Resumo

A transposição de documentos para o meio digital se constitui em uma importante atividade para a sociedade nos dias atuais. A principal razão disso é a preservação, visto que essa forma de armazenamento é bem mais resistente à ação do tempo do que o papel. Para possibilitar a divulgação de conteúdo dos documentos, já que as imagens ocupam grande espaço de armazenamento em memória, surgem as aplicações de Reconhecimento Óptico de Caracteres (OCR), as quais visam a transformar o conteúdo dos documentos digitalizados em texto editável. Além disso, elas permitem a busca por palavras-chave além de facilitar a indexação e catalogação dos acervos. Uma etapa crucial para os sistemas de OCR é a segmentação de caracteres. Quando se tratam de documentos manuscritos, as dificuldades envolvidas nesse processo aumentam bastante. Os principais problemas dessa tarefa se devem às variações de escrita tanto de pessoas diferentes, como de uma mesma pessoa com o passar do tempo. Como principais obstáculos à separação de caracteres, podem-se citar casos de toque, disjunção e sobreposição entre eles. Este trabalho faz um estudo acerca da segmentação de dígitos manuscritos. Quatro algoritmos de segmentação são expostos e um estudo detalhado em um deles é apresentado, sendo implementado em MATLAB. Alguns experimentos também são apresentados ao final.

Abstract

The transposition of paper documents to digital media constitutes an important task to society nowadays. The main reason is the preservation of its contents, since this manner of storage has much greater durability than paper. Optical Character Recognition (OCR) aims to transform document digital images into editable text. These applications appear to allow the divulgation of document contents, since images require large storage space in memory. Besides, they give the possibility of searching for keywords and making indexation easier. Character segmentation is a crucial stage in OCR systems. When one treats manuscript documents, this process becomes more complex. Variations in handwriting of different or even the same person are the principal source of difficulties. Touching, disjointed and overlapping characters can be pointed out as the main problems to character separation. This work makes a study about segmentation of handwritten digits. It presents four segmentation algorithms and a detailed study in one of them, which was implemented in MATLAB. Some results and experiments are also presented.

Sumário

Índice de Figuras

Índice de Tabelas

Tabela de Símbolos e Siglas

ASCII – *American Standard Code for International Interchange* (Padrão de Codificação Americano para Intercâmbio Internacional).

BMP – Windows bitmap (mapa de bits do Windows)

CD – *Compact Disc* (Disco Compacto).

DVD – *Digital Video Disc* (Disco de Vídeo Digital).

OCR – *Optical Character Recognition* (Reconhecimento Óptico de Caracteres).

Capítulo 1

Introdução

O papel foi uma das maiores revoluções tecnológicas da humanidade. Ele substituiu todas as formas anteriores de armazenamento e difusão da informação e ainda hoje, é o meio mais utilizado para esse fim. No entanto, sua fragilidade, grande quantidade de espaço demandada para seu armazenamento, e também, a dificuldade de busca por informações específicas nos documentos são suas principais desvantagens. Atualmente, uma alternativa viável e vantajosa para a solução de tais problemas é a utilização de recursos computacionais.

A criação de dispositivos digitalizadores (*scanners* e câmeras digitais) tornou possível a transposição de documentos para computadores na forma de imagens. Dessa forma, é possível o armazenamento de imagens de documentos, visando a uma proteção mais eficiente ao desgaste provocado pelo tempo. Armazenados em dispositivos de memória secundária: sejam ópticos, como discos Blu-Ray, DVD's ou CD's, ou magnéticos como discos rígidos, fitas, discos Zip, etc., é inteiramente viável a cópia de acervos completos de documentos para outro dispositivo de armazenamento sem nenhuma perda de dados, posterior à digitalização.

Talvez, o ponto crítico dessa tecnologia seja o grande espaço em *bytes* necessário para armazenar as imagens. Uma imagem de documento digitalizado pode ocupar mais de 4 *megabytes*, quando armazenada no formato padrão do sistema operacional Windows, o BMP (*bitmap*). Quando essas imagens são relativas a texto, técnicas de análise e processamento de documentos [1][2] podem convertê-las para o formato de texto editável. E o arquivo texto contendo a mesma informação da imagem, pode ocupar menos de 100 *kilobytes* na memória.

Uma transposição não automática é inaceitável devido aos custos envolvidos, à baixa velocidade e a baixa confiabilidade do processo. Um sistema automático deve reconhecer os caracteres presentes no documento, diferenciando-os de imagens ou outros dados que possam estar presentes, e transpô-los para caracteres ASCII (padrão computacional de caracteres). Esse processo é chamado de Reconhecimento Óptico de Caracteres (OCR).

O espaço ocupado por um arquivo de texto é centenas de vezes menor que o ocupado por uma imagem, além de possibilitar a execução de mecanismos de busca por palavras-chave. Para o processo de digitalização, as dificuldades estão, basicamente, em dois pontos: a escolha do melhor método para realizar a transposição e os melhores ajustes de parâmetros para digitalização de resolução, brilho, contraste, número de cores, etc.

Definições errôneas podem provocar uma atuação ineficiente dos algoritmos de reconhecimento e podem requerer uma nova digitalização ou um processamento específico da imagem. Ainda mais, por diversas vezes, um processamento na imagem só se torna viável quando conhecidos os ajustes utilizados na digitalização, o que nem sempre acontece.

O objetivo do Reconhecimento Óptico de Caracteres é o mesmo tanto para documentos manuscritos, quanto para datilografados. Porém, devido à grande quantidade de características distintas inerentes à escrita de cada pessoa, o reconhecimento de textos manuscritos ainda é um ponto em estudo. Para documentos datilografados, dependendo da qualidade do papel e da tinta, o problema já está praticamente resolvido com altas taxas de acerto.

Além do uso de OCR's para diminuição do espaço de armazenamento, a criação de livros digitais surge hoje como uma área em grande expansão. Embora o ser humano esteja mais acostumado com o uso de papel e ainda o considere o melhor método para leitura, os livros digitais têm evoluído bastante em termos de interface ao longo dos anos. Novos livros podem ser gerados diretamente no computador, mas a transposição da literatura já existente para o universo digital necessita do uso de ferramentas eficientes para reconhecimento automático de caracteres.

Chegou a se pensar que com o emprego da tecnologia digital o papel deixaria de ser utilizado, o que não ocorreu. Na realidade, o papel ainda é muito utilizado, sendo uma das principais formas de mídia e armazenamento de informações e o seu consumo ainda aumenta devido à grande quantidade de informação gerada na atualidade.

Apesar da importância do papel para a civilização humana, ele apresenta uma série de desvantagens na sua utilização. Em primeiro lugar, a sua fabricação

depende da celulose, material extraído de árvores e, desta forma, leva ao desmatamento desenfreado de grandes áreas florestais. Em segundo lugar, o papel se desgasta com o tempo tornando-o amarelado e frágil. Caso não seja conservado em um ambiente adequado, pode sofrer a ação de fungos e insetos.

Outra desvantagem é o custo relacionado ao papel. O valor do papel é extremamente barato e, portanto, pode acarretar em um consumo descontrolado. Além disso, o valor gasto com materiais relacionados à sua utilização, como por exemplo, cartuchos de impressoras, lápis, pastas, clipes e pranchetas, é elevado.

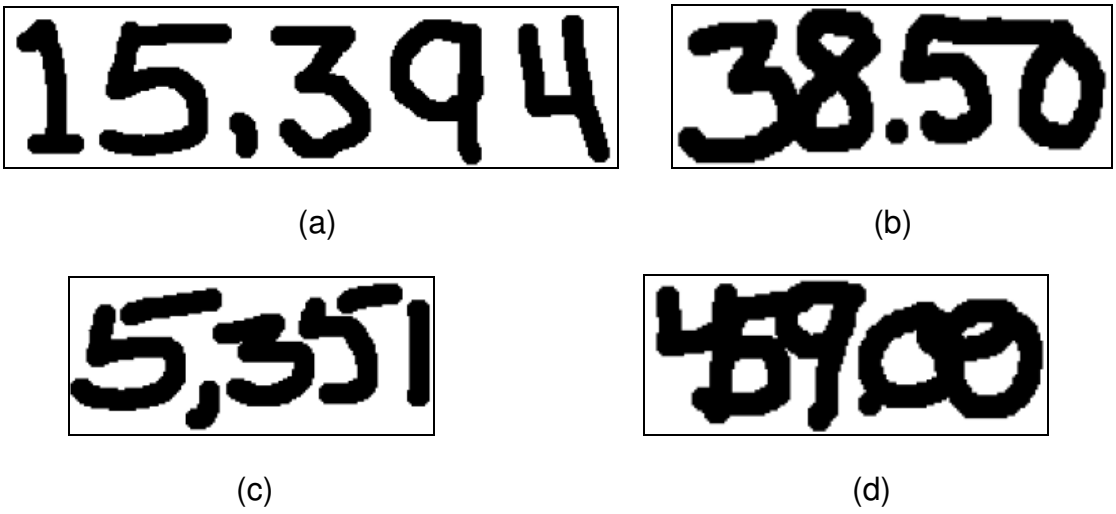
Por último, o papel ocupa um grande espaço físico. Em empresas públicas e privadas, tornou-se comum o uso de estantes e salas de arquivos para armazenar e organizar os documentos mais importantes, como por exemplo, contratos, notas fiscais, relatórios, manuais, etc. As empresas maiores, com grandes quantidades de papéis, terceirizam o serviço de armazenamento e organização dos documentos que são guardados em galpões.

Em alguns domínios, as vantagens do uso da tecnologia digital se multiplicam. Um bom exemplo são os documentos históricos, pois a digitalização facilita o acesso a um acervo, preserva os documentos originais, já que os mesmos não precisam ser manipulados constantemente. Além disso, ainda possibilita maior divulgação do acervo, por exemplo, através da Internet. Outro exemplo a ser citado são os cheques bancários, já que os mesmos têm valor financeiro diretamente agregado e os processos de reconhecimento precisam ser bastante eficazes, além de documentos e cartas diversas que precisem ser armazenados seja por obrigação legal, preservação de conteúdo, ou outra razão específica.

No contexto de OCR, a segmentação [1] é o primeiro passo de muitas aplicações de processamento de imagens e visão computacional. Uma etapa crucial para que haja uma classificação correta dos caracteres presentes no texto é a segmentação de caracteres. Isso porque mesmo um bom classificador pode não corrigir os erros vindos da fase de segmentação. Observa-se então que é necessário apresentar os padrões corretos para que eles sejam classificados de forma satisfatória.

Os principais obstáculos para essa tarefa são caracteres degradados, conectados ou sobrepostos. Além do fato da segmentação de caracteres

manuscritos, como estudado nesse projeto ser mais complexa devido a variações no estilo da escrita, ângulo e espaçamento entre os caracteres do texto. Esses problemas são comuns em documentos ou campos de documentos como códigos postais em cartas, ou o campo numérico de cheques bancários. A Figura 1 mostra exemplos de imagens de dígitos e problemas decorrentes da escrita manuscrita.



(a) (b)

(c) (d)

Figura 1. Exemplos de imagens de dígitos manuscritos, (a) bem escritos, (b) conectados, (c) disjuntos, (d) sobrepostos.

Em documentos históricos, um exemplo da importância de segmentar os dígitos corretamente, para obviamente reconhecê-los posteriormente, é a data do documento a qual é utilizada para fins de organização de acervo ou para métodos de busca automáticos [2]. Em cheques bancários, para um processo mais eficiente, em diversas aplicações, processa-se apenas o campo numérico (dimensões reduzidas em relação ao total do cheque) [3], salientando assim como é importante segmentar dígitos de maneira correta.

1.1 Objetivos

A partir do exposto, o presente trabalho tem como objetivo principal o estudo sobre segmentação de caracteres, de forma a dar suporte a estudos e ao desenvolvimento de novos métodos e trabalhos na área, como por exemplo, o trabalho desenvolvido em [4]. Como objetivos específicos, pode-se citar a implementação do algoritmo de segmentação baseado na arquitetura de multiagentes inteligentes [5] proposta por

Alhajj e Elnagar em [6], e o estudo das técnicas propostas em [7-9]. No caso, é feita a avaliação e comparação entre os métodos estudados e implementados.

