UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Sistema Computacional de Apoio à Rotina de Usuários Tetraplégicos

Daniela Martins Bretas

Itajubá, agosto de 2010

UNIVERSIDADE FEDERAL DE ITAJUBÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Daniela Martins Bretas

Sistema Computacional de Apoio à Rotina de Usuários Tetraplégicos

Dissertação submetida ao Programa de Pós-Graduação em Ciência e Tecnologia da Computação como parte dos requisitos para obtenção do Título de Mestre em Ciência e Tecnologia da Computação.

Área de concentração: Matemática da Computação

Orientador: Prof. Dr. Carlos Alberto Murari Pinheiro

Agosto de 2010

Itajubá - MG

Ficha catalográfica elaborada pela Biblioteca Mauá –
Bibliotecária Cristiane N. C. Carpinteiro- CRB_6/1702

B844s

Bretas, Daniela Martins

Sistema computacional de apoio à rotina de usuários tetraplégicos / Daniela Martins Bretas. -- Itajubá, (MG) : [s.n.], 2010.

97 p. : il.
Orientador: Prof. Dr. Carlos Alberto Murari Pinheiro.
Dissertação (Mestrado) – Universidade Federal de Itajubá.

1. Visão computacional. 2. Interação homem-computador. 3. Tecnologias assistivas. 4. Detecção de piscadas. 5. Deficiência física. 6. Usuário tetraplégico. I. Pinheiro, Carlos Alberto Murari, orient. II. Universidade Federal de Itajubá. III. Título.

Dedicatória

Aos meus pais, João e Doralice.

Agradecimentos

A meus pais e minhas irmãs, que sempre me apoiaram e incentivaram em todos os momentos de minha vida. Em especial, agradeço a compreensão pelas muitas ausências durante a dedicação a este trabalho.

Ao meu amigo Glauco, pela imensa prestatividade e ajuda na pesquisa bibliográfica.

A minha tia Marilda, que também muito me auxiliou graças à sua competência na área de biblioteconomia.

Ao meu orientador, Prof. Dr. Carlos Alberto Murari Pinheiro, pela imensa paciência e pelas muitas palavras de incentivo e tranquilidade.

Resumo

Técnicas de visão computacional utilizadas para a construção de interfaces apresentam grande aplicabilidade na interação entre homem e computador, especialmente na construção de tecnologias assistivas para pessoas portadoras de deficiências físicas severas, como a tetraplegia, que as incapacitam para o uso do mouse ou do teclado.

Para alguns portadores de tetraplegia, os movimentos dos olhos são os únicos movimentos do corpo sobre o qual se tem razoável controle. Para esses usuários, a única possibilidade de interação com um sistema computadorizado é através da análise e rastreamento da imagem de seus olhos.

Este trabalho apresenta um sistema computacional que captura as imagens da face do usuário em tempo real usando uma câmera digital do tipo webcam. Um método de detecção de piscadas voluntárias possibilita que usuários tetraplégicos controlem uma interface com menus do tipo *click-down* sem o auxílio de suas mãos. O uso do método proposto na integração com aplicativos ou sistemas automatizados desenvolvidos para tetraplégicos é capaz de facilitar a rotina desses usuários. O método pode ser integrado, por exemplo, a acionadores eletrônicos de cadeiras de rodas, campainhas e diversos outros equipamentos que possam dar suporte à vida do usuário.

O sistema foi testado com o auxílio de voluntários não portadores de deficiências e mostrou excelentes resultados mesmo em condições de baixa iluminação. Por não necessitar de nenhum hardware especial como dispositivo de entrada, além de uma webcam, o sistema apresenta menor custo e maior conforto na operação se comparado a sistemas que se utilizam de óculos, capacetes, eletrodos especiais ou iluminação infravermelha.

Palavras-chave: Visão computacional. Interação Homem-Computador. Tecnologias Assistivas. Detecção de piscadas. Deficiência física. Usuário tetraplégico.

Abstract

Computer vision techniques used to build interfaces have great applicability in human-computer interaction, especially in the construction of assistive technologies for people with severe physical disabilities, such as the quadriplegia, who are not able to use the mouse or keyboard.

For some quadriplegic people, eye movements are the only body movements on which they have reasonable control. For these users, the only possibility of interaction with a computer system is through the image analysis and tracking of their eyes.

This paper presents a simple system that captures user face images in real time using a webcam. A detection method of voluntary blinks enables quadriplegic users to control a interface with click-down menus without the aid of their hands. The use of this proposed method integrated into applications or automated systems developed for quadriplegic is able to facilitate the routine of these users. For example, the method can be integrated into electronic triggers of wheelchairs, bells, and many equipments that can support user's life.

The system was tested on healthy volunteers and has shown excellent results even in low light conditions. It does not require any special hardware apart from a webcam. Therefore, it has lower cost and provides greater comfort to users than systems that use glasses, helmets, special electrodes or infrared lighting.

Key-words: Computer vision. Human-computer interaction. Assistive technologies. Blink detection. Physical disability. Quadriplegic users.

Sumário

Lista de figuras

1 Introdução

A tetraplegia é a condição que apresenta um indivíduo portador de paralisia de membros superiores e inferiores. Vários fatores podem levar um indivíduo a este quadro. Os mais frequentes, entretanto, são o acidente vascular cerebral (AVC) e as lesões traumáticas da medula a nível cervical causadas por acidentes automobilísticos ou mergulho em regiões de pequena profundidade.

Não existe no Brasil e no mundo um dado censitário que revele com segurança o número de pessoas que vivem hoje com esta condição. Estima-se, entretanto, que este número esteja na ordem dos milhares. Estas pessoas dependem da ajuda de cuidadores (familiares, enfermeiros, etc.) para a realização de atividades cotidianas elementares como se vestir, alimentar, tomar banho ou fazer uso de sanitários, por exemplo. O grau de dependência do tetraplégico depende da severidade de sua lesão.

A maioria dos portadores de tetraplegia preservam a sua capacidade de comunicação através da fala o que lhes permite interagir com o mundo exterior, expressar e manter o aprimoramento de suas habilidades intelectuais e cognitivas. Entretanto, apesar de preservarem suas faculdades mentais, muitos tetraplégicos não apresentam nenhuma autonomia ou possibilidade de manifestação de seus pensamentos e de suas vontades por terem também o comprometimento da fala, e vivem completamente dependentes de seus cuidadores.

Para aumentar a autonomia e a integração destes indivíduos à sociedade, são projetados e desenvolvidos recursos capazes de diminuir suas limitações e superar suas dificuldades. Estes recursos, e outros capazes de auxiliar o desenvolvimento dos potenciais e capacidades de indivíduos portadores de diferentes deficiências, são conhecidos como Tecnologias Assistivas. Existem

Tecnologias Assistivas para o treinamento e aprendizado de capacidades, para a proteção e o cuidado pessoal, para a mobilidade pessoal, para a realização de atividades domésticas, para a comunicação e a informação e para a recreação (Portal Nacional de Tecnologia Assistiva).

No âmbito das tecnologias assistivas para a interação de portadores de deficiências físicas severas com o computador, seja para atividades recreativas ou de comunicação e aprendizagem, existem diversos dispositivos facilitadores do uso do teclado ou mouses adaptados. Para portadores de tetraplegia que mantêm sua capacidade de verbalização existem softwares que permitem o acionamento de funções por meio de comandos de voz. Usuários tetraplégicos que não possuem a capacidade de utilizar teclados ou mouses adaptados nem de se comunicarem através da fala podem se beneficiar de sistemas baseados em visão computacional.

Uma interface baseada em visão computacional utiliza, como dispositivo de entrada, imagens digitalizadas do usuário capturadas por uma câmera dispensando o uso do mouse e do teclado. As imagens capturadas são processadas em tempo real, permitindo a interação do usuário com o computador através da interpretação de movimentos de partes do seu corpo (mãos, olhos, nariz, boca, etc.).

O uso destas técnicas em IHM (Interação Homem-Máquina) não apenas contribuem para a produção de tecnologias assistivas para pessoas portadoras de deficiências físicas, como também acenam para a produção de uma nova classe de interfaces capazes de revolucionar os ambientes virtuais no futuro.

Para os portadores de tetraplegia incapazes de verbalização e cujos movimentos dos olhos são os únicos movimentos do corpo sobre o qual se tem razoável controle, a única possibilidade de interação com um sistema computadorizado é através da análise e rastreamento da imagem de seus olhos. Portanto, a pesquisa de tecnologias de visão computacional, focadas no rastreamento dos olhos do usuário (*eye tracker*), são extremamente relevantes, principalmente por possibilitarem o desenvolvimento de sistemas capazes de os auxiliarem em suas rotinas diárias diminuindo o seu grau de dependência e aumentando sua auto-estima.

Sistemas baseados em *eye tracker* por iluminação infravermelha se encontram em estágio de desenvolvimento mais avançado e sua precisão para o rastreamento possibilita que o usuário movimente um cursor na tela de um computador usando apenas seu olhar. Entretanto, estes sistemas apresentam as desvantagens de um custo mais elevado por necessitarem de equipamentos

especiais para a captura das imagens e dos efeitos nocivos que a iluminação infravermelha pode causar ao usuário com o uso prolongado e frequente.

A motivação deste trabalho é o desenvolvimento de técnicas que possam auxiliar a rotina de pessoas portadoras de tetraplegia que não são capazes de falar e movimentar nenhuma parte do corpo além de seus olhos, mas que permanecem conscientes e cognitivamente ativas, por intermédio da utilização de um sistema computadorizado baseado em visão computacional que não utilize equipamentos de hardware especiais para a captura e processamento das imagens.

Com este objetivo foi desenvolvido um sistema computacional que captura as imagens da face do usuário em tempo real usando uma câmera digital de baixo custo do tipo webcam. As imagens adquiridas são processadas por técnicas de processamento digital de imagens visando, primeiramente, a extração de informações sobre a localização dos olhos. Duas técnicas são utilizadas nesta etapa: segmentação por detecção de pontos e segmentação baseada em estudos morfológicos. Após a inicialização do sistema, o usuário é capaz de navegar em um menu de opções do tipo *click-down* através do controle das piscadas de seus olhos. As piscadas involuntárias, de curtíssima duração, não são consideradas. Piscadas voluntárias de curta duração alteram o foco de seleção na interface, e piscadas voluntárias mais longas acionam o botão que detém o foco no instante da piscada. A técnica apresentada pode ser aplicada a qualquer aplicativo ou sistema automatizado desenvolvido para tetraplégicos. A interface poderia ser integrada, por exemplo, a sistemas projetados para se acionar atuadores eletrônicos em um sistema de suporte à rotina de um usuário deficiente físico, como cadeiras de rodas, campainhas, etc.

Esta dissertação está organizada da seguinte forma: o capítulo 2 apresenta uma discussão sobre o uso de técnicas de visão computacional na construção de interfaces assistivas para portadores de deficiências. No capítulo 3, é realizada uma revisão sobre técnicas de segmentação de imagens. Diferentes técnicas são discutidas. Devido à relevância dos estudos morfológicos para este trabalho, o capítulo 4 é dedicado à uma revisão sobre as operações e aplicações da morfologia matemática em processamento digital de imagens. No capítulo 5 é feita a apresentação do sistema projetado e dos métodos utilizados em seu desenvolvimento. Também é apresentado um exemplo de integração do método a um aplicativo para comunicação através do uso de um teclado virtual. O capítulo final apresenta as conclusões do trabalho e as perspectivas futuras para o projeto.

2 Visão Computacional em IHM para Portadores de Deficiências

Computadores pessoais estão cada vez mais presentes na vida das pessoas. Hoje são utilizados não apenas como ferramenta de trabalho, mas também como meio de entretenimento, comunicação e inserção social. Seja qual for a sua utilidade, a interface de comunicação entre homem e computador ou Interface Homem-Máquina (IHM) é de extrema relevância para a ciência da computação e para a usabilidade dos sistemas.

De uma maneira geral, os trabalhos em IHM estudam as formas como um usuário se comunica com sistemas computadorizados, ou seja, investiga a capacidade de interatividade dos sistemas computacionais com os usuários. Computadores são capazes de interpretar apenas sinais elétricos e convertê-los para um sistema digital binário. Por outro lado, usuários são capazes de interpretar apenas sons, imagens, toques, paladar e cheiros. É preciso então que haja uma interface que intermedeie a comunicação entre estes dois agentes (Fig. 2.1).

A maioria dos computadores pessoais utilizam um mouse, um teclado e um monitor como dispositivos padrões para a comunicação entre usuário e computador. Entretanto, no contexto da inserção social, estes dispositivos não são os mais funcionais para uma parcela da população que apresenta algum tipo de deficiência, principalmente motora ou visual. A maior parte das pessoas não apresenta nenhuma dificuldade em utilizar o mouse como dispositivo de entrada, entretanto, a simples tarefa de arrastar o mouse para mover o cursor ou clicar um botão pode ser muito difícil ou mesmo impossível para algumas pessoas.

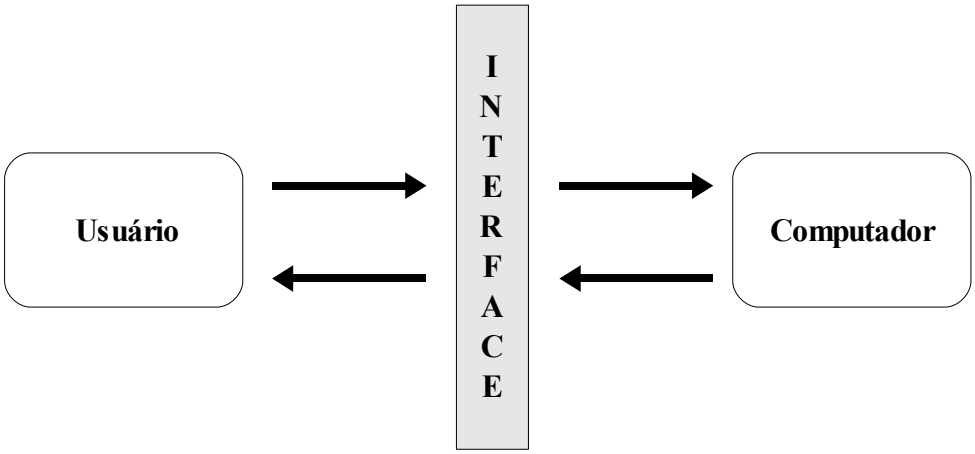


Fig. 2.1 – Fluxograma de interação entre homem-computador.

As interfaces conhecidas atualmente são baseadas em displays e menus gráficos do tipo *click-down* ou *pull-down* acionados por um mouse ou um teclado (BYRNE et al., 1999). Porém, para Jaimes e Sebe (2007), o modelo de um usuário sentado na frente de um computador está mudando rapidamente. Os computadores estão sendo integrados, cada vez mais rapidamente, a dispositivos de entrada equivalentes aos sentidos humanos como câmeras, microfones, monitores touch screen, etc., e a pesquisa por interfaces mais naturais e intuitivas é fundamental. Interfaces perceptivas possibilitarão o surgimento de novos ambientes virtuais no futuro.

Os estudos de IHM também investigam a usabilidade de sistemas computacionais. O princípio da usabilidade está relacionado à facilidade, conforto de uso e eficiência dos sistemas. Um monitor padrão não é o melhor meio de comunicação com um usuário portador de deficiência visual, assim como um mouse ou um teclado também não é o melhor meio de comunicação com um usuário portador de alguma limitação ou incoordenação motora.

Para Bergman e Johnson (1995), usuários deficientes estão sendo negligenciados no projeto de interfaces. Projetar softwares que levam em consideração as necessidades de usuários deficientes aumenta a usabilidade dos sistemas para todos, inclusive para aqueles usuários sem nenhuma incapacidade. Usuários que precisam manter os olhos em suas tarefas como controladores de tráfego aéreo, por exemplo, podem se beneficiar de sistemas que se interagem por meio de voz assim como usuários com deficiência visual. Projetar interfaces voltadas para deficientes visuais ou para usuários que não podem manter a atenção em uma tela de computador por alguma particularidade de sua atividade não implica em relevantes diferenças porque as necessidades são as mesmas.

Embora usuários portadores de deficiências auditivas tenham menores problemas com a usabilidade de sistemas computadorizados porque as interfaces usuais têm forte componente visual e, geralmente, não são dependentes da condição do usuário ter ouvido alguma notificação sonora, interfaces projetadas para melhor atenderem esta parcela da população também podem beneficiar usuários de computadores em ambientes de elevado ruído, como o ambiente de produção de fábricas, metrô, etc.

Além disto, tecnologias assistivas que incluem meios de entrada e saída especializados podem incluir pessoas deficientes no mercado de trabalho além de aumentar a capacidade laborativa de usuários com incapacidades temporárias. Um usuário que fratura um pulso ou um braço passa por um período com as mesmas capacidades que muitos deficientes físicos permanentes.

Ainda para Bergman e Johnson (1995), muitos usuários são excluídos do mundo digital porque este é voltado para pessoas com elevada capacidade motora e sensorial. Entretanto, este mesmo mix de áudio e vídeo pode aumentar a acessibilidade dos sistemas digitais por serem capazes de gerar múltiplas formas de entrada e saída da informação.

É preciso, então, oferecer alternativas efetivas para usuários que não podem acessar ou usufruir das facilidades do mundo digital através dos dispositivos convencionais. Usuários cegos podem interagir com sistemas computacionais através de navegação por teclado Braille ou softwares leitores de texto, por exemplo. Portadores de deficiências físicas de membros superiores podem utilizar mouses adaptados para os pés, tecnologia para *eye tracker*, etc.

Muitas tecnologias assistivas, como teclados virtuais, softwares para reconhecimento de voz e leitura de textos, já estão sendo embarcadas em softwares reconhecidos no mercado. Interfaces multimodais (Multimodal Human-Computer Interaction – MMHCI), ou seja, que respondem a entradas em mais de uma modalidade ou canal de comunicação, possibilitam ao usuário escolher a melhor forma de interação e, consequentemente, melhoram a acessibilidade (JAIMES; SEBE, 2007) e tornam os sistemas mais amigáveis.

Fornecer acessibilidade significa remover barreiras que dificultam pessoas com deficiências a participarem de atividades cotidianas substanciais, incluindo o uso de serviços, produtos e informações. É, portanto, por definição, uma categoria da usabilidade pois softwares que não são acessíveis para uma classe de usuários em particular, têm baixa usabilidade para eles (BERGMAN; JOHNSON, 1995).

Explorar os sentidos humanos na construção de interfaces torna a interação homem-computador cada vez mais natural. E ainda, se vários sentidos são explorados como no caso de interfaces multimodais, maior será a acessibilidade da aplicação. Pessoas com deficiências em um ou outro sentido, ainda serão capazes de utilizar e aplicar o sistema.

Então, ao se projetar um sistema digital é necessário um estudo detalhado de quais problemas ele se proporá a resolver e que tipos de usuários irão utilizá-lo. Projetar interfaces mais acessíveis ou que apresentem múltiplos canais de comunicação envolve maiores custos. Em razão disto, estas interfaces são muitas vezes relegadas a um segundo plano.

Uma alternativa ao elevado custo das tecnologias assistivas usuais é a adaptação dos próprios dispositivos de entrada e saída mais comuns às necessidades dos usuários portadores de deficiências. Além do menor custo, a vantagem desta abordagem é a facilidade de suporte e a extensão das atualizações das tecnologias dos dispositivos usuais também aos dispositivos adaptados.

Bergman e Johnson (1995) citam melhoramentos nos dispositivos de I/O comuns que podem melhorar a usabilidade para deficientes físicos. Alguns exemplos são o travamento das teclas Shift e Control, o atraso na repetição de teclas, displays braille, sons de áudio para indicar a posição do usuário, closed captioning, etc.

Shih, Chang e Shih (2009) discutem que novos drivers para mouses podem ser projetados para se adaptarem aos movimentos existentes e limitados de diferentes deficientes físicos permitindo configurações diversas das usuais para as funções disponíveis no mouse tradicional. As funções de clicar, arrastar, etc., podem ser adaptadas e distribuídas aos movimentos possíveis para o usuário. Entretanto, novos projetos de drivers para ajustar e colocar novas funções em mouses tradicionais são raramente propostos por ser algo bem mais complexo.

Segundo Kennedy et al. (2000), pacientes presos a deficiências físicas que não podem se mover nem falar são alertas e cognitivamente intactos. Então, usar o movimento dos olhos como mecanismo de entrada para guiar a interação com interfaces gráficas possibilita que estas pessoas utilizem sistemas computadorizados em suas rotinas. Os autores descrevem uma técnica de BCI (Brain Computer Interface) que usa eletrodos especiais, implantados em uma camada externa do córtex cerebral, permitindo que o paciente guie um cursor sobre a tela de um computador à sua frente através de um controle eletroencefalográfico provocado por movimento dos olhos, piscadas,

ou movimentos de alguns músculos da face. Esta técnica, entretanto, é extremamente invasiva.

Outras técnicas, menos invasivas mas ainda pouco confortáveis para o uso, utilizam eletrodos montados na pele ao redor dos olhos que medem a diferença do potencial elétrico para detectar o movimento do globo ocular. Jacob (1991) ressalta que a cabeça do usuário deve permanecer imóvel para garantir que qualquer medida de movimento represente realmente a movimentação dos olhos do usuário e não de sua cabeça.

Os sistemas mais modernos são baseados em visão computacional e utilizam imagens de vídeo para capturar os movimentos do usuário através de softwares de processamento de imagens e evitarem o problema da invasividade dos sistemas mais antigos. Técnicas para *head tracking*, *nose tracking*, *eye tracking*, etc., utilizadas para a construção de interfaces *hands-free* para portadores de deficiências físicas fornecem um meio alternativo ao mouse e teclado para moverem um cursor na tela, selecionar e arrastar objetos, mover barras de rolagem, etc.

Para Porta (2002), uma interface baseada em visão computacional ou VBI (Vision Based Interface) é uma interface perceptiva que explora o sentido da visão como canal de comunicação entre usuário e computador. Sistemas que utilizam VBIs interpretam informações adquiridas por meio de câmeras digitais sobre os movimentos do corpo do usuário (expressão facial, gestos, *gaze*, etc.) como intenções de comandos em aplicações computacionais. O *gaze* revela o ponto de visão ou a direção do olhar do usuário e é uma forte indicação do objeto de atenção naquele instante.

Técnicas de visão computacional baseadas no rastreamento do *gaze* do usuário (*gaze tracker*) são especialmente importantes para usuários portadores de tetraplegia. Estes usuários são pouco assistidos na interação com computadores em virtude da severidade de sua condição física limitar ou mesmo impossibilitar o uso de quase todos os dispositivos de interação conhecidos. Para algumas destas pessoas, os movimentos dos olhos são os únicos movimentos do corpo sobre o qual se tem razoável controle. Portanto, a pesquisa de tecnologias para *eye tracker* ou *gaze tracker* são extremamente relevantes para estes usuários por possibilitarem o desenvolvimento de sistemas capazes de os auxiliarem em suas rotinas diárias. Neste trabalho, os termos *eye tracker* e *gaze tracker* serão utilizados intercambiavelmente.

Para Betke, Gips e Fleming (2002), os sistemas baseados em visão computacional, por dispensarem qualquer tipo de hardware especial, são preferidos por usuários com deficiências

físicas também por chamarem menos a atenção para suas deficiências. Os autores desenvolveram um sistema supervisionado que capta os movimentos do usuário através de uma câmera de vídeo e converte estes movimentos em movimentos do cursor na tela de um computador. O sistema pode ser usado com qualquer software comercial e é bastante flexível podendo-se escolher a região do corpo que será rastreada, como nariz, olhos, boca, ponta dos dedos, etc. Nesse caso, o rastreamento dos olhos não significa rastrear o ponto de observação do usuário (*gaze*) e sim o deslocamento dos olhos efetuado em conjunto com movimentos de cabeça. O sistema foi testado em 20 pessoas não portadoras de deficiências e em 12 portadores de deficiências físicas severas e exige que o usuário tenha a capacidade de coordenar o movimento de alguma parte do corpo que será utilizada para o rastreamento. Apenas 3 portadores de deficiências físicas não obtiveram êxito nos testes.

Usuários com limitações físicas que não possuem a coordenação do movimento de nenhuma parte do corpo, mas que têm a capacidade de emitir sons, ou falar, podem fazer usos de sistemas com reconhecimento de voz. Porém, quando a verbalização também não é possível, os únicos movimentos para rastreamento que restam ao usuário são os movimentos de seus olhos e as técnicas de *eye tracker* ou *gaze tracker* são a única opção viável nesses casos.

A maioria dos sistemas para *eye tracker* são baseados em LEDs que emitem um feixe de luz na faixa do infravermelho (IR) e que devem ser direcionados para dentro dos olhos do usuário. Segundo Poole e Ball (2006), a luz infravermelha é utilizada por não causar o ofuscamento causado pela luz visível. A luz entra na retina e em sua grande parte é refletida de volta, fazendo a pupila aparecer como um ponto brilhante conhecido como efeito *bright pupil*. A reflexão corneana também é gerada por luz infravermelha, aparecendo como um pequeno, mas intenso ponto brilhante (Fig. 2.2).

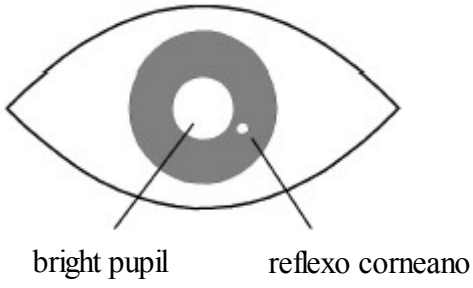


Fig. 2.2 – Imagem do reflexo corneano e *bright pupil* provocados por um LED infravermelho.
Fonte: POOLE e BALL, 2006.

Embora sistemas com IR sejam mais precisos, há a preocupação com os efeitos da exposição prolongada ao seu feixe de luz (JAIMES; SEBE, 2007). Sistemas que não utilizam IR precisam de imagens com maior resolução para sua eficiência e ainda se encontram em estágio de desenvolvimento menos avançado. Estes sistemas utilizam modelos matemáticos para encontrar os olhos do usuário, determinar sua orientação e rastrear sua trajetória em uma imagem digitalizada.

Interfaces baseadas em *gaze tracker* são um caminho natural em IHM não apenas pelo seu enorme potencial de acessibilidade para usuários portadores de necessidades especiais, mas também por ser um meio rápido e fácil para auxiliar a comunicação entre usuários e computador. Uma vez que o usuário direciona sua atenção para um objeto por meio de seus olhos, dispositivos *eye tracker* tornam uma interface muito mais perceptiva e natural por permitirem que objetos em uma tela de computador sejam selecionados simplesmente com o olhar.

Varona, Manresa-Yee e Perales (2008) apresentam uma interface *hands-free* projetada especificamente para pessoas com deficiências incapacitantes em mãos ou membros superiores. O objetivo da nova interface é substituir completamente o tradicional mouse na interação com computadores. O sistema utiliza *nose tracking* para o controle do cursor e o reconhecimento das picadas com olho direito e olho esquerdo para o acionamento dos links. O método foi testado em dois grupos de usuários, um que já havia sido treinado para a sua utilização e outro, que nunca tinha tido contato com o sistema. Os resultados mostraram 97,3% de eficiência no reconhecimento das piscadas de olhos dos usuários do primeiro grupo; e 85,9%, no segundo grupo.

Piscadas voluntárias são utilizadas para substituir a ação do mouse em menus do tipo *click-down* em diversos trabalhos como o de (CHAU; BETKE, 2005). A identificação das piscadas dos usuários também é muito utilizada em softwares de monitoramento de motoristas de automóveis e detectores de mentira. A detecção do movimento das pálpebras durante uma piscada também é útil para a localização automática da posição dos olhos na imagem em sistemas *eye tracker* (MORRIS; BLENKHORN; ZAIDI, 2002).

Grauman et al. (2003) usam piscadas voluntárias como um dispositivo binário de entrada em aplicações que não requerem nenhum movimento de cursor, ou seja, aplicações operadas apenas por clicks do mouse. Estas aplicações apresentam uma opção de cada vez na tela do computador e o usuário deve então selecionar a opção desejada quando esta aparecer (*interfaces scanning*). Um padrão de piscadas do usuário pode ainda ser desenvolvido para construir uma codificação para a construção de mensagens textuais.

