

uai_le_linha(ce,0,bi,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);

for(i= 4; i((j[3]+4); i++){
  uai_le_linha(ce,i,b5,jj);
  for(k= 0; k(j[1]; k++){
    for(z=0; z49; z++){
      v[z]= 0.;
```

```
  med[z]= 0.;
  medq[z]= 0.;
}
med[0]= (float)(bi[k]+bi[k+1]+b2[k]+b2[k+1]+b2[k+2]+b3[k+1]+b3[k+2])/7.;
medq[0]= (bi[k]*bi[k])+(bi[k+1]*bi[k+1])+(b2[k]*b2[k]);
medq[0]= medq[0]+(b2[k+1]*b2[k+1])+(b2[k+2]*b2[k+2])+(b3[k+1]*b3[k+1]);
medq[0]= (float)(medq[0] + (b3[k+2]*b3[k+2]))/7.;

med[1]= (float)(bi[k+1]+bi[k+2]+bi[k+3]+b2[k+1]+b2[k+2]+b2[k+3]+b3[k+2])/7.;
medq[1]= (bi[k+1]*bi[k+1])+(bi[k+2]*bi[k+2])+(bi[k+3]*bi[k+3]);
medq[1]= medq[1]+(b2[k+1]*b2[k+1])+(b2[k+2]*b2[k+2])+(b2[k+3]*b2[k+3]);
medq[1]= (float)(medq[1] + (b3[k+2]*b3[k+2]))/7.;

med[2]= (float)(bi[k+3]+bi[k+4]+b2[k+2]+b2[k+3]+b2[k+4]+b3[k+2]+b3[k+3])/7.;
medq[2]= (bi[k+3]*bi[k+3])+(bi[k+4]*bi[k+4])+(b2[k+2]*b2[k+2]);
medq[2]= medq[2]+(b2[k+3]*b2[k+3])+(b2[k+4]*b2[k+4])+(b3[k+2]*b3[k+2]);
medq[2]= (float)(medq[2] + (b3[k+3]*b3[k+3]))/7.;

med[3]= (float)(b2[k]+b2[k+1]+b3[k]+b3[k+1]+b3[k+2]+b4[k]+b4[k+1])/7.;
medq[3]= (b2[k]*b3[k])+(b2[k+1]*b2[k+1])+(b3[k]*b3[k]);
medq[3]= medq[3]+(b3[k+1]*b3[k+1])+(b3[k+2]*b3[k+2])+(b4[k]*b4[k]);
medq[3]= (float)(medq[3] + (b4[k+1]*b4[k+1]))/7.;

med[4]= b2[k+1]+b2[k+2]+b2[k+3]+b3[k+1]+b3[k+2]+b3[k+3]+b4[k+1];
med[4]= (float)(med[4]+b4[k+2]+b4[k+3])/9.;
medq[4]= (b2[k+1]*b2[k+1])+(b2[k+2]*b2[k+2])+(b2[k+3]*b2[k+3]);
medq[4]= medq[4]+(b3[k+1]*b3[k+1])+(b3[k+2]*b3[k+2])+(b3[k+3]*b3[k+3]);
medq[4]= medq[4]+(b4[k+1]*b4[k+1])+(b4[k+2]*b4[k+2]);
medq[4]= (float)(medq[4] + (b4[k+3]*b4[k+3]))/9.;

med[5]= (float)(b2[k+3]+b2[k+4]+b3[k+2]+b3[k+3]+b3[k+4]+b4[k+3]+b4[k+4])/7.;
medq[5]= (b2[k+3]*b2[k+3])+(b2[k+4]*b2[k+4])+(b3[k+2]*b3[k+2]);
medq[5]= medq[5]+(b3[k+3]*b3[k+3])+(b3[k+4]*b3[k+4])+(b4[k+3]*b4[k+3]);
medq[5]= (float)(medq[5] + (b4[k+4]*b4[k+4]) )/7.
med[6]= (float)(b3[k+1]+b3[k+2]+b4[k]+b4[k+1]+b4[k+2]+b5[k]+b5[k+1])/7.;
medq[6]= (b3[k+1]*b3[k+1])+(b3[k+2]*b3[k+2])+(b4[k]*b4[k]);
medq[6]= medq[6]+(b4[k+1]*b4[k+1])+(b4[k+2]*b4[k+2])+(b5[k]*b5[k]);
medq[6]= (float)(medq[6] + (b5[k+1]*b5[k+1]) )/7.;