Este trabalho faz parte de uma das linhas de pesquisa do Programa de Pós-Graduação em Engenharia da Computação do Departamento de Sistemas e Computação da Escola Politécnica de Pernambuco. O trabalho também se insere nos esforços do Grupo de Pesquisa em Reconhecimento de Padrões da Universidade de Pernambuco.

1.2 Organização do Trabalho

Este trabalho está organizado em cinco capítulos. Neste Capítulo 1, expõem-se o contexto do problema tratado, mostrando sua importância e dificuldades, bem como a motivação e objetivos do trabalho. No Capítulo 2, a fim de se ter uma fundamentação teórica para o entendimento do trabalho, mostram-se as etapas do processamento de imagens de documentos, além dos problemas relacionados às mesmas, principalmente, os relacionados à segmentação.

No Capítulo 3, explicam-se em detalhes, as técnicas abordadas no presente trabalho. O Capítulo 4 apresenta os resultados obtidos com os experimentos realizados como também a análise deles. Por fim, o Capítulo 5 apresenta as considerações finais, conclusões, contribuições e propostas de trabalhos futuros a esta monografia.

Capítulo 2

Processamento de Imagens de Documentos

O processo de Reconhecimento Óptico de Caracteres não se resume à classificação de caracteres. Na realidade, um sistema de processamento de documentos para OCR é composto por diversas fases [2], a saber: (i) Limiarização ou Binarização, (ii) Pré-processamento, (iii) Segmentação de Documento, (iv) Segmentação de Texto, (v) Extração de Características e por fim, (vi) Classificação. A Figura 2 mostra o fluxo entre as diversas fases do processamento de documentos para OCR.

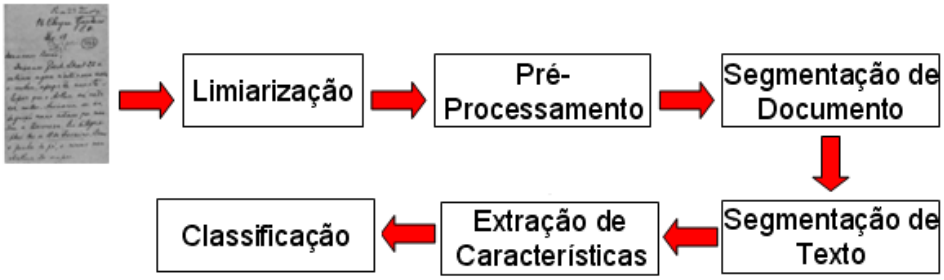


Figura 2. Principais etapas de um sistema de processamento de documentos para Reconhecimento Óptico de Caracteres.

Cada etapa nesse processo tem objetivos, características e problemas peculiares, os quais serão abordados ao longo deste capítulo. É importante salientar que o desempenho de cada fase afeta os resultados da etapa seguinte. Sendo assim, para um sistema de OCR eficiente, é necessário que haja resultados satisfatórios em cada parte do processo. De outra forma, pode-se dizer que a entrada para determinada fase deve ser correta para que seja possível executá-la com sucesso.

2.1 Binarização

O processo de binarização ou limiarização [10] consiste em reduzir para dois o número de níveis de cor em uma imagem digital. O nome limiarização vem do fato

de se definir, neste tipo de operação, um limiar o qual é utilizado para definir quais *pixels*[1] serão convertidos em um tom ou em outro. Já o nome binarização decorre do fato de a imagem resultante ter dois níveis de cor, sendo assim uma imagem binária. Geralmente, se utilizam a cor preta e a cor branca para denotar os dois níveis resultantes da binarização. Em documentos escritos, sejam datilografados, impressos ou manuscritos, usualmente a cor branca representa o fundo da imagem e a cor preta o texto.

Em várias aplicações, os níveis de cinza do fundo da imagem são bem distintos ou pelo menos notadamente diferentes dos níveis de cinza dos objetos da imagem (sejam figuras ou texto) [11]. Em imagens de documentos, também se percebem diferenças entre os níveis de cor entre os *pixels* referentes aos objetos e aos referentes ao fundo da imagem. E para documentos, esse fato torna viável o uso desse tipo de operação como um processo de segmentação já que se presta a separar os objetos e o fundo da imagem.

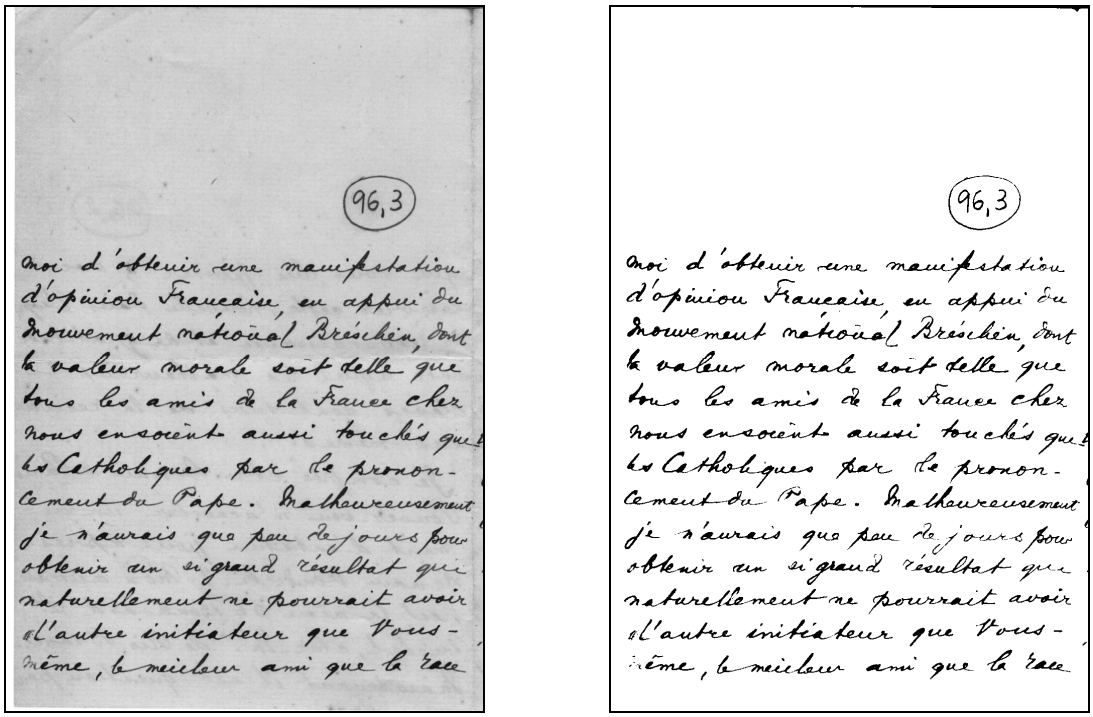
Normalmente, têm-se imagens em tons de cinza como entrada para os diversos métodos de limiarização, embora isso não seja obrigatório. Sendo assim, quando se tem como entrada imagens no formato *true color*[2], geralmente se aplica um processo de quantização [12] o qual visa a diminuir o espaço de cores da imagem original. Nesse caso específico, uma redução para 256 tons de cinza. E no caso, a binarização é de fato um processo de quantização para dois níveis.

Quando os tons de cinza que constituem o fundo e os objetos da imagem estão bem separados, a tarefa de binarizar é relativamente simples. A Figura 3 ilustra um exemplo no qual o processo de binarização obtém resultados satisfatórios. Observa-se que a imagem original é de boa qualidade, ou seja, não apresenta distorções ou ruídos que não representam informação na imagem.

---

[1] *Pixel* (*Picture element* ou em português, elemento da imagem) corresponde a cada ponto ou coordenada de uma imagem digital.

[2] *True color* é um formato de codificação de cores de imagens digitais utilizado nos computadores. Nesse formato, cada *pixel* é representado por 24 bits, possibilitando mais de 16 milhões de cores para o mesmo.

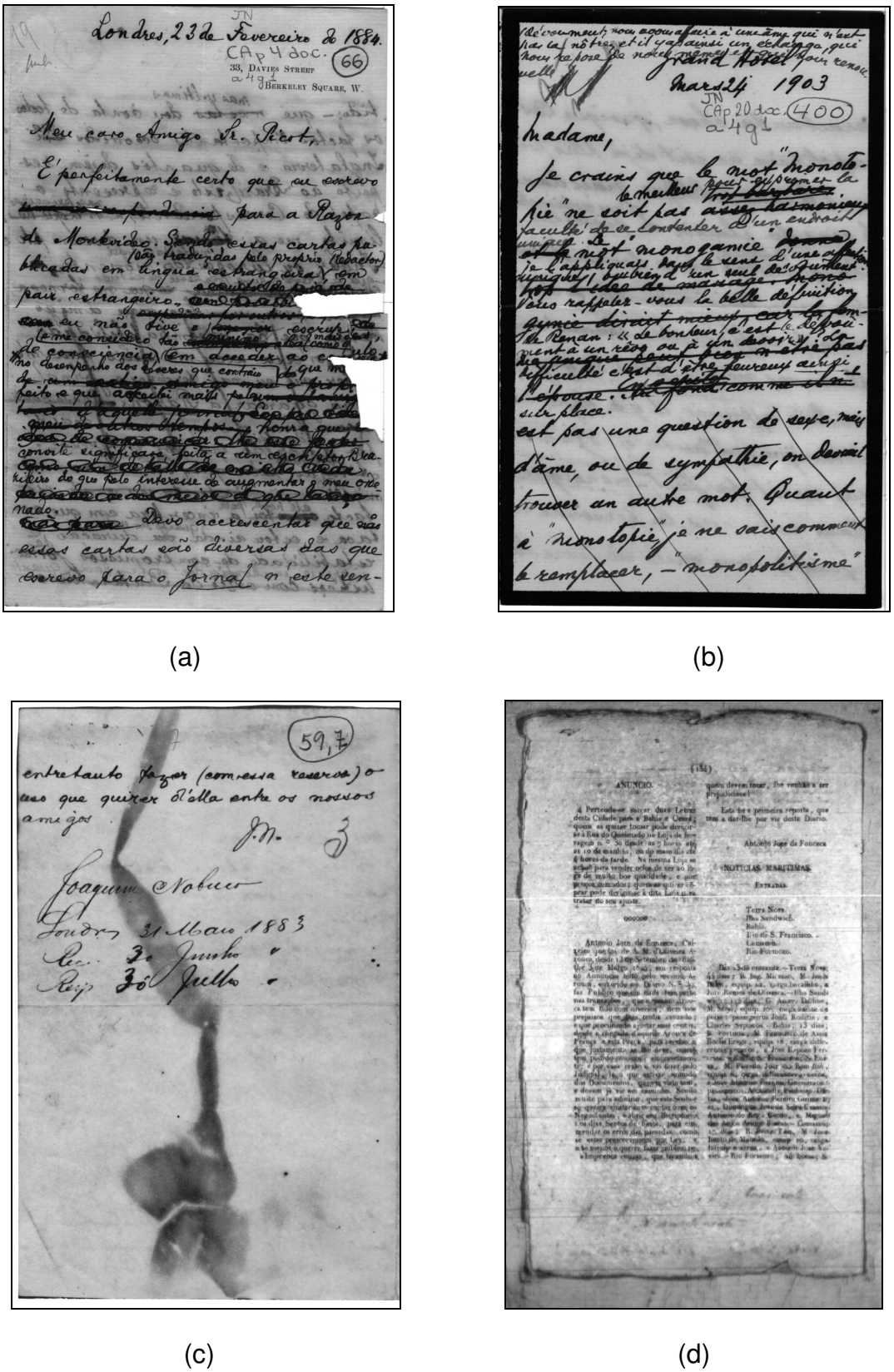


(a) (b)

Figura 3. Exemplo de aplicação da binarização, (a) imagem original, (b) imagem binarizada.