Atualmente a velocidade da comunicação entre homem e máquina só não é maior porque o tempo de resposta do lado do homem pode ser considerado, relativamente, constante. Ou seja, do lado da máquina as velocidades de processamento dos computadores estão cada vez maiores com o surgimento de processadores mais velozes e com múltiplos núcleos; entretanto, do lado do usuário o tempo de resposta só pode ser alterado com treinamento para o uso mais rápido dos dispositivos de entrada padrões (mouse, teclado ou joystick). A possibilidade de se introduzir o movimento dos olhos do usuário como meio de entrada pode, entretanto, diminuir o tempo de resposta do usuário e revolucionar os ambientes virtuais do futuro.

De acordo com Ware e Mikaelian (1987), dispositivos de seleção baseados em *eye tracker* são mais rápidos quando comparados ao mouse ou qualquer outro dispositivo convencional. Byrne et al. (1999) observam que nenhum movimento do mouse é realizado até que o objeto de interesse tenha sido localizado pelo usuário economizando-se o tempo entre a percepção e a ação de deslocar o mouse e clicar o botão. Ou seja, a ação de deslocamento do mouse e de apertar o botão é eliminada em técnicas de *eye tracker* e nenhum esforço adicional é realizado uma vez que o usuário já percorreu a interface com seus olhos.

Jacob e Sibert (2000) utilizam dois experimentos que comparam a seleção de objetos usando o mouse e um sistema *eye gaze* para mostrar que os olhos se movem mais rápido que as mãos e que nós movemos primeiro os olhos e depois as mãos para selecionar algum item na tela de um computador tradicional. Muitas características do olho humano podem ser utilizadas para se inferir o ponto de observação de um usuário de computador, tal como a reflexão corneana, o centro da íris ou da pupila, o limite entre a íris e a esclera e a relação entre o reflexo corneano e o centro da pupila. Esta última técnica, por relacionar duas características permite desassociar os movimentos da cabeça dos cálculos do *gaze* permitindo uma certa liberdade de movimentos por parte do usuário. Estes sistemas necessitam de um procedimento de calibração onde o usuário fixa o olhar para um determinado ponto na tela e o sistema, então, grava o relacionamento entre o centro da pupila e o reflexo corneano como correspondente à coordenada (x, y) do ponto de calibração. Este procedimento é repetido algumas vezes dependendo da implementação.

A utilização de técnicas de *eye tracker* fornecendo a possibilidade de se operar interfaces com os olhos é extremamente útil não apenas para deficientes físicos mas também para agilizar tarefas em ambientes onde os usuários precisam manter suas mãos ocupadas com outras atividades. O movimento dos olhos é um movimento natural e requer pouco esforço consciente do usuário. Entretanto, incorporar *eye tracker* em um sistema interativo requer modelos matemáticos

precisos. Além disto, a maioria das tecnologias existentes atualmente apresentam um elevado custo por exigirem hardware e software especializado.

Outra dificuldade na construção de interfaces *eye tracker* é evitar o efeito Midas Touch. Assim como o personagem da mitologia que tudo que toca vira ouro, na ocorrência de um Midas Touch, tudo o que usuário olha ativa um comando, ou seja, o usuário não pode olhar para qualquer lugar sem que isto seja interpretado como um comando (JACOB, 1993). Muitos movimentos dos olhos são involuntários e alguns ícones podem ser ativados não intencionalmente tornando a intenção do usuário difícil de se interpretar (POOLE; BALL, 2006).

O rastreamento do movimento dos olhos pode ainda ser útil na avaliação da usabilidade de interfaces gráficas de sistemas comerciais e páginas web. Neste caso, a determinação dos pontos de fixação do usuário é utilizada para se identificar as áreas de interesse na tela do computador. O ponto de fixação (*gaze*) pode, por exemplo, determinar quais anúncios ou regiões estão chamando mais a atenção do usuário e, desta forma, aprimorar o desenvolvimento de interfaces (POOLE; BALL, 2006).

3 Segmentação

Em geral, interfaces baseadas em visão computacional apresentam quatro etapas. Primeiramente, é necessário determinar a localização na imagem da região do corpo que será utilizada para o rastreamento (1 – Segmentação). A seguir, as mudanças de posição do objeto em questão devem ser reportadas em tempo real e traduzidas para uma nova posição do cursor na tela (2 – Rastreamento). Ao mesmo tempo em que o rastreamento, ou *tracking*, é realizado, características tais como olho aberto, olho fechado, mãos abertas, mãos fechadas devem ser identificadas para que na fase seguinte possam ser interpretadas como comandos (3 – Classificação). E, finalmente, as classes identificadas na fase anterior são interpretadas gerando algum comando no aplicativo como um clique ou um duplo clique do mouse, por exemplo (4 – Interpretação). Este capítulo se dedica a revisar algumas técnicas de segmentação encontradas na literatura.

A segmentação é a primeira e mais importante etapa para o processamento de imagens digitais e tem grande relevância para sistemas de visão computacional. O procedimento de segmentação simplifica a análise das imagens por reduzir consideravelmente a quantidade de dados a serem processados, ao mesmo tempo em que preserva informações sobre a estrutura e os limites dos objetos. Então, para a validade e o sucesso das etapas subsequentes do processamento, é preciso que o processo de segmentação seja confiável.

Neste contexto possui aplicações diversas tais como em medicina para a análise de exames radiológicos, em monitoramento por satélite, biometria, reconhecimento de faces e caracteres, robótica, etc.

Para Carvalho (2006), “segmentação de imagens é um processo que tipicamente divide em partições o domínio espacial de uma imagem em subconjuntos mutuamente exclusivos, chamados segmentos ou regiões. Cada região é homogênea e uniforme considerando-se algumas propriedades como tom ou textura e cujos valores diferem em alguns aspectos e significados, das propriedades de cada região vizinha”, ou seja, a segmentação é o processo de subdividir uma imagem em suas partes ou objetos constituintes. O procedimento termina quando os objetos de interesse na aplicação tiverem sido isolados (GONZALES; WOODS, 2000). Para Ziou e Tabbone (1998), um bom método de segmentação deve encontrar os objetos de interesse com pouco esforço computacional.

De uma forma geral, a segmentação pode ser vista como um procedimento de classificação que classifica os pixels como pertencentes à região dos objetos ou à região do plano de fundo.

As principais técnicas de segmentação de imagens em escala de cinzas se baseiam nas propriedades de descontinuidade (seção 3.1) ou similaridade dos valores de intensidade dos pixels (seção 3.2). A descontinuidade se refere às mudanças bruscas nos níveis de cinza e é a propriedade considerada pelos algoritmos de detecção de pontos, linhas ou bordas. Já a similaridade é a propriedade base para os algoritmos de limiarização por histograma ou *thresholding*, divisão e fusão de regiões (como algoritmos do tipo *quadtree* e *watersheds*). Há também técnicas que utilizam a variação da posição dos objetos no tempo para executar ou aprimorar o processo de segmentação (seção 3.3).

De uma forma geral, as técnicas para segmentação de imagens monocromáticas ou binárias também poderiam ser aplicadas para imagens coloridas executando-se os mesmos algoritmos em cada canal de cor e fazendo uma interseção entre as imagens resultantes. Entretanto, esta extensão seria bastante simplista por não levar em consideração a significância de cada canal em cada um dos diferentes espaços utilizados para representação de imagens coloridas. Existem técnicas diferenciadas para estas imagens, mas que são pouco abordadas na literatura. Skarbek e Koschan (1994) relacionam diversas destas técnicas. Lezoray e Cardot (2002) propõem um método confiável e auto-adaptável, excelente para espaços de cor com canais correlacionados, que modifica a influência das informações de cada canal, iterativamente, de acordo com critérios locais e globais avaliados em tempo real.

Técnicas de morfologia matemática também são frequentemente utilizadas para

segmentação. Estas técnicas analisam a estrutura geométrica inerente ao objeto de interesse e assim são capazes de extrair objetos com formatos semelhantes em um determinado cenário. São excelentes para o caso de segmentação de objetos claramente distinguíveis através de sua forma, como é o caso da íris ocular que é facilmente distinguida em imagens de face em virtude de seu formato circular. Por sua relevância para o método proposto por este trabalho, seus conceitos e técnicas serão discutidos separadamente no Capítulo 4.

3.1 Segmentação por Detecção de Descontinuidades

A maioria dos algoritmos de segmentação utiliza uma matriz 3 X 3 como máscara sobre os pixels da imagem original para detecção de descontinuidades (Fig. 3.1). Cada pixel em questão é avaliado juntamente com seus oito pixels vizinhos multiplicando-se o valor de sua intensidade de cinza pela constante determinada pela máscara sobreposta à imagem original. A somatória (1) destes valores representa a saída do pixel de posição central, onde *c* representa a máscara e *z* a intensidade de cinza da imagem. Este valor de saída é então comparado com um valor de *threshold* para determinar quais pixels são mais prováveis de fazerem parte de uma linha ou de um ponto, por exemplo.

$$S = \sum_{i=0}^{8} c_i z_i \tag{1}$$

C_2	C_1	C_8
C_3	C_0	C_7
C_4	C_5	C_6

Fig. 3.1 - Representação de uma máscara para detecção de descontinuidades.

Uma correta determinação do valor de *threshold* é, então, crucial para a implementação de máscaras de detecção de descontinuidades. Valores muito baixos poderão produzir uma imagem de saída muito borrada, e valores muito altos poderão perder detalhes significantes para a segmentação (ROBINSON, 1977).

As máscaras de detecção varrem a imagem, da primeira à última linha, avaliando uma

região quadrada de pixels de cada vez. Quando os pixels analisados pertencem à extremidade da imagem original (Fig. 3.2), os pixels ausentes na sobreposição da máscara são preenchidos com zero ou duplicados da primeira linha da imagem (no caso de pixel da margem superior), última linha da imagem (margem inferior), primeira coluna (margem esquerda) ou última coluna da imagem (margem direita).

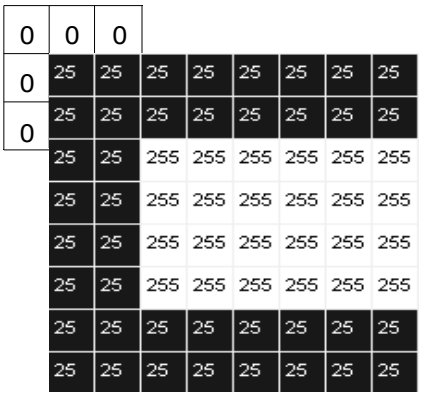


Fig. 3.2 - Sobreposição de uma máscara 3 X 3 ao pixel I(1,1) de uma imagem 8 X 8.

Uma vantagem de se utilizar máscaras 3 X 3 é que o pixel é analisado em relação a todos seus circunvizinhos. Máscaras 2 X 2, por exemplo, analisam apenas os pixels a 0°, 270° e 315° e, por analisarem um espaço menor, são muito mais sensíveis a ruídos. Máscaras do tipo J X J analisam *(j* x *j – 1)* pixels ao redor do pixel de interesse.

3.1.1 Detecção de Pontos

A detecção de pontos em uma imagem é conseguida com a utilização da máscara da Fig. 3.3 (GONZALES; WOODS, 2000). Cada pixel é considerado parte de um ponto se a saída |S| é maior que o valor de *threshold*. A idéia é que cada pixel avaliado isoladamente terá um peso maior em relação aos seus circunvizinhos e será destacado pela máscara se seu nível de intensidade variar acima de um determinado limite. A detecção de pontos, então, está condicionada ao valor de *threshold* escolhido. Regiões homogêneas no nível de intensidade produzirão saídas próximas a zero.

-1	-1	-1
-1	8	-1
-1	-1	-1

Fig. 3.3 - Máscara para detecção de pontos.

O resultado é uma imagem binária onde os pixels brancos revelam a presença dos pontos detectados; e os pixels pretos, o plano de fundo. A cada translação da máscara, a seguinte comparação é realizada:

$$S_2(x, y) = \begin{cases} 1 \; se \; S_1(x, y) > t \\ 0 \; se \; S_1(x, y) \leq t \end{cases} \quad (2)$$

Onde t é o valor de limiar, ou *threshold*, escolhido, $S_1(x, y)$ o valor de saída após a aplicação da máscara de detecção de pontos e $S_2(x, y)$ o valor de saída após a limiarização. Desta forma, pequenos desvios na intensidade do tom não são notados. Apenas descontinuidades maiores que o valor definido para o *thresholding* são identificadas.

3.1.2 Detecção de Linhas

O princípio utilizado para a detecção de pontos é repetido para a detecção de linhas, agora com máscaras sensíveis à identificação de linhas horizontais (Fig. 3.4(a)), verticais (Fig. 3.4(b)) ou diagonais (Fig. 3.4(c) e 3.4(d)). No caso da Fig. 3.4(a), por exemplo, um valor máximo de |S| é alcançado quando a linha do meio da máscara sobrepõe os pixels de uma linha horizontal na imagem original, já a Fig. 3.4(b) produz um valor elevado de |S| quando a coluna do meio da máscara sobrepõe uma linha vertical na imagem.

Embora não constitua uma regra, a idéia central na construção de máscaras de detecção de descontinuidades é que a soma de seus coeficientes deve ser anulada estabelecendo-se valores positivos para os pixels que se deseja destacar e, valores negativos para o seu complemento. Assim, em todas as máscaras da Fig. 3.4, temos que: $C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_7 + C_8 + C_9 = 0$.

-1	-1	-1
2	2	2
-1	-1	-1

(a)

-1	2	-1
-1	2	-1
-1	2	-1

(b)

-1	-1	2
-1	2	-1
2	-1	-1

(c)

2	-1	-1
-1	2	-1
-1	-1	2

(d)

Fig. 3.4 - Máscara para detecção de linha horizontal (a), linha vertical (b) e linhas diagonais (c) e (d).

Segundo Robinson (1977), utilizando uma matriz 3 X 3, é possível ainda determinar a direção das arestas, em imagens em tons de cinza, nos oito sentidos a seguir (Fig. 3.5): norte (1), noroeste(2), oeste (3), sudoeste(4), sul(5), sudeste(6), leste(7), e nordeste(8). O autor considera que a parte mais brilhante da aresta está sempre à esquerda e testa oito máscara direcionais correspondentes aos oito sentidos mencionados, a máscara que produzir o maior resultado de saída determinará a direção da aresta.

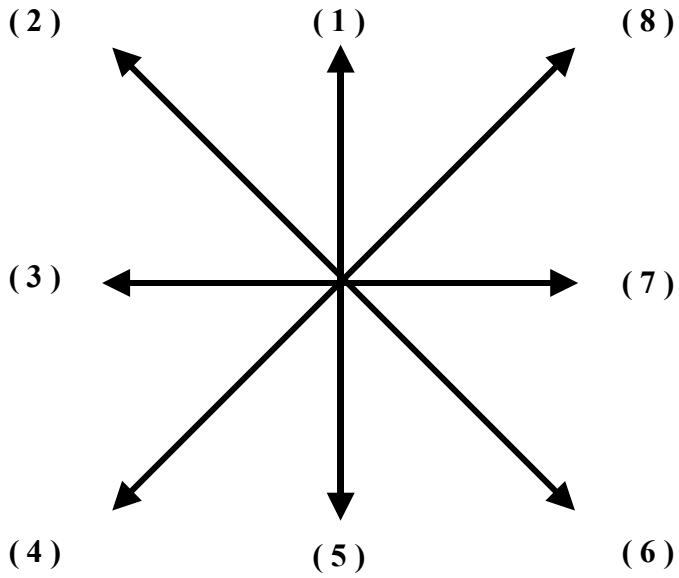
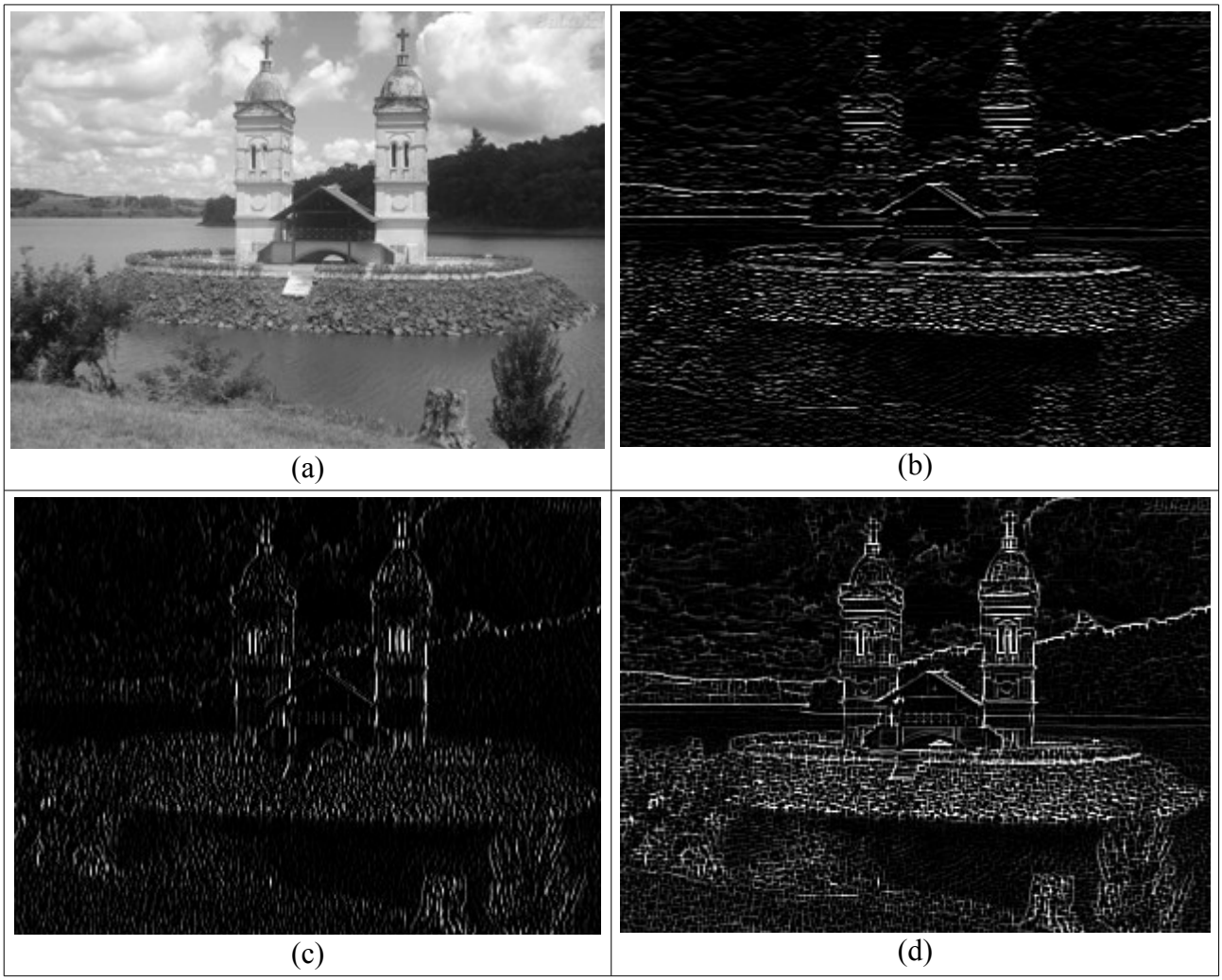


Fig. 3.5 – Sentidos obtidos com máscaras direcionais 3 X 3.

Um bom método de detecção é aquele que não identifica falsas linhas e nem perde linhas verdadeiras. As características dos contornos das imagens que se deseja analisar determinam a melhor máscara a ser utilizada. É possível também combinar duas ou mais máscaras, conseguindo a segmentação em mais de um sentido em uma mesma imagem (Fig. 3.6).



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

Fig. 3.6 – Exemplo de aplicação de máscara de detecção de linhas: (a) Imagem original; (b) Interação com máscara para detecção de linhas horizontais; (c) Interação com máscara para detecção de linhas verticais; (d) combinação das máscaras de (b) e (c).

Resumindo, as características das imagens do espaço analisado devem influenciar no método de segmentação escolhido. Imagens de auto-estrada, por exemplo, são bem segmentadas por algoritmos de detecção de linhas, entretanto, em imagens com características geométricas mais complexas deve-se escolher outra abordagem para a segmentação.

3.1.3 Detecção de Bordas

"Uma borda é o limite entre duas regiões com propriedades relativamente distintas de nível de cinza" (GONZALES; WOODS, 2000). Uma borda ou *edge* é uma região de fronteira entre um objeto e o plano de fundo (*background*), e também indica a fronteira entre objetos sobrepostos. Uma vez que a identificação precisa das bordas em uma imagem pode identificar objetos com

exatidão, técnicas para detecção de bordas são ferramentas essenciais para o processo de segmentação. A detecção de bordas, entretanto, não determina quais pixels fazem parte do objeto e quais pixels fazem parte do plano de fundo, ela apenas determina o limite entre as regiões identificadas (PARKER, 1996), ou seja, quais pixels fazem parte de um contorno e quais não fazem parte do contorno.

Segundo Vincent e Folorunso (2009), as bordas de uma imagem contêm informações relevantes para o processamento digital e representar uma figura pela imagem de suas bordas tem a vantagem de reduzir a quantidade de dados para posterior análise, ao mesmo tempo em que preserva informações sobre a estrutura da imagem.

É difícil projetar um algoritmo para detecção de bordas que seja eficiente para qualquer tipo de imagem e que capture adequadamente os traços desejados para as etapas subsequentes do processamento. Neste contexto, uma variedade de detectores de bordas têm sido projetados no desenvolvimento de técnicas para processamento digital de imagens (ZIOU; TABBONE, 1998).

Matematicamente, as bordas correspondem a regiões da imagem onde ocorre uma mudança de intensidade em certo intervalo do espaço e em certa direção. Isto corresponde a regiões de alto valor da derivada da função *f(x, y)* que expressa intensidade de pixels em uma imagem. Com esta informação, podemos concluir que os pontos onde esta derivada é alta correspondem a pixels pertencentes às bordas em uma determinada imagem.

A noção de gradiente, bastante popular em programas de edição de imagens, tem sido estendida para a solução de alguns problemas em segmentação de imagens. Neste contexto, o termo gradiente é usado para caracterizar uma mudança gradual de cor na imagem (Fig. 3.7). Como pixels pertencentes ao contorno dos objetos apresentam uma mudança brusca na intensidade de cor da imagem, podem ser detectados através de um elevado valor de gradiente, ou seja, uma elevada taxa de mudança no nível de intensidade da função *f(x, y)* que representa a imagem digital. Vários autores apresentam técnicas diferentes para se calcular o gradiente (CHAUDHURI; CHANDA, 1984).

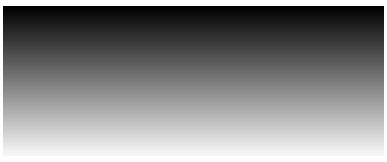


Fig. 3.7 – Gradiente em tons de cinza.

Com estes conceitos, a implementação matemática do gradiente (3) e (4) é, então, útil para a detecção das regiões de elevada derivada na função *f(x, y)* que representam as bordas de uma imagem. Computacionalmente, usa-se um par de máscaras, geralmente 3 X 3, onde uma máscara avalia o gradiente ao longo do eixo *x*, e a outra, avalia o gradiente na direção do eixo *y* (VINCENT; FOLORUNSO, 2009).

$$G_x = \triangle x = \frac{f(x+dx, y) - f(x, y)}{dx} \tag{3}$$

$$G_y = \triangle y = \frac{f(x, y+dy) - f(x, y)}{dy} \tag{4}$$

Onde G_x e G_y representam os gradientes nas direções dos eixos *x* e *y*, respectivamente, e *dx* e *dy* medem a distância entre os pixels, também ao longo dos eixos *x* e *y*. Como imagens digitais são representadas em espaços discretos, podem-se considerar *dx* e *dy*, em termos numéricos, iguais ao valor 1(um) como sendo a distância entre dois pixels vizinhos. A combinação dos dois gradientes Δx e Δy resulta no gradiente da função representado por Δf. Na maioria das implementações, o valor de Δf é, então, comparado a um valor limiar para se identificar os pixels que apresentam elevada taxa de mudança na função da imagem (Fig. 3.8). A imagem resultante de algoritmos de detecção de bordas representam um mapa dos contornos dos objetos e, por isto, são chamadas frequentemente de *edge map*. Neste trabalho, será sempre referenciada desta forma.

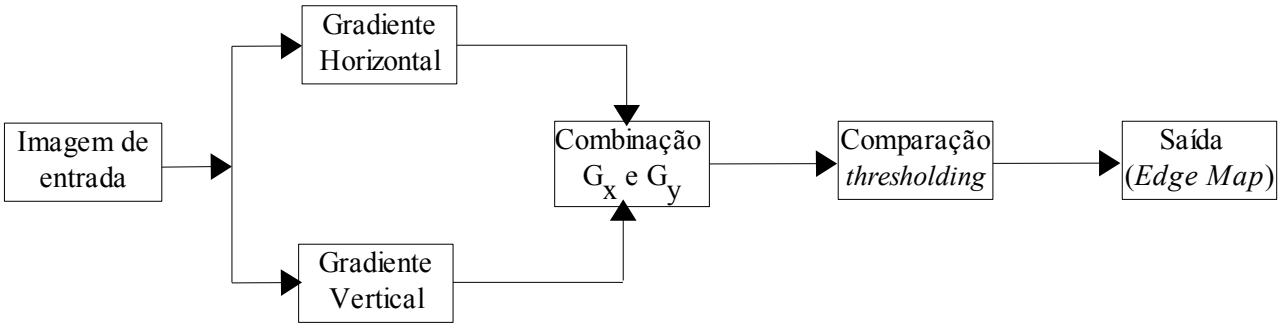


Fig. 3.8 – Fluxograma da aplicação dos conceitos de G_x e G_y.

A combinação de G_x e G_y para se detectar uma descontinuidade na função *f(x, y)* é frequentemente referenciada na literatura através do conceito de magnitude. Também representada por Δf, a magnitude combina os gradientes, nas direções horizontal e vertical, tornando possível a detecção de bordas em ambas as direções (5).

$$\triangle f = mag(\triangle f) = \left[G_x^2 + G_y^2 \right]^{\left(\frac{1}{2}\right)} \tag{5}$$

Em geral, a implementação dessas equações não é trivial. O que se observa na prática é uma combinação do resultado dos esforços na análise teórica e muitos refinamentos práticos para se alcançar o efeito desejado. Desta forma, a equação (5) pode ser aproximada por $\Delta f \approx | G_x | + | G_y |$, com a intenção de ter sua implementação bastante simplificada. Para uma máscara 3 X 3 [c_1 c_2 c_3; c_4 c_5 c_6; c_7 c_8 c_9] a derivação poderia ser implementada como se verifica em (6).

$$\triangle f \approx \left|(c_7 + c_8 + c_9) - (c_1 + c_2 + c_3)\right| + \left|(c_3 + c_6 + c_9) - (c_1 + c_4 + c_7)\right| \tag{6}$$

Onde G_x é representado pela diferença entre a terceira e a primeira linha, aproximando a derivada na direção do eixo *x*, e G_y, pela diferença entre a terceira e a primeira coluna, aproximando a derivada na direção do eixo *y* (GONZALES; WOODS, 2000). As máscaras utilizadas para computar G_x e G_y da forma acima (6) são chamadas de operadores de Prewitt e são ilustradas, respectivamente, pelas Fig. 3.9(a) e Fig. 3.9(b). Para realizar a detecção de bordas utilizando estes operadores, aplicam-se as duas máscaras separadamente sobre a imagem original e, posteriormente, sobrepõem-se os dois resultados.

-1	0	1
-1	0	1
-1	0	1

(a)

-1	-1	-1
0	0	0
1	1	1

(b)

Fig. 3.9 - Operadores de Prewitt.

De uma forma geral, a implementação da derivação é feita através de máscaras 3 X 3 com a fórmula da Fig. 3.10 (ZIOU; TABBONE, 1998). No caso do operador de Sobel, outro

operador muito conhecido na literatura também por utilizar o conceito de gradiente da função, o valor típico de *a* é igual a 2. Segundo Gonzales e Woods (2000), esta diferença em relação ao operador de Prewitt tem a maior capacidade de suavizar eventuais ruídos ao mesmo tempo em que detecta as bordas da imagem. Embora implementado por uma máscara 2 X 2, o operador de Robert é um exemplo de um operador *4-neighbourhood* também muito utilizado para derivação (CHAUDHURI; CHANDA, 1984) (Fig. 3.11). A Fig. 3.12 ilustra a aplicação dos três operadores citados.

$$\triangle x = \begin{bmatrix} -1 & 0 & 1 \\ -a & 0 & a \\ -1 & 0 & 1 \end{bmatrix} \qquad \triangle y = \begin{bmatrix} -1 & -a & -1 \\ 0 & 0 & 0 \\ 1 & a & 1 \end{bmatrix}$$

(a) (b)

Fig. 3.10 – Máscaras para operação do gradiente no eixo *x* (a) e no eixo *y* (b).