med[7]= (float)(b3[k+2]+b4[k+1]+b4[k+2]+b4[k+3]+b5[k+1]+b5[k+2]+b5[k+3])/7.;
medq[7]= (b3[k+2]*b3[k+2])+(b4[k+1]*b4[k+1])+(b4[k+2]*b4[k+2]);
medq[7]= medq[7]+(b4[k+3]*b4[k+3])+(b5[k+1]*b5[k+1])+(b5[k+2]*b5[k+2]);
medq[7]= (float)(medq[7] + (b5[k+3]*b5[k+3]) )/7.;

med[8]= (float)(b3[k+2]+b3[k+3]+b4[k+2]+b4[k+3]+b4[k+4]+b5[k+3]+b5[k+4])/7.;
medq[8]= (b3[k+2]*b3[k+2])+(b3[k+3]*b3[k+3])+(b4[k+2]*b4[k+2]);
medq[8]= medq[8]+(b4[k+3]*b4[k+3])+(b4[k+4]*b4[k+4])+(b5[k+3]*b5[k+3]);
medq[8]= (float)(medq[8] + (b5[k+4]*b5[k+4]) )/7.;

for(w=0; w<9; w++) {
  v[w]= medq[w] - (med[w]*med[w]);
  if(v[w] < 0 ) v[w]= -v[w];
}
vmin= v[0];
vind= 0;
```

```
    for(t=0; t<9; t++){
      if(v[t] <= vmin){
        vmin= v[t];
        vind= t;
      }
    }
    bs[k]= med[vind]+0.5;
  }
  is= i-4;
  uai_esc_linha(cs,is,bs,j);
  for(r= 0; r< (j[i]+4); r++){
    b1[r]= b2[r];
    b2[r]= b3[r];
    b3[r]= b4[r];
    b4[r]= b5[r];
  }
}
}
```

```
#include <stdio.h>
#include <uai.h>
#include <math.h>
#include <dia.h>
main()
{
/*
**
**              FILTRO  SIGMA  NAO  POLARIZADO
**
*/
float delta,desvio,p,pq,dpq,dp,vv;
int   som,medp,i,k,ce,cs,u,z,r,t,op,s,is;
int   j[4], jj[4], jd[4], dif1, dif2, dif3, dif4, dif5;
unsigne char  c, b1[512], b2[512], b3[512];
unsigne char b4[512], b5[512], bs[512], bdv[512];


limpahelp();
msg();
pegdadsig();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= (j[0]-2); jj[1]= (j[1]+4);
jj[2]= (j[2]-2); jj[3]= (j[3]+4);

delta= 2.* desvio;
uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);
```

```
for(i=4; i(jj[3]; i++){
  uai_le_linha(ce,i,b5,jj);
    for(k=0; k(j[i]; k++){
      r= 0;  som= 0;
      for(t=k; t((k+5); t++){  /* TESTA 2SIG SUP E */
                          INF E SALVA EM VETORES  */
        dif1= (b3[k+2] - bi[t]);
        if ( dif1 ( 0 ) dif1= -dif1;
        if( dif1 ( delta ){
          som= som + bi[t];
          r++;
        }
        dif2= (b3[k+2] - b2[t]);
        if ( dif2 ( 0 ) dif2= -dif2;
        if( dif2 (delta ){
          som= som + b2[t];
          r++;
        }
        dif3= (b3[k+2] - b3[t]);
        if ( dif3 ( 0 ) dif3= -dif3;
        if( dif3 (delta ){
         som= som + bi[t];
          r++;
        }
        dif4= (b3[k+2] - b4[t]);
        if ( dif4 ( 0 ) dif4= -dif4;
        if( dif4 (delta ){
          som= som + bi[t];
          r++;
        }
        dif5= (b3[k+2] - b5[t]);
        if ( dif5 ( 0 ) dif5= -dif5;
        if( dif5 (delta ){
          som= som + bi[t];
          r++;
        }
      }
      if( r ) 2 ê medp= som/r;
      else{
        medp= b2[k+1]+b2[k+2]+b2[k+3]+b3[k+1]+b3[k+2];
        medp' (medp+b3[k+3]+b4[k+1]+b4[k+2]+b4[k+3])/9.+0.5;
      }
      bs[k]= medp + 0.5;
    }
    is= i-4;
  uai_esc_linha(cs,is,bs,j);
  for(s=0; s(jj[1]; s++){
    bi[s]= b2[s];
    b2[s]= b3[s];
    b3[s]= b4[s];
    b4[s]= b5[s];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**          AGUCAMENTO  EXTREMO   P/MASCARAS  3x3