Porém, em diversas aplicações, surgem problemas que dificultam bastante o processo de binarização. Exemplos disso são os documentos históricos que em muitos casos, apresentam os seguintes problemas: (i) tinta bastante degradada o que aproxima os tons de cinza do texto e do fundo da imagem; (ii) interferência da parte de trás escrita quando o documento é escrito dos dois lados de uma folha; (iii) rasuras ou manchas contidas no documento original ou decorrentes da digitalização. A Figura 4 mostra imagens com os problemas apresentados no texto.

Em cheques bancários [3], a inserção de padrões aleatórios e marcas d'água no fundo da imagem por questões de segurança se tornam obstáculos à binarização. Além disso, esses padrões são diferentes nas partes específicas dos cheques, além do que, cada banco utiliza seus próprios padrões, não havendo uma uniformização geral. Para cheques de um mesmo banco, os padrões empregados geralmente são similares [13].



(a) (b)

(c) (d)

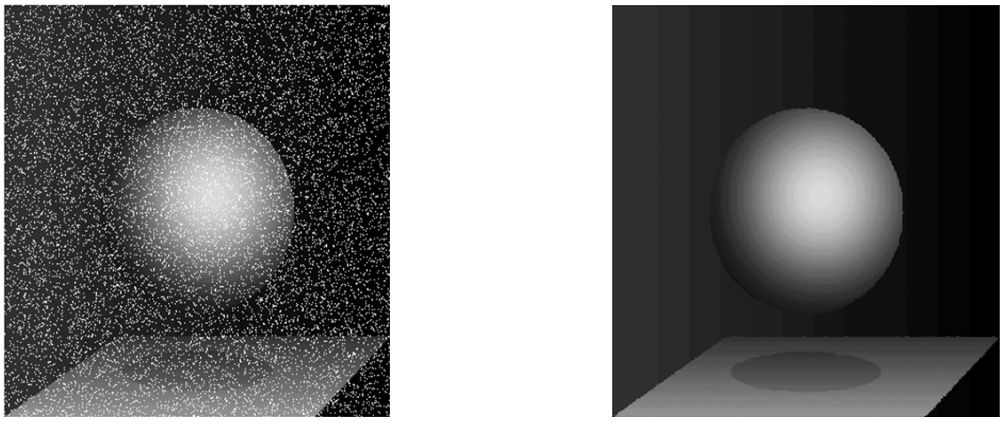
Figura 4. Principais problemas enfrentados pelas técnicas de binarização em documentos históricos, (a) interferência, (b) bordas pretas, (c) manchas, (d) tinta degradada e variação de iluminação.

Há grandes dificuldades e peculiaridades de cada aplicação onde se aplica a binarização, e na prática, a impossibilidade de se criar um método universal o qual obtenha bons resultados para todo tipo de imagem. Por conta disso, há a tendência de se propor algoritmos para aplicações específicas como documentos históricos [10][14-17], cheques bancários [3][18], entre outros.

2.2 Pré-processamento

Em muitos casos, a binarização não consegue separar completamente o texto e o fundo do documento, restando ruídos (elementos que não se constituem em informação da imagem), como manchas por exemplo. O pré-processamento busca retirar ou diminuir tanto quanto possível os ruídos tanto decorrentes do processo de limiarização quanto da própria digitalização da imagem.

Várias técnicas podem ser utilizadas para esse fim. Um bom exemplo disso é a aplicação de filtros digitais [12][19]. A Figura 5 ilustra a aplicação de um filtro digital para redução de ruído, nesse caso específico o filtro estatístico da mediana [12]. Um sério problema encontrado nesta fase é que em muitas vezes, a aplicação de filtros requer o conhecimento do tipo de ruído presente na imagem o que quase sempre não é possível, principalmente quando os ruídos são advindos do processo de digitalização.



(a) (b)

Figura 5. Exemplo de filtragem digital para redução de ruído, (a) imagem original com ruído, (b) imagem filtrada.

Outro problema o qual é tratado nesta etapa é a correção de orientação do documento. Por diversas vezes, seja pelo processo de digitalização, seja pela própria caligrafia do autor, o texto ou a própria imagem não apresenta um enquadramento perfeito. Assim, nessa fase se procura corrigir a orientação da imagem quando necessário.

Um algoritmo bastante utilizado para esse fim é a Transformada de Hough [19] a qual fornece como resposta o ângulo de inclinação da imagem, possibilitando assim a correção na orientação. A Figura 6 mostra a aplicação da Transformada de Hough para detectar a inclinação de uma imagem de documento a qual é posteriormente corrigida [19].

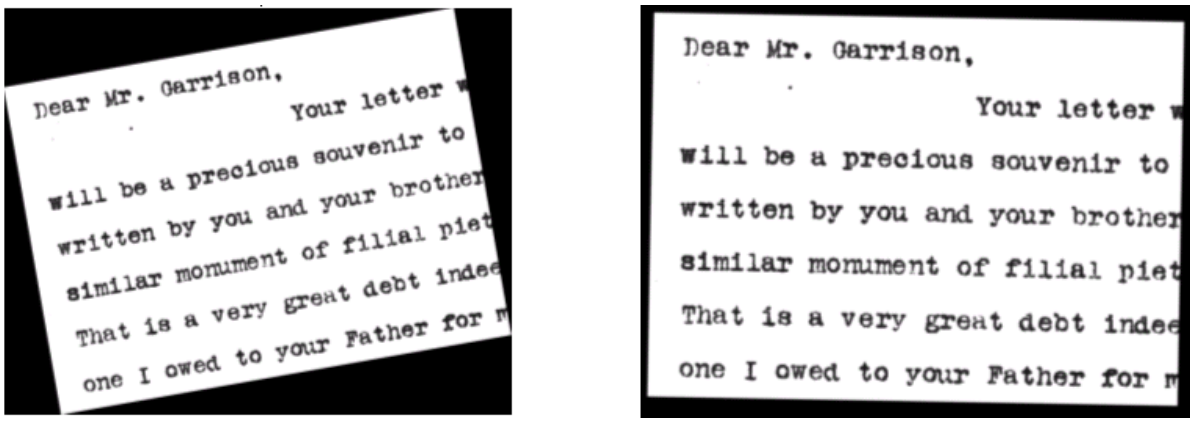


Figura 6. Aplicação da Transformada de Hough para detecção de inclinação da imagem: (a) imagem original, (b) imagem com a orientação corrigida.

2.3 Segmentação de Documento

A segmentação de documento [20] visa a identificar e assim poder separar áreas de texto e de objetos gráficos em uma imagem de documento. Isso porque uma imagem de documento pode conter mais do que caracteres apenas; pode ter gravuras, fotos ou se tratar de um cartão postal, e esses elementos não devem ser tratados pelo algoritmo de reconhecimento de caracteres.

Por isso, esse procedimento é bastante comum em ferramentas de OCR. Isso se deve ao fato de não se desejar fornecer ao reconhecedor de caracteres, elementos que não sejam de fato caracteres, pois isso acarreta desperdício de

processamento e erros no reconhecimento. A Figura 7 apresenta uma simulação de segmentação de documento, delimitando por linhas vermelhas as regiões detectadas em (a) e pode-se ver em (b) a classificação das regiões encontradas.

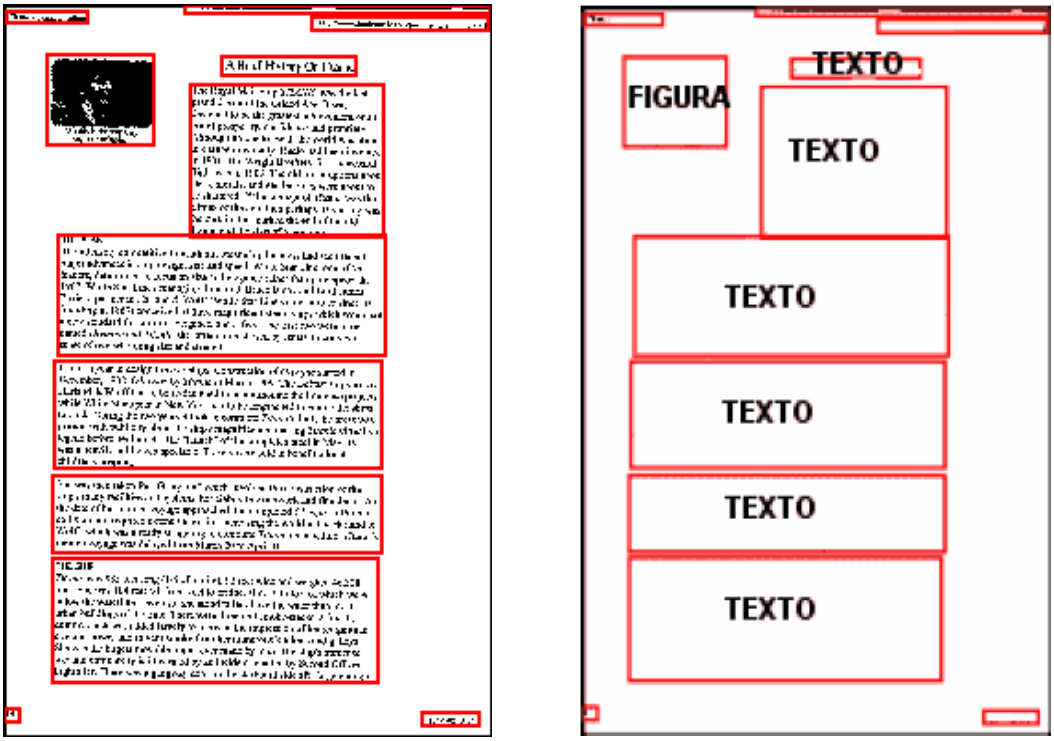


Figura 7. Simulação de segmentação de documento, (a) identificação de regiões através de análise da estrutura física do documento, (b) resultado: estrutura lógica com a classificação das regiões como figura ou texto[3].

2.4 Segmentação de Texto

A segmentação de texto é de certa forma análoga à segmentação de documento, no entanto, busca delimitar elementos de texto apenas. De forma mais detalhada e para obter melhores resultados, a segmentação de texto pode ser dividida em segmentação de linhas, de palavras e de caracteres.

Como sugere o nome, a segmentação de linhas visa a separar as regiões que formam as linhas de texto do documento. Isso é feito para facilitar o processo de segmentação de caracteres. Um estudo sobre segmentação de linhas para imagens de documentos históricos pode ser encontrado em [22].

[3] Figura adaptada da referência [21].

Para documentos datilografados, devido ao espaçamento regular entre as linhas, o processo de separação de linhas é relativamente simples e pode ser resolvido por análise de projeção. Já para documentos manuscritos, são encontradas dificuldades adicionais sendo as principais delas conexão e sobreposição entre as linhas. A Figura 8 mostra o resultado da aplicação de uma das técnicas de separação de linhas abordadas em [22].