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

(a) (b)

Fig. 3. 11 – Operadores de Robert.

Existem, na literatura, muitos refinamentos e adaptações da aplicação destes operadores considerados como o fundamento dos algoritmos para detecção de bordas. Alguns deles também comparam o valor da magnitude do gradiente com um valor de *thresholding* (Fig. 3.8) para precisar a localização do pixel de borda na imagem; outros, analisam a função de derivação para identificação dos pontos de máximo positivo e de mínimo negativo (ZIOU; TABBONE, 1998). No mesmo trabalho, os autores mostram que a maior parte destes algoritmos realizam, inicialmente, uma operação de suavização sobre as imagens. Este procedimento serve para a redução do ruído da imagem original, mas tem a desvantagem de poder causar a perda de pixels relevantes para a correta identificação das bordas. Um bom procedimento de suavização é aquele que produz uma relação aceitável entre a eliminação do ruído e a preservação da estrutura da imagem.

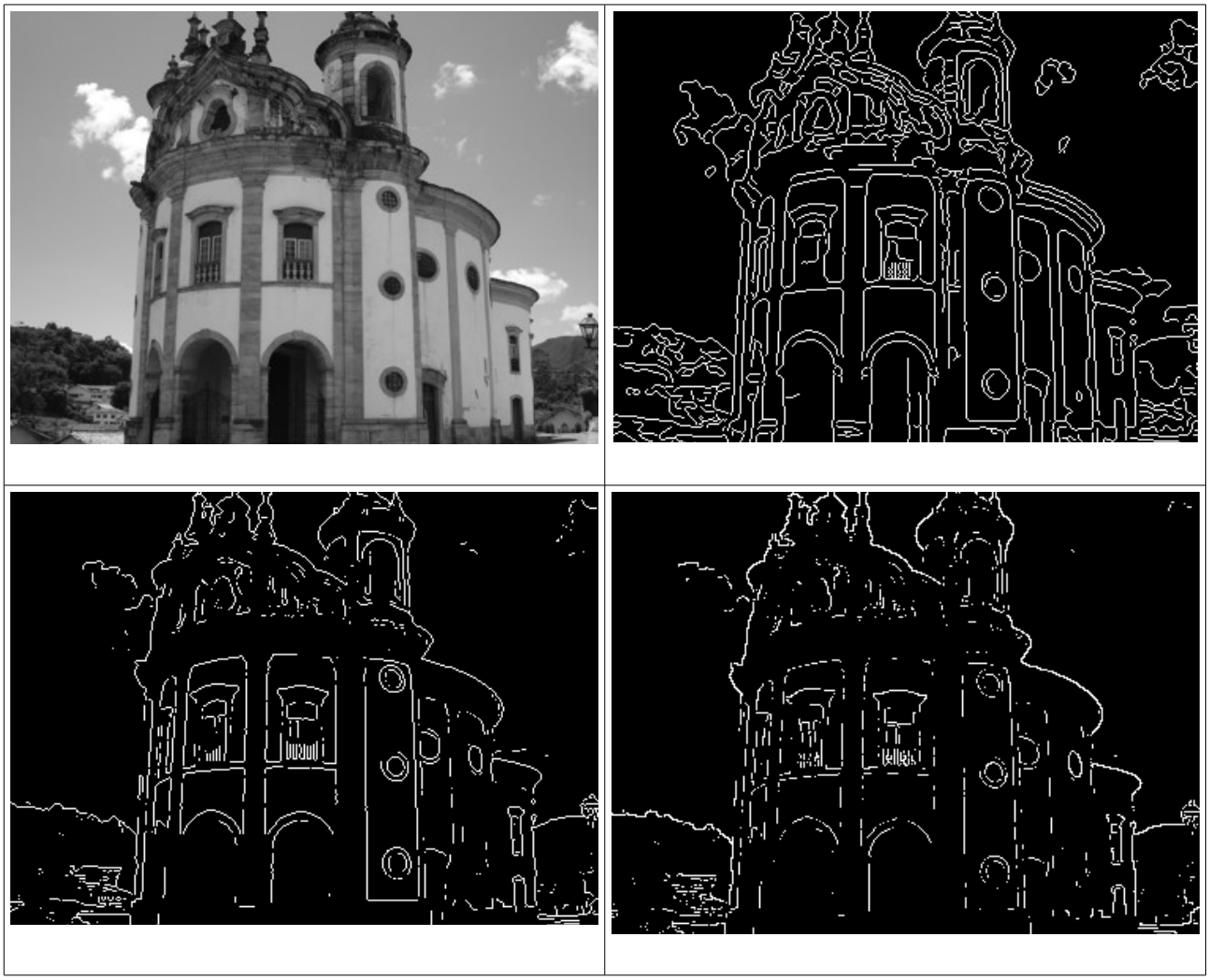


Fig. 3.12 – Aplicação de algoritmos de detecção de bordas: (a) Imagem original; (b) Operador de Prewitt; (c) Operador de Sobel; (d) Operador de Robert.

Outras técnicas utilizam a segunda derivada para a detecção de bordas em uma imagem. O conceito se baseia no fato de que, em regiões de elevada taxa de mudança no nível de cinza, a segunda derivada apresenta sinal diferente da primeira derivada. A forma mais aproximada de se implementar a derivação de segunda ordem é através do uso do Laplaciano definido pela equação (7). Então, os algoritmos para detecção de bordas que utilizam o Laplaciano varrem a imagem à procura de pontos em que a segunda derivada cruza o eixo *x* da função, chamados de pontos *zero-crossing*, e detectam bordas tanto no sentido horizontal quanto vertical (VINCENT; FOLORUNSO, 2009).

$$\triangle^2 f = (\partial^2 f / \partial x^2) + (\partial^2 f / \partial y^2) \tag{7}$$

As formas mais encontradas na prática para se implementar computacionalmente a

equação anterior são dadas por $\Delta^2 f \approx 4c_5 - (c_2 + c_4 + c_6 + c_8)$ (GONZALES; WOODS, 2000) ou $\Delta^2 f \approx c_2 + c_4 + c_6 + c_8 - 4c_5$ (ZIOU; TABBONE, 1998), também considerando a máscara como [c_1 c_2 c_3; c_4 c_5 c_6; c_7 c_8 c_9], por motivo de simplificação. As Fig. 3.13(a) e Fig.3.13(b) ilustram, respectivamente, as fórmulas citadas acima.

0	-1	0
-1	4	-1
0	-1	0

(a)

0	1	0
1	-4	1
0	1	0

(b)

Fig. 3.13 - Máscaras para implementação do Laplaciano.

Outra implementação bastante simples compara os sinais da primeira e segunda derivadas de três pixels consecutivos. Se uma mudança de sinal é verificada, neste ponto existe um *zero-crossing*.

Em aplicações práticas com imagens do mundo real, as características de interesse dos objetos podem ocorrer em uma infinidade de escalas e orientações, por isto, um único operador pode não funcionar para todos os casos. O algoritmo Marr-Hildreth contorna este problema tratando separadamente diferentes escalas de variação e combinando as informações obtidas de forma a identificar as mudanças de intensidade de cinza que representem bordas em uma imagem (MARR; HILDRETH, 1980).

As taxas de variação de intensidades nas imagens ocorrem em dois domínios: frequência ($\nabla\omega$) e espaço (∇x). Além disso, funções gaussianas são mais adequadas para representar taxas de variação não lineares. Devido a essas considerações, o algoritmo Marr-Hildreth, inicialmente aplica uma função Gaussiana bi-dimensional à imagem *I* através de um operador de convolução (8).

$$G(x, y) * I(x, y) \tag{8}$$

Onde *G(x,y)* representa a função Gaussiana bi-dimensional e * o operador de convolução.

A idéia é decompor a imagem original em um conjunto de cópias, cada uma delas usando uma função Gaussiana com variância (σ) diferente e, depois, identificando a mudança de

intensidade em cada uma destas cópias ou canais. Considerando que a taxa de ruído não é a mesma em toda a imagem, esta decomposição produz o efeito de suavização por reduzir o número de escalas de variação sobre a função *I(x, y)*.

Após a operação de convolução, um operador de derivação de segunda ordem é aplicado com o objetivo de se detectarem os contornos da imagem. Toda vez que ocorrer uma mudança de intensidade na função *I(x, y),* a derivação de primeira ordem corresponderá a um pico no gráfico da função resultante; e a derivação de segunda ordem, a um cruzamento no eixo *x* do gráfico (*zero-crossing*), pois dois pixels vizinhos, e com significante taxa de variação, têm derivadas de sinal diferente. Então, a tarefa de detecção de bordas pode ser reduzida à procura por cruzamentos por zero na segunda derivada (D^2) da função *I(x, y)*, onde *I(x, y)* representa a intensidade de cinza na coordenada (*x, y*) (9).

$$f(x,y) = D^2[G(x,y) * I(x,y)] \tag{9}$$

O operador de derivação utilizado é também o Laplaciano (∇^2) e o método, então, se limita à identificação dos cruzamentos por zero que identificam as mudanças no nível de intensidade da imagem. Operadores lineares como o Laplaciano são associativos e comutativos em relação à convolução e, por isto, as fórmulas (9) e (10) produzem o mesmo resultado (ZIOU; TABBONE, 1998).

$$f(x,y) = D^2 G(x,y) * I(x,y) \tag{10}$$

Este procedimento é repetido isoladamente para cada canal utilizado. Se os cruzamentos por zero têm a mesma posição e orientação em cada canal com variância (σ) distinta, estas variações de intensidade podem ser identificadas como bordas. Entretanto, cruzamentos por zero com posição ou orientação distintas irão representar a existência de sombras. O mínimo de canais requeridos para o algoritmo são 2 (dois) e, desde que estes canais sejam razoavelmente separados no domínio da frequência e seus cruzamentos por zero coincidam, eles indicarão corretamente a presença de bordas na imagem.

De acordo com Gonzalez e Woods (2000), a orientação da primeira derivada e também do cruzamento por zero na segunda derivação é capaz de identificar a direção de mudança no nível de intensidade de uma borda como ilustrado pela Fig. 3.14. A Fig. 3.14(c) demonstra que a transição de uma região escura para um região clara produz valores positivos de derivadas com o

ponto de máximo correspondendo ao limite da transição, ou seja, à borda. Da mesma forma, a transição de uma região clara para uma região escura produz valores negativos de derivadas com o ponto de mínimo correspondendo à borda. A Fig. 3.14(d) ilustra o comportamento da primeira e segunda derivadas, nas mesmas situações descritas.

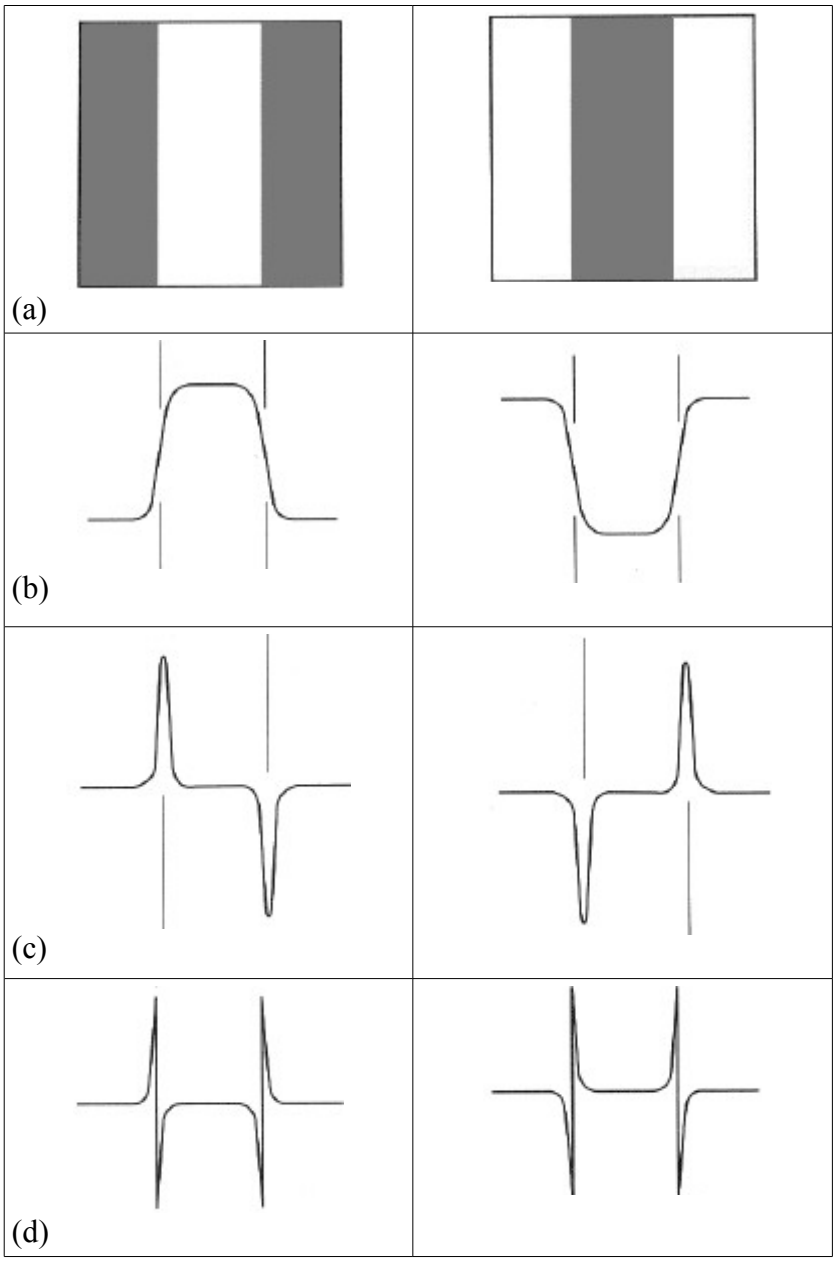


Fig. 3.14 – Perfil de derivação na mudança de gradiente: (a) Imagem; (b) Perfil da mudança de intensidade de uma linha vertical; (c) Primeira derivada; (d) Segunda derivada. Fonte: GONZALEZ e WOODS, 2000.

Resumindo, o método de detecção de bordas proposto por Marr e Hildreth primeiramente filtra a imagem através de ao menos dois canais $G(x,y) * I(x, y)$ independentes e,

após aplicar o Laplaciano, compara as imagens de saída procurando por segmentos *zero-crossing* coincidentes em todos os canais.

De uma forma geral, nenhum algoritmo detector de bordas resume características que o tornam eficiente para a resolução de qualquer problema em sistemas de processamento de imagens. As características geométricas dos objetos que se deseja extrair das imagens orientam a escolha do melhor método de detecção a ser utilizado na aplicação de interesse. O importante é que o método satisfaça aos critérios de baixa taxa de erro e precisa tradução da localização da borda na imagem para a *edge map*. Por uma taxa de erro nula entende-se que ruídos e outros fenômenos de iluminação adversos não serão confundidos com verdadeiras *edges* e que todas as *edges* verdadeiras serão detectadas. Em outras palavras, não serão detectadas bordas falso positivas ou falso negativas. Também é importante que uma mesma borda não produza múltiplos sinais de saída. Canny (1986) faz uma análise matemática detalhada destes critérios.

Em geral, os algoritmos para detecção de bordas funcionam bem, mesmo em condições diversas de iluminação. Isto se explica pelo fato de que, geralmente, a taxa de variação nos contornos dos objetos é muito maior que a taxa de variação causada pelos efeitos de iluminação (DAI et al., 1996). Entretanto, mudanças graduais no nível de intensidade de cinza entre duas regiões tornam as bordas mais difíceis de serem detectadas.

3.2 Segmentação por Detecção de Similaridades

Inversamente, alguns métodos de segmentação avaliam a igualdade entre pixels ou segmentos. É o caso da limiarização por histograma, das técnicas de divisão e fusão de regiões e dos algoritmos *watershed*. Estes métodos procuram reunir pixels em uma imagem de acordo com algum critério de similaridade ou homogeneidade. Entretanto, apesar destas técnicas compartilharem o conceito essencial de homogeneidade, elas diferem no modo como realizam o processo de segmentação (MUÑOZ et al., 2003).

De modo semelhante a um processo de classificação, as técnicas de segmentação por detecção de similaridades procuram definir se um pixel pertence a uma determinada região ou a outra. Geralmente, um valor limiar (*thresholding*), ou semente, irá definir a qual região o pixel pertence, se ele se agrega a um dado segmento ou se junta a outro. Portanto, o resultado da

segmentação depende de uma boa escolha desses valores. Existem diversos métodos para a escolha de um valor de limiar ótimo de forma automática. Uma relação destes trabalhos pode ser encontrada em Carvalho (2006).

Por ser adequada para resolver problemas que envolvem diversas informações qualitativas, a lógica difusa (*fuzzy*) pode ser uma alternativa para avaliar os critérios de homogeneidade em imagens digitais. Existem na literatura alguns estudos que utilizam a lógica difusa com o objetivo de segmentação (MANSOOR et al., 2007).

3.2.1 Limiarização por Histograma

O histograma de uma imagem revela a distribuição dos níveis de cinza ou a distribuição de cada componente de cor para imagens coloridas (como em sistemas RGB), e produz informação útil para fazer realce e análise de imagens (CARVALHO, 2006). É representado por um gráfico do tipo nº de pixels *versus* intensidade de cor, onde os componentes de cor de maior frequência na imagem são representados graficamente por picos; e os componentes de cor de menor prevalência, por vales.

O método de limiarização por histograma é o processo de divisão de uma imagem em diferentes regiões, baseado na distribuição de seu histograma (GONZALES; WOODS, 2000). O número de regiões depende do número de picos presentes na distribuição. Este método pode, ainda, ser utilizado nos procedimentos de classificação e melhoramento de imagens. Um método também baseado no histograma da imagem foi utilizado neste trabalho para a identificação de olhos abertos ou olhos fechados após o procedimento de segmentação.

Devido à baixa complexidade, a limiarização por histogramas é largamente utilizada. Entretanto, histogramas perdem qualquer informação espacial sobre as imagens, além de serem sensíveis a ruídos (NORIEGA; BASCLE; BERNIER, 2006).

Em imagens que apresentam apenas duas regiões bem definidas de níveis de cinza o histograma é bimodal, ou seja, tem a predominância de apenas dois tons, um dado pelo objeto e o outro, pelo fundo da cena (Fig. 3.15). Para a segmentação dessas imagens, o método de limiarização por histograma é o mais simples e menos complexo a ser utilizado. Basicamente, o problema consiste em decidir se um determinado pixel pertence a um objeto ou ao plano de fundo.

Selecionando-se um valor de limiar que esteja entre os dois tons estatisticamente predominantes, a imagem *I* , de resolução *M X N*, é percorrida de *I (0, 0)* a *I (M, N),* tendo seus valores modificados conforme seus níveis iniciais sejam maiores ou menores que o limiar *t* (11).

$$\begin{aligned} &\textit{if } I(x,y) >= t \\ &\qquad S(x,y) = 1 \\ &\textit{else} \\ &\qquad S(x,y) = 0; \end{aligned} \tag{11}$$

Onde *t* é o limiar selecionado e *S(x, y)* a imagem após a limiarização. Se a imagem contiver um objeto claro sobre um fundo escuro, a imagem limiarizada será uma imagem binária com um objeto branco sobre um fundo preto. Esta técnica é também utilizada na conversão de imagens em tons de cinza para imagens binárias.

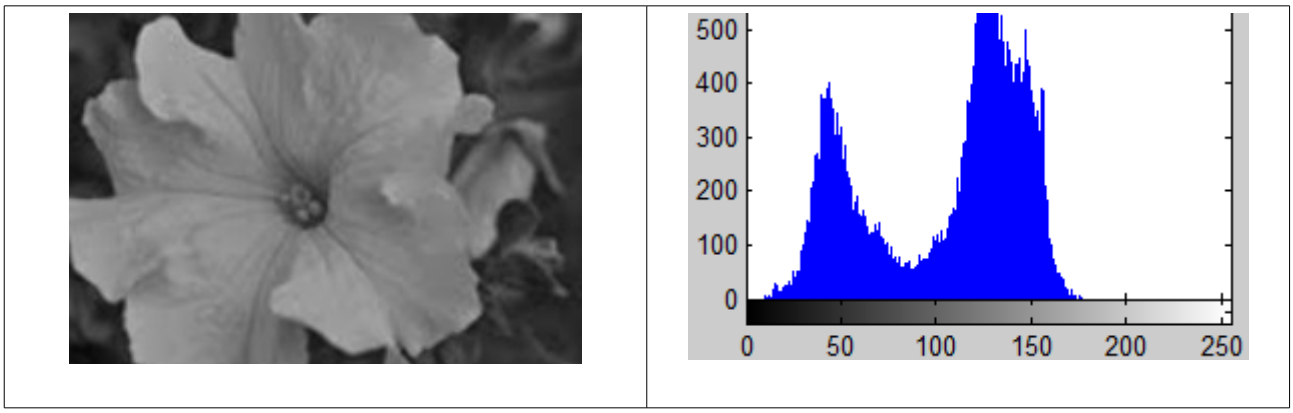


Fig. 3.15 – Exemplo de histograma bimodal.

A maioria das imagens do mundo real, entretanto, apresentam uma maior variação entre os níveis de intensidade de seus pixels e, portanto, seus histogramas são multimodais (Fig. 3.16). Este fato aumenta a complexidade da segmentação por limiarização, uma vez que, se um único valor de *thresholding* for utilizado, muita informação será perdida durante o procedimento.

Nesses casos, utilizam-se múltiplos valores de *thresholding* para a limiarização. Cada região é rotulada com um determinado nível de cinza e o resultado, portanto, não é uma imagem binária como na limiarização bimodal. Nain et al. (2008) apresentam um método que calcula o número ótimo de valores de *thresholding* baseado no número de significantes picos presentes no histograma de uma imagem. O número de regiões em que a imagem será dividida depende então do

número de picos presentes em seu histograma. E cada região é descrita pelo menor e maior valor de seu nível de intensidade. Rao e Prasad (1995) descrevem uma técnica de limiarização de baixa complexidade e elevada flexibilidade que pode ser usada tanto para imagens com histogramas bimodais quanto para imagens com qualquer cardinalidade de picos.

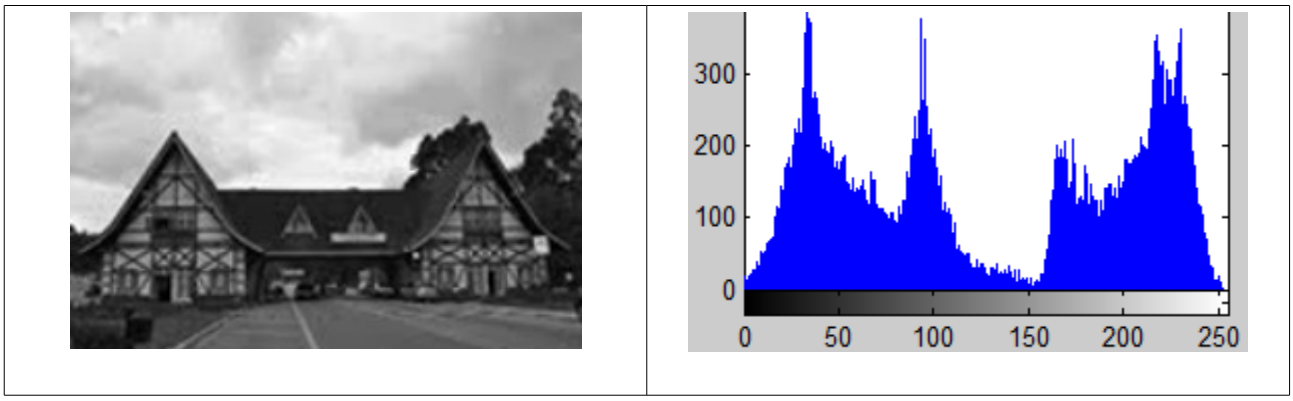


Fig. 3.16 – Exemplo de histograma multimodal.

Existem outras técnicas de limiarização que contornam um pouco das limitações da técnica de limiarização simples. Contudo, a complexidade destes algoritmos é bem maior. Um exemplo é o método de limiarização automática citado por Carvalho (2006), em que o limiar é definido automaticamente por meio de funções que buscam maximizar o critério da separabilidade para a definição de limiares ótimos. Além disto, a limiarização é frequentemente utilizada em associação com outras técnicas. Usualmente, a maioria dos detectores de bordas aplicam um limiar à imagem de saída para decidir quais *edges* são significantes. Se uma borda apresenta níveis de intensidade superiores ao limiar, ela é definitivamente marcada na imagem de saída (*edge map*). Esta associação procura minimizar o efeito de ruídos sobre a imagem segmentada.

Outro aperfeiçoamento do método de limiarização por histograma se relaciona ao conceito de *thresholding* global, local ou dinâmico. As técnicas citadas anteriormente lidam com *thresholding* global, ou seja, um mesmo valor de limiar é aplicado a toda imagem. Nas técnicas que utilizam *thresholding* local, a imagem é primeiramente subdividida em regiões ou segmentos e, então, um valor de *thresholding* é escolhido para cada região. Já nas técnicas que utilizam *thresholding* dinâmico, um valor diferente de limiar pode ser determinado para cada coordenada da imagem. Técnicas de *thresholding* dinâmico costumam ser mais eficientes para a limiarização de imagens (GAMBA, LODOLA; MECOCCI, 1997), sobretudo, aquelas sob condições diversas de iluminação.

Dai et al. (1996) apresentam uma técnica de *thresholding* dinâmico que determina valores limiares iniciais a partir de uma *edge map* da imagem de entrada. Considerando que os pixels pertencentes às bordas dos objetos representam o limite entre objetos e o plano de fundo, o seu valor de intensidade de cinza pode ser considerado como um valor de *thresholding* ótimo para estes mesmos pixels. Terminada esta etapa, os valores de limiar são propagados para as regiões vizinhas através de uma operação morfológica de dilatação (ver seção 4.3.1). Esta operação é repetida iterativamente até que a *thresholding map* esteja completa, ou seja, até que todos os pixels da imagem sejam rotulados com um valor de limiar. Nesse momento, a imagem pode ser limiarizada de acordo com (12).

$$S(x,y)=\begin{cases}1\ se\ I(x,y)>t(x,y)\\0\ se\ I(x,y)\leq t(x,y)\end{cases} \tag{12}$$

Onde *I (x, y)* representa o nível de intensidade de cinza na imagem *I* no ponto *(x, y)*; *t (x, y)* o limiar na *thresholding map* para a mesma coordenada (*x, y*) e S *(x, y)*, a imagem limiarizada.

3.2.2 Divisão e Fusão de Regiões (S*plit-and-Merge*)

As técnicas de segmentação que usam divisão e fusão de regiões ou, simplesmente, crescimento de regiões, procuram dividir uma imagem em regiões conexas e disjuntas sob a influência de algum critério de homogeneidade. Uma destas técnicas escolhe, aleatoriamente, alguns pixels como semente e faz crescer as regiões a partir destes pontos iniciais, agregando pixels vizinhos e similares. Então, selecionados os pixels sementes e as propriedades ou critérios que identificarão um pixel como pertencente a uma ou outra região, a imagem é varrida a partir das sementes, incrementando o tamanho da região a cada iteração do procedimento. Algoritmos que escolhem os pixels sementes de maneira não aleatória, geralmente produzem melhores resultados. As técnicas que no final do procedimento fundem as regiões conectadas e similares em algum critério também são mais eficientes.

Karatzas e Antonacopoulos (2004) utilizam um procedimento de segmentação do tipo divisão e fusão de regiões em imagens coloridas. No estágio de divisão, a imagem é dividida em

camadas de acordo com seu histograma no espaço de cor HLS. No estágio da fusão, as regiões das camadas inferiores se agregam de acordo com a similaridade das regiões vizinhas e o resultado é copiado para a camada imediatamente acima. O processo continua até atingir novamente a primeira camada.