**
*/
int   i,k,z,r,smin,smax,is,t,dif1,dif2,ce,cs;
int   jj[4], b[9], j[4];
unsigne char  bi[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= j[0]-1; jj[1]= j[1]+2; jj[2]= j[2]-1; jj[3]= j[3]+2;

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i(jj[3]; i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    b[0]= bi[k]; b[1]= bi[k+1]; b[2]= bi[k+2];
    b[3]= b2[k]; b[4]= b2[k+1]; b[5]= b2[k+2];
    b[6]= b3[k]; b[7]= b3[k+1]; b[8]= b3[k+2];
    smax= b[0];                /* ENCONTRA EXTREMOS        */
    smin= b[0];
    for(r=0; r(9; r++){
      if(b[r] ) smax) smax= b[r];
      if(b[r] ( smin) smin= b[r];
    }
    dif1= abs(b2[k+1]-smin);
    dif2= abs(b2[k+1]-smax);
    if( dif1 ( dif2 ) bs[k]= smin;
    if( dif1 ) dif2 ) bs[k]= smax;
    else bs[k]= b2[k+1];
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t(jj[1]; t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#includ (uai.h)
#includ (dia.h)
#includ (stdio.h)
#includ (math.h)
main()
{
/*
**
**      SUAVIZACAO LOGARITMICA
**
*/
float f[8];
int   tmed,med[8],medq[8],v[8],vmin;
int   i,k,fatm,w,t,p,z,r,ce,cs;
int   j[4], jj[4];
unsigne char  bi[512], b2[512], b3[512];
unsigne cha  b4[512], b5[512], bs[512];

limpahelp();
msg();
pegdadlog();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= (j[0]-2); jj[1]= (j[1]+4);
jj[2]= (j[2]-2); jj[3]= (j[3]+4);