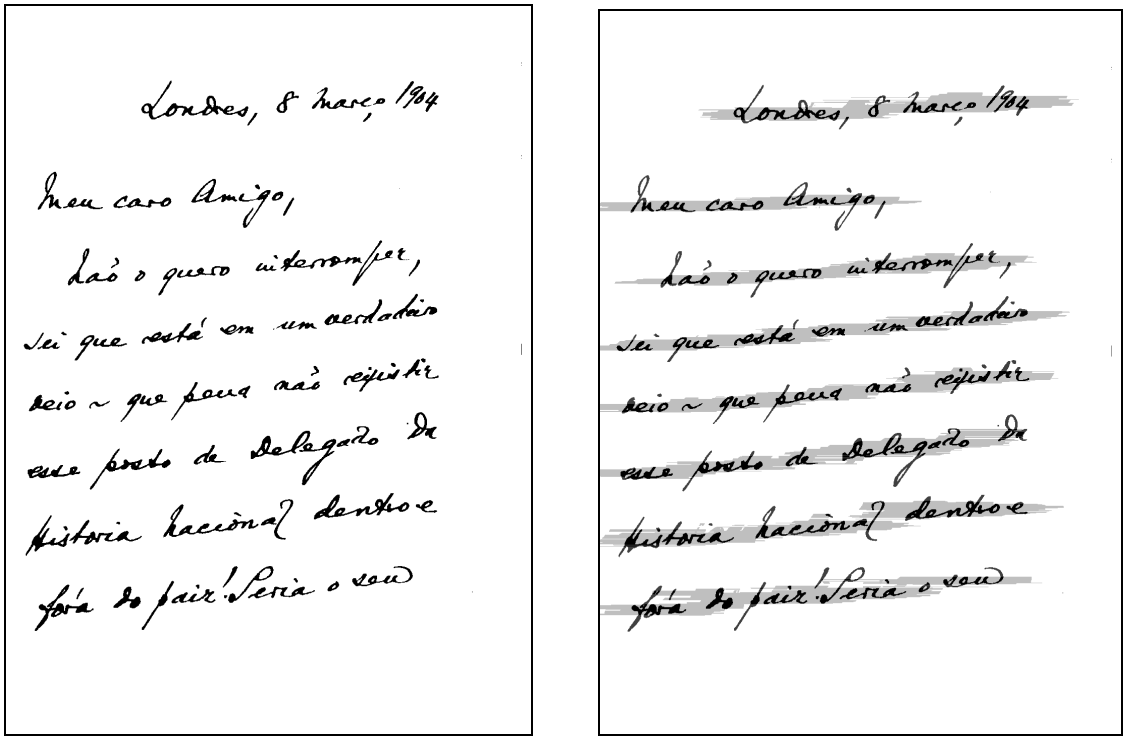
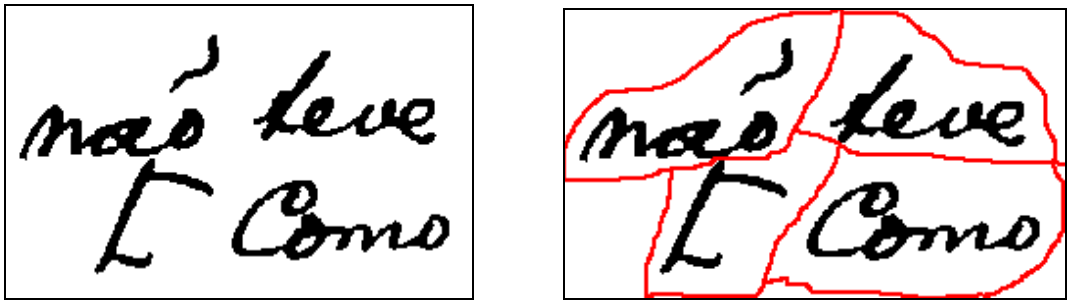


Figura 8. Exemplo de aplicação da segmentação de linhas, (a) imagem original binarizada, (b) regiões das linhas em tom de cinza.

A segmentação de palavras se presta a separar as regiões que compõem as palavras presentes no texto do documento. Sendo assim, como resultado desse processo se tem as partes referentes às palavras contidas no texto do documento. A Figura 9 ilustra uma simulação de segmentação de palavras para um pedaço de documento. Nesse exemplo, tem-se um caso ideal, ou seja, no fim cada palavra corresponde a uma região. Isso pode ser visto pela marcação de cada região por linhas vermelhas.

No entanto, nem sempre é usado esse passo, e então se vai diretamente para a segmentação dos caracteres. Esta visa a separar cada caractere existente na imagem. Isso obviamente de forma correta para então fornecê-los como entrada

para as ferramentas de reconhecimento de caracteres. Nesse ponto, é muito importante ressaltar, como já exposto na Introdução do trabalho, que erros nessa etapa acarretam insucesso ao reconhecimento de caracteres mesmo que se utilize um bom classificador.



(a) (b)

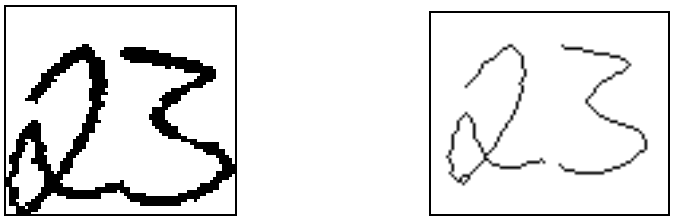
Figura 9. Simulação de segmentação de palavras, (a) imagem original, (b) resultado: regiões referentes às palavras do texto.

Nesse ponto, é preciso fazer uma ressalva. Há processos de segmentação que segundo o exposto em [8], podem ser classificados como holísticos, nos quais não se busca separar cada caractere isoladamente, e sim, aplicar o reconhecimento diretamente nas palavras encontradas pela segmentação de palavras.

Para caracteres naturalmente separados como no caso de documentos datilografados ou apenas porque quando escritos ficaram separados, algoritmos de projeção ou a aplicação de algoritmos de atribuição de legenda a componentes conectados [23] é suficiente para fornecer as partes da imagem referentes aos caracteres. E dessa forma, para esse cenário, o problema da segmentação já é resolvido com altas taxas de acerto.

Porém, para caracteres com problemas de disjunção, toque, sobreposição como mostrado na Figura 1, a segmentação de caracteres se mostra como grande desafio para aplicações de processamento de documentos. Portanto, ela é uma área de pesquisa bastante ativa. Existem várias técnicas e diversos algoritmos são criados a fim de resolver o problema como um todo, ou pelo menos em uma categoria. Por exemplo, o método desenvolvido em [4] foi projetado para dígitos sobrepostos. Já em [6], a técnica foi construída para dígitos conectados.

A segmentação de caracteres, e mais especificamente, a aplicação em dígitos manuscritos constitui o foco deste trabalho. No caso, algumas técnicas de segmentação de dígitos manuscritos são apresentadas no Capítulo 3. A Figura 10 mostra um caso de sucesso na aplicação da segmentação de caracteres já que pode se observar em (b), que os dígitos resultantes são os originais, só que agora separados. Nesse exemplo, foi usada a técnica de segmentação baseada em agentes implementada neste trabalho.



(a) (b)

Figura 10. Exemplo de aplicação da segmentação de caracteres, (a) imagem original: dígitos conectados (b) dígitos segmentados corretamente.

2.5 Extração de Características

A fase de extração de características tem como objetivo selecionar os dados mais significativos das unidades textuais dadas como entrada, para então fornecê-los como entrada para o processo de classificação. Esse fato acontece porque na maior parte dos casos, a utilização de todos os dados de entrada, ao contrário do que pode se imaginar, acaba atrapalhando a classificação. Isso tanto em relação a prejuízos de desempenho em termos de tempo de processamento, quanto na precisão da classificação dos caracteres.

Consequentemente, para melhorar o desempenho e precisão do processo de reconhecimento de caracteres, técnicas de extração de características são estudadas e desenvolvidas. Um estudo sobre o assunto, inclusive para o caso de dígitos manuscritos, pode ser consultado em [24].

2.6 Classificação

Enfim, a classificação é a etapa final e objetivo primordial da aplicação de OCR. Esse processo busca identificar e classificar corretamente cada caractere (já segmentado) presente no texto do documento. Geralmente, utilizam-se técnicas de aprendizado de máquina para executar essa tarefa. Na literatura, verifica-se que vários classificadores são utilizados para se reconhecer caracteres em documentos [25], tais como o Algoritmo dos K-vizinhos mais Próximos, Redes Neurais Artificiais, Máquinas de Vetor-Suporte [26].

São encontradas diversas dificuldades na tarefa de reconhecimento de caracteres. Uma delas é a variação da escrita de cada pessoa, o que torna praticamente inviável o desenvolvimento de uma técnica genérica, para qualquer acervo de documentos, já que isso inclui forma, inclinação, estilo e espessura. Ainda mais, a mesma pessoa escreve de maneiras diferentes com o passar do tempo.

A Figura 11 apresenta amostras do número 1896 escritas pela mesma pessoa. Analisando esta Figura, pode-se ver a variação na escrita, mesmo tendo sido feita pelo mesmo indivíduo, fato que dificulta bastante o reconhecimento dos caracteres.

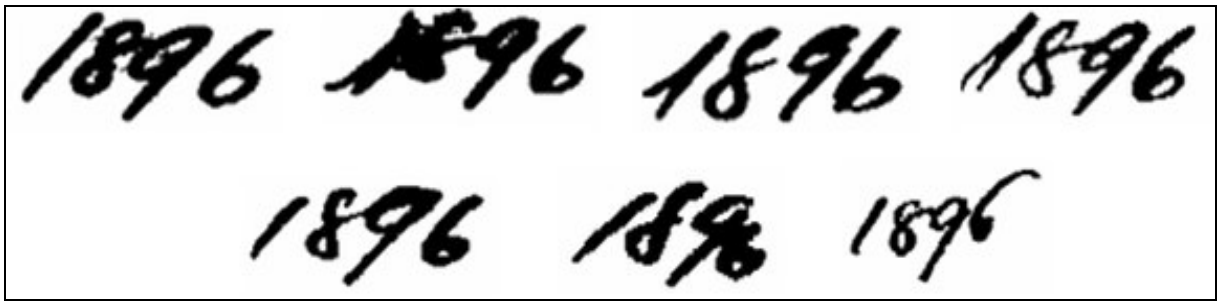


Figura 11. Exemplos de escrita à mão livre do número 1896 efetuada pela mesma pessoa[4].

[4] Figura adaptada de [24].

Capítulo 3
Segmentação de Dígitos Manuscritos

Como exposto no Capítulo 2, a segmentação de caracteres é uma etapa crucial para o sucesso do reconhecimento de caracteres. Muitas vezes, para fins de simplificação, algoritmos de segmentação são criados para dígitos apenas, em vez de para caracteres em geral.

Isso pode ser explicado primeiramente, pelo fato de existirem apenas dez dígitos (de zero a nove), fato que diminui as dificuldades referentes às peculiaridades de cada caractere. Segundo, também facilita a fase de reconhecimento já que os dados de treinamento e de teste serão bem menores. Além disso, é mais comum encontrar apenas pares de dígitos conectados ou sobrepostos, o que no caso de caracteres, muitas vezes temos uma palavra inteira conectada, ou sobreposição de linhas, etc.

Existe um dilema encontrado pelas diversas técnicas de segmentação em geral (principalmente se tomando como base o processo de OCR como descrito no Capítulo 2). Este é o paradoxo de Sayre [27], ou paradoxo segmentação-reconhecimento, o qual enuncia: um caractere não pode ser segmentado sem ser reconhecido; e não pode ser reconhecido sem ser segmentado.

Sendo assim, o Capítulo 3 trata de algumas técnicas de segmentação de dígitos manuscritos. É importante salientar que se buscou nesse trabalho analisar técnicas clássicas e também métodos mais recentes na tarefa de separação de dígitos.

3.1 *Hit and Deflect*

No Capítulo 2, viu-se que para caracteres separados, análise de projeção ou atribuição de legendas a componentes conectados resolvem o problema da segmentação. No caso de conexão ou sobreposição, técnicas mais elaboradas são

necessárias. Uma ideia inicial seria traçar uma linha vertical que dividisse a imagem separando os dígitos da mesma. Porém, para o caso em que os caracteres estão sobrepostos ou inclinados, essa saída não se mostra precisa ou eficaz.

A estratégia de segmentação *Hit and Deflect* [8] busca por um caminho ótimo de separação analisando o contorno dos dígitos. Esse tipo de algoritmo pode ser definido como sendo técnicas capazes de traçar um caminho curvo de segmentação, através da movimentação iterativa de um ponto "de varredura". No início esse ponto é determinado como o pico máximo do perfil inferior da imagem (incluindo apenas a metade inferior da imagem). Essa pode ser considerada já uma técnica clássica para a separação de dígitos.