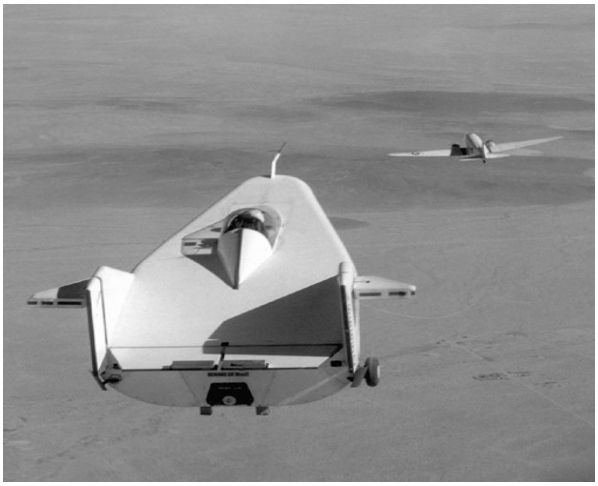
Outra técnica subdivide uma imagem em quadrantes na tentativa de estabelecer as regiões que satisfaçam a uma determinada condição ou propriedade e, depois, funde as regiões adjacentes que possuam a mesma propriedade. Toda vez que a propriedade *P* não é satisfeita em um determinado quadrante, este é novamente subdividido em quatro partes até que *P* seja satisfeita. Geralmente, a propriedade analisada é a homogeneidade. Terminada a etapa de subdivisões sucessivas, as regiões adjacentes, com propriedades idênticas, são fundidas. Esta técnica é convenientemente representada por uma *quadtree*, uma árvore em que cada nó tem exatamente quatro filhos. A imagem original representa a raiz e, a cada iteração do procedimento, quatro novos nodos (ou regiões) são acrescentados à *quadtree* (Fig. 3.17). Esta técnica é também muito utilizada para algoritmos de compressão de imagens.

A técnica de baixa complexidade para limiarização através de múltiplos valores de *thresholding* descrita por Rao e Prasad (1995), e citada na seção anterior, é recomendada pelos autores para a fase de divisão das imagens para se aumentar o desempenho de algoritmos do tipo divisão e fusão de regiões.

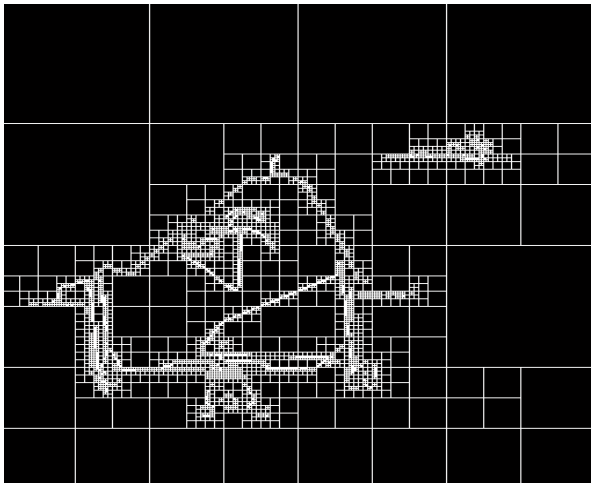
Em geral, os limites das regiões segmentadas por técnicas de divisão e fusão de regiões não coincidem com os limites dos objetos nas imagens, resultando em erro de segmentação. Além disso, pontos iniciais de crescimento (sementes) e adequados critérios de parada são difíceis de ser definidos sem qualquer outro tipo de informação (MUÑOZ et al., 2003). Ainda há o fato de que a maioria das abordagens para segmentação são testadas em ambientes com características bem definidas e são menos úteis para a interpretação de imagens complexas do mundo real (GAMBA; LODOLA; MECOCCI, 1997). Para minimizar estes erros de segmentação e, consequentemente, aumentar a confiabilidade no processamento de qualquer tipo de imagem, alguns autores usam métodos que integram técnicas de detecção de bordas com técnicas de crescimento de regiões, durante a segmentação.

Muñoz et al. (2003) relacionam diversas dessas técnicas classificando-as em duas categorias de acordo com o momento em que a integração ocorre. Na integração embutida, as bordas são extraídas primeiramente e esta informação é usada para definir novos critérios de

decisão para os algoritmos de crescimento de regiões, como a localização de sementes ou um novo critério de parada ou de controle do crescimento. Por exemplo, um critério de homogeneidade ou propriedade analisada para o crescimento de regiões poderia ser a ausência de pixels de bordas na região em crescimento. Se o próximo pixel a ser agregado na iteração coincide com um pixel de contorno, significa que a região chegou ao seu limite e o crescimento deve parar. Este método compara a consistência da região em crescimento com a *edge map*. A integração pós-processamento é realizada após ambas as técnicas de segmentação terem processado a imagem paralelamente. Um procedimento final de fusão é utilizado para integrar os resultados produzidos por cada um dos métodos testando a consistência dos dois resultados e produzindo uma nova imagem de saída mais precisa e confiável.



(a)



(b)

Fig. 3.17 – Segmentação por divisão e fusão de regiões: (a) Imagem original; (b) Imagem segmentada.

Fonte: MATLAB, 2008.

É muito difícil obter cadeias de bordas fechadas. Frequentemente, os contornos dos objetos são representados por linhas quebradas e incompletas. Para Gamba, Lodola e Mecocci (1997), se regiões são consideradas isoladamente torna-se difícil a interpretação da cena, pois o resultado final é um cenário completamente fragmentado. *Edge maps* geralmente transportam mais informações, pois mesmo com bordas incompletas, tem-se uma idéia mais precisa do cenário (Fig. 3.18). Além disto, observadores humanos são capazes de reconstruir um cenário a partir destas bordas fragmentadas e, assim, reconhecer objetos. Os autores combinam as duas técnicas, detecção de bordas e crescimento de regiões, superpondo a *edge map* da imagem original à imagem segmentada por divisão e fusão de regiões. Com esta associação consegue-se obter resultados mais aprimorados na segmentação.

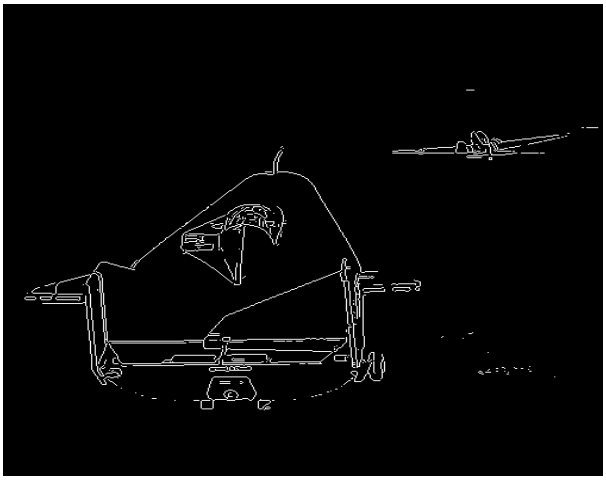


Fig. 3.18 – Imagem da Fig. 3.17(a) após detecção de bordas.

3.2.2.1 Algoritmos *Watershed*

Uma classe de algoritmos de detecção de similaridades por crescimento de regiões são os chamados *watersheds*. Existem várias implementações deste procedimento descritas na literatura. Para se compreender melhor tais abordagens é preciso comparar as imagens a superfícies topográficas onde o valor da intensidade de cinza no ponto *p(x, y)* corresponde à sua altitude na imagem vista como um relevo. Assim, regiões mais claras na imagem correspondem a regiões de elevada altitude, ou picos, e regiões mais escuras correspondem a baixas altitudes, ou vales. O crescimento das regiões é realizado a partir da simulação de um procedimento de inundação.

O primeiro passo é definir os pontos de mínimo local. Um pixel *p* é dito ser um ponto de mínimo local se a altitude de *p* é menor que a altitude de todos os seus circunvizinhos, ou seja, $p(altitude) < p_{n=1...8}(altitude)$, onde p_n representa a vizinhança de *p* (Fig. 3.19).

p_1	p_8	p_7
p_2	p	p_6
p_3	p_4	p_5

Fig. 3.19 - Conceito de vizinhança.

Selecionados os pontos de mínimo, é preciso imaginar a imagem como uma maquete 3D na forma de um relevo onde os estes pontos mínimos apresentam furos. A imagem é então lentamente submersa em água. A água penetraria por esses furos, inundaria primeiramente regiões de vales e, gradualmente, avançaria para regiões de maiores altitudes, formando diversos lagos. Nos pontos onde lagos oriundos da inundação de vales diferentes se encontrariam, represas seriam construídas prevenindo a fusão dos lagos. O processo de segmentação se completaria quando toda a superfície do relevo fosse coberta por água. Imaginado o comportamento de um algoritmo *watershed*, cada lago representa uma região na imagem e as represas, ou linhas divisórias de água, os limites ou bordas entre os segmentos. Nos procedimentos para *watersheds*, cada segmento é sempre separado do outro, por um contorno fechado.

Esses algoritmos produzem imagens simbólicas representativas das imagens de entrada semelhantes a mosaicos ou mapas (Fig. 3.20). Cada região no mapa é homogênea em seu nível de intensidade de cor, pois cresce a partir de um mesmo ponto de mínimo. Entretanto, o algoritmo deve preservar o contraste entre as regiões. Segundo Najman, Couprie e Bertrand (2005), quando dois mínimos são separados por uma "elevação" na imagem original, esses pontos devem continuar separados por uma "elevação" de mesma altitude no mosaico. Uma definição formal da preservação de contraste é dada em Najman e Couprie (2003). Essa definição lida com o conceito de *pass value*, que é o menor caminho ou menor altitude que separa dois mínimos. Bertrand (2004) define algumas outras propriedades formais para *watersheds* de imagens em tons de cinza.

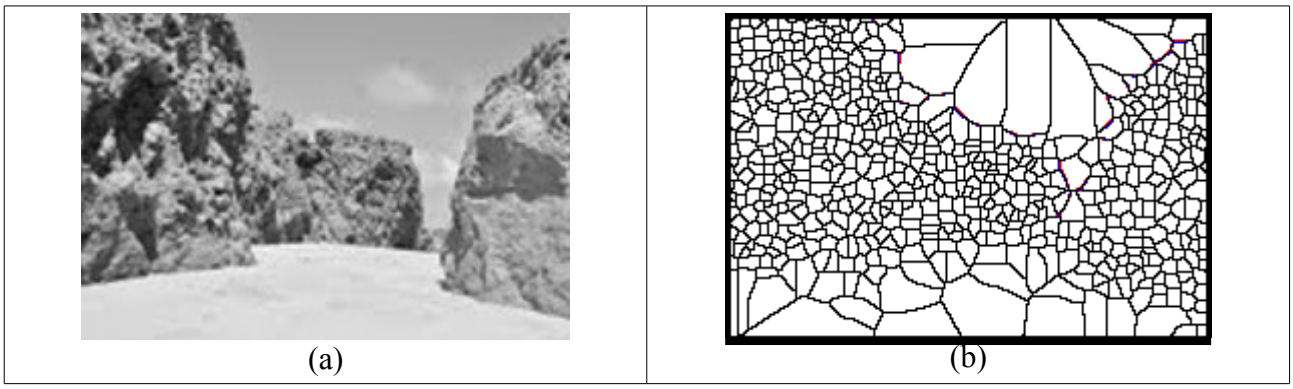


Fig. 3.20 – *Watershed*: (a) Imagem original; (b) W*atershed*.

Najman, Couprie e Bertrand (2005) introduzem também o conceito formal de emergência, que é um paradigma de inundação inverso ao dos algoritmos *watersheds* tradicionais. Neste novo paradigma, os pixels são processados em ordem decrescente de altitude, como uma inundação da superfície que ocorre a partir de gotas d'água que caem sobre o relevo e realizam um caminho descendente até alcançarem um ponto de mínimo (WRIGHT; ACTON, 1997). Entretanto, a implementação desse novo conceito é mais difícil de ser alcançada na prática (NAJMAN; COUPRIE, 2003). A complexidade do método está em se definir a zona de influência de cada ponto de mínimo ou o domínio de atração da água. Gee e Abidi (1995) relacionam algumas situações onde a zona de influência de cada mínimo local pode alterar a complexidade deste modelo. A Fig. 3.21(a) mostra um ponto minimal que exerce atração total sobre a gota d'água, e as Figs. 3.21(b) e 3.21(c) mostram duas situações em que dois pontos mínimos exercem atração sobre uma mesma gota d'água. Nesses casos, devem ser formalizadas regras para se definir o caminho que a água irá seguir.

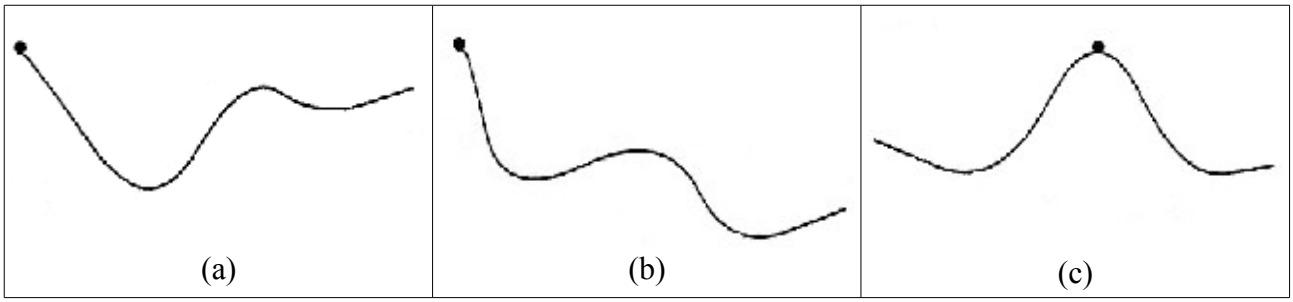


Fig. 3.21 – Domínio de atração da água em algoritmo *watershed*:: (a) minimal; (b) mínimo regional; (c) múltiplos mínimos. Fonte: GEE e ABIDI, 1995.

Os algoritmos *watersheds*, entretanto, produzem contornos que não são necessariamente os mais significantes contornos da imagem original (NAJMAN; COUPRIE, 2003). Este fato, aliado

à idéia de que o número de regiões encontradas é igual ao número de pontos de regional mínimo existentes na imagem, provoca um fenômeno conhecido como sobre-segmentação (*over-segmentation*) (Fig. 3.20). É preciso empregar alguma técnica de fusão e crescimento de regiões para agrupar segmentos que satisfaçam a algum critério de homogeneidade e, assim, reduzir o problema da sobre-segmentação.

Para Felzenszwalb e Huttenlocher (1998), uma sobre-segmentação ocorre quando existe algum par de regiões vizinhas na imagem segmentada onde a variação entre as regiões é relativamente menor que a variação dentro de cada uma delas. Dessa forma, estas regiões poderiam ser fundidas em um único segmento. Uma sub-segmentação, ao contrário, ocorre quando existe uma maneira de dividir alguns segmentos em sub-regiões de forma que o resultado da segmentação não produza uma imagem sobre-segmentada.

A técnica de *watershed* também possui a sua versão morfológica que será discutida na seção 4.4.2. Vincent e Soille (1991) relatam alguns procedimentos *watersheds* baseados na morfologia matemática e nos paradigmas de inundação e emergência e comparam a eficiência e precisão desses a um novo algoritmo projetado pelos próprios autores.

3.3 Segmentação por Detecção de Movimento

Esta técnica usa a variação da posição dos objetos entre dois quadros de uma mesma imagem tomados em instantes distintos. Uma maneira simples de se fazer isso é realizar a subtração entre os dois quadros. Entretanto, uma imagem de diferença frequentemente contém pontos isolados relativos a ruídos, e não a movimentos de objetos. A influência desses pontos de ruídos pode ser minimizada pela remoção de elementos com conectividade menor que quatro ou oito, por exemplo. Um problema dessa abordagem é que pequenos objetos em movimento podem ser eliminados. Outra maneira de minimizar a ocorrência desses pequenos ruídos é comparar um número maior de quadros para observar se a mudança no pixel ocorreu apenas em alguns quadros, o que denotaria um ruído, ou se em grande parte deles, o que indicaria o movimento de algum objeto (GONZALES; WOODS, 2000).

Horpraset, Harwood e Davis (2000) apresentam um algoritmo de subtração de fundo para a detecção de objetos em movimento capaz de tratar as variações de iluminação em um fundo

estático. A idéia é separar os componentes de brilho e cromaticidade no espaço de cor RGB, uma vez que as propriedades espectrais das cores não mudam, mesmo que sob diferentes condições de iluminação.

Huang, Paulus e Niemann (2000) propuseram um método que combina duas técnicas de segmentação; uma estática, baseada no algoritmo *watershed,* e outra, baseada em um modelo de movimento dominante. O algoritmo *watershed* é usado para subdividir a imagem em pequenas regiões. Em seguida é utilizado um método para se determinar o movimento em cada uma das regiões segmentadas.

4 Morfologia Matemática

A morfologia matemática é um conjunto de teorias e técnicas que usa conceitos de álgebra (teoria dos conjuntos, lattices, álgebra de Boole) e de geometria (translação, reflexão, espaço euclidiano, convexidade) para estudar a estrutura geométrica de partículas. Foi desenvolvida, inicialmente, em trabalhos de mineralogia que estudavam propriedades físicas e mecânicas em rochas e minerais; e, hoje, é amplamente pesquisada e aplicada também para processamento e análise de imagens digitais.

Em processamento e análise de imagens é aplicada principalmente em imagens binárias, mas também é aplicável a imagens em intensidades de cinza e a imagens coloridas. Porém, há uma diferença de enfoque nestes dois últimos casos. O foco deste texto será sobre a aplicação da morfologia matemática sobre imagens binárias, uma vez que foi a técnica utilizada para o desenvolvimento do método proposto neste trabalho. A aplicação da morfologia matemática sobre imagens em tons de cinza será brevemente discutida apenas como caráter ilustrativo. As técnicas sobre imagens coloridas são pouco discutidas na literatura e serão apenas citadas.

Os operadores e filtros morfológicos têm-se mostrado eficientes para o uso em técnicas de segmentação de imagens (inclusive, detecção de bordas), remoção de ruídos, suavização, reconstrução e esqueletonização de imagens. No caso da segmentação, é especialmente útil quando se tem um conhecimento prévio do formato do objeto que se deseja extrair da cena.

O presente trabalho tem a intenção de introduzir os conceitos de morfologia matemática em imagens binárias, sem requerer nenhum conhecimento prévio na área para que o leitor possa alcançar um completo entendimento do método proposto.

4.1 Elemento Estruturante

Operações morfológicas fazem interagir a imagem original com uma imagem menor e com determinada forma que é conhecida como elemento estruturante (EE) ou estrutura morfológica. Segundo Burgeth et al. (2004), o uso de matrizes facilita a implementação e a representação de formas geométricas. Então, um elemento estruturante pode ser definido como uma matriz de 0's e 1's escolhida de maneira a representar o formato geométrico que se deseja aplicar à imagem.

O elemento estruturante percorre, pixel a pixel, a imagem original, em um movimento de translação, processando alguma operação morfológica de interesse sobre essa imagem. O elemento na matriz, definido como a origem do EE, corresponde ao ponto de sobreposição durante o deslocamento (translação) sobre a imagem. Deste modo, é muito importante definir a célula da matriz que irá corresponder à origem. Uma mesma estrutura, mas com célula de origem definida de maneira diferente, irá produzir um resultado de processamento também diferente. A forma mais comum é definir a origem do EE como o centro da matriz de 0's e 1's.

A função do elemento estruturante é definir o domínio de análise para cada pixel na imagem (COSTER; CHERMANT, 2001). Esse domínio, ou vizinhança, é definido pelas células de valor 1(um). A Fig. 4.1 representa um elemento estruturante com formato cruciforme. A célula (2,2) da matriz corresponde à origem (0, 0) do EE. Sendo assim, para essa estrutura, são analisados 4 pixels na vizinhança também identificada como *4-neighbourhood* ou N_4 : $(x - 1, y)$, $(x + 1, y)$, $(x, y - 1)$ e $(x, y + 1)$. Neste trabalho, por convenção, a origem do EE será sempre definida como o centro da matriz.

0	1	0
1	1	1
0	1	0

Fig.4.1 - Elemento estruturante com vizinhança de 4 pixels (*4-neighbourhood* ou N_4).

Marchand-Maillet e Sharaiha (2000) demonstram o uso de lattices triangulares, hexagonais e quadradas, ao invés de matrizes, para a definição de vizinhança (Fig. 4.2). Entretanto, lattices triangulares e hexagonais não representam satisfatoriamente a topologia digital da maioria das imagens. Lattices quadradas são mais utilizadas pela sua semelhança com matrizes. Lattices triangulares e hexagonais são mais úteis quando imagens são representadas por grafos. Nesta representação, cada nó no grafo corresponde a um pixel na imagem.

Elementos estruturantes podem ainda ser tri-dimensionais quando apresentam pesos associados às suas células. Esse tipo de estrutura é chamado de elemento estruturante volumétrico e não é usado para imagens binárias. Já o elemento estruturante bi-dimensional da Fig. 4.1 é chamado plano, para diferenciar do elemento estruturante volumétrico, e tanto pode ser usado para imagens binárias quanto para imagens em intensidades de cinza.

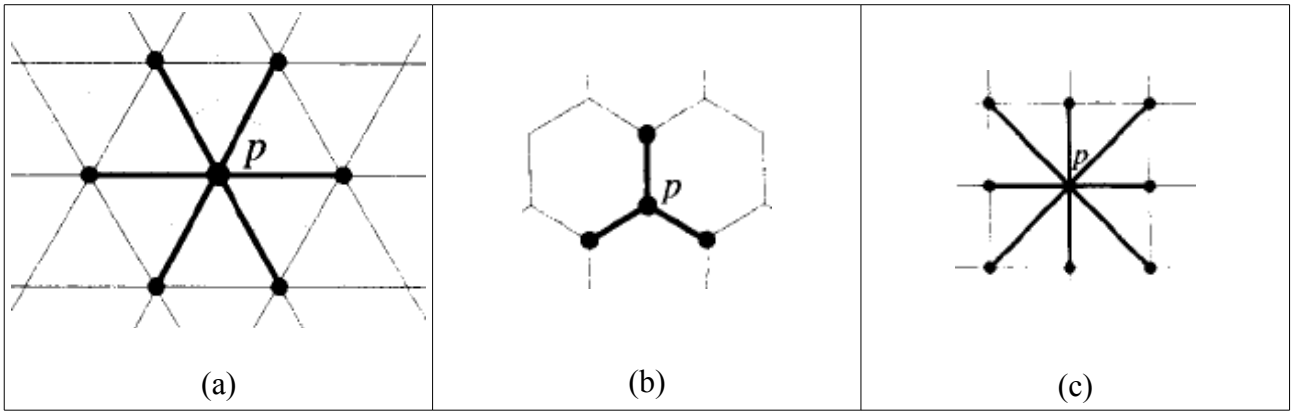


Fig. 4.2 - Vizinhança definida como lattices: (a) N_6 em lattice triangular; (b) N_3 em lattice hexagonal; (c) N_8 em lattice quadrada. Fonte: MARCHAND-MAILLET e SHARAIHA, 2000.

4.2 Componentes Conectados

Outro conceito importante para a compreensão da aplicação das técnicas de morfologia matemática sobre imagens binárias é o de componentes conectados (*connected components*). Toda imagem binária pode ser dividida em dois grandes conjuntos: o conjunto dos pixels pertencentes ao primeiro plano da imagem, e o dos pixels pertencentes ao plano de fundo (por convenção, os pixels de valor zero). Para uma imagem binária *Ib*, analisada como um conjunto *BW* de pixels, tem-se (13).

$$BW = \{ p \mid (p = 0 \ \ or \ \ p = 1) \ \forall p \in Ib \} \tag{13}$$

Dividindo o conjunto *BW* no conjunto *F*, que representa o plano dos objetos contidos na imagem (*foreground*) e no conjunto *B,* que representa o plano de fundo (*background*), obtém-se (14) e (15). Dessas formulações pode-se facilmente concluir que o conjunto *F* é o complemento de *B*, e vice-versa. Representaremos o conjunto complemento com a seguinte notação: $F = B^c$.

$$F = \{ p \mid \ p = 1 \ and \ p \in BW\} \tag{14}$$

$$B = \{ p \mid \ p = 0 \ and \ p \in BW\} \tag{15}$$

A imagem da Fig. 4.3 representa, matricialmente, uma imagem *Ib* e os conjuntos *B* e *F*. Observe que, neste caso, o conjunto *F* apresenta dois sub-conjuntos disjuntos. Cada um destes sub-conjuntos representa um componente conectado que é, então, definido como um conjunto de pixels no qual existe um caminho entre todos os seus elementos. Com este conceito é possível enumerar e diferenciar cada um dos objetos existentes no primeiro plano da imagem. A união de todos os objetos de uma imagem formam, então, o conjunto do primeiro plano (16).

$$F_1 \cup F_2 \cup F_n = F \tag{16}$$

Onde *n* representa o número de componentes conectados presentes no conjunto *F*. O plano de fundo também pode apresentar mais de um componente conectado. Entretanto, para a análise morfológica, interessa apenas os componentes conectados do primeiro plano da imagem.

0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0

Fig. 4.3 – Representação matricial de uma imagem *Ib*. Em branco, o conjunto *B* e, em cinza, o conjunto *F*.

Outra definição para componentes conectados leva em consideração a conectividade de cada pixel pertencente ao conjunto *F*. Segundo Gonzalez, Woods e Eddins (2004), *p* e *q* são ditos *4-connected* se existe um caminho *4-connected* entre eles ou, 8-*connected* se existe um caminho *8-connected* entre eles, consistindo inteiramente de pixels do conjunto *F*. Um caminho *4-connected* significa dizer que existe uma conexão entre *p(x, y)* e *q(x',y')* levando em consideração apenas as adjacências (*x – 1, y*), (*x + 1, y*), (*x, y – 1*) e (*x, y + 1*) de cada pixel pertencente ao caminho. Já em um caminho *8-connected,* os oito pixels imediatamente adjacentes são considerados. Observe na Fig. 4.4 como este conceito pode interferir na identificação dos objetos em uma cena. Na Fig. 4.4(a) foram identificados 2 componentes conectados; já na Fig. 4.4(b), foram identificados 4 componentes ou objetos. Neste trabalho, os termos objetos e componentes conectados serão

tratados, intercambiavelmente, como sinônimos, uma vez que irão se referir à mesma estrutura.

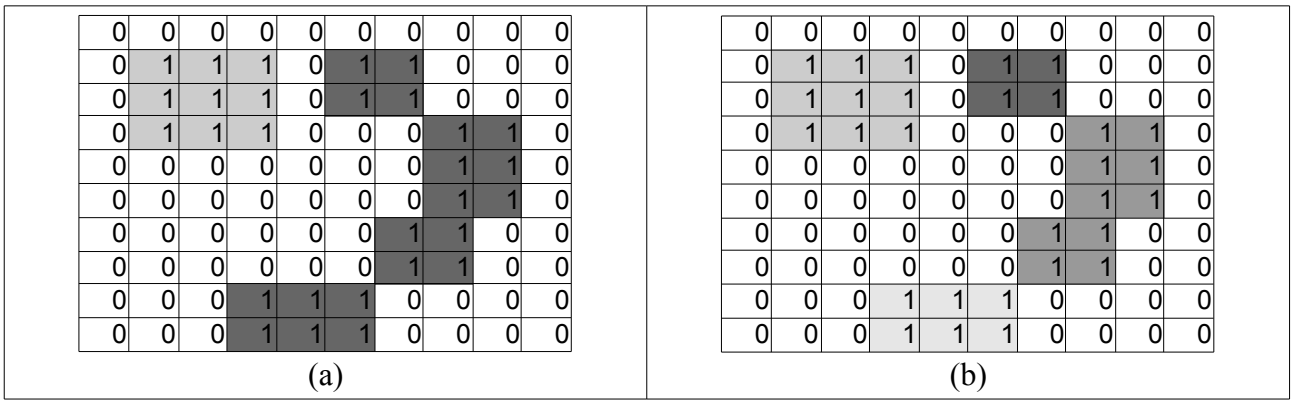


Fig. 4.4 – Componentes conectados de acordo com a conectividade entre os pixels: (a) utilizando um caminho *8-connected;* (b) utilizando um caminho *4-connected*.