uai_le_linha(ce,0,bi,jj);  /* LE LINHAS INICIAIS  */
uai_le_linha(ce,1,b2,jj);
uai_le_linha(ce,2,b3,jj);
uai_le_linha(ce,3,b4,jj);
for(i=4; i((j[3]+4); i++){ /* LOOP VARREDURA VERT */
  uai_le_linha(ce,i,b5,jj);
  for(k=0; k(j[1]; k++){ /* LOOP VARREDURA HORIZ    */
                      /* CALC E[X] E E[X] QUAD   */
    med[0]= (bi[k]+bi[k+1]+bi[k+2]+b2[k+1]+b2[k+2]+b3[k+3])/6. + 0.5;
    medq[0]= (bi[k]*bi[k])+(bi[k+1]*bi[k+1])+(bi[k+2]*bi[k+2])+(b2[k+1]*b2[k+1]);
    medq[0]= (medq[0] + (b2[k+2]*b2[k+2])+(b3[k+3]*b3[k+3]))/6. + 0.5;
    med[1]= (bi[k+2]+bi[k+3]+bi[k+4]+b2[k+2]+b2[k+3]+b3[k+2])/6. + 0.5;
    medq[1]= (bi[k+2]*bi[k+2])+(bi[k+3]*bi[k+3])+(bi[k+4]*bi[k+4])+(b2[k+2]*b2[k+2]);
    medq[1]= (medq[1] + (b2[k+3]*b2[k+3])+(b3[k+2]*b3[k+2]))/6. + 0.5;
    med[2]= (bi[k+4]+b2[k+3]+b2[k+4]+b3[k+2]+b3[k+3]+b3[k+4])/6. + 0.5;
    medq[2]= (bi[k+4]*bi[k+4])+(b2[k+3]*b2[k+3])+(b2[k+4]*b2[k+4])+(b3[k+2]*b3[k+2]);
    medq[2]= (medq[2] + (b3[k+3]*b3[k+3])+(b3[k+4]*b3[k+4]))/6. + 0.5;
    med[3]= (b3[k+2]+b3[k+3]+b3[k+4]+b4[k+3]+b4[k+4]+b5[k+4])/6. + 0.5;
    medq[3]= (b3[k+2]*b3[k+2])+(b3[k+3]*b3[k+3])+(b3[k+4]*b3[k+4])+(b4[k+3]*b4[k+3]);
    medq[3]= (medq[3] + (b4[k+4]*b4[k+4])+(b5[k+4]*b5[k+4]))/6. + 0.5;
    med[4]= (b3[k+2]+b4[k+2]+b4[k+3]+b5[k+2]+b5[k+3]+b5[k+4])/6. + 0.5;
    medq[4]= (b3[k+2]*b3[k+2])+(b4[k+2]*b4[k+2])+(b4[k+3]*b4[k+3])+(b5[k+2]*b4[k+2]);
    medq[4]= (medq[4] + (b5[k+3]*b5[k+3])+(b5[k+4]*b5[k+4]))/6. + 0.5;
    med[5]= (b3[k+2]+b4[k+1]+b4[k+2]+b5[k]+b5[k+1]+b5[k+2])/6. + 0.5;
    medq[5]= (b3[k+2]*b3[k+2])+(b4[k+1]*b4[k+1])+(b4[k+2]*b4[k+2])+(b5[k]*b5[k]);
    medq[5]= (medq[5] + (b5[k+1]*b5[k+1])+(b5[k+2]*b5[k+2]))/6. + 0.5;
    med[6]= (b3[k]+b3[k+1]+b3[k+2]+b4[k]+b4[k+1]+b5[k])/6. + 0.5;
```

```
    medq[6]= (b3[k]*b3[k])+(b3[k+1]*b3[k+1])+(b3[k+2]*b3[k+2])+(b4[k]*b4[k]);
    medq[6]= (medq[6] + (b4[k+1]*b4[k+1])+(b5[k]*b5[k]))/6. + 0.5;
    med[7]= (bi[k]+b2[k]+b2[k+1]+b3[k]+b3[k+1]+b3[k+2])/6. + 0.5;
    medq[7]= (bi[k]*bi[k])+(b2[k]*b2[k])+(b2[k+1]*b2[k+1])+(b3[k]*b3[k]);
    medq[7]= (medq[7] + (b3[k+1]*b3[k+1])+(b3[k+2]*b3[k+2]))/6. + 0.5;
                              /Ë CALCULñ Dä V(xê  */
    for(w= 0; w( 8; w++)   v[w]= abs(medq[w] - (med[w]*med[w]));
    vmin= v[0];
    for(t= 0; t( 8; t++){  /* MINIMA VARIANCIA */
      if(vmin ) v[t])   vmin= v[t];
    }
    tmed= 0;
    for(p= 0; p( 8; p++){  /* CALC PONDERADO PIXEL  */
      f[p]= vmin/v[p];
      tmed= tmed + ( med[p] * pow((double)f[p],(double)fatm) ) + 0.5;
    }
    if( tmed ) 255 ) tmed= 255;
    bs[k]= tmed;
  }
  z= i-4;
  uai_esc_linha(cs,z,bs,j);  /* ESC LINHA PROCESSADA     */
  for(r=0; r((j[1]+4); r++){   /* DESLOCA CONTEUDO BUFFERS */
    bi[r]= b2[r];
    b2[r]= b3[r];
    b3[r]= b4[r];
    b4[r]= b5[r];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**          REALCADOR  DE  LINHAS
**
*/
int   i,k,is,t,med,ce,cs;
int   jj[4],j[4],dif;
unsigne char  bi[5i2],b2[5i2],b3[5i2],bs[5i2];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
```

```
jj[0]= j[0]-1; jj[1]= j[1]+2; jj[2]= j[2]-1; jj[3]= j[3]+2;