Então, o ponto é movido para cima, e ao encontrar um *pixel* ativo[5], ele sofre uma deflexão, mudando assim a trajetória do movimento. O algoritmo segue um conjunto de regras em seu movimento, as quais procuram maximizar as chances de colisões e deflexões resultarem no caminho de segmentação exato. A Figura 12 ilustra um exemplo de separação utilizando uma estratégia *Hit and Deflect*. Observa-se em vermelho, o caminho de segmentação encontrado pelo algoritmo.

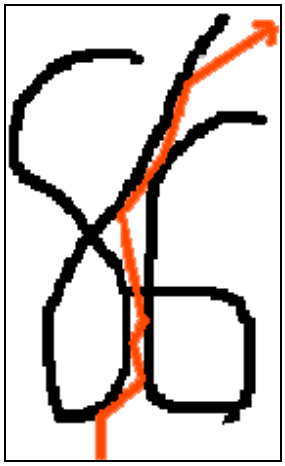


Figura 12. Segmentação por *Hit and Deflect*.

Um exemplo de utilização dessa estratégia foi mostrado em [28] para separação de cadeias de dígitos. Basicamente, o algoritmo se divide em dois passos:

[5] Um *pixel* é dito ativo quando faz parte do objeto de interesse da imagem. Nas figuras apresentadas neste trabalho, os *pixels* ativos se referem aos caracteres, exibidos na cor preta.

i. Faz-se uma varredura vertical a partir do centro da imagem, a qual se supunha conter dígitos conectados. Se a varredura se completa em uma coluna com duas ou nenhuma transição de preto para branco ou de branco para preto, os dígitos não estão conectados ou são conectados em um só ponto, e sendo assim, esse já é o caminho de segmentação.

ii. Se houve no caminho mais de duas transições, o algoritmo considera que provavelmente os dígitos estão inclinados, e então uma estratégia *Hit and Deflect* é empregada. O algoritmo inicia a partir de um pico na parte inferior do contorno do caractere conectado, e então procura subir na imagem baseado em um conjunto de regras que minimiza o número de cortes ao longo do caminho de segmentação. Isso significa que só é feito um corte quando não há outro movimento possível. Quando o algoritmo alcança o topo da imagem, o caminho está completo e constitui o segmento de separação entre os dígitos.

Menciona-se que em boa parte dos casos, apenas um corte é necessário para separar caracteres inclinados ou conectados por um único ponto [8].

3.2 *Drop-Fall*

O algoritmo *Drop-Fall* [9] é outro algoritmo clássico de segmentação de dígitos. Conceitualmente, ele simula um objeto, como uma gota de água ou uma esfera, caindo de cima dos caracteres e deslizando ao longo dos seus contornos. A Figura 13 ilustra exemplos de caminhos seguidos pelos contornos dos dígitos, dependendo do ponto inicial adotado. Iniciando-se na posição (1), o algoritmo procede para a esquerda do primeiro caractere. A partir da posição (2), o algoritmo procede para a direita do segundo caractere.

No caso, a esfera “cai” até que não possa mais se movimentar livremente (quando encontra um *pixel* ativo) e a partir daí, continua para baixo através do contorno do caractere da mesma maneira que o *Hit and Deflect* descrito na seção 3.1. A propósito, o *Drop-Fall* pode ser visto como um tipo especial de estratégia *Hit and Deflect*. O algoritmo encontra um ponto de partida, e a partir de um conjunto de regras determinam o caminho a ser seguido pela esfera.

O algoritmo *Drop-Fall* completo é dividido em dois níveis. No primeiro passo, aplica-se um detector de componentes conectados a fim de se identificar blocos isolados. Cada bloco é passado para um classificador, e os blocos que foram bem reconhecidos, ou seja, apresentaram resposta satisfatória pelo classificador, são separados e não mais utilizados na segmentação. Esse passo inclusive poderia ser considerado em qualquer técnica de segmentação, já que consiste em não se aplicar uma estratégia de separação a dígitos já separados na imagem original.

No segundo passo, então de fato é aplicada a estratégia *Drop-Fall* nos blocos que não foram reconhecidos no primeiro passo. Sendo assim, aplicam-se os quatro algoritmos *drop-fall* (explicados na seção 3.2.1) em sequência. Cada algoritmo escolhe um ponto de partida e define determinado caminho de separação. No fim do procedimento de segmentação, o bloco é dividido em dois sub-blocos os quais são passados para o classificador novamente.

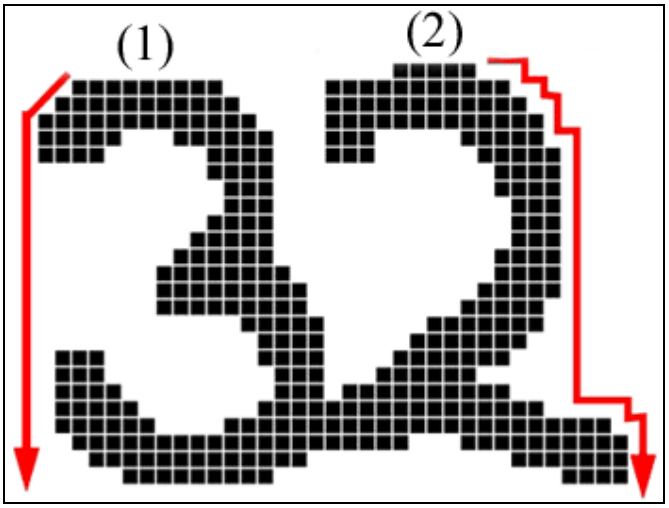


Figura 13. Exemplos de caminhamentos feitos pelo algoritmo *Drop-Fall* em função dos pontos de partida (1 e 2) do algoritmo.

E então, são possíveis três situações: (i) se os dois blocos são rejeitados, então se aplica um algoritmo *drop-fall* adicional. Se os quatro algoritmos são aplicados sem sucesso, a sequência de números inteira é rejeitada; (ii) se um dos blocos é reconhecido, reprocessam-se os sub-blocos restantes para segmentação adicional usando todos os algoritmos; (iii) se os dois blocos são reconhecidos, o processo retorna ao primeiro passo de segmentação para analisar os blocos restantes, se existirem.

3.2.1 Estratégias *drop-fall*

Nesse ponto, explicam-se as quatro estratégias ou algoritmos *drop-fall* os quais, como informado no texto, são aplicados como segundo nível de segmentação no método *Drop-Fall* completo. Eles são classificados de acordo com sua orientação e ponto de partida da esfera. E assim, as quatro orientações são: superior-esquerda (o padrão), superior-direita, inferior-esquerda e inferior-direita.

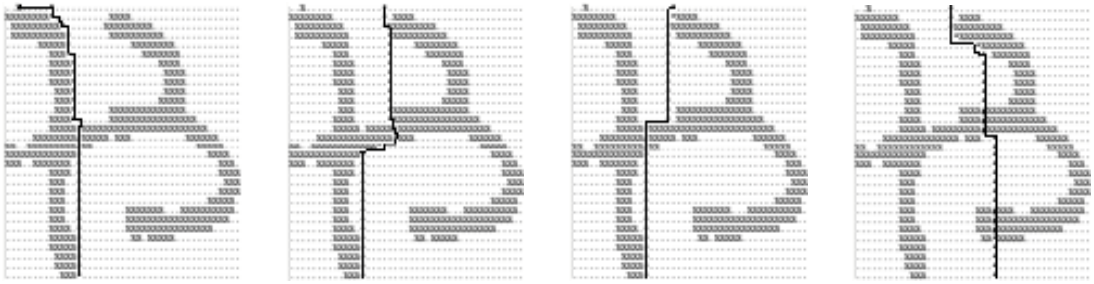
Cada versão do algoritmo inicia a busca pelo ponto de partida pelo canto da imagem indicado pelo seu nome, avançando para o canto oposto. As variantes que começam na parte de baixo da imagem “caem” para cima. Isso é equivalente a rotacionar a imagem sobre o eixo horizontal, a aplicar o *drop-fall* superior-esquerdo. Para as variantes que iniciam do lado direito, faz-se uma rotação sobre o eixo vertical. Dessa forma, detalha-se um pouco o algoritmo superior-esquerdo, e os outros são aplicados em imagens rotacionadas como descrito no texto.

As quatro orientações são utilizadas, se necessário, por conta do ponto inicial ser um parâmetro muito importante para o sucesso na separação dos dígitos. O ponto de partida é definido como o *pixel* inativo imediatamente à direita da borda esquerda. A borda é encontrada, a partir da procura do primeiro *pixel* ativo a ter um espaço inativo imediatamente à direita, e outro *pixel* ativo mais à direita.

Ressalta-se a importância da escolha adequada do ponto de partida do algoritmo, pois pontos iniciais errados levam a insucesso na segmentação, como pode ser visto na Figura 13.

As regras de movimentação do algoritmo podem ser consultadas em [9]. É importante salientar, que a esfera não pode “subir” já que o movimento simula um objeto caindo. Além disso, quando nenhuma posição predita pelas regras não é possível, então a esfera “corta” o dígito (passando por um *pixel* ativo).

A Figura 14 mostra as diferentes orientações do *Drop-Fall* aplicadas ao mesmo par de dígitos. Fica evidente com a análise desta Figura, que o sucesso na segmentação depende da orientação utilizada e, portanto, do ponto inicial empregado. Nesse caso específico, observa-se que o melhor resultado foi obtido com a orientação inferior-direita, em (b).



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

Figura 14. Aplicação das quatro orientações do algoritmo *Drop-Fall*, (a) superior-esquerda, (b) inferior-direita, (c) superior-esquerda, (d) inferior-direita.

3.3 Renaudin *et al.*

O método proposto por Renaudin *et al.* [7] é bem recente, data de 2007, e tem pontos bastante interessantes, motivo pelo qual foi abordado neste trabalho. O primeiro deles é a tentativa de evitar o paradoxo de Sayre, utilizando uma estratégia que une segmentação e reconhecimento, em vez de um processo seqüencial como na maior parte das aplicações. O segundo é que ele tenta também ser o mais geral possível, no sentido de não ser aplicável apenas a dígitos, mas também a caracteres e símbolos musicais ou matemáticos. Além disso, o algoritmo se presta tanto para caracteres conectados quanto sobrepostos.

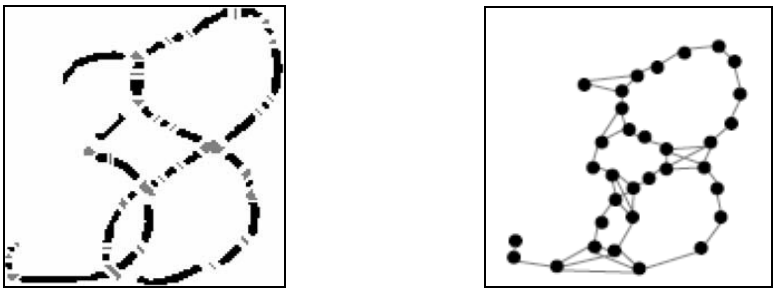
Sobre o algoritmo em si, ele utiliza uma abordagem baseada na sobre-segmentação[6] dos caracteres. Assim, a imagem é dividida em subpartes dos símbolos originais. Para gerar essa sobre-segmentação, é utilizado um algoritmo de decomposição de traços em áreas singulares e regulares [29]. De forma simples, áreas singulares são aquelas nas quais existem certas características como interseção, alta curvatura, variações de espessura, etc. Já as áreas regulares são aquelas situadas entre áreas singulares. Essa distinção é importante, pois os pontos de segmentação, supostamente estão nas áreas singulares.

Áreas regulares são conectadas umas às outras através de áreas singulares. Sendo assim, a sobre-segmentação passa a ser representada por um grafo não-

[6] Sobre-segmentação consiste em se dividir a imagem ou traço em muito mais regiões ou segmentos do que realmente os constituem.

direcionado [30], no qual os vértices representam as áreas regulares e as arestas, áreas singulares. Há também a possibilidade de se criar arestas entre áreas regulares, sem ser da forma descrita, dependendo de um critério de proximidade [7]. A Figura 15 mostra, em (a), a decomposição por sobre-segmentação de um par de dígitos em áreas regulares (em preto) e singulares (em cinza), e em (b) o grafo construído a partir dos segmentos gerados.