Dividir uma imagem binária em seus componentes conectados é extremamente útil em morfologia matemática para se efetivar a busca por determinados critérios geométricos. Os componentes conectados podem ainda ser distinguiveis uns dos outros através de um procedimento de rotulação (*connected component labeling*). Cada pixel pertencente a um mesmo componente conectado recebe um identificador, ou rótulo, numérico que o diferencia dos demais pixels também pertencentes ao conjunto *F* (*foreground*), mas que pertençam a um componente conectado diferente. Desse modo, a imagem da Fig. 4.3, contendo dois objetos no primeiro plano, seria representada pela Fig. 4.5, onde os pixels rotulados com o valor 1 identificam o primeiro objeto; e os rotulados com o valor 2, o segundo objeto.

0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	0	2	2	2	0
0	0	0	0	2	2	2	0
0	0	0	0	2	2	2	0

Fig. 4.5 – Imagem *Ib* após rotulação.

De acordo com Marchand-Maillet e Sharaiha (2000), cada sub-conjunto é tratado como uma entidade separada dentro da imagem, o que facilita a aplicação dos estudos morfológicos. Os autores ainda descrevem duas abordagens para o procedimento de rotulação, uma baseada em

matrizes e outra em grafos. Os respectivos estudos de complexidade também são apresentados.

A divisão do conjunto F em *connected component labeling* permite ainda o cálculo de medidas de interesse para o processamento dos objetos analisados individualmente, como perímetro, área, centro de massa, raio, diâmetro, etc. Essas medidas podem ser muito úteis para diferenciar e distinguir os objetos uns dos outros ou para compará-los com algum outro objeto tomado como referência.

Basicamente, existem duas regras para qualquer procedimento de rotulação:

a) Label(p) = 0, se $p \in B$ (onde B representa o conjunto do plano de fundo);

b) Label(p) = Label(q), se $p \in F_n$ e $q \in F_m$, $\forall\ n = m$ (onde F representa o conjunto do plano dos objetos).

4.3 Operações em Morfologia Matemática

As operações morfológicas são procedimentos que varrem, sequencialmente, uma imagem aplicando, através de um elemento estruturante com forma e tamanho escolhidos de acordo com o objetivo da aplicação, algum tipo de transformação nos componentes conectados de uma imagem binária ou nos pixels componentes de uma imagem em intensidades de cinza. As operações de dilatação e erosão formam a base da morfologia matemática. A combinação destes operadores básicos permite a construção de operações mais complexas. A seguir, apresentamos as operações morfológicas mais comuns.

4.3.1 Dilatação ($\oplus$) e Erosão ($\ominus$)

Tipicamente, a operação de dilatação adiciona uma camada de pixels sobre as bordas de um objeto em uma imagem e a operação de erosão remove uma camada de pixels. O número de pixels adicionados ou removidos da borda do objeto depende do tamanho e formato do elemento estruturante. Como o resultado é sempre uma imagem do mesmo tamanho que a original, a dilatação dos componentes conectados do primeiro plano implica em uma consequente erosão do

plano de fundo e vice-versa.

Posicionado o centro, ou origem, de um elemento estruturante simétrico sobre cada pixel na imagem binária *I*, o valor do pixel na imagem de saída dilatada será 1 se qualquer um dos pixels de sua vizinhança for 1, e 0 se nenhum dos pixels de sua vizinhança for igual a 1 (Fig. 4.6). Em outras palavras, o resultado da dilatação de um objeto consiste na união de todas as células que, quando sobrepostas pela origem do elemento estruturante, têm em seu domínio de análise pelo menos uma célula pertencente ao objeto original. Na dilatação de imagens em tons de cinza usando-se elementos estruturantes planos, o valor do pixel de saída será o máximo valor de intensidade de cinza entre os pixels da vizinhança analisada. Note que caso de imagens binárias, este raciocínio pode ser comparado à operação OR para portas lógicas; e no caso de imagens em tons de cinza, à operação de união (∪) em lógica difusa.

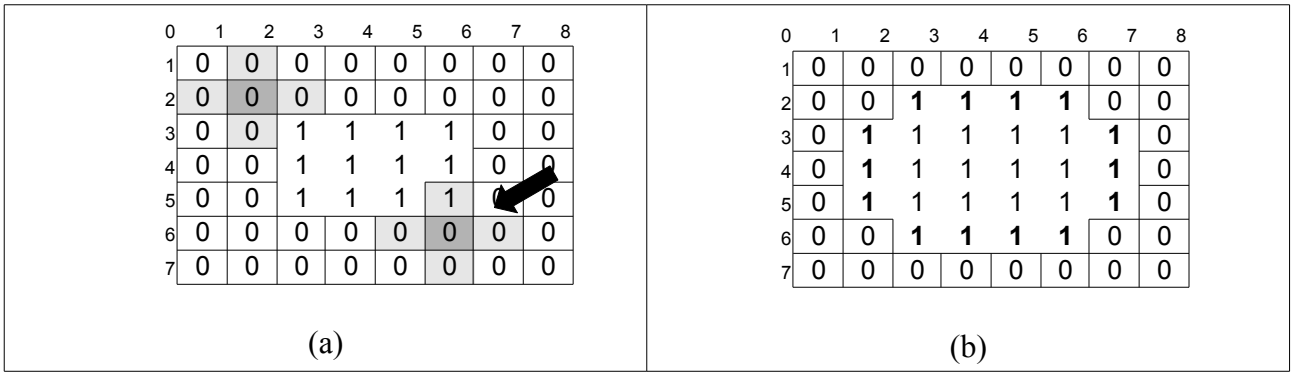


Fig. 4.6 – Exemplo de dilatação de um objeto retangular (3 X 4) em uma imagem (7 X 8) com um EE N_4: (a) imagem original mostrando duas translações do EE; (b) imagem dilatada (Id), pixels em negrito representando os pixels que foram acrescentados à imagem original. (Note que a translação do EE sobre a célula (2,2) resultará em Id (2,2) = 0 e que a translação do EE sobre a célula (6,6) resultará em Id (6,6) = 1).

Para a dilatação de imagens com elementos estruturantes assimétricos, o EE deve ser primeiramente refletido. A Fig. 4.7 mostra a dilatação da mesma imagem da Fig. 4.6, porém utilizando agora um elemento estruturante assimétrico. A operação de dilatação para um elemento estruturante plano pode, então, ser definida pela fórmula (17):

$$X \oplus E = \left\{ z \middle| (\hat{E})_z \cap X \neq \emptyset \right\} \tag{17}$$

Onde *X* representa um componente conectado, *E* o elemento estruturante, *z* todos os pixels de uma imagem *I(x, y)*, e E_z a translação de *E* sobre *z*. A Fig. 4.8 ilustra a dilatação de um componente

quadrado por um elemento estruturante cruciforme de forma semelhante à Fig. 4.6. Pode-se observar a influência do formato da estrutura morfológica sobre o resultado final da dilatação. Dependendo do tamanho do elemento estruturante, objetos próximos podem ser fundidos em um único componente e buracos internos podem ser preenchidos.

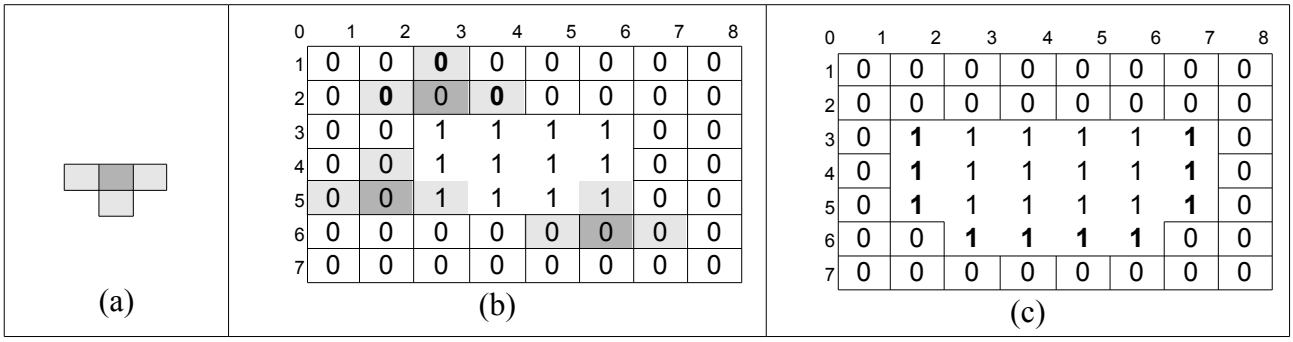


Fig. 4.7 – Exemplo de dilatação de um objeto retangular (3 X 4) em uma imagem (7 X 8) com um EE N_4 não simétrico: (a) EE; (b) imagem original mostrando três translações do EE refletido; (c) imagem dilatada .

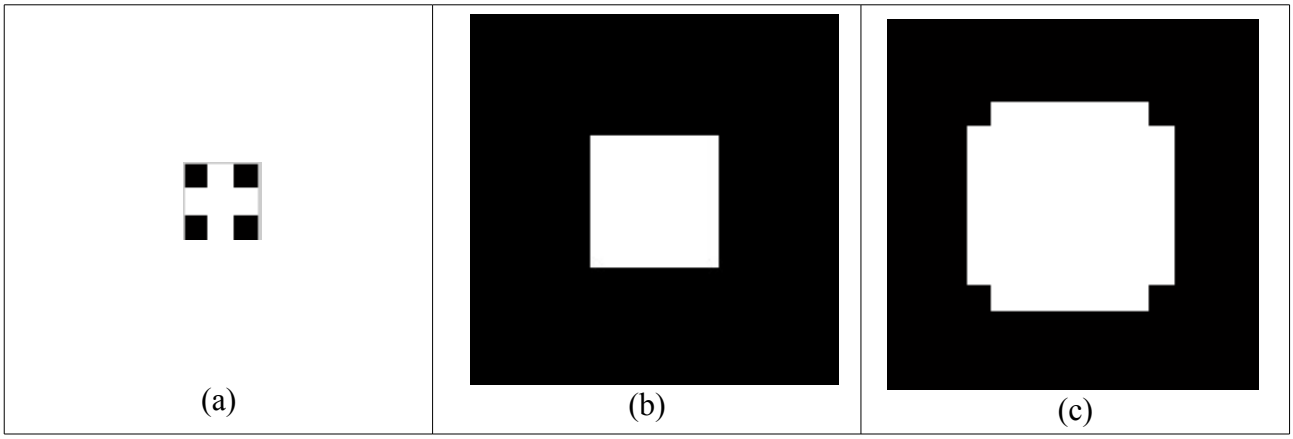


Fig. 4.8 – Exemplo de dilatação: (a) elemento estruturante; (b) imagem original; (c) imagem dilatada.

A operação de erosão é inversa à dilatação e tem sua fórmula definida como (18):

$$X \ominus E = \left\{ z \middle| (B)_z \subseteq X \right\} \tag{18}$$

Note que no caso da erosão, entretanto, o elemento estruturante não é refletido. Então, na operação de erosão de uma imagem binária *I* o valor do pixel na imagem de saída será 0, se qualquer um dos pixels de sua vizinhança for 0; e 1, se nenhum dos pixels de sua vizinhança for igual a 0. Similarmente à dilatação, na prática, a erosão pode ser definida como a união de todas as

células que, quando sobrepostas pela origem do elemento estruturante, não têm em seu domínio de análise nenhuma célula pertencente ao plano de fundo da imagem. A erosão de um objeto por ele mesmo resultaria, então, em um único pixel central.

Utilizando-se a mesma comparação feita para a operação de dilatação, na teoria das operações lógicas, a erosão de imagens binárias pode ser definida em termos da operação AND e, na teoria de conjuntos *fuzzy*, a erosão de imagens em intensidades de cinza pode ser definida em termos da operação de interseção (∩). A Fig. 4.9 mostra o resultado da erosão de um objeto, utilizando o mesmo elemento estruturante da Fig. 4.8.

Embora o número de translações que o elemento estruturante execute sobre a imagem seja igual ao número de pixels da imagem original, nem todas as translações necessitariam ser consideradas. No caso da erosão, basta que sejam analisadas as translações que tenham a origem do elemento estruturante posicionada sobre pixels pertencentes aos componentes conectados e, no caso da dilatação, apenas aquelas que tenham a origem do EE refletido posicionada sobre pixels do plano de fundo. Esta consideração é importante para otimizar os procedimentos.

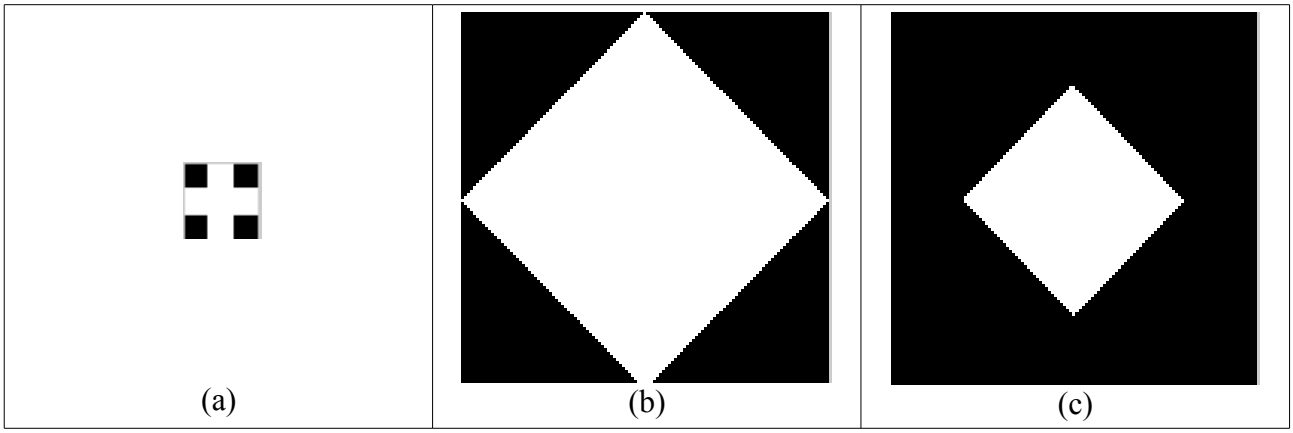


Fig. 4.9 – Exemplo de erosão: (a) elemento estruturante; (b) imagem original; (c) imagem erodida.

Quando se faz uma analogia com a teoria dos conjuntos, a operação de erosão corresponde à interseção (∩) entre todas as translações do componente conectado sobre o elemento estruturante refletido (Fig. 4.10(c)); e a dilatação, à união (∪) entre todas as translações do componente conectado sobre o elemento estruturante não refletido (Fig. 4.10(d)). Estas equivalências se comparam às operações da álgebra de Minkowski.

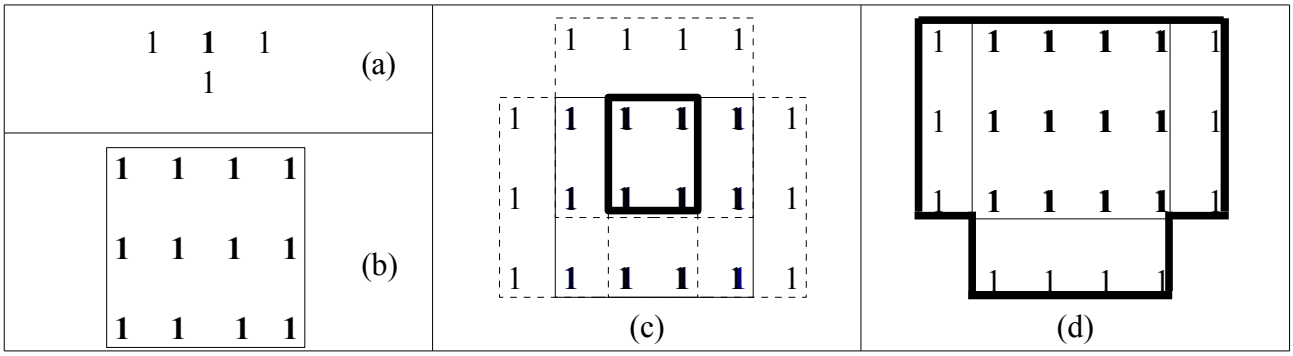


Fig. 4.10 – Analogia à teoria dos conjuntos: (a) elemento estruturante com origem em negrito; (b) componente conectado; (c) imagem erodida; (d) imagem dilatada.

A origem da teoria da morfologia matemática reside justamente na álgebra de Minkowski e as operações de dilatação e erosão equivalem às suas clássicas operações de adição e subtração, respectivamente (ROERDINK, 1994). Sendo assim, podem também ser representadas pelas fórmulas:

$$I \oplus E = \bigcup_{e \in E} I_e \qquad \text{e} \qquad I \ominus E = \bigcap_{e \in E} I_{-e}$$

Onde $X_y = \{ x + y \mid x \in X \}$ significa a translação do conjunto X ao longo do vetor y e $-x$ a reflexão de X em relação à sua origem.

Dos exemplos e teorias discutidos acima, podemos chegar a algumas conclusões. Considere L um objeto em uma imagem qualquer, LD a versão dilatada deste objeto, e LE, a sua versão erodida:

a) $LD \cap L \neq \emptyset$

b) $LE \subset L$

c) $LE \subset L \subset LD$

As operações de morfologia matemática sobre imagens em tons de cinza são uma extensão natural das operações binárias. Como uma faixa de novos valores inteiros são introduzidos, a noção de operações lógicas desaparece e fica apenas a noção de mínimo e máximo.

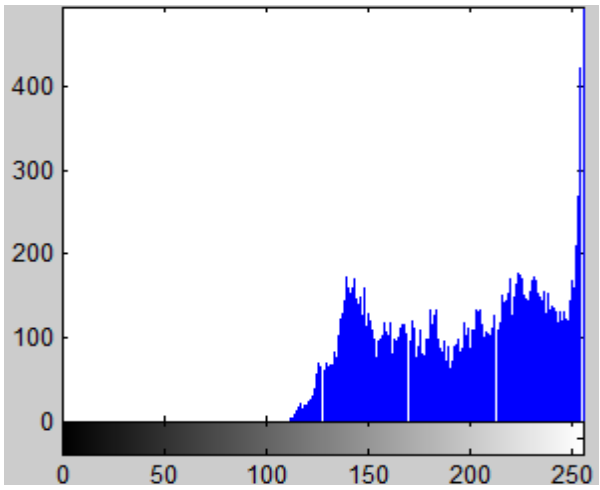
Uma vez que os algoritmos para operações morfológicas para imagens binárias e

imagens em intensidades de cinza não são os mesmos, os efeitos das operações de dilatação e erosão sobre estas imagens são também diferentes. No lugar de observarmos expansão ou encolhimento dos objetos, observamos um aumento ou diminuição das regiões mais claras ou mais escuras da imagem. Isto porque em imagens de intensidades de cinza não existe o conceito de componentes conectados. Os operadores morfológicos varrem a imagem maximizando ou minimizando o nível de intensidade dos pixels de acordo com a vizinhança determinada pelo elemento estruturante, ou seja, não existe a remoção ou inclusão de pixels e, sim, a modificação de seu nível de intensidade.

A Fig. 4.11(a) apresenta a imagem de um paraglider com nuvens mais brilhantes e nuvens mais escuras; a Fig. 4.11(b), o histograma da imagem. A erosão dessa imagem por um elemento estruturante no formato de um disco é mostrada na Fig. 4.12. É possível notar o destaque das nuvens mais escuras do cenário e que as imagens do piloto e do paraglider, apesar de borradas, se destacaram bastante. Já a Fig. 4.13 apresenta o resultado da dilatação da mesma imagem pelo mesmo elemento estruturante. Nota-se que houve um grande crescimento das áreas de nuvens mais brilhantes, suprimindo, inclusive, a imagem do piloto e do paraglider.

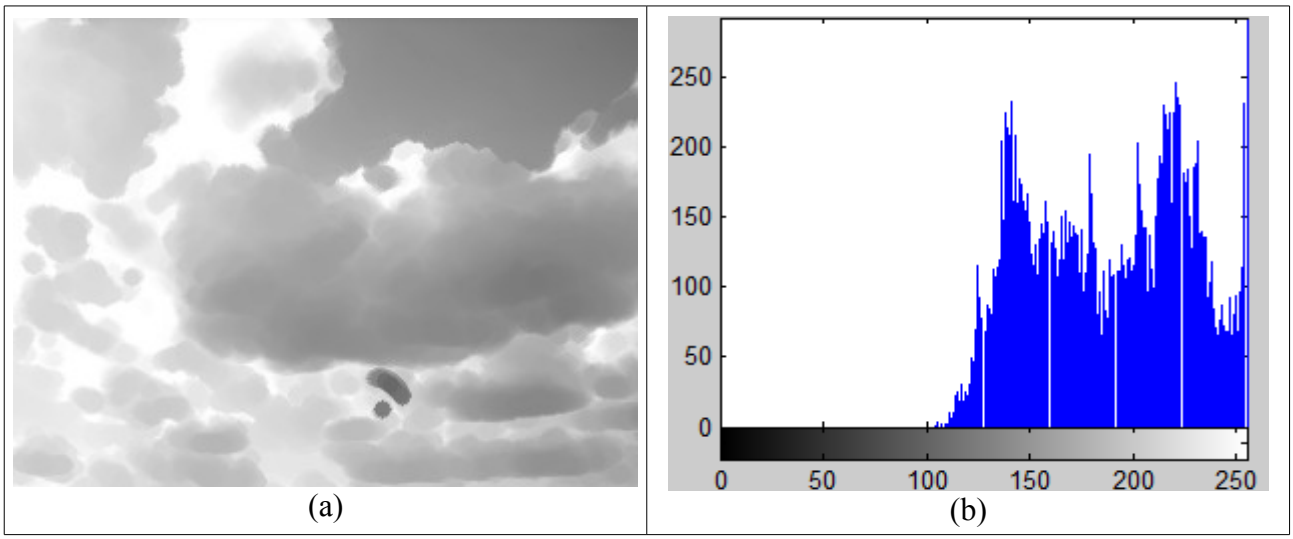
Os exemplos acima demonstram que a erosão de imagens em tons de cinza ressalta regiões escuras da imagem original, suprimindo regiões mais claras, menores que o elemento estruturante e, ao contrário, as imagens dilatadas ressaltam regiões mais claras e eliminam detalhes escuros e menores que o elemento estruturante. Os histogramas das Fig. 4.11, 4.12 e 4.13 ilustram essa característica.

(a)



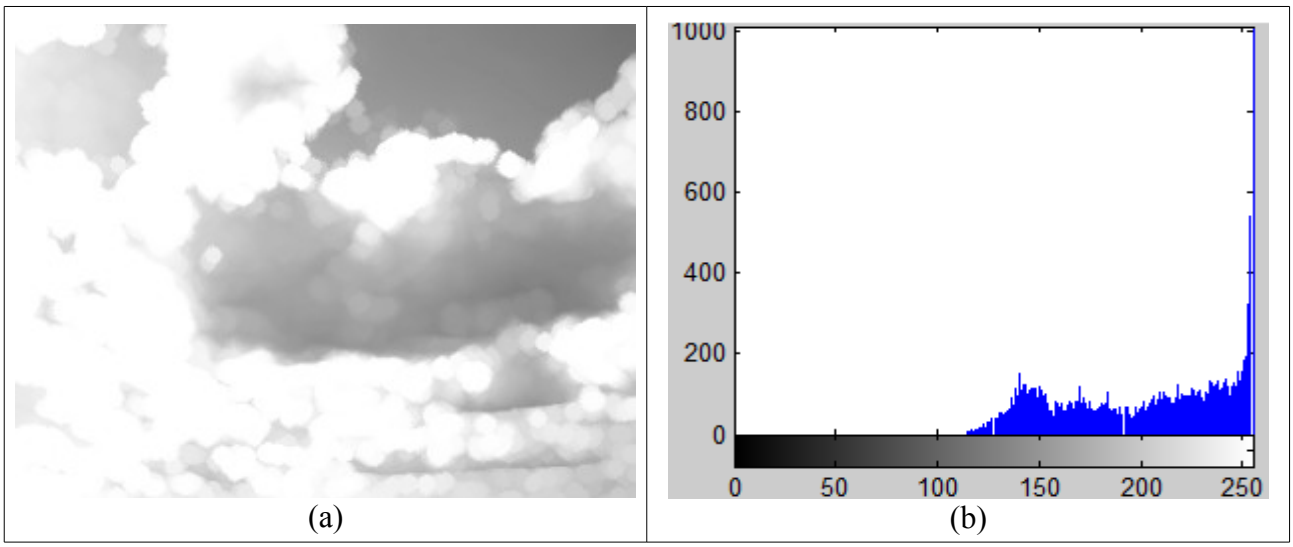
(b)

Fig. 4.11 – Imagem original e histograma (b).



(a) (b)

Fig. 4.12 – Imagem erodida e histograma (b).



(a) (b)

Fig. 4.13 – Imagem dilatada e histograma.

Aplicando-se o conceito de elementos estruturantes volumétricos, as operações de dilatação e erosão em imagens em tons de cinza são expressas pelas fórmulas (19) e (20):

$$(I \oplus E) = \left\{ max(I(x-x', y-y') + E(x', y')) \middle| I(x-x', y-y') \in D_I \, e \, E(x', y') \in D_E \right\} \quad (19)$$

$$(I \ominus E) = \left\{ min(I(x+x', y+y') - E(x', y')) \middle| I(x-x', y-y') \in D_I \, e \, E(x', y') \in D_E \right\} \quad (20)$$

Onde E representa o elemento estruturante aplicado, $I(x, y)$ representa a intensidade de cinza na coordenada da imagem I sobreposta por $E(0, 0)$, $E(x', y')$ representa o peso de E no ponto (x', y'), D_I

o domínio da imagem I e D_E o domínio do elemento estruturante. Note que a fórmula (19) prevê a reflexão do EE.

Diferente das transformações lineares em imagens em tons de cinza, as transformações morfológicas são caracterizadas pelo seu caráter irreversível (STERNBERG, 1986). O mesmo pode ser dito para as operações morfológicas sobre imagens binárias, ou seja, operações morfológicas removem, acrescentam ou alteram valores de intensidade de pixels, e, onde isto ocorre, a informação sobre a imagem original é perdida.

Para Beucher (1992), apesar da morfologia matemática ser por excelência uma técnica definida para aplicação em imagens binárias, os resultados alcançados com a morfologia matemática sobre imagens em tons de cinza são melhores que os alcançados em imagens binárias porque a perda de informação durante o processo de transformação é melhor controlada.

O uso de imagens coloridas introduz uma maior complexidade tanto no conceito quanto na implementação da morfologia matemática. Isso porque uma imagem pode ser representada de diversas formas, dependendo do espaço de cor utilizado (RGB, CMYK, HSV, etc.). Seriam necessários algoritmos diferentes para cada espaço de cor. Além disso, tratar individualmente cada canal de cor não seria apropriado, já que as informações de cada canal são fortemente correlacionadas. Essa abordagem acarretaria grande perda de informação (BURGETH et al., 2004).

4.3.1.1 Propriedades das Operações de Dilatação e Erosão

As operações morfológicas de dilatação e erosão não são inversas no sentido de que quando uma imagem *I* é dilatada por um elemento estruturante *E*, a erosão da imagem dilatada pelo mesmo elemento estruturante não recupera a imagem *I* original. Transformações morfológicas perdem a informação sobre os pixels que foram modificados. Entretanto, essas operações compartilham algumas propriedades e podem, ainda assim, ser classificadas como duais sobre alguns aspectos. Ambas as operações satisfazem as propriedades associativa (a), distributiva (b) e de invariância em relação à translação (c) (HEIJMANS, 1995; ROERDINK, 1994):

a) $(I \oplus E_1) \oplus E_2 = I \oplus (E_1 \oplus E_2)$ ou $(I \ominus E_1) \ominus E_2 = I \ominus (E_1 \ominus E_2)$

b) $\left(\bigcup_{x \in X} I_x \right) \oplus E = \bigcup_{x \in X} (I_x \oplus E)$ ou $\left(\bigcap_{x \in X} I_x \right) \ominus E = \bigcap_{x \in X} (I_x \ominus E)$

c) $(I \oplus E)_x = I_x \oplus E$ ou $(I \ominus E)_x = I_x \ominus E$

De acordo com as formulações em (b), a dilatação é distributiva em relação à união; a erosão, em relação à interseção. A propriedade de associação (a) é importante por permitir a decomposição de elementos estruturantes maiores em outros dois menores, aumentando a velocidade de processamento em relação às operações de dilatação ou erosão realizadas com um único elemento estruturante. A Fig. 4.14 ilustra a decomposição de uma estrutura (10 X 10) em duas outras menores.