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i(jj[3]; i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    med= b1[k]+b1[k+1]+b1[k+2]+b2[k]+b2[k+2];
    med= (med+b2[k+1]+b3[k]+b3[k+1]+b3[k+2])/9.;
    dif= b2[k+1]-med;
    bs[k]= b2[k+1]+dif;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t(jj[1]; t++){
    b1[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**  OPERADOR  DIFERENCIAL  DE SOBEL
**
*/
int   i,k,is,t,gx,gy,b,sob,sb,ce,cs;
int   jj[4], j[4];
unsigne char  b1[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]=(j[0]-1); jj[1]=(j[1]+2);
jj[2]=(j[2]-1); jj[3]=(j[3]+2);

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i((j[3]+2); i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    gx= b1[k]+2*b1[k+1]+b1[k+2];
    gx= abs(gx - (b3[k]+2*b3[k+1]+b3[k+2]));
    gy= b1[k]+2*b2[k]+b3[k];
```

```
    gy= abs(gy - (bi[k+2]+2*b2[k+2]+b3[k+2]));
    sb= abs(gx*gx) + abs(gy*gy);
    sob= sqrt( (double) sb );
    bs[k]=  sob ;
    if( bs[k] ) 255 ) bs[k]= 255;
    if( bs[k] ( 0 ) bs[k]= 0;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t(j[i]+2; t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**          OPERADOR  DIRECIONAL  DE  ROBINSON
**          ( MASC SIMPLES DE 3 NIVEIS  )
**
*/
int   z,i,k,is,t,n,m,x,y,ce,cs;
int   jj[4], j[4], g[8],gmax;
unsigne char  bi[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]= (j[0]-1); jj[1]= (j[1]+2);
jj[2]= (j[2]-1); jj[3]= (j[3]+2);

uai_le_linha(ce,0,bi,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i(j[3]+2; i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){ /ë GRADIENTEõ DIô CARDEAIõ  */
    g[0]= abs(bi[k]+bi[k+1]+bi[k+2]-b3[k]-b3[k+1]-b3[k+2]);
    g[1]= abs(bi[k]+bi[k+1]+b2[k]-b2[k+2]-b3[k+1]-b3[k+2]);
    g[2]= abs(bi[k]-bi[k+2]+b2[k]-b2[k+2]+b3[k]-b3[k+2]);
    g[3]= abs(-bi[k+1]-bi[k+2]+b2[k]-b2[k+2]+b3[k]+b3[k+1]);
    g[4]= abs(-bi[k]-bi[k+1]-bi[k+2]+b3[k]+b3[k+1]+b3[k+2]);
    g[5]= abs(-bi[k]-bi[k+1]-b2[k]+b2[k+2]+b3[k+1]+b3[k+2]);
    g[6]= abs(-bi[k]+bi[k+2]-b2[k]+b2[k+2]-b3[k]+b3[k+2]);
```

```
    g[7]= abs(bi[k+1]+bi[k+2]-b2[k]+b2[k+2]-b3[k]-b3[k+1]);
    for(z=0; z(8; z++){
      if(g[z] ) 255) g[z]= 255;
    }
    gmax= g[0];
    for(t=0; t(8; t++){  /* MAXIMO GRADIENTE */
      if(g[t] ) gmax)   gmax= g[t];
    }
    bs[k]= gmax;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t((j[i]+2); t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)

main()
{
/*
**
**          OPERADOR  DIRECIONAL  DE  KIRSCH
**
*/
int   z,i,k,is,t,n,m,x,y,ce,cs;
int   jj[4], j[4], g[8],gmax;
unsigne char  bi[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= (j[0]-1); jj[1]= (j[1]+2);
jj[2]= (j[2]-1); jj[3]= (j[3]+2);

uai_le_linha(ce,0,bi,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i(j[3]+2; i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    g[0]= 5*(bi[k]+bi[k+1]+bi[k+2])-3*(b2[k]+b2[k+2]+b3[k]+b3[k+1]+b3[k+2]);
    g[1]= 5*(bi[k]+bi[k+1]+b2[k])-3*(bi[k+2]+b2[k+2]+b3[k]+b3[k+1]+b3[k+2]);
```