A partir do grafo resultante da decomposição dos caracteres em áreas singulares e regulares, criam-se candidatos a símbolos como determinados subgrafos da estrutura. Estes consistem em diferentes junções das áreas regulares que incluem as áreas singulares associadas. Por ser um algoritmo baseado em grafos, no pior caso se têm 2^n candidatos, para n áreas regulares. Logo, aplicam-se algumas heurísticas para se reduzir o número de candidatos [7], fato que inclusive pode ser considerado como ponto negativo da técnica.



(a) (b)

Figura 15. Aplicação da sobre-segmentação do algoritmo Renaudin *et al.*, (a) resultado da sobre-segmentação, (b) o grafo associado[7].

O próximo passo do método é classificar os diversos segmentos candidatos para determinar quais provavelmente constituem os símbolos originais. Então, como normalmente se faz em processamento de documentos, extraem-se características e então, elas são fornecidas para um classificador. No trabalho em questão, para a extração de características, empregaram-se os momentos de Zernike [31]; para a classificação, utilizou-se ;uma Rede de Função de Base Radial, ou rede RBF [26].

Para cada candidato, a rede RBF associa uma “pontuação” a qual indica a probabilidade de um candidato pertencer à determinada classe. Essa pontuação é

[7] Figura adaptada de [7].

utilizada para dois propósitos: (i) para descartar candidatos e (ii) para estabelecer uma hierarquia entre os candidatos "bem" reconhecidos. Supõe-se que o candidato com a pontuação mais alta é o qual corresponde a um símbolo real.

Considerando a hipótese de que a imagem inicial contém um par de símbolos conectados ou sobrepostos, há uma fase de seleção. Esta consiste em determinar qual par de candidatos, entre todos aqueles que foram bem reconhecidos, melhor corresponde aos símbolos iniciais.

Basicamente, para esse fim, é utilizado um critério de complementaridade (dependente do tipo de aplicação) que se baseia no número de áreas regulares compartilhadas pelos símbolos. Além disso, utiliza-se a pontuação obtida por cada candidato na classificação. O par com a pontuação mais alta (obtida através do produto da pontuação de cada um) é considerado como o par mais provável de corresponder aos símbolos iniciais. A Figura 16 mostra o fluxo, em alto nível, do algoritmo Renaudin *et al.* (os passos do método e seus resultados).

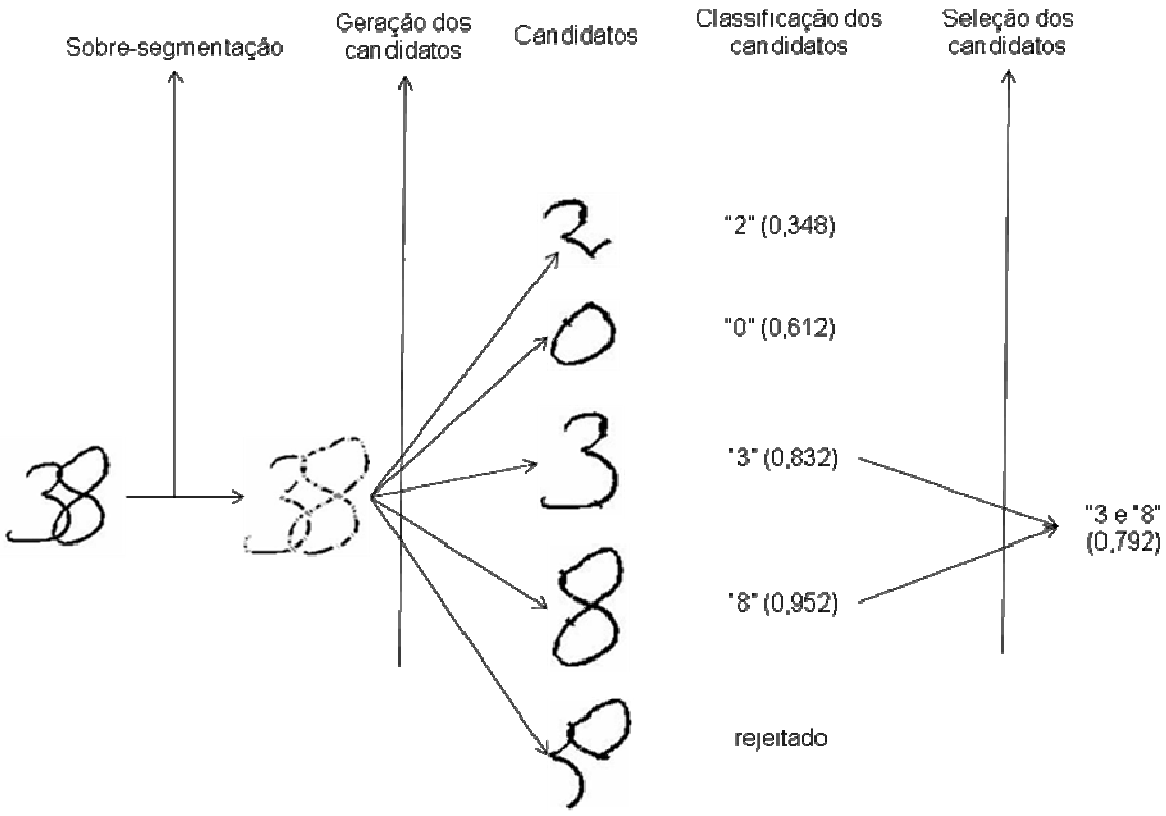


Figura 16. Esquema geral da aplicação do algoritmo Renaudin *et al.*[8].

[8] Figura adaptada de [7].

3.4 Alhajj e Elnagar

A técnica proposta por Alhajj e Elnagar [6] se constitui no foco deste trabalho, já que foi o método de segmentação de caracteres implementado. Também é um trabalho relativamente recente, data de 2003, e tem uma proposta inovadora baseada na utilização de multiagentes inteligentes [5]. Especificamente, os dois agentes envolvidos no processo negociam sobre qual será o ponto de segmentação entre os dígitos, baseados em uma medida de confidência.

No entanto, ressalta-se logo que neste trabalho não se programaram o algoritmo de restauração como também o de eliminação de ruídos empregado no algoritmo original. Isso se deve principalmente à má especificação desses processos em [6]. Logo, devido ao caráter deste trabalho, resolveu-se não implementar tal característica, embora se saiba do possível impacto disso no desempenho do método como um todo.

Sendo assim, nesta seção detalha-se o algoritmo em suas diversas partes (o modelo de agente empregado, a procura pelo ponto de segmentação, a estratégia de negociação) e também aspectos de sua implementação.

De forma geral, em alto nível, pode-se explicar o algoritmo da seguinte maneira (a Figura 17 ilustra essa explicação). Primeiramente, aplica-se a operação de esqueletização [1] na imagem, para que os segmentos constituintes dos dígitos tenham apenas um *pixel* de espessura. Isso é feito para facilitar o caminhamento dos agentes.

Depois, são marcados alguns pontos especiais na imagem, chamados de pontos característicos [6]. Eles se dividem em três tipos: (i) terminação – pontos que contém apenas um vizinho ativo; (ii) ramificação – pontos em que o segmento se ramifica, ou seja, há mais de uma trajetória possível se considerando um caminhamento sobre o dígito e (iii) cruzamento – pontos os quais têm quatro vizinhos ativos, tendo assim aparência de uma cruz.

Então, empregam-se dois agentes para o processo de segmentação. Um se concentra na parte superior e o outro na parte inferior da imagem. O primeiro seleciona como ponto de separação, o ponto característico mais próximo ao centro

do vale mais profundo. O segundo seleciona como ponto de segmentação o ponto característico mais próximo ao centro do morro mais alto.

Dessa forma, cada agente escolhe um ponto de separação. Se o ponto selecionado pelos agentes foi o mesmo, ou pelo menos eles se encontram na mesma coluna da imagem, um deles é escolhido como ponto de segmentação. Quando isso não acontece, o ponto de separação final é selecionado através de um processo de negociação entre os dois agentes. A decisão se baseia em um grau de confidência de cada ponto candidato.

Aplicam-se dois agentes para a separação entre os dígitos porque em diversos casos, um deles pode falhar. Assim, a cooperação entre os dois agentes permite maior eficácia do algoritmo; de outra forma, aumenta sua taxa de sucesso.

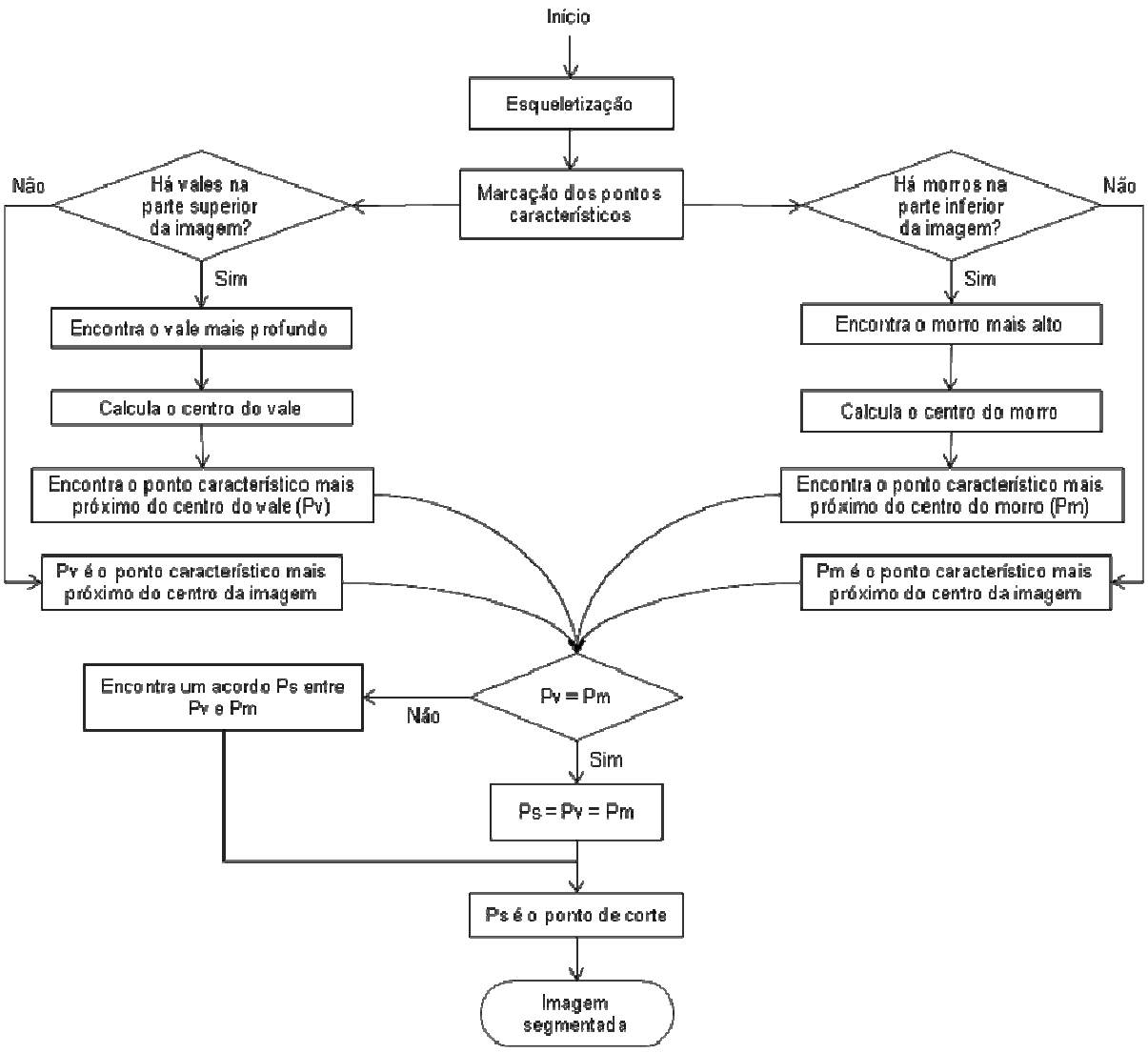
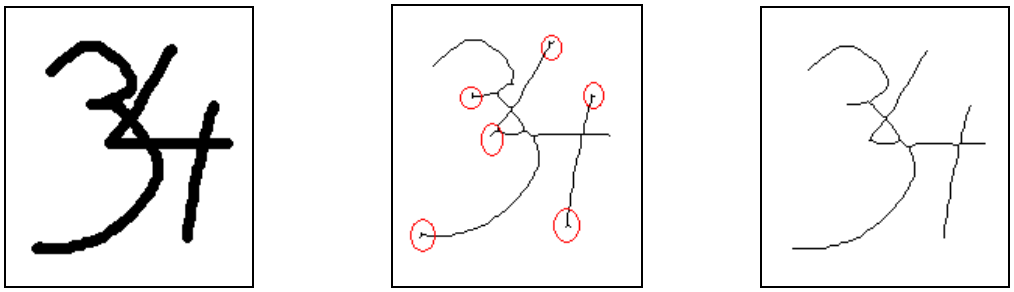


Figura 17. Fluxograma do algoritmo Alhajj e Elnagar.