As operações de dilatação e a erosão são também duais em relação ao complemento (HEIJMANS, 1995), ou seja, o complemento da dilatação de um conjunto de pixels que formem um componente conectado produz o mesmo efeito que a erosão do complemento deste mesmo componente conectado com o mesmo elemento estruturante, porém refletido (21).

$$(I \oplus E)^c = I^c \ominus \hat{E} \quad \text{e} \quad (I \ominus E)^c = I^c \oplus \hat{E} \tag{21}$$

Onde ^{c} representa complemento, $\hat{}$ representa reflexão e E o elemento estruturante. Ainda segundo Heijmans (1995), pode-se dizer que as operações de dilatação são duais também em relação à adjunção (22):

$$I_1 \oplus E \subseteq I_2 \quad \Leftrightarrow \quad I_1 \subseteq I_2 \ominus E \tag{22}$$

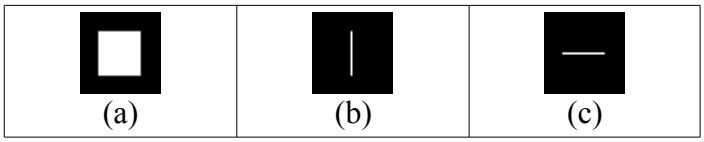


Fig. 4.14 – Decomposição de EE: (a) EE (10 X 10); (b) EE (10 X 1); (c) EE (1 X 10).
Note que: (b) $\oplus$ (c) = (a)

4.3.2 Hit-or-Miss ($\otimes$)

A operação hit-or-miss difere das operações de dilatação e erosão por utilizar dois elementos estruturantes que obrigatoriamente devem satisfazer à seguinte condição:

$$EE_1 \cap EE_2 = \emptyset$$

É geralmente utilizada para detecção de pontos específicos nas imagens, tais como extremidades de segmentos. O primeiro elemento é chamado de EE de acerto porque tenta se ajustar dentro do objeto e o segundo elemento é chamado de EE de erro porque tenta se ajustar em seu complemento, ou seja, no plano de fundo. Os elementos estruturantes da Fig. 4.15 foram escolhidos de tal maneira que pudessem identificar o canto superior direito do objeto presente na imagem. De acordo com Heijmans (1995), o operador de hit-or-miss, ou de acerto e erro, pode ser formulado como (23):

$$X \otimes (A, B) = \{ h \in E^d \mid A_h \subseteq X \text{ and } B_h \subseteq X^c \} \tag{23}$$

Onde E^d representa um espaço de d dimensões e $A_h = \{ a + h \mid a \in A \}$ representa a translação do conjunto A ao longo do vetor h. Esta fórmula pode ser simplificada pela seguinte (24) (GONZALES; WOODS; EDDINS, 2004):

$$X \otimes (A, B) = (X \ominus A) \cap (X^c \ominus B) \tag{24}$$

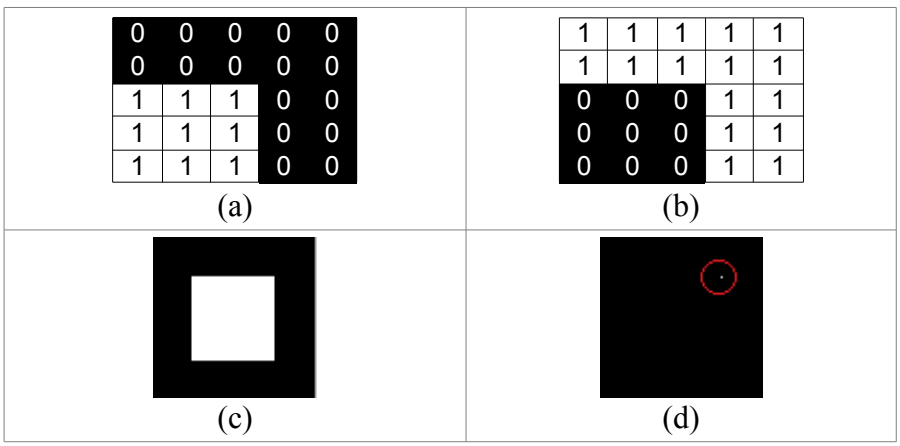


Fig. 4.15 – Hit-or-miss: (a) EE de acerto; (b) EE de erro; (c) imagem original; (d) detecção do canto superior direito.

A operação hit-or-miss não tem uma extensão para imagens em intensidades de cinza.

4.3.3 Fechamento e Abertura

A combinação das duas operações morfológicas básicas, erosão e dilatação, e a ordem dessa combinação é o que define as operações de abertura (*opening*) e fechamento (*closing*).

Assim, a operação de fechamento é definida como a erosão da imagem dilatada de I usando-se o mesmo elemento estruturante E (25):

$$I \bullet E = (I \oplus E) \ominus E \tag{25}$$

A operação de abertura é inversa à operação de fechamento, ou seja, é a dilatação da imagem erodida de I (26). No final obtêm-se objetos menores que os originais.

$$I \circ E = (I \ominus E) \oplus E \tag{26}$$

O fechamento afeta a imagem de maneira a preencher pequenos buracos no interior dos objetos além de suavizar seus contornos externos. Um buraco é definido como um conjunto desconexo de pixels do plano de fundo, ou seja, uma área de pixels pretos circundada por pixels brancos.

O procedimento de abertura é eficiente para a remoção de ruídos, além disso, tende a abrir buracos na imagem e remover as irregularidades de contorno. Portanto, ambas as operações são utilizadas para suavização de imagens. A diferença está no fato de que a abertura suaviza os contornos dos objetos por remover as irregularidades menores que o elemento estruturante; e o fechamento, por preencher os espaços vazios entre irregularidades e menores que o EE. Então, o mesmo pode ser concluído para a remoção de ruídos que, dependendo da operação utilizada, pode ser removido ou preenchido. O tamanho adequado do elemento estruturante é muito importante. Se for projetado muito pequeno, os ruídos podem não ser removidos. Há ainda o problema de que pequenos objetos podem ser interpretados como ruídos e removidos indesejadamente.

A abertura de um objeto X por um elemento estruturante Y pode ser também definida como a união das translações de Y completamente contidas em X, e o fechamento de X por Y, como

o complemento das translações de *Y* refletido que estão completamente contidas em X^c (STERNBERG, 1986). Estas considerações podem ser facilmente observadas através da Fig. 4.16, onde a imagem original passou por um procedimento de abertura e de fechamento utilizando um elemento estruturante de formato esférico. Em outras palavras, na abertura, o elemento estruturante deve se ajustar dentro do objeto. Regiões pertencentes aos componentes conectados, e que não comportam inteiramente o EE dentro de si mesmas, são removidas. Na operação de fechamento, ao contrário, o EE deve se ajustar fora do objeto e o resultado final da operação é a imagem complemento deste ajuste.

Outra forma muito clara de se visualizar os procedimentos de abertura e fechamento, e também a Fig. 4.16 , é através do algoritmo *Rolling Ball*. Este algoritmo é descrito na prática como o efeito do rolamento de uma esfera sobre as superfícies das imagens imaginadas como um relevo geográfico. Assim, a operação de fechamento suaviza os contornos de um objeto através do procedimento de "rolar a bola" sobre sua superfície externa (Fig. 4.16 (b)). Similarmente, a operação de abertura pode ser visualizada como o ato de "rolar a bola" sob as superfícies dos objetos (Fig. 4.16 (c)) e tem o efeito de suavização inverso ao do procedimento de fechamento, ou seja, a abertura suaviza por remoção de pixels; e o fechamento, pelo acréscimo de pixels à borda do objeto. O grau desta suavização depende do raio da esfera escolhida. Esferas menores conseguem percorrer maiores profundidades nos vales. Nas Fig. 4.16 (b) e (c) foram utilizadas esferas de 9 pixels de raio. Nota-se que nas Fig. 4.16 (d) e (e), onde foram utilizadas esferas de 5 pixels de raio, a esfera percorreu mais profundamente os vales e também preencheu melhor o espaço interno, sob a superfície do objeto original.

De acordo com Heijmans (1995), podem-se listar as seguintes propriedades para os operadores de abertura e fechamento:

a) Os operadores de abertura e fechamento são idempotentes;

b) $X \circ A \subseteq X$;

c) $(X \circ A) \ominus A = X \ominus A$;

d) $(X \bullet A) \oplus A = X \oplus A$;

e) Os operadores de abertura são um a negação do outro:

$$(X \circ A)^c = X^c \bullet \hat{A} \qquad \text{e} \qquad (X \bullet A)^c = X^c \circ \hat{A} \text{ ;}$$

f) A operação de abertura é anti-extensiva, e o fechamento é extensivo.

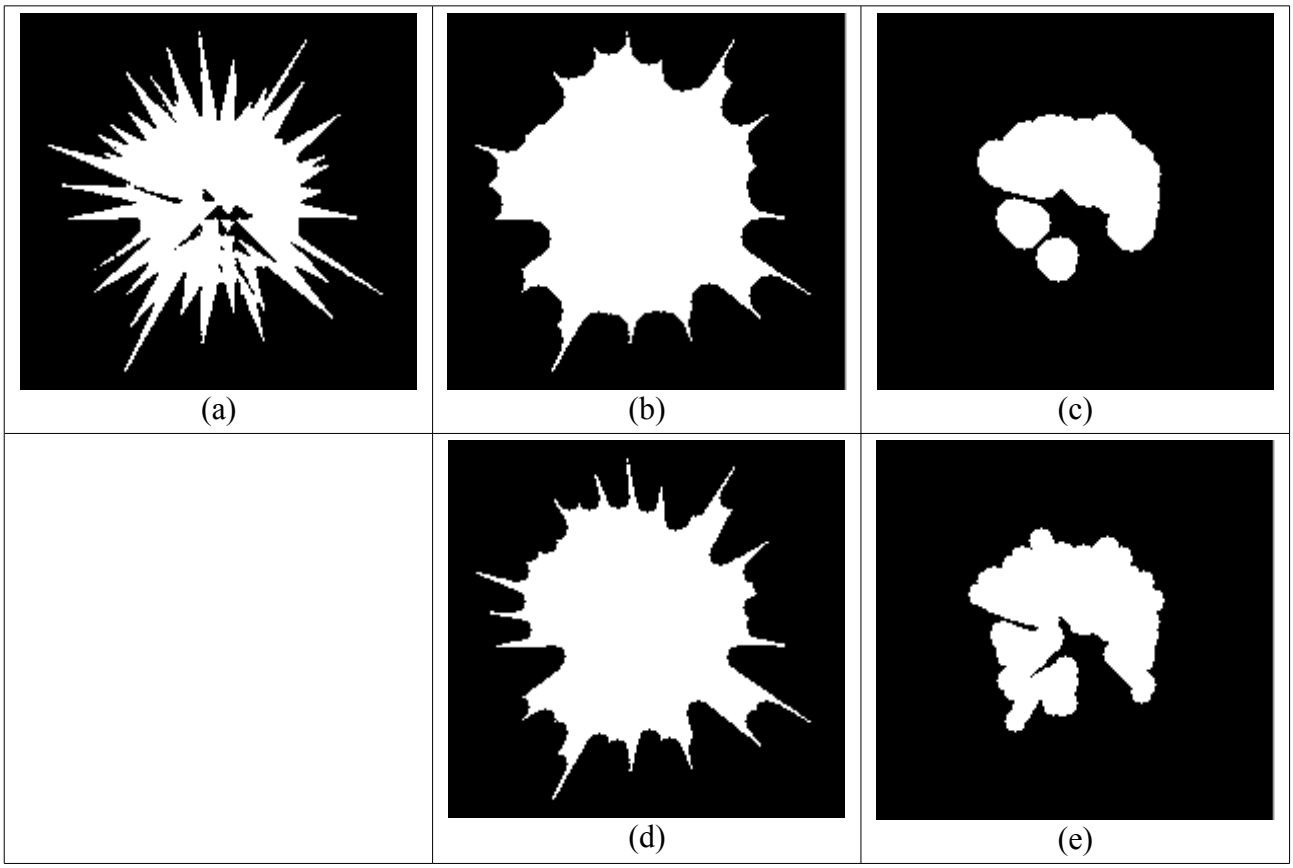
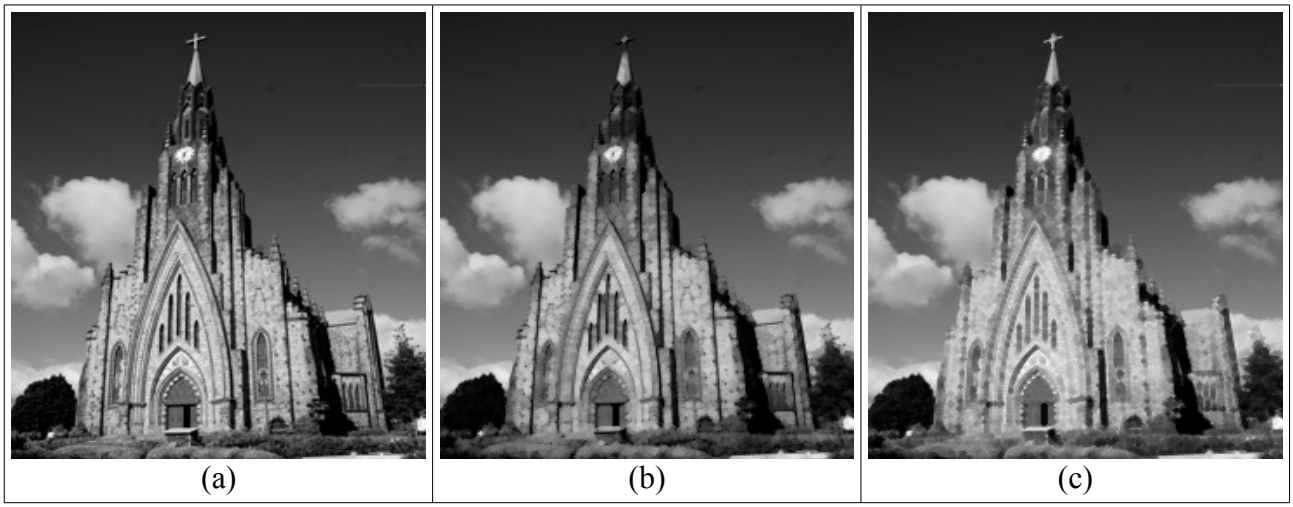


Fig. 4.16 – Algoritmo *Rolling Ball*: (a) Imagem original; (b)fechamento por EE esférico de 9 pixels de raio; (c) abertura por EE esférico de 9 pixels de raio; (d) fechamento por EE esférico de 5 pixels de raio; (e) abertura por EE esférico de 5 pixels de raio.

O efeito da abertura e fechamento em imagens em tons de cinza é similar ao provocado pela erosão e dilatação destas imagens. A abertura realça regiões mais escuras na imagem original, e o fechamento, por sua vez, realça regiões de maior brilho. Nota-se que na Fig. 4.17 (b), a abertura praticamente eliminou a cruz no alto da catedral e também realçou os contornos mais escuros da construção. Observe também o brilho destacado no relógio no alto da torre na Fig. 4.17 (c).



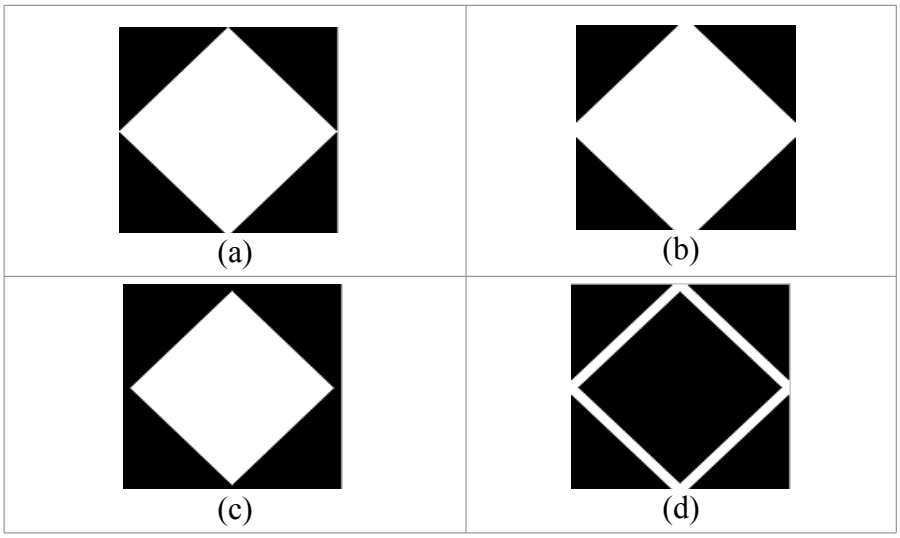
(a) (b) (c)

Fig. 4.17 – Abertura e fechamento: (a) Imagem original; (b) imagem após abertura ; (c) imagem após fechamento.

4.4 Outras Aplicações da Morfologia Matemática

4.4.1 Detecção de Bordas

É possível ainda implementar a detecção de bordas utilizando a subtração da imagem dilatada pela imagem erodida usando o mesmo elemento estruturante. Esta diferença é chamada de gradiente morfológico. A Fig. 4.18 mostra a aplicação deste conceito. As imagens das Fig. 4.18 (b) e (c) foram transformadas por um elemento estruturante quadrado e de altura de 10 pixels.



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

Fig. 4.18 – Exemplo de detecção de bordas usando dilatação e erosão: (a) imagem original; (b) imagem dilatada; (c) imagem erodida; (d) gradiente morfológico.

O conceito de gradiente morfológico também é eficiente para detecção de bordas em imagens em tons de cinza como mostra a Fig. 4.19.

(a) (b)

Fig. 4.19 – Exemplo de detecção de bordas usando gradiente morfológico em imagem em tons de cinza: (a) imagem original; (b) gradiente morfológico.

4.4.2 Algoritmo *Watershed* Morfológico

O algoritmo *watershed*, muito utilizado em segmentação de imagens, também tem sua versão morfológica. Este algoritmo trata uma imagem em intensidades de cinza como se fosse uma superfície topográfica, tons mais claros são vistos como regiões de maior altitude ou picos, e tons mais escuros são vistos como regiões de vales. Os picos de uma imagem I são os vales da imagem complemento de I (I^c) (STERNBERG, 1986). Com esta noção de picos e vales, imagine a imagem como uma maquete sendo lentamente submersa e um furo em cada regional mínima. A água entraria a partir destes furos e formaria lagos. Com a evolução da inundação, dois ou mais lagos poderiam se encontrar e se fundir. Para evitar este encontro são construídas represas, também chamadas de divisores de águas. Esta é uma explicação ilustrativa do algoritmo *watershed*.

Fica clara a percepção de que o algoritmo *watershed* é um procedimento para segmentação de imagens por detecção de similaridades. Regiões de picos e vales são identificadas a partir de sua regional máxima ou mínima. O resultado é uma imagem segmentada semelhante a um mosaico.

Versões morfológicas do algoritmo *watershed* substituem o conceito de regional mínima pelo conceito de marcadores. Os marcadores funcionam como sementes para o crescimento

das regiões, similarmente à regional mínima no conceito do algoritmo original. Gee e Abidi (1995) executam sucessivas erosões na imagem para definir marcadores para cada objeto. Para Coster e Chermant (2001), o uso de regionais mínimas geralmente produz uma "sobre-segmentação" e a correta escolha dos marcadores é fundamental para a qualidade da segmentação.

Beucher (1992) descreve o algoritmo *watershed* a partir da implementação do conceito de gradiente morfológico. Entretanto, esta técnica também produz uma "sobre-segmentação". Este efeito é contornado através da seleção de marcadores antes da aplicação do procedimento *watershed* morfológico.

4.4.3 Esqueletonização

Esqueletonização é uma forma de representação da imagem de objetos através de uma unidade de largura e que se assemelha a um grafo. A esqueletonização simplifica detalhes complexos em uma figura e pode ser utilizada para diversos propósitos como reconhecimento de padrões e compressão de imagens. A idéia central é produzir um esqueleto do objeto que o represente de forma única. Na literatura, existe uma enorme variedade de algoritmos para esqueletonização.

Uma exigência dos algoritmos para esqueletonização é que a imagem resultante do procedimento deve reter informação sobre a forma original dos objetos. Como, de uma forma geral, as operações morfológicas são caracterizadas por perderem informação sobre a imagem original, os algoritmos morfológicos para esqueletonização são pouco utilizados.

Para Marchand-Maillet e Sharaiha (2000), o esqueleto S de uma imagem I deve satisfazer às seguintes propriedades (considerando F_n os n componentes conectados presentes em I e IS a imagem resultante após a esqueletonização):

a) IS deve ter o mesmo número de componentes conectados que I;

b) IS^c deve ter o mesmo número de componentes conectados que I^c;

c) $S_n \subseteq F_n$;

d) S_n tem um pixel de largura;

e) A partir de S_n é possível reconstruir F_n.

Essas propriedades significam dizer que S_n e F_n são estruturas topológicas equivalentes, e que uma pode ser mapeada para a outra através de alguma transformação contínua. Nem sempre todas as propriedades são satisfeitas. No caso da compressão de imagens, por exemplo, a propriedade (d) pode ser relaxada, mas a propriedade (e) deve, obrigatoriamente, ser cumprida.

As versões morfológicas, em geral, são implementadas através de sucessivas operações de erosão que provocam um emagrecimento dos objetos. A cada iteração do procedimento, a validade das propriedades deve ser novamente verificada.

A Fig. 4.20 apresenta um exemplo da aplicação de um algoritmo de esqueletonização morfológica sobre uma imagem binária. Os componentes conectados possuem um pixel de largura, mas não é possível recuperar fielmente a imagem original através do uso de operações morfológicas inversas. A principal preocupação dos algoritmos de esqueletonização morfológica é a possibilidade de desconexão dos componentes após algumas iterações do procedimento.

(a)

(b)

Fig. 4.20 – Esqueletonização: (a) imagem original; (b) imagem após esqueletonização.

4.4.4 Filtros Morfológicos

Convolução, transformada de Fourier, filtros passa-altas e filtros passa-baixas são exemplos de transformações que pertencem à classe dos filtros lineares. Do outro lado desta classificação encontram-se os filtros morfológicos cujos exemplos mais simples são as próprias operações de abertura e fechamento. Entretanto, com a combinação em série de várias dilatações e/ou erosões, aberturas e/ou fechamentos, usando o mesmo EE ou, algumas vezes, um EE diferente, uma infinidade de filtros pode ser construída. No geral, a construção do filtro depende do objetivo que se pretende na aplicação e da forma geométrica de interesse na imagem a ser processada. O número de operações morfológicas utilizadas pode, ainda, ser projetado para ser definido

automaticamente. Por exemplo, pode-se projetar um filtro que realize sucessivas erosões até que um determinado critério seja satisfeito.

Alguns outros exemplos de filtros morfológicos já bastante conhecidos na literatura são o *top-hat*, o *bottom-hat* e o próprio operador *hit-or-miss*. A transformação *top-hat* subtrai a imagem original de sua imagem de abertura, e a *bottom-hat* subtrai uma imagem de fechamento de sua imagem original. A combinação destes dois filtros pode melhorar o contraste de uma imagem (GONZALES; WOODS; EDDINS, 2004).

De acordo com Beucher (1992), existem duas condições para que uma transformação morfológica seja considerada um filtro:

a) se $X \subseteq Y$ então $\Phi(X) \subseteq \Phi(Y)$ (onde $\Phi(X)$ representa o conjunto X transformado pelo filtro Φ);

b) a transformação aplicada deve ser idempotente, ou seja, $\Phi(\Phi(X)) = \Phi(X)$.

Operadores morfológicos também são utilizados para reconstrução. A reconstrução morfológica envolve duas imagens e um elemento estruturante. Uma das imagens é utilizada como imagem de referência. A reconstrução é, então, obtida através de sucessivas dilatações e comparações com a imagem de referência que, então, funciona como limitadora do número de iterações do procedimento. As iterações terminam quando a interseção com a imagem de referência não mais se altera após uma nova iteração, ou seja, quando o procedimento alcança a idempotência. Ao final, a imagem reconstruída tem que estar contida na imagem original.

Existem vários algoritmos morfológicos para reconstrução. Em geral, a precisão da restauração depende da similaridade entre os objetos na imagem e o elemento estruturante. Para Gonzales, Woods e Eddins (2004), o ponto chave é selecionar adequadamente a imagem de referência, também chamada de máscara, e o EE adequado para se alcançar o efeito desejado. A operação pode ser definida como (27):

$$R_{k+1} = (R_k \oplus E) \cap I \tag{27}$$

Onde R_k representa *k-ésima* iteração do algoritmo e E o elemento estruturante. O procedimento pára quando $R_{k+1} = R_k$.

5 Desenvolvimento de Sistema Computacional de Apoio à Usuário Tetraplégico

5.1 Introdução

O objetivo deste trabalho é desenvolver um sistema computadorizado de baixo custo, sem qualquer requisição de hardware especial em contato com o corpo do usuário, capaz de ser operado por usuários tetraplégicos que não possuem coordenação motora para a utilização de um mouse ou um teclado, pacientes cujo único movimento do corpo sobre o qual se tem controle é o movimento dos seus olhos e que também são incapazes de utilizar softwares de reconhecimento de voz. Diversas condições patológicas ou traumáticas podem levar um paciente à condição descrita, como distrofia muscular severa, paralisia cerebral, acidentes com injúria da medula espinhal, acidente vascular cerebral ou mesmo condições congênitas.

O meio mais óbvio de se projetar uma interface livre de contato para se satisfazerem as necessidades destes usuários é usar uma câmera digital interfaceando com um computador pessoal, traduzindo movimentos do corpo em movimentos do cursor e interpretando mudanças na expressão corporal ou facial para emular o click de um mouse.

Assim, o trabalho propõe um método de identificação da localização dos olhos do usuário na imagem digitalizada de sua face capturada por uma webcam comum e de reconhecimento de suas piscadas voluntárias. As piscadas identificadas como voluntárias são ainda classificadas em piscadas de curta duração e piscadas de longa duração. Piscadas curtas são interpretadas como um comando para mudança de foco no menu da interface do aplicativo emulando o movimento de um mouse. Piscadas longas reproduzem a ação do click do mouse.

A interface experimentada apresenta um menu do tipo *click-down* com nove opções. Cada opção pode ser associada a uma função que se deseja aplicar ao sistema para apoio à rotina do usuário deficiente como o acionamento de uma campanhia, a ação de ligar ou desligar algum equipamento, etc. A interface apresenta ainda três outras opções que não são controláveis pelos olhos e que devem ser acionadas por um segundo agente que funcionaria como um supervisionador ou facilitador do processo. Estas opções permitem localizar os olhos do usuário, iniciar e encerrar o sistema. A Fig. 5.1 apresenta um diagrama da interface do aplicativo de teste. O componente texto é utilizado para fornecer informações textuais de saída para o aplicativo. Pode-se informar, por exemplo, qual função está ativada em um dado momento ou apresentar mensagens explicativas para o usuário.

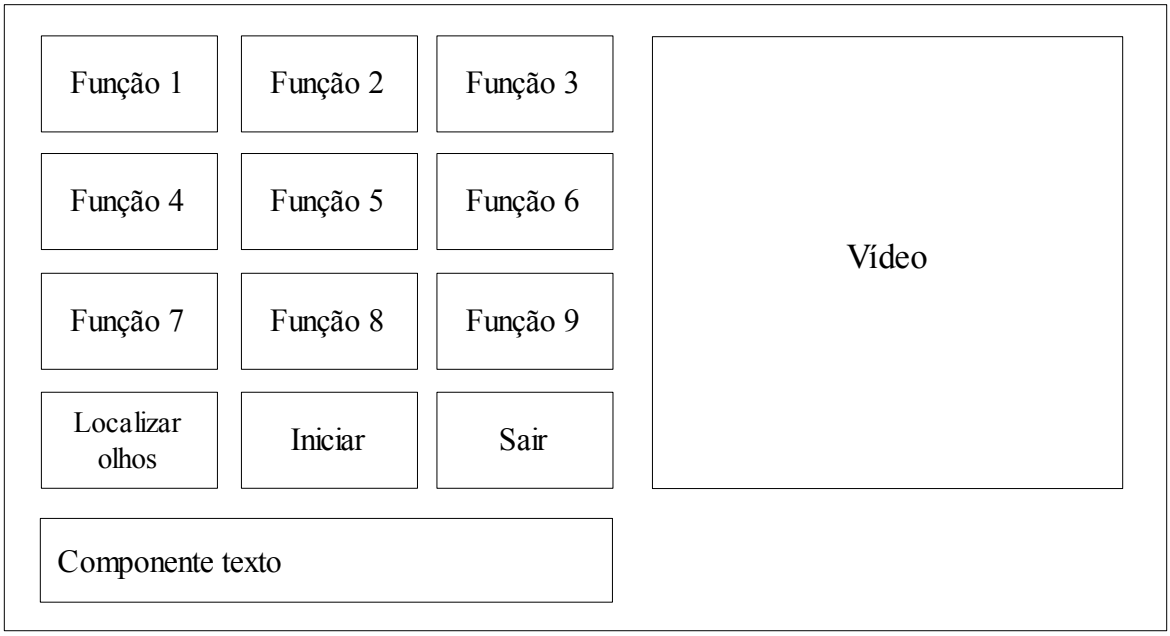


Fig. 5.1 - Diagrama da interface de teste utilizada.