```
    g[2]= 5*(bi[k]+b2[k]+b3[k])-3*(bi[k+1]+bi[k+2]+b2[k+2]+b3[k+1]+b3[k+2]);
    g[3]= 5*(b2[k]+b3[k]+b3[k+1])-3*(bi[k]+bi[k+1]+bi[k+2]+b2[k+2]+b3[k+2]);
    g[4]= 5*(b3[k]+b3[k+1]+b3[k+2])-3*(bi[k]+bi[k+1]+bi[k+2]+b2[k]+b2[k+2]);
    g[5]= 5*(b2[k+2]+b3[k+1]+b3[k+2])-3*(bi[k]+bi[k+1]+bi[k+2]+b2[k]+b3[k]);
    g[6]= 5*(bi[k+2]+b2[k+2]+b3[k+2])-3*(bi[k]+bi[k+1]+b2[k]+b3[k]+b3[k+1]);
    g[7]= 5*(bi[k+1]+bi[k+2]+b2[k+2])-3*(bi[k]+b2[k]+b3[k]+b3[k+1]+b3[k+2]);
    for(t=0; t<8; t++){
      if( g[t] < 0 ) g[t]= -g[t];
    }
    gmax= g[0];
    for(t=0; t<8; t++){
      if(g[t] > gmaxÊ  gmax= g[t];
    }
    if(gmax > 255) gmax= 255;
    bs[k]= gmax;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t<(j[i]+2); t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)

main()
{
/*
**
**          DETETOR DE BORDAS HIBRIDO
**
*/
float v1,v2,guardv,edgev,gv;
float h1,h2,guardh,edgeh,gh;
int   i,k,is,t,ce,cs,ev,gv,eh,gh,z;
int   jj[4], j[4];
unsigne char  bi[512], b2[512], b3[512], bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);

jj[0]= (j[0]-1); jj[1]= (j[1]+2);
jj[2]= (j[2]-1); jj[3]= (j[3]+2);

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i(j[3]+2; i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    guardv= 0.; edgev= 0.;
    v1= (bi[k]+bi[k+1]+b2[k]+b2[k+1]+b3[k]+b3[k+1])/2.;
    v2' (bi[k+2]+b2[k+2]+b3[k+2]);
    guard ' (v1-v2)/3.;
    gv= guardv + 0.5;
    edgev= (bi[k]+b2[k]+b3[k]-bi[k+2]-b2[k+2]-b3[k+2])/3.;
    ev= edgev + 0.5;
    if ( gv)0. && ev)0. ){
      if( gv )= ev ) gv=  ev;
      else gv= gv;
    }
    else if ( gv(0. && ev(0.) {
      gv= -gv;   ev= -ev;
      if( gv )= ev ) gv=  ev;
      else gv= gv;
    }
    else {
      gv= 0;
    }
    guardh= 0.; edgeh= 0.;
```

```
    hi= (bi[k]+bi[k+1]+bi[k+2]+b2[k]+b2[k+1]+b2[k+2])/2.;
    hô ' b3[k]+b3[k+1]+b3[k+2];
    guard ' (vi-v2)/3.;
    gv= guardh + 0.5;
    edgeh= (bi[k]+bi[k+1]+bi[k+2]-b3[k]-b3[k+1]-b3[k+2])/3.;
    eh= edgeh + 0.5;
    if ( gh)0. && eh)0. ){
      if( gh )= eh ) yh=  eh;
      else yh= gh;
    }
    else if ( gh(0. && eh(0.) {
      gh= -gh;   eh= -eh;
      if( gh )= eh ) yh=  eh;
      else yh= gh;
    }
    else {
      yh= 0;
    }
    bs[k]= yh + yv + 0.5;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t((j[i]+2); t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
 }
 }
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**                                      0 -1  0
**   OPERADOR  DIFERENCIAL  LAPLACIANO  -1  4 -1
**                                      0 -1  0
*/
int   i,k,is,t,b,ce,cs,laplac;
int   jj[4],j[4];
char  bi[512],b2[512],b3[512],bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
```