3.4.1 Algoritmo de esqueletização

Ressalta-se que neste trabalho, o algoritmo utilizado não possui associação à transformação do eixo médio [32]. Dessa forma, obtém-se um esqueleto mais suave, com menos traços externos ou “pontas”.

A Figura 18 ilustra a diferença, para o caso de aplicação da esqueletização em um par de dígitos, com ou sem a transformação do eixo médio. Observam-se “pontas” nas regiões circuladas em vermelho (b), as quais não existem em (c). Isso justifica a escolha de um algoritmo que não utiliza a transformação do eixo médio.



(a) (b) (c)

Figura 18. Diferença entre algoritmos de esqueletização, (a) imagem original, (b) esqueleto da imagem com transformação do eixo médio, (c) esqueleto da imagem sem a transformação do eixo médio.

3.4.2 Modelo de agente inteligente

O algoritmo Alhajj e Elnagar utiliza dois agentes inteligentes para separar dígitos conectados. O modelo de agente é basicamente idêntico, e na verdade, apenas alguns cálculos são modificados (explicam-se essas alterações mais a frente nesta seção) e se aplica o mesmo modelo de agente na imagem invertida verticalmente. Sendo assim, detalha-se o agente que procura por vales na parte superior da imagem e posteriormente, os ajustes feitos no caso do agente que procura por morros na parte inferior da imagem.

Sobre o agente que se concentra nos vales, a ideia básica é que o caminho de segmentação está próximo ao vale mais profundo encontrado. Mais detalhadamente, o ponto característico mais próximo ao centro do vale mais profundo está em algum lugar sobre o caminho de segmentação. Logo, o sucesso desse agente depende da existência de pelo menos um vale com origem na parte

superior da imagem. Se não, como mostrado na Figura 17, o agente indica como ponto de segmentação o ponto característico mais próximo do centro da imagem. Nesse caso, esse ponto apresenta um grau de confidência baixo e provavelmente, isso será resolvido no processo de negociação, desde que o outro agente consiga um desempenho melhor.

Nesse ponto, explica-se de fato como o agente que procura por vales funciona. Ele segue o contorno da imagem esqueletizada (o que de certa forma equivale a caminhar pelo dígito em si) procurando por vales na imagem, sempre mantendo a informação de quem é o vale mais profundo.

O ponto de início do caminhamento é o ponto de terminação mais acima e à esquerda da imagem, e ele é feito da esquerda para a direita. Neste trabalho, em virtude da base de dígitos utilizada e também para otimizar o código, a procura pelo ponto de terminação é feita apenas no primeiro quadrante da imagem (considerando uma divisão da imagem em quatro partes, esse trecho se refere ao primeiro na parte superior e à esquerda).

Ressalta-se também que na referência base, o processo de se encontrar vales não é bem especificado, de forma que foi preciso definir neste trabalho como isso seria feito. Na implementação realizada, o caminhamento foi feito nos traços dos dígitos. Um problema que aparece nesse caso, é que nos pontos de ramificação e de cruzamento há mais de um caminho possível, de forma que o agente não saberia o caminho a seguir. Isso foi resolvido, escolhendo sempre como próximo ponto, o situado mais à direita. Para evitar que o agente visitasse um ponto já percorrido, a imagem é marcada à medida que vai se caminhando nos dígitos.

Além disso, foi preciso adotar uma estratégia para o caso em que não há um ponto de terminação para o início do caminhamento. No trabalho base, menciona-se que há casos em que é possível não se encontrar um vale (pode-se observar isso no fluxograma da Figura 17), e sendo assim, o ponto de separação é o ponto característico mais próximo do centro da imagem. Dessa maneira, considerou-se neste trabalho que quando não há um ponto de terminação para ser o início do caminhamento, não foi possível encontrar um vale e, portanto o ponto candidato será o ponto característico mais próximo do centro da imagem. E nesse caso, não

precisa se aplicar os cálculos descritos a seguir. A Figura 19 mostra um par de zeros conectados, caso em que não há pontos de terminação.

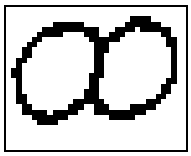


Figura 19. Exemplo em que não há pontos de terminação.

A profundidade de um vale é calculada como a diferença entre o ponto mais alto onde o contorno muda para baixo, e o ponto mais baixo onde o contorno muda para cima. Ressalta-se que se consideram os pontos mais altos dos dois lados do vale no cálculo da profundidade. Formalmente, pode-se dizer que há três pontos importantes em um vale: os pontos mais altos nos dois lados do vale e o ponto mais profundo do vale, denotados respectivamente por *A*, *B* e *C*.

No trabalho original, considera-se que cada ponto tem coordenadas *x* e *y*, tendo como origem dos eixos coordenados, o ponto de terminação mais baixo e à esquerda da imagem. Neste trabalho, essas considerações não foram feitas da mesma maneira, e se considerou o sistema de eixos padrão para imagens digitais [12]. E então, algumas mudanças foram feitas nos cálculos frisando-se, no entanto que os resultados são equivalentes. Assim, a profundidade de um vale é calculada da seguinte maneira (considerando *max* uma função que calcula o valor máximo entre dois valores):

$$D = y_C - \max\left(y_A, y_B\right) \tag{1}$$

Na versão original do algoritmo, emprega-se uma heurística para eliminar possíveis vales existentes em um único dígito, e também os que possam existir devido a traços ruidosos. Essa heurística não foi implementada neste trabalho, até porque, não se percebeu a utilidade dela, como descrito em [6].

O centro de um vale é calculado se considerando as duas bordas do mesmo. Isto significa que se utilizam pontos da borda esquerda *L*, e da borda direita do vale *R*. Mais especificamente, o centro do vale é definido como a distância média entre suas duas bordas, do ponto *C* (o mais profundo de um vale) até o ponto *M*, o qual é definido como:

$$M = \max(A, B) \tag{2}$$

Dessa maneira, formalmente, o centro de um vale é calculado assim:

$$x_{centro} = média\left(\sum_{C}^{M}(x_R - x_L)\right) \tag{3}$$

$$y_{centro} = y_C + \frac{D}{2} \tag{4}$$

O ponto característico mais próximo do centro do vale *F*, ponto que é apontado pelo agente como candidato para separar os dígitos, é definido como (onde *min* é uma função que calcula o valor mínimo entre dois valores):

$$F = \min(abs(x_F - x_{centro})) \tag{5}$$

Se mais de um valor corresponder a equação (5), então se aplica o seguinte cálculo para decidir o ponto candidato:

$$F = \min(abs(y_F - y_{centro})) \tag{6}$$

A confidência ou grau de confidência do ponto *F*, medida que representa o quanto o agente acredita que terá sucesso com esse ponto na segmentação, é calculada da seguinte maneira (onde D representa a profundidade de um vale ou a altura de um morro, dependendo do agente em questão):

$$Confidência(F) = \frac{D}{D + \min(abs(x_F - x_{centro}))} \tag{7}$$

Observa-se que o sucesso de um agente depende do grau de confidência do ponto candidato por ele apresentado. Um grau de confidência baixo leva a insucesso na segmentação. Já, à medida que esse grau cresce, a segmentação passa a ter mais sucesso. Inclusive, por conta do fato de um agente poder reportar um ponto candidato com baixa confidência, é que se utilizam dois agentes na tarefa de separação dos dígitos.

Como já mencionado no início da seção, o modelo de agente é o mesmo tanto para procurar vales como para procurar morros, no entanto, aplica-se o processo com a imagem invertida. Contudo, devido a isso, alguns cálculos são

ajustados para se ter as coordenadas dos pontos na imagem não-invertida. Basicamente, o ajuste ocorre no cálculo do centro do morro, da seguinte maneira:

$$x_{centro_m} = x_{centro_v} \tag{8}$$

$$y_{centro_m} = H - y_{centro_v} \tag{9}$$

3.4.3 Negociação e resolução de conflito

Após cada agente apresentar um ponto candidato, *i.e.*, ponto característico com maior grau de confidência, para a separação dos dígitos, esses pontos servem como entrada para a negociação e resolução de conflito, caso haja algum conflito, o qual será resolvido pela cooperação entre os agentes. Nesse contexto, há três possibilidades. Os dois agentes podem reportar: (i) o mesmo ponto candidato; (ii) pontos candidatos diferentes, porém com a mesma coordenada *x*, e coordenada *y* diferente; (iii) pontos candidatos com as duas coordenadas diferentes.

Apenas no último caso é preciso aplicar a negociação. Nos dois primeiros, a imagem é segmentada na coordenada *x* dos pontos (é a mesma em ambos candidatos). No entanto, ressalta-se que devido ao fato de não se utilizar o mesmo esquema de coordenadas que no artigo original, nesse ponto é adicionado um *desvio*, que na verdade corresponde ao deslocamento relacionado à origem dos eixos coordenados do trabalho de referência. E dessa forma, esse *desvio* é igual à coordenada *x* do ponto de terminação mais à esquerda.

Uma confidência maior faz o agente que a possui ter prioridade no processo de negociação. Conforme pode se observar em (7), o valor máximo da confidência para um ponto candidato é a unidade. Isso acontece quando a coordenada *x* do ponto candidato é igual à do centro do vale. Por outro lado, se a profundidade ou altura de um vale ou morro for igual a zero, a confidência será zero, fato que dará ao agente em questão baixa prioridade durante a negociação. O Quadro 1 mostra como se faz a negociação entre os agentes e as equações para escolha do ponto de corte.

Pode-se observar em (11) e (12) que há um termo na equação (a fração entre as confidências) que pode ser somado ou subtraído. A operação a ser utilizada depende do ponto candidato que está mais à direita. No caso, quando o ponto de menor confidência está mais à direita, usa-se a adição e caso contrário, a subtração.

Apresentado o algoritmo, a Figura 20 ilustra o funcionamento do algoritmo com um exemplo. Embora para essa imagem o resultado não tenha sido satisfatório, pode-se observar a sequência das operações como descrito nesta seção, e mostrado na Figura 17. Em (c), observa-se a marcação dos pontos característicos da imagem – em magenta os pontos de terminação e em verde, os pontos de ramificação e cruzamento (marcados da mesma maneira como no algoritmo original). A partir da análise de (d) e (e) é possível observar os conceitos de vale e morro tão importantes nesse algoritmo (o que não é mostrado no trabalho original).