O método proposto neste trabalho foi desenvolvido usando a ferramenta de desenvolvimento MATLAB 7.6.0 (R2008a). Os *toolboxes* utilizados foram: Image Processing Toolbox, Image Acquisition Toolbox e Data Acquisition Toolbox.

O MATLAB é uma plataforma ideal para se avaliar a viabilidade de algoritmos e aplicativos em processamento digital de imagens. Os métodos propostos aqui podem posteriormente ser transcritos para qualquer outro ambiente de desenvolvimento.

A implementação do sistema proposto neste trabalho apresenta as seguintes fases:

1 – Segmentação dos olhos;
2 – Rastreamento em tempo real;
3 – Identificação das piscadas voluntárias;
4 – Interpretação e tradução do comando.

As seções seguintes discutem cada um destes tópicos.

5.2 Segmentação dos Olhos

Não existe um método perfeito de segmentação que seja eficiente para qualquer tipo de imagem. Além disto, a segmentação de imagens digitais usualmente requer um grande número de passos de processamento e a maioria destes passos são frequentemente muito dependentes da aplicação (NAJMAN; COUPRIE; BERTRAND, 2005). Muñoz et al. (2003) observam ainda que a integração de duas ou mais técnicas parece ser uma tendência nos estudos de processamento digital de imagens.

O método proposto neste trabalho para localização da posição dos olhos do usuário segue esta tendência utilizando uma combinação de duas técnicas de segmentação. O método combina uma técnica modificada para detecção de pontos com técnicas baseadas nos conceitos de morfologia matemática. A segmentação morfológica é ideal em casos onde se deseja segmentar objetos claramente distinguíveis por seu formato como é o caso da íris ocular.

Na primeira etapa do processo de segmentação, uma máscara 3 X 3 é utilizada para a detecção de pontos na imagem. O uso desta máscara também é eficiente na diminuição da complexidade do plano de fundo da imagem tornando menos complexas as próximas etapas da segmentação (Fig. 5.2). Máscaras para detecção de linhas horizontais e verticais também foram eficientes mostrando resultados similares à máscara de detecção de pontos utilizada. Antes desta etapa, as imagens capturadas no espaço de cor RGB foram transformadas em imagens em intensidades de cinza e processadas por um filtro de realce para incrementar o contraste.

A idéia seguinte foi explorar o formato evidentemente circular da imagem da íris para separá-la das demais regiões segmentadas utilizando técnicas de morfologia matemática. Entretanto, nos testes de diversas imagens de faces de usuários diferentes, a imagem circular das narinas era também destacada pela máscara de detecção de descontinuidades. Este problema foi contornado por

meio de testes de consistência baseados na análise de medidas antropométricas de tamanho e distância entre as íris como será explicado nos próximos parágrafos.

Fig. 5.2 - Pré-segmentação usando máscara de detecção de pontos.

Para a aplicação dos estudos morfológicos, a imagem resultante da aplicação da máscara de detecção de pontos é invertida seguindo o princípio de que os pixels pertencentes ao plano dos objetos possuem valor 1 (*foreground*), e os pixels pertencentes ao plano de fundo possuem valor 0 (*background*). Realizada a inversão, é aplicado um procedimento morfológico para preenchimento de buracos ou lacunas na imagem binária (Fig. 5.3(a)). Um buraco é um conjunto de pixels pertencentes ao plano de fundo totalmente circundado por pixels do primeiro plano, ou seja, é um sub-conjunto disjunto do plano de fundo. Note que a imagem da Fig. 5.2(b) apresenta diversos buracos no interior da imagem da íris.

Um elemento estruturante de formato esférico é, então, aplicado à imagem para se extrair o formato circular da íris. A operação morfológica utilizada é a abertura. No procedimento de abertura o elemento estruturante deve se ajustar dentro dos objetos, desta forma todo objeto ou ruído da imagem que não comporte o formato esférico do elemento estruturante é eliminado. Observe nas Fig. 5.2(b) e Fig. 5.3(a) que a imagem circular das narinas do usuário foi também destacada mas que a área destes componentes é visivelmente diferente. A segmentação morfológica destas regiões é evitada através da adequada escolha do tamanho do EE e de testes de consistência que avaliam a distância média entre as íris na face. Experimentalmente, observou-se que o uso de um disco com raio de quatro pixels, geralmente, é suficiente para a eliminação dos componentes

representativos das narinas. O resultado após a operação de abertura são todos os componentes conectados que comportem o formato e tamanho do EE utilizado (Fig.5.3(b)).

(a) (b)

Fig. 5.3 – Segmentação por técnicas de morfologia matemática: (a) imagem após preenchimento de buracos; (b) imagem após operação de abertura.

Isoladas todas as regiões que comportem o mínimo tamanho observado para a íris, e com a finalidade de se determinar quais componentes conectados dentre os extraídos pela operação de abertura realmente correspondem aos olhos do usuário, novamente a consistência do tamanho da íris é testada. Os componentes são rotulados (*connected component labeling*) para que se possa identificar o centro de massa e a área de cada um deles. De acordo com medidas antropométricas, os componentes são então avaliados aos pares para se identificar o par de componentes consistente com o tamanho máximo da íris e a posição relativa entre os olhos no rosto. Para o critério da posição dos olhos, são comparadas as distâncias tanto no eixo horizontal quanto vertical. Desta forma, mesmo que em alguns usuários as narinas pudessem ser segmentadas elas seriam descartadas nesta fase do procedimento dada a proximidade entre elas ser inconsistente com a separação das íris no rosto humano. As medidas utilizadas foram baseadas em experimentações com diversos usuários posicionados a uma distância entre 30cm a 40cm da câmera.

A Fig. 5.4 mostra que, para a eficácia do método de segmentação, é então necessário que o usuário mantenha a cabeça relativamente ereta sobre o eixo horizontal durante a inicialização do sistema. Rotações laterais acentuadas podem comprometer a identificação dos olhos na imagem durante a fase de avaliação das medidas antropométricas. Note nas Figs. 5.4 (c) e (d) que a rotação lateral da face aproxima os componentes conectados correspondentes às imagens das íris no eixo *x*

e, em compensação, a distância no eixo *y* entre estes mesmos componentes é aumentada. A flexão frontal da face, entretanto, não compromete o método (Figs. 5.4 (e) e (f)). Após a identificação da posição dos olhos, ou seja, durante o rastreamento e execução do aplicativo, rotações laterais lentas não comprometem a eficácia do sistema.

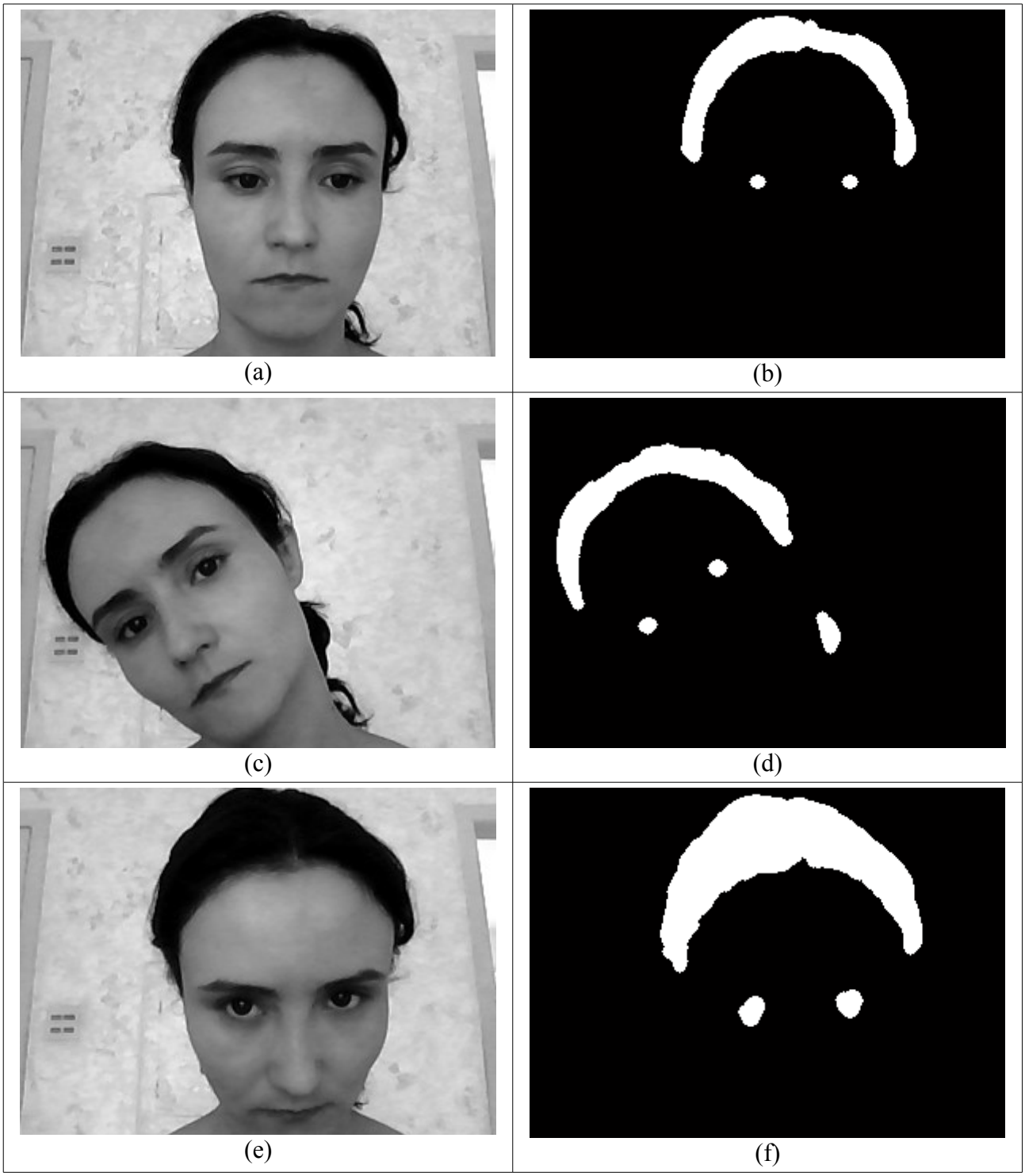


Fig. 5.4 – Efeito da rotação de cabeça do usuário: (a) e (b) cabeça ereta; (c) e (d) rotação lateral; (e) e (f) rotação frontal.

O procedimento usa um único frame para o processo de segmentação. Se o procedimento falha na localização do par de olhos, novo frame deve ser capturado e o procedimento é, então, repetido. Havendo sucesso nesta etapa, o sistema está pronto para ser inicializado. Duas saídas são produzidas neste procedimento de localização inicial da posição dos olhos do usuário:

a) coordenadas do centro de massa do olho, direito ou esquerdo, escolhido pelo supervisionador ou facilitador do processo;
b) coordenadas de um recorte de 200 X 160 pixels tendo as coordenadas do ítem (a) como centro.

A coordenada do centro do olho identificada nesta etapa inicial de segmentação é tomada como a primeira localização do olho para a etapa seguinte de rastreamento dos olhos em tempo real. O recorte de 200 X 160 pixels é gravado e utilizado durante a completa execução do sistema para limitar a região de rastreamento e diminuir a complexidade dos procedimentos seguintes. Esta restrição é possível porque o usuário alvo deste projeto possui movimentos de cabeça muito limitados ou mesmo ausentes. A Fig. 5.5 resume a etapa inicial de segmentação .

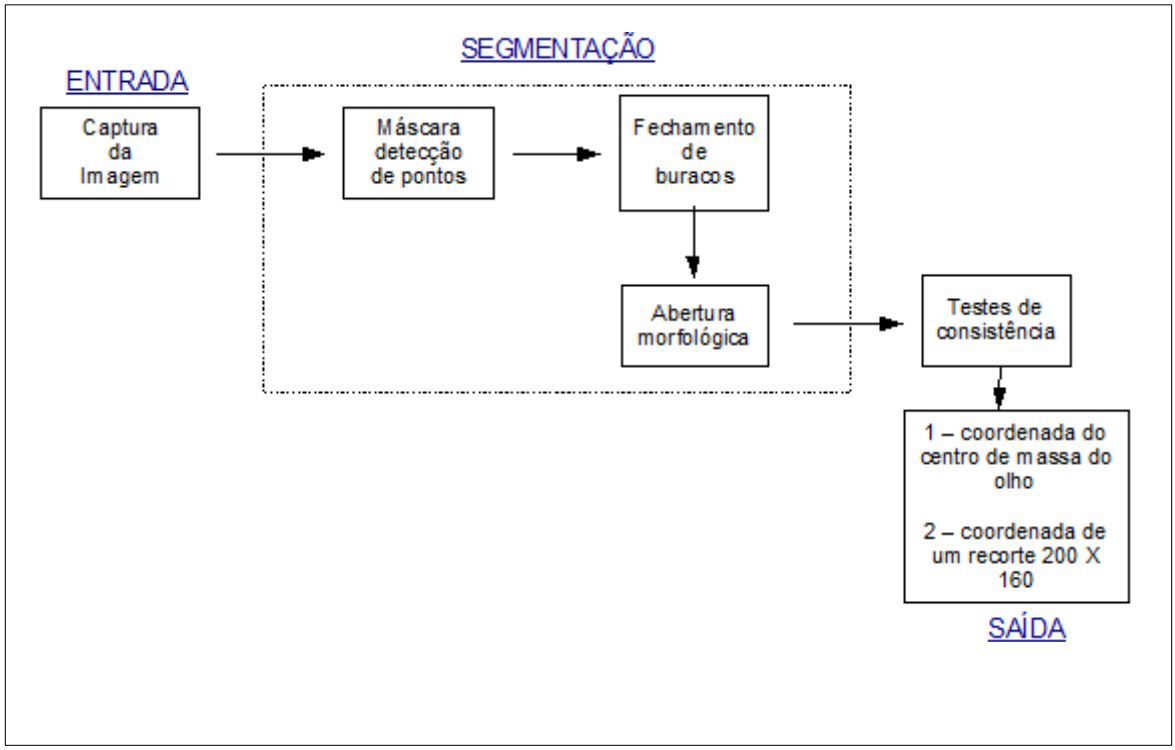


Fig. 5.5 – Diagrama do procedimento inicial de segmentação.

5.3 Rastreamento

As imagens são capturadas e digitalizadas a uma resolução de 640 X 480, entretanto, apenas a região do recorte de 200 X 160 pixels gravada na etapa anterior é processada durante toda a execução do sistema. A taxa de captura utilizada é de 10 frames/seg.

A cada 0,10 seg., um novo quadro é capturado e o processo de segmentação é repetido, porém sempre limitado à região do recorte de 200 X 160 pixels. No primeiro frame capturado após a inicialização do sistema, cada componente conectado encontrado é comparado aos valores de centro de massa e área do olho segmentado na etapa de localização dos olhos. Identificada a nova posição do olho do usuário, seus valores são atualizados de forma que a comparação é feita sempre com os dados do frame anterior. Quando o método não encontra a imagem da íris, ou seja, os olhos do usuário estão fechados mantém-se a coordenada da última posição da íris para a comparação no frame seguinte. Esta abordagem permite um certo grau de liberdade de movimentos do usuário suportando movimentos lentos que revelem mudança gradual na posição dos olhos e com a condição de que o olho do usuário esteja sempre localizado dentro da faixa de recorte definida. Movimentos bruscos podem levar à falha do rastreamento. Como o sistema é projetado para usuários com limitados ou mesmo ausência de movimentos de cabeça este fato não é considerado uma desvantagem relevante. A Fig. 5.6 mostra a área de liberdade de movimentos do usuário. A figura também destaca que o procedimento de segmentação durante o rastreamento ocorre em uma área significativamente menor que a segmentação inicial para a localização do par de olhos do usuário diminuindo a complexidade do método e aumentando o desempenho do sistema. A Fig. 5.7 apresenta um diagrama que ilustra os procedimentos executados no rastreamento.

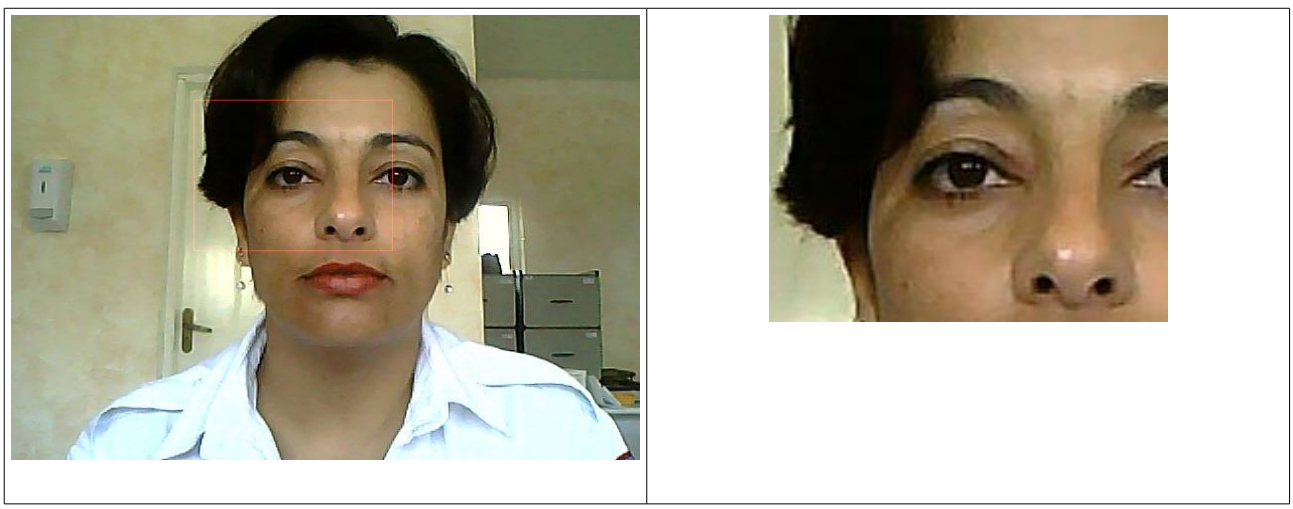


Fig. 5.6 – Região processada durante rastreamento: (a) imagem original; (b) recorte (200 X 160).

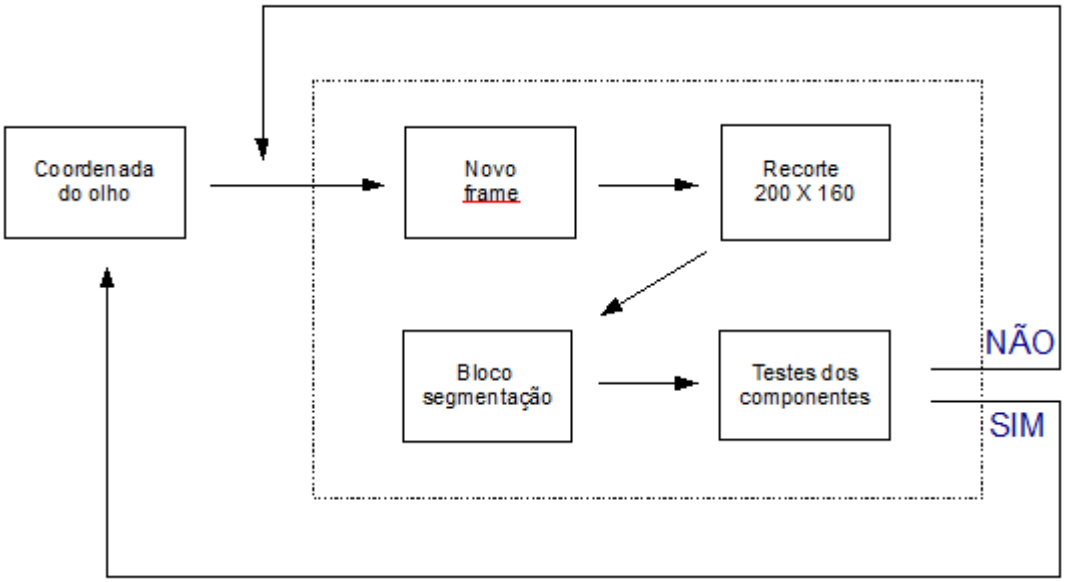


Fig. 5.7 – Diagrama da etapa de rastreamento.

5.4 Identificação das Piscadas

Após a localização da nova posição do olho do usuário em cada frame, usa-se um novo recorte agora de 80 X 40 pixels a partir de seu centro de massa para se identificar as piscadas voluntárias. Este novo recorte é feito na imagem de saída do processo completo de segmentação, ou seja, após a operação morfológica de abertura. Esta imagem, portanto, é uma imagem binária com duas regiões bem definidas, o plano de fundo e o componente conectado representativo da imagem da íris do usuário. Quando o olho do usuário se encontra fechado, ausência da imagem da íris, observa-se apenas a região do plano de fundo.

Para se identificar as piscadas de olhos foi então, utilizado o histograma da imagem separando-se os pixels pertencentes ao plano dos objetos e os pixels pertencentes ao plano de fundo da imagem. A presença ou ausência da imagem da íris representada por pixels brancos na imagem binária classifica o olho como aberto ou fechado. Um valor de *thresholding* é definido para se descartar eventuais ruídos na imagem e, consequentemente, a identificação de piscadas falso negativas. Esta condição é geralmente verificada em condições de baixa iluminação. Em situações com boa iluminação sem grandes ocorrências de sombras, quando os olhos do usuário estão fechados nenhum pixel de valor 1 é identificado na imagem dos olhos, revelando a completa ausência da imagem da íris. O valor de *thresholding* definido foi de 130 pixels baseados em testes

experimentais com 10 usuários. A Fig. 5.8 ilustra o procedimento de detecção de piscadas.

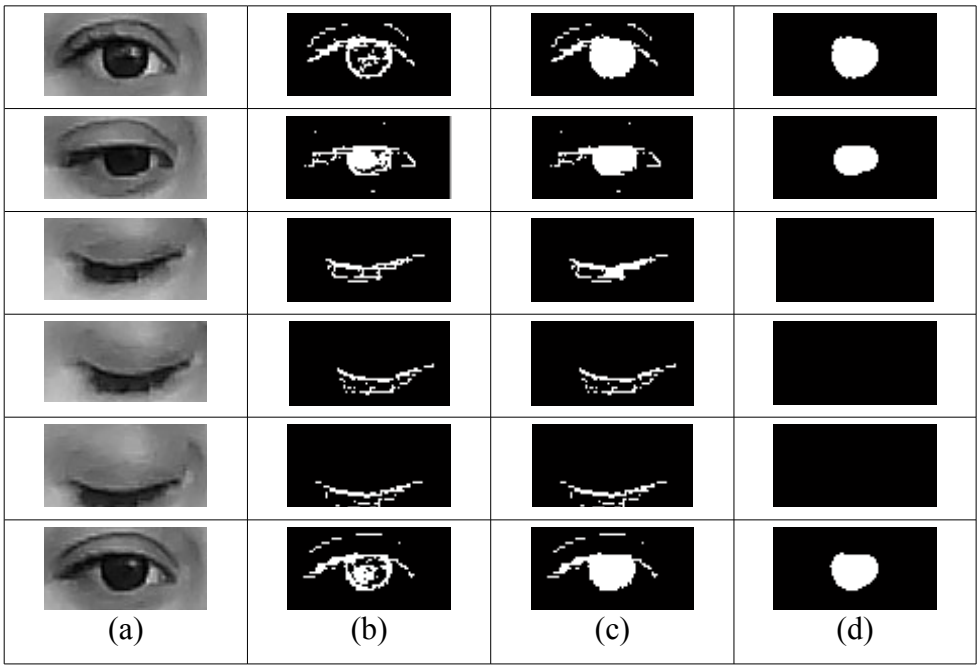


Fig. 5.8 – Processamento para reconhecimento de piscadas: (a) Imagem original convertida para intensidades de cinza; (b) imagem invertida após aplicação da máscara de detecção de pontos; (c) imagem após preenchimento de buracos; (d) imagem após abertura.

Para efeito de interpretação da intenção do usuário é necessário ainda classificar as piscadas como voluntárias ou involuntárias, e se voluntárias, como de curta ou longa duração. Piscadas naturais e involuntárias têm a duração média de 0,2 seg. A um desempenho de análise de 10 frames a cada segundo, uma piscada natural têm a duração, portanto, de 2 frames em média. Baseado neste fato, o sistema classifica a identificação do olho fechado por mais que 4 frames consecutivos como piscadas provocadas voluntariamente pelo usuário. Além disso, piscadas com a duração entre 4 a 10 frames são classificadas como curtas, e piscadas com a duração de 10 ou mais frames são classificadas como piscadas de longa duração. A Fig. 5.9 ilustra um gráfico do método de reconhecimento de piscadas. Além das piscadas identificadas com os marcadores da legenda, pode-se observar outras 3 piscadas voluntárias curtas e 2 piscadas involuntárias. As piscadas involuntárias, frames 66 e 67, e frames 69 a 71, são um exemplo típico de quando o usuário não consegue manter a piscada pelo tempo necessário para a classificação pelo sistema.

O procedimento de detecção das piscadas é robusto porque é executado apenas dentro da área segmentada dos olhos e o procedimento de classificação utilizado, embora simples, é

suficiente para o nível de análise desejada.

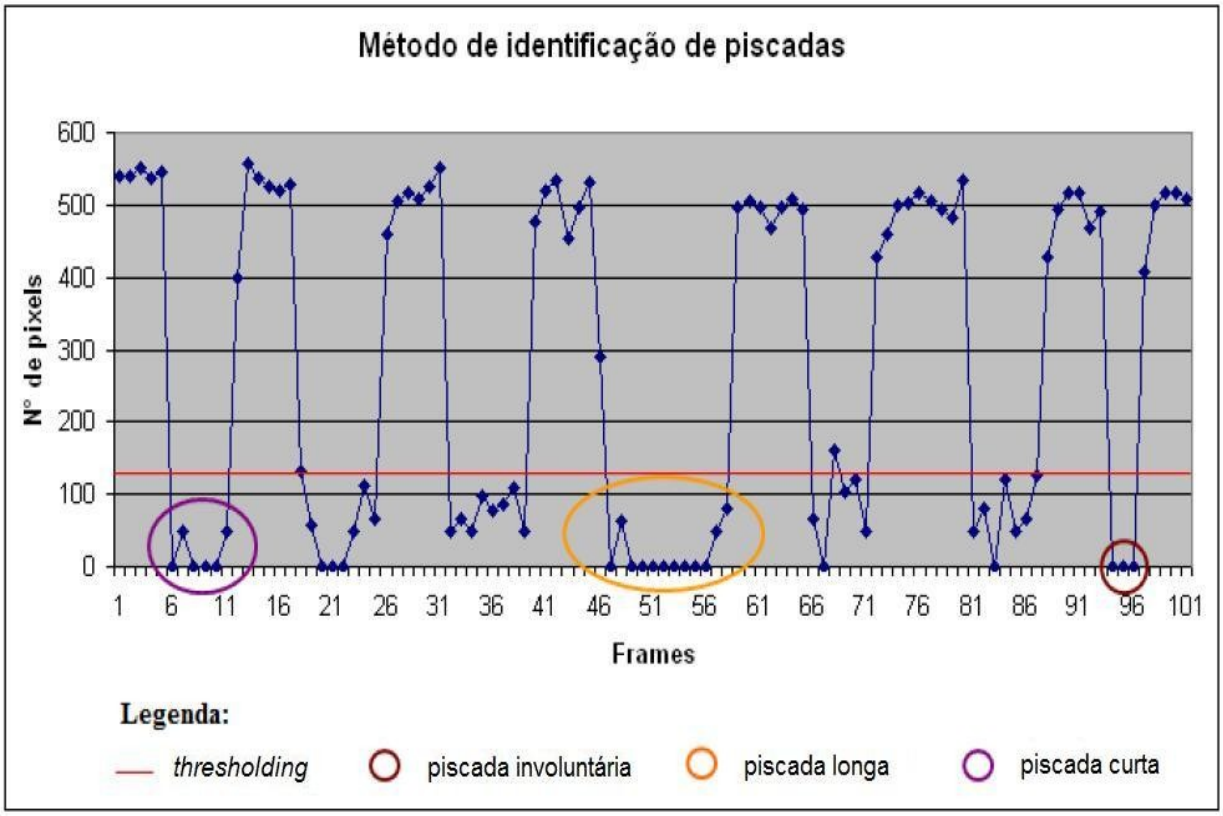


Fig. 5.9 – Reconhecimento de piscadas.