```
jj[0]=(j[0]-1); jj[1]=(j[1]+2);
jj[2]=(j[2]-1); jj[3]=(j[3]+2);

uai_le_linha(ce,0,b1,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i((j[3]+2); i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    laplac= -b1[k+1]-b2[k]+4*b2[k+1];
    laplac= laplac-b2[k+2]-b3[k+1];
    if( laplac ( 0 ) laplac= -laplac;
    if( laplac ) 255 ) laplac= 255;
    bs[k]=  laplac;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t(jj[1]; t++){
    b1[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
**
**   OPERADOR  DIF  LAPLACIANO   -1-1-1
**                               -1-8-1
**                               -1-1-1
**
*/
int   i,k,is,t,b,ce,cs,laplac;
int   jj[4],j[4];
unsigne char  b1[512],b2[512],b3[512],bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]=(j[0]-1); jj[1]=(j[1]+2);
jj[2]=(j[2]-1); jj[3]=(j[3]+2);

uai_le_linha(ce,0,b1,jj);
```

```
uai_le_linha(ce,1,b2,jj);
for(i=2; i((j[3]+2); i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    laplac= -bi[k]-bi[k+1]-bi[k+2]-b2[k];
    laplac= laplac-b2[k+2]-b3[k]-b3[k+1]-b3[k+2];
    laplac= laplac + 8*b2[k+1];
    if( laplac ( 0 ) laplac= -laplac;
    if( laplac ) 255 ) laplac= 255;
    bs[k]=  laplac;
  }
  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t(j[1]+2; t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
}
}
```

```
#include (uai.h)
#include (dia.h)
#include (stdio.h)
#include (math.h)
main()
{
/*
** OPERADOR  DIFERENCIAL  LAPLACIANO    1 -2  1
**                                      2  4 -2
**                                      1 -2  1
*/
int   i,k,is,t,b,ce,cs,laplac;
int   jj[4],j[4];
char  bi[512],b2[512],b3[512],bs[512];

limpahelp();
msg();
pegdados();

j[0]=200; j[1]=128; j[2]=200; j[3]=128;
uai_cursor(ce,j);
jj[0]=(j[0]-1); jj[1]=(j[1]+2);
jj[2]=(j[2]-1); jj[3]=(j[3]+2);

uai_le_linha(ce,0,bi,jj);
uai_le_linha(ce,1,b2,jj);
for(i=2; i((j[3]+2); i++){
  uai_le_linha(ce,i,b3,jj);
  for(k=0; k(j[1]; k++){
    laplac= bi[k]-2*bi[k+1]+bi[k+2]-2*b2[k];
```

```
    laplac= laplac-2*b2[k+2]+b3[k]-2*b3[k+1]+b3[k+2];
    laplac= laplac + 4*b2[k+1];
    if ( laplac < 0 ) laplac= -laplac;
    if ( laplac > 255 ) laplac= 255;
    bs[k]=  laplac;

  is= i-2;
  uai_esc_linha(cs,is,bs,j);
  for(t=0; t<(j[1]+2); t++){
    bi[t]= b2[t];
    b2[t]= b3[t];
  }
}
}



#include <stdio.h>
#include <uai.h>
#include <dia.h>
main(argv)
{
        int i,l,ce;
        int j[4];
        char buffer[512],nome[80];
        unsigned fp;


limpahelp();
msg();
for(i=12; i<21; i++){
        dia_poe_texto(i,37,"                                          ",0,1);
}
        dia_poe_texto(13,39," Selecione arquivo imagem ",3,0);
        dia_poe_texto(14,39," --) ",3,0);
        argv=nome;
        getstr(nome,80);
        if( nome[0] == 'v' ) goto fim;
        dia_pega_int(15,39,"  CANAIS :,  0  1  ou 2, ","," Selecione canal  --) ",&ce);
        if( ce == 'v' ) goto fim;
        dia_poe_texto(18,43,"  Selecione janela na UVI  ",0,3);
        dia_poe_texto(19,43,"     atraves do cursor     ",0,3);

        j[0]=200; j[1]=128; j[2]=200; j[3]=128;
        uai_cursor(ce,j);

        fp=fopen( argv ,"rqv");
        for (l=0; l<(j[3]); l++ ) {

                fget(buffer,j[1],fp);
                uai_esc_linha(ce,l,buffer,j);

        zero (buffer,512);
        }
        fclose(fp);
fim:
}
```

```
#include (stdio.h)
#include (uai.h)
#include (dia.h)
main(argv)
{
        int i,l,ce;
        int j[4];
        char buffer[512],nome[80];
        unsigned fp;