Quadro 1. Algoritmo de negociação entre os agentes.

```
NEGOCIAR(Fv, Fm)
início
   se confidência(Fv) = confidência(Fm) então
```

$$x_{corte} = \frac{\left(x_{Fv} + desvio_v\right) + \left(x_{Fm} + desvio_m\right)}{2} \tag{10}$$

```
   mas se confidência(Fv) > confidência(Fm) então
```

$$x_{corte} = \left(x_{Fv} + desvio_v\right) \times \left(\frac{1 \mp \frac{confidência(F_m)}{confidência(F_v)}}{2} \right) \tag{11}$$

```
   mas se confidência(Fm) > confidência(Fv) então
```

$$x_{corte} = \left(x_{Fm} + desvio_m\right) \times \left(\frac{1 \mp \frac{confidência(F_v)}{confidência(F_m)}}{2} \right) \tag{12}$$

```
fim
```

Na Tabela 1, apresentam-se os valores para as confidências dos agentes, suas colunas candidatas, e a coluna na qual a imagem foi dividida. Isso é feito a fim de se mostrar a influência do grau de confidência na negociação e, por conseguinte, na escolha do ponto de separação entre os dígitos. Sendo assim, para os dois agentes empregados na segmentação, o que procura vales o que procura morros, mostra-se: a coluna candidata (pois o algoritmo separa a partir de uma coluna, não importando praticamente a linha em que o ponto está; esta servindo apenas como critério de desempate quando os pontos apresentam a mesma confidência) e o grau

de confidência do ponto. Ainda mais, mostra-se o ponto selecionado pelo processo de negociação.

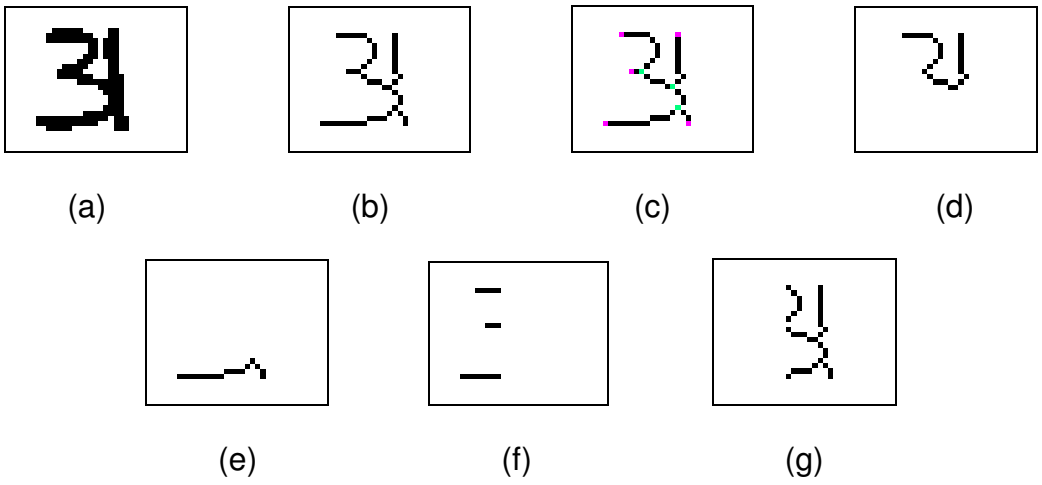


Figura 20. Aplicação do algoritmo Alhajj e Elnagar, (a) imagem original, (b) esqueleto da imagem, (c) marcação dos pontos característicos, (d) vale, (e) morro, (f) segmento correspondente a um dígito, (g) segmento correspondente ao outro dígito.

Tabela 1. Resultados individuais de cada agente e da negociação entre eles.

	Agente que procura vales	Agente que procura morros	Negociação
Coluna candidata	14	21	-
Confidência	0,8403	0,3214	-
Coluna selecionada	-	-	15

Capítulo 4

Experimentos

Este capítulo trata dos experimentos realizados e resultados obtidos neste trabalho. Mais detalhadamente, analisam-se os resultados de três das técnicas abordadas no Capítulo 3, a saber: (i) *Drop-Fall* [9], (ii) Renaudin *et al*. [7], e principalmente por ter sido o método implementado, (iii) Alhajj e Elnagar [6], descritas respectivamente nas Seções 3.2, 3.3 e 3.4.

Para tal, implementou-se o algortimo Alhajj e Elnagar utilizando o *software* MATLAB [33]. A escolha desse programa e de sua linguagem de programação se deu devido à facilidade de se tratar matrizes e consequentemente imagens, além de o *software* fornecer diversas funcionalidades para o Processamento Digital de Imagens.

Os resultados dos outros dois algoritmos foram extraídos de implementações as quais não foram realizadas no presente trabalho. No caso do algoritmo Renaudin *et al*., o autor do trabalho, o Sr. Cristophe Renaudin gentilmente testou a técnica com as imagens da base de dígitos empregada neste trabalho. Para o *Drop-Fall*, utilizou-se uma implementação do Grupo de Pesquisa em Reconhecimento de Padrões (grupo no qual este trabalho se insere) na linguagem de programação Java [34]. Ressalta-se que nessa implementação a orientação do *Drop-Fall* é a superior-esquerda.

Ressalta-se desde já a importância na escolha dos métodos abordados, já que se trata de uma técnica já clássica para segmentação de dígitos, e de métodos recentes na literatura. Pode-se citar como grande contribuição deste trabalho a comparação entre essas técnicas, já que isso é inédito.

4.1 Material de Estudo

As imagens utilizadas no presente trabalho foram geradas a partir da base de dados de dígitos manuscritos MNIST [35]. Ela é composta por dígitos isolados, e salienta-se que há vários padrões para um mesmo dígito, fato coerente já que as pessoas

escrevem de forma variada. Para os propósitos deste trabalho e das pesquisas envolvidas, por meio de um algoritmo criaram-se dígitos conectados e sobrepostos. A Figura 21 apresenta alguns exemplos de dígitos isolados contidos na base MNIST.

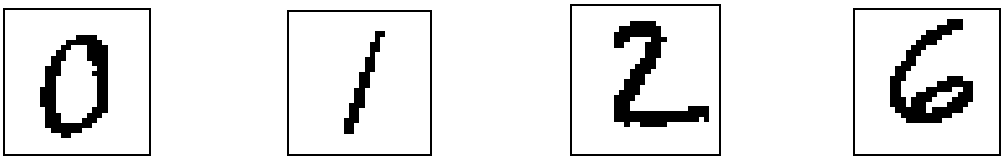


Figura 21. Amostras de dígitos manuscritos da base MNIST.

Já a Figura 22 ilustra as vinte e quatro imagens de sobreposição e conexão entre dígitos criados a partir da base de dados MNIST utilizadas neste trabalho (foram criadas muito mais imagens, só que selecionamos as vinte e quatro mostradas na Figura 22). As imagens foram salvas em formato BMP e possuem dimensões 35x28 *pixels* (neste texto algumas estão ampliadas para facilitar a visualização).

A escolha das imagens não foi aleatória, e um critério adotado para tal foi garantir todos os dígitos (de 0 a 9) presentes nas imagens. Além disso, também se buscaram casos com uma segmentação complexa já que, para casos simples, técnicas menos elaboradas podem resolver o problema de forma satisfatória.

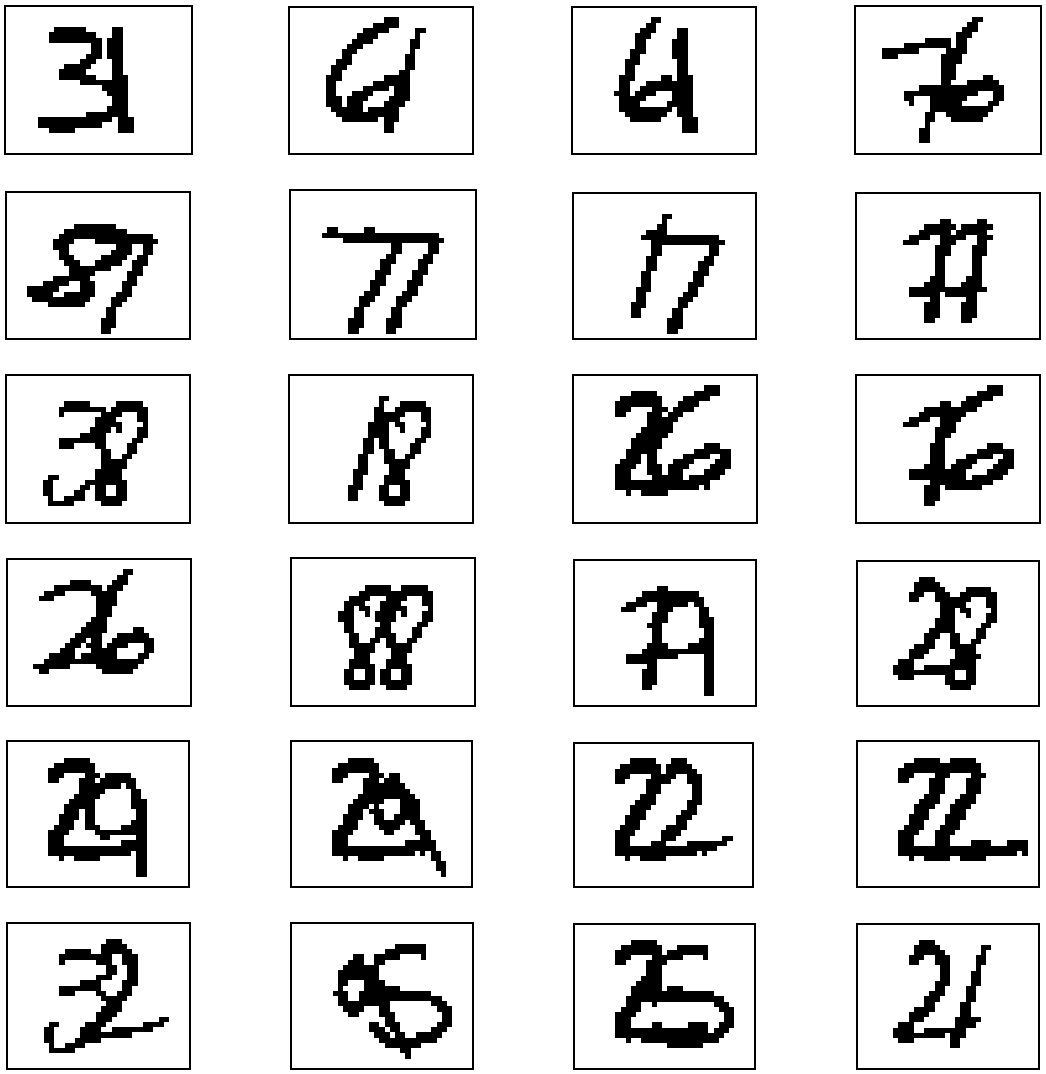


Figura 22. Imagens de dígitos conectados e sobrepostos utilizadas nos testes realizados neste trabalho.

4.2 Resultados

Realizou-se a análise entre os resultados através de inspeção visual, ou seja, observa-se a segmentação obtida e então se classifica como correta ou não. Isso foi feito para não se fugir do escopo do trabalho, já que uma comparação automática necessitaria da implementação de um classificador apropriado. Nesse caso, o trabalho entraria bastante na área de Reconhecimento de Padrões, especificamente de caracteres (Seção 2.6).

A Tabela 2 apresenta os resultados obtidos na aplicação dos três métodos de segmentação abordados nos testes. Na primeira coluna, têm-se as entradas para os

algoritmos, e posteriormente, separadas por linhas triplas, os resultados de cada técnica. Para cada técnica, duas imagens, que no caso ideal, devem corresponder aos dígitos originais. Os campos da tabela grafados com 'x' indicam que, para aquela entrada, o algoritmo não forneceu resposta. Usou-se um 'x' nesse caso para não haver confusão com algum traço de dígito. Lembra-se que para o algoritmo Alhajj e Elnagar, os resultados estão esqueletizados, devido a um dos passos do método (Seção 3.4).

Tabela 2. Resultados dos experimentos.

Entrada	*Drop-Fall*		Renaudin *et al.*		Alhajj e Elnagar	
						
			x	x	x	x
						
					x	x
						
			x	x	x	x
						
					x	x
					x	x
					x	x

Entrada	*Drop-Fall*