5.5 Resultados

O sistema foi testado com 11 pessoas não portadoras de deficiências físicas que foram orientadas para não se moverem bruscamente diante da tela do computador simulando a condição apresentada por um usuário tetraplégico. Também foi solicitado que o usuário se sentasse na frente da câmera a uma distância entre 30cm a 40cm e mantivesse a cabeça ereta no sentido lateral. Foi utilizado um notebook com processador Intel Core 2 Duo (1,67 Ghz), 2 GB de memória RAM e webcam integrada.

O sistema foi capaz de reconhecer os olhos de 10 usuários testados mesmo quando suas faces não se localizavam no centro da imagem capturada pela webcam. Porém, em algumas situações com iluminação não favorável, foram necessárias mais de uma tentativa para a correta

localização dos olhos. Antes de capturar o frame utilizado para a segmentação dos olhos, o sistema apresenta uma caixa de diálogo onde o usuário é solicitado a escolher o olho, direito ou esquerdo, que gostaria de utilizar para o controle da interface. A Fig. 5.10 mostra a interface de teste do método proposto logo após o procedimento de identificação da localização do olho direito da usuária. Durante a execução do sistema, o olho do usuário deverá sempre estar contido na região delimitada pela caixa vermelha. Note também que, na primeira localização, o olho escolhido se encontra no centro da área delimitada de 200 X 160 pixels.

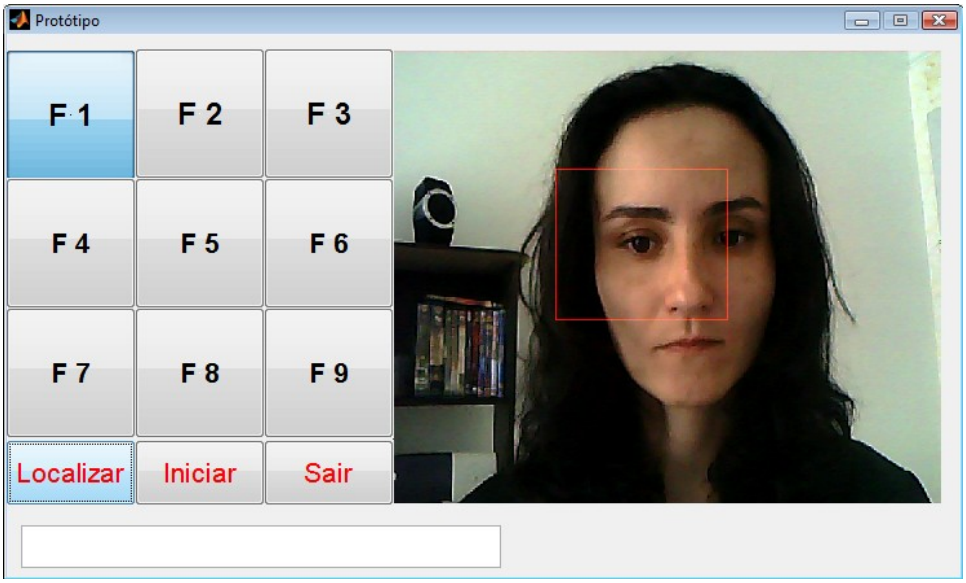


Fig. 5.10 – Interface do sistema de testes.

O método de segmentação se mostrou bastante robusto pois lida sempre com o mesmo tipo de imagem, imagens da face de um usuário, e objetiva segmentar sempre a mesma forma geométrica. Este é um cenário ideal para o uso de técnicas baseadas em morfologia matemática. O sistema é eficiente mesmo com pessoas de olhos não muito claros porque é baseado principalmente no formato dos olhos e, menos na distribuição de cor da íris. Nos testes, dois usuários possuíam olhos esverdeados.

O sistema também foi avaliado sob diferentes condições de iluminação mostrando excelentes resultados. Mesmo em situações de baixa iluminação como os testes realizados em períodos noturnos o sistema foi capaz de segmentar os olhos e identificar as piscadas dos usuários. Nem todos os 10 usuários realizaram os testes noturnos. Em situações de pouca iluminação o sistema costuma segmentar uma área maior que a região da íris do usuário em virtude da ocorrência

de sombras. Entretanto, a simplicidade do método aliada ao valor de *thresholding* definido para a classificação entre olhos abertos e olhos fechados faz com que o tamanho da região segmentada não interfira no sistema.

O sistema se mostrou falho apenas em situações de iluminação não uniforme nos dois lados da face do usuário. Este fato se justifica em virtude de ser o método de identificação dos olhos baseado na simetria da face humana. Também em virtude da não uniformidade de iluminação, quando o frame capturado apresenta regiões de elevada iluminação, como lâmpadas acesas, janelas abertas, etc., o método não se mostra eficaz.

Havendo sucesso na segmentação dos olhos, algumas piscadas falso negativas e falso positivas foram detectadas e justificadas pelo fato do usuário não segurar a piscada pelo tempo necessário para o sistema classificá-las ou por se mover de forma brusca ou para além da área de processamento da imagem capturada. Após um maior tempo de treinamento do usuário, as piscadas, tanto curtas quanto longas, foram corretamente classificadas.

Resumindo, o sistema proposto apresenta apenas dois sinais de entrada: olhos abertos ou olhos fechados. A combinação destes sinais faz com que um usuário tetraplégico consiga interagir com uma interface com menu do tipo *click-down*. Para esta abordagem nenhum aparato especial é necessário. O único hardware adicional utilizado como dispositivo de entrada é uma webcam que além de apresentar baixo custo, também não apresenta contato físico com o usuário.

A utilização de iluminação infravermelha facilita o processo de localização dos olhos do usuário além de produzir resultados mais precisos. A precisão na detecção do centro da pupila permite determinar a coordenada *x* e y que o usuário está olhando na tela possibilitando o controle do movimento do cursor apenas com o olhar. Entretanto, estes sistemas possuem as desvantagens de elevado custo e dos efeitos nocivos que esta iluminação pode causar ao usuário com o uso prolongado e frequente.

O rastreamento do ponto de visão do usuário não foi alcançado com o método proposto em virtude da baixa resolução da imagem utilizada no sistema. Em uma resolução de 640 X 480 pixels a diferença no centro de massa da imagem da íris do usuário olhando para o canto superior esquerdo da tela e a seguir para o canto superior direito mostrou ser apenas de 30 pixels. E a diferença no eixo vertical, ou seja, com o usuário olhando para o canto superior esquerdo da tela e a seguir, para o canto inferior também esquerdo, ficou em torno de 3 pixels (Fig. 5.11). Alguns

trabalhos como o de Betke, Gips e Fleming (2002), que realizam o rastreamento sem a utilização de iluminação infravermelha exploram o movimento do globo ocular conjugado com o movimento de cabeça do usuário. Liberdade de movimento logicamente é altamente desejado para o maior conforto do usuário não tetraplégico. Entretanto, o público alvo deste trabalho não possui movimentos de cabeça para a realização deste tipo de rastreamento.

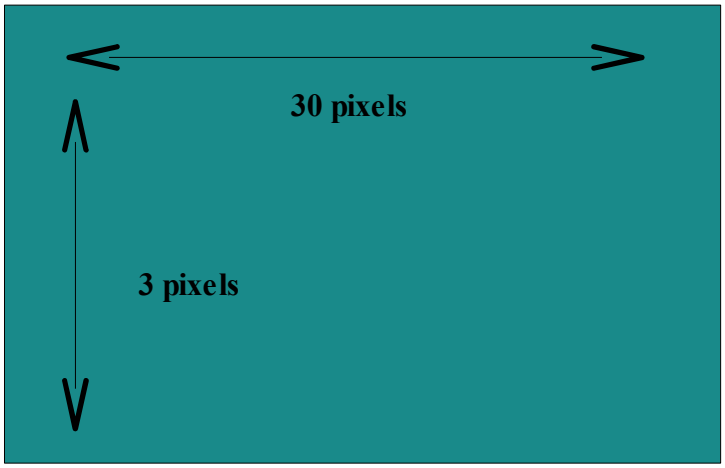


Fig. 5.11 – Dificuldade para rastreamento.

Também foi testado um periférico USB de I/O de baixo custo (Fig. 5.12) como interface eletrônica para o teste do acionamento de dispositivos externos como lâmpadas, campainhas, etc. A idéia consiste em usuários tetraplégicos poderem se beneficiar da integração desta interface com sistemas eletrônicos, aumentando sua independência e facilitando sua interação com o ambiente em que vivem. O controle do dispositivo foi realizado com a ferramenta Data Acquisition Toolbox do MATLAB.

Fig. 5.12 – Periférico eletrônico utilizado para os testes de integração.

5.6 Exemplo de Aplicação (Teclado Virtual)

Um aplicativo simulador de um teclado físico também foi brevemente testado com o método de detecção de piscadas proposto. O objetivo é apenas o de ilustrar as diversas aplicações que podem ser integradas com o método.

Para o maior conforto do usuário o aplicativo agrupou as letras do alfabeto de acordo com estudos sobre a sua maior prevalência na língua portuguesa (BRAGA, 2003). Com este agrupamento o caractere que exige um maior número de piscadas para ser alcançado é o caractere "W" , ou seja, o menos utilizado no português juntamente com os caracteres "Y" e "K". A Tabela 5.1 apresenta a taxa de ocorrência de cada caractere e o número correspondente de piscadas necessárias para que cada um deles seja alcançado no teclado virtual projetado.

Tabela 5.1 – Prevalência de caracteres do alfabeto na língua portuguesa e sua relação com o número de piscadas necessárias no teclado virtual.

Caractere	%	Nº Piscadas	Caractere	%	Nº Piscadas
A	14,64	2	**P**	2,58	6
E	12,70	3	**V**	1,68	6
O	10,78	3	**H**	1,42	7
S	7,97	4	**G**	1,28	7
R	6,88	4	**B**	1,16	7
I	5,90	4	**Q**	1,09	7
D	4,97	5	**F**	1,02	7
N	4,85	5	**Z**	0,42	8
M	4,71	5	**J**	0,32	8
U	4,42	5	**X**	0,23	8
T	4,26	6	**K**	0,01	9
C	3,76	6	**Y**	0,01	9
L	2,95	6	**W**	0,01	10

A Fig. 5.13 mostra a interface do aplicativo desenvolvido. Cada item selecionado no menu inicial apresenta um novo menu de opções com os caracteres pertencentes ao grupo

selecionado. Assim, para se acionar a letra “H” da palavra “Hello”, por exemplo, é preciso primeiro acionar o grupo “I M L H Z” e, dentro do sub-menu que surgirá a seguir, selecionar então o caractere “H”. Observe também na Fig.5.13 que são necessárias 3 piscadas para se acionar o grupo “I M L H Z” sendo 2 piscadas curtas e uma longa, e 4 piscadas (3 curtas e 1 longa) para acionar o caractere “H” no sub-menu da Fig. 5.14, perfazendo um total de 7 piscadas como ilustra a Tabela 5.1. Cada sub-menu apresenta também uma função para “Voltar” ao menu principal para o caso de detecção de piscadas falso positivas.

Figura 5.13 – Teclado virtual.

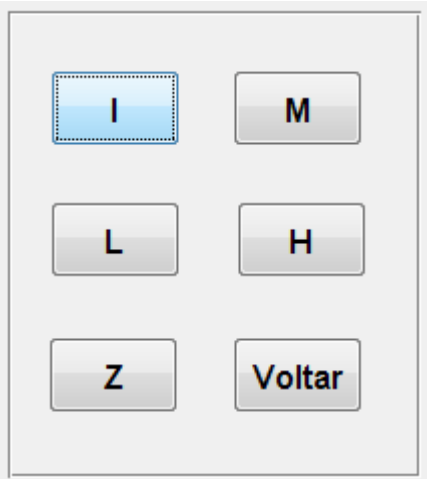


Figura 5.14 – Sub-menu: Grupo “I M L H Z”.

O uso do aplicativo em longas frases pode ser cansativo para o usuário mostrando que o método de detecção de piscadas tem maior aplicabilidade na integração com funções de suporte à vida do usuário deficiente. Entretanto, o aplicativo pode ainda ser considerado como uma alternativa para a comunicação e expressão destas pessoas. Além disto, algoritmos de predição de palavras baseados em técnicas de Inteligência Artificial integrados ao método podem contornar um pouco do problema de desgaste do usuário.

6 Conclusão

O estudo de técnicas computacionais alternativas para o auxilio a usuários portadores de tetraplegia que vivem na condição de dependência total de seus cuidadores foi o incentivo principal deste trabalho. Estas pessoas são, evidentemente, incapazes de utilizarem softwares de apoio com dispositivos de entrada padrões, como mouse e teclado, e também não possuem a capacidade de verbalização e consequente utilização de softwares com reconhecimento de comandos por voz.

O sistema desenvolvido explora, portanto, o único movimento sobre o qual estes usuários possuem controle, ou seja, o movimento de seus olhos; e pode ser integrado a sistemas diversos para o apoio à rotina de usuários tetraplégicos. Neste trabalho foram testadas a integração com um acionador de equipamentos eletrônicos e com a interface de um teclado virtual.

A técnica utiliza como dispositivo de entrada imagens digitalizadas do usuário capturadas por uma câmera. As imagens capturadas são processadas em tempo real com objetivo de se identificar as piscadas voluntários do usuário, permitindo a interação com um sistema de auxílio que independe de movimentos do mouse para navegação. O sistema executa em microcomputadores pessoais sem a necessidade de nenhum recurso de hardware especial além da webcam ou de acionadores eletrônicos para o caso da integração com equipamentos como cadeira de rodas, lâmpadas, campainhas, etc. Como a câmera não mantém contato com a cabeça do usuário, o sistema é confortável de se operar.

Entretanto, na integração do método com a interface do teclado virtual testado, o uso do aplicativo na escrita de períodos longos pode ser cansativo para o usuário. Algoritmos de predição de palavras baseados em técnicas de Inteligência Artificial integrados ao método podem diminuir

esta desvantagem.

O método proposto não utiliza nenhum algoritmo de reconhecimento de face, ou pixels representativos de pele para a localização automática dos olhos do usuário e, pode ser uma contribuição para outros sistemas dotados de técnicas de *eye tracker* que não se utilizam de iluminação infravermelha.

A iluminação pouco influenciou a eficiência do sistema. Apenas em condições de iluminação não uniforme ou de iluminação muito forte, o sistema não foi capaz de reconhecer a posição dos olhos do usuário. Movimentos de cabeça lentos e de pequena amplitude também não alteraram o desempenho e a eficiência do sistema proposto. Maior grau de rotação da cabeça, uso de óculos ou cabelos sobre a face do usuário podem, entretanto, influenciar negativamente o procedimento de segmentação.

O uso dos movimentos naturais dos olhos para se controlar o cursor pode ser mais confortável e intuitivo para o usuário. Em trabalhos futuros pretende-se tentar rastrear o foco de visão do usuário para que ele possa selecionar as funções da interface gráfica do sistema apenas com o seu olhar. Esta evolução possibilitaria a comunicação mais confortável do usuário através de um teclado virtual padrão. Este desenvolvimento facilitaria ainda a integração com sistemas comerciais como browsers permitindo que o usuário deficiente, por exemplo, adquira conhecimento ativamente navegando sozinho na Internet.

Referências Bibliográficas

BERGMAN, E.; JOHNSON, E. Towards Accessible Human-Computer Interaction. In: NIELSON, J. **Advances in Human-Computer Interaction.** New Jersey: Ablex Publishing Corporation, 1995. Cap. 4.

BERTRAND, G. Some properties of topological greyscale watersheds. In: IS&T/SPIE SYMPOSIUM ON ELECTRONIC IMAGING, 2004, San Jose. **Vision Geometry XII**, v. 5300, [S. l.: S. n.], 2004, p.127-137.

BETKE, M.; GIPS, J.; FLEMING, P. The Camera Mouse: Visual Tracking of Body Features to Provide Computer Access for People With Severe Disabilities. **IEEE Transactions on Neural Systems and Rehabilitation Engineering**, v. 10, n. 1, Mar., 2002.

BEUCHER, S. **Recent Advances in Mathematical Morphology.** Fontainebleau,[FR] Ecole des Mines de Paris, Centre de Morphologie Mathématique, 1992.

BRAGA, B. R. **Análise de Freqüências de Línguas.** Rio de Janeiro: COPPE / UFRJ. 24 de Março, 2003. Disponível em: <http://www.lockabit.coppe.ufrj.br/downloads/academicos/analise_freq.pdf> Acesso em : 30 maio 2010.

BYRNE, M. et al. Eye tracking the visual search of clickdown menus. In: CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS – CHI, 1999, Pittsburgh. **Proceedings...** New York: ACM Press, 1999, p. 402-409

BURGETH, B. et al. Morphological Operations on Matrix-Valued Images. In.: EUROPEAN CONFERENCE ON COMPUTER VISION, 8, 2004, Prague. **Proceedings**, Part IV, [S. l.: S. n.], 2004, p.155-167.

CANNY, J. A. Computational Approach to Edge Detection. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v.8, n.6, p.679-698, Nov., 1986.

CARVALHO, J. E. R. de. **Uma abordagem de segmentação de placas de automóveis baseada**

em morfologia matemática. 2006. 101 f. Dissertação (Mestrado em Computação) – Universidade Federal Fluminense, Niterói, 2006.

CHAU, M.; BETKE, M. **Real time eye tracking and blink detection with usb cameras.** 2005. 10 f. Technical Report – Boston University Computer Science, Boston, 2005.

CHAUDURI, B. B.; CHANDA, B. The Equivalence of Best Plane Fit Gradient with Robert's, Prewitt's and Sobel's Gradient for Edge Detection and a 4-Neighbour Gradient with Useful Properties. **Signal Processing**, North-Holland, v.6, n.2, p.143-151, Apr., 1984.

COSTER, M.; CHERMANT, J.-L. Image analysis and mathematical morphology for civil engineering materials. **Cement & Concrete Composites**, v.23, n.2-3, p.133-151, Apr.-June, 2001.

DAI, M. et al. Image segmentation by a dynamic thresholding using edge detection based on cascaded uniform filters. **Signal Processing**, v.52, n.1, p.49-63, July, 1996.

FELZENSZWALB, P. F.; HUTTENLOCHER, D. P. Image Segmentation Using Local Variation. In: IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 1998, Santa Barbara. **Proceedings...** 1998. p.98-106.

GAMBA, P.; LODOLA, R.; MECOCCI, A. Scene interpretation by fusion of segment and region information. **Image and Vision Computing**, v.15, n.7, p.499-509, July, 1997.

GEE, L. A.; ABIDI, M. A. Segmentation of Range Images Using Morphological Operations: Review and Examples. In: CONF. ON INTELLIGENT ROBOTS AND COMPUTER VISION XIV, 1995, Philadelphia. **Proceedings...** [S. l.]: SPIE, v.2588, 1995. p.734-746.

GONZALEZ, R. C.; WOODS, R. E. **Processamento de Imagens Digitais**. São Paulo: Edgard Blücher, 2000.

GONZALEZ, R. C.; WOODS, R. E.; EDDINS, S. L. **Digital Image Processing Using MATLAB.** New Jersey: Prentice Hall, 2004.

GRAUMAN, K. et al. Communication via eye blinks and eyebrow raises: video-based human-computer interfaces. **Universal Access In The Information Society**, v.2, n.4, p.359–373, Nov., 2003.

HEIJMANS, H. J. A. M. Mathematical Morphology: Basic Principles. In: SUMMER SCHOOL ON MORPHOLOGICAL IMAGE AND SIGNAL PROCESSING, 1995, Zakopane, [PL]. **Proceedings...** [S. l.: S. n.], 1995.

HORPRASERT, T.; HARWOOD, D.; DAVIS, L. S. A Robust Background Subtraction and Shadow Detection. In: ASIAN CONFERENCE ON COMPUTER VISION, 4, 2000, Taipei. **Proceedings...** [S. l.: S. n.], 2000. p.983-988.

HUANG, Y.; PAULUS, D.; NIEMANN, H.; Background-Foreground Segmentation Based on Dominant Motion Estimation and Static Segmentation. **Journal of Computing and Information Technology**, Zagreb, v.8, n.4, p. 349-353, 2000. Special Issue on ITI 2000 - Information Technology Interfaces.

JACOB, R. J. K. The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at Is What You Get. **ACM Transactions on Information Systems**, v.9, n.2, p. 152-169, Apr.,1991.

JACOB, R. J. K. Eye Movement-Based Human-Computer Interaction Techniques: Toward Non-Command Interfaces. **Advances in Human-Computer Interaction**, v.4, p. 151-190, 1993.

JACOB, R. J. K., SIBERT, L. E. Evaluation of Eye Gaze Interaction. In: SIGCHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 2000, Netherlands. **Proceedings...** [S. l.: S. n.], 2000, p. 281-289.

JAIMES, A.; SEBE, N. Multimodal Human Computer Interaction: a survey. **Computer Vision and Image Understanding**, v.108, n.1-2, p.116-134, Oct.-Nov., 2007.

KARATZAS, D.; ANTONACOPOULOS, A. Text Extraction from Web Images Based on A Split-and-Merge Segmentation Method Using Colour Perception. In: INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION, 17, 2004, Cambridge. **Proceedings...** [S. l.]: IEEE CS Press, v. 2, 2004, p.634-637.

KENNEDY, P. R. et al. Direct Control of a Computer from the Human Central Nervous System. **IEEE Transactions on Rehabilitation Engineering**, v.8, n.2, p.198-202, June, 2000.

LEZORAY, O.; CARDOT, H. Histogram and watershed based segmentation of color images. In: EUROPEAN CONFERENCE ON COLOR IN GRAPHICS, IMAGING AND VISION, 1, 2002, Poitiers. **Proceedings...** [S. l.: S. n.], 2002. p.358-362.

MANSOOR, A. B. et al. Fuzzy Morphology for Edge Detection and Segmentation. In: INTERNATIONAL SYMPOSIUM ON VISUAL COMPUTING, 3, 2007, Lake Tahoe, [USA]. **Proceedings…** [Berlin]: Springer, 2007. p.811-822. (Lecture Notes in Computer Science, v.4842).

MARCHAND-MAILLET, S.; SHARAIHA, Y. M. **Binary Digital Image Processing**: A Discrete Approach. San Diego: Academic Press, 2000.

MARR, D.; HILDRETH, E. Theory of Edge Detection. In: ROYAL SOCIETY BIOLOGICAL SCIENCES **Proceedings B…** [S. l.]: Royal Society Publishing, v.207, n. 1167, p.187-217, Feb., 1980.

MORRIS, T.; BLENKHORN, P.; ZAIDI, F. Blink detection for real-time eye tracking. **Journal of Network and Computer Applications**, v.25, n.2, p.129-143, Apr., 2002.

MUÑOZ, X. et al. Strategies for image segmentation combining region and boundary information. **Pattern Recognition Letters**, v.24, n.1-3, p.375-392, Jan., 2003.

NAIN, N. et al. Dynamic Thresholding Based Edge Detection. In: WORLD CONGRESS ON ENGINEERING, 2008, London. **Proceedings...** [S. l.: S. n.], v. 1, 2008. p.694-699.

NAJMAN, L.; COUPRIE, M. Watershed algorithms and contrast preservation. In: INTERNATIONAL CONFERENCE ON DISCRETE GEOMETRY FOR COMPUTER IMAGERY, 11, 2003, Naples. **Proceedings...** Berlin: Springer, v. 2886, 2003. p. 62–71.

NAJMAN, L.; COUPRIE, M.; BERTRAND, G. Watersheds, mosaics, and the emergence paradigm. **Discrete Applied Mathematics**, v. 147, n.2-3, p. 301-324, Apr., 2005.

NORIEGA, P.; BASCLE, B.; BERNIER, O. Local Kernel Color Histograms for Background Subtraction. In: INTERNATIONAL CONFERENCE ON COMPUTER VISION THEORY AN APPLICATIONS – VISAPP, 1, 2006, Setúbal. **Proceedings...** [S. l.]: INSTICC Press, v.1, 2006, p.213–219.

PARKER, J. R. **Algorithms for Image Processing and Computer Vision**. New York: Wiley Computer Pub., 1996.

POOLE, A.; BALL, L. J. Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects. In: GHAOUI, C. **Encyclopedia of Human-Computer Interaction.** Hershey: Idea Group, 2006. p.211-219.

PORTA, M. Vision-based user interfaces:methods and applications. **International Journal of Human-Computer Studies,** v.57, n.1, p. 27-73, July, 2002.

Portal Nacional de Tecnologia Assistiva. Ministério da Ciência e Tecnologia. Espaço de informação e troca de conhecimento sobre Tecnologia Assistiva. Disponível em: <http://www.assistiva.org.br/>. Acesso em: 04 fev. 2010.

RAO, R. L.; PRASAD, L. **Segmentation by Multiresolution Histogram Decomposition.** 1995. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.49.6385>. Acesso em: 02 mar. 2010.

ROBINSON, G. S. Detection and Coding of Edges Using Directional Masks. **Optical Engineering**, v.16, n.6, p.580-585, Nov.-Dec. 1977.

ROERDINK, J. B. T. M. Computer Vision and Mathematical Morphology. In: THEORETICAL FOUNDATIONS OF COMPUTER VISION, 7, 1994, Dagstuhl. **Proceedings...** [S. l.]: Springer, 1996. p.131-148. (Computing Supplement, v.11).

SHIH, C-H.; CHANG, M-L.; SHIH, C-T. Assisting people with multiple disabilities and minimal motor behavior to improve computer pointing efficiency through a mouse wheel. **Research in**

Developmental Disabilities, v.30, n.6, p.1378-1387, Nov.-Dec. 2009.

SKARBEK, W.; KOSCHAN, A. **Colour Image Segmentation** : a survey. 1994. 81 f. Technical Report – Technical University of Berlin, Berlin, 1994.

STERNBERG, S. R. Grayscale Morphology. **Computer Vision, Graphics, and Image Processing**, v.35, n.3, p.333-355, Sept. 1986.

VARONA, J.; MANRESA-YEE, C.; PERALES, F. J. Hands-free vision-based interface for computer accessibility. **Journal of Network and Computer Applications**, v.31, n.4, p.357-374, Nov. 2008.

VINCENT, L.; SOILLE, P. Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v.13, n.6, p.583-598, June, 1991.

VINCENT, O. R.; FOLORUNSO, O. A Descriptive Algorithm for Sobel Image Detection. In: INFORMING SCIENCE & IT EDUCATION CONFERENCE – InSITE, 2009, Macon, [US]. **Proceedings...** [S. l.: S. n.], 2009. p.97-107.

WARE, C.; MIKAELIAN, H. H. An evaluation of an eye tracker as a device for computer input. In: CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS AND GRAPHICS INTERFACE – CHI/GI, 1987, Toronto. **Proceedings...** New York: ACM Press, 1987. p.183-188.

WRIGHT, A. S.; ACTON, S. T. Watershed Pyramids for Edge Detection. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING, 1997, Santa Barbara. **Proceedings...** [S. l.: S. n.], v.2, 1997. p.578-581.

ZIOU, D.; TABBONE, S. Edge Detection Techniques -: an overview. **Pattern Recognition and Image Analysis**, v.8, n.4, p.537-559, Dec. 1998.

Livros Grátis

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

Milhares de Livros para Download:

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