limpahelp();
msg();
for(i=12; i(21; i++){
        dia_poe_texto(i,37,"                                        ",0,1);
}
        dia_poe_texto(13,39," Selecione nome para imagem ",3,0);
        dia_poe_texto(14,39," --) ",3,0);
        argv=nome;
        getstr(nome,80);
        if( nome[0] == 'v' ) goto fim;
        dia_pega_int(15,39,"  CANAIS :,  0  1  ou 2, ","," Selecione canal  --) ",&ce);
        if( ce == 'v' ) goto fim;
        dia_poe_texto(18,43,"  Selecione janela na UVI  ",0,3);
        dia_poe_texto(19,43,"     atraves do cursor     ",0,3);

        j[0]=200; j[1]=128; j[2]=200; j[3]=128;
        uai_cursor(ce,j);

        fp=fopen( argv ,"wqv");
        for (l=0; l(j[3]; l++ ) {

                uai_le_linha(ce,l,buffer,j);
                fput(buffer,j[1],fp);

        zero (buffer,512);
        }
        fclose(fp);
fim:
}
```

```
for(i=12; i(21; i++){
        dia_poe_texto(i,37,"                                        ",0,1);
}
dia_pega_int(13,39,"  CANAIS :,  0  1  ou 2, ","," Canal entrada  --) ",&ce);
dia_pega_int(14,39,"  CANAIS :,  0  1  ou 2, ","," Canal saida  --) ",&cs);
dia_poe_texto(18,43,"  Selecione janela na UVI  ",0,3);
dia_poe_texto(19,43,"     atraves do cursor     ",0,3);
```

```
for(i=12; i(21; i++){
      dia_poe_texto(i,37,"                                          ",0,1);
}
dia_pega_int(13,39,"  CANAIS :,  0  1  ou 2, ","," Canal entrada  --) ",&ce);
dia_pega_int(14,39,"  CANAIS :,  0  1  ou 2, ","," Canal saida  --) ",&cs);
dia_pega_float(15,39,"  REAL    ,  0. a 255. , ","," Desvio padrao  --) ",&desvio);
dia_poe_texto(18,43,"  Selecione janela na UVI  ",0,3);
dia_poe_texto(19,43,"     atraves do cursor     ",0,3);
```

```
for(i=12; i(21; i++){
      dia_poe_texto(i,37,"                                          ",0,1);
}
dia_peega_int(13,39,"  CANAIS :,  0  1  ou 2, ","," Canal entrada  --) ",&ce);
dia_pega_int(14,39,"  CANAIS :,  0  1  ou 2, ","," Canal saida  --) ",&cs);
dia_pega_int(15,39,"  TAMANHOS,  3  5  ou 7, ","," Tamanho da mascara  --) ",&ta);
dia_pega_int(16,39,"          ,          , ","," Ordem  --) ",&ordem);
dia_poe_texto(18,43,"  Selecione janela na UVI  ",0,3);
dia_poe_texto(19,43,"     atraves do cursor     ",0,3);
```

```
for(i=12; i(21; i++){
      dia_poe_texto(i,37,"                                          ",0,1);
}
dia_pega_int(13,39,"  CANAIS :,  0  1  ou 2, ","," Canal entrada  --) ",&ce);
dia_pega_int(14,39,"  CANAIS :,  0  1  ou 2, ","," Canal saida  --) ",&cs);
dia_pega_int(15,39,"  TAMANHO:,  3  5  ou 7, ","," Tamanho da mascara --) ",&ta);
dia_pega_int(16,39,"          ,          , ","," Numero de vizinhos ---) ",&kk);
dia_poe_texto(18,43,"  Selecione janela na UVI  ",0,3);
dia_poe_texto(19,43,"     atraves do cursor     ",0,3);
```

```
limpahelp()
{
        int i;
        for(i=3; i(26; i++){
                dia_poe_texto(i,0,"                ",0,0);
        }
}



msg()
{

        dia_poe_texto(21,0,"                ",0,1);
        dia_poe_texto(22,0,"  digite opcao  ",0,1);
        dia_poe_texto(23,0,"                ",0,1);
}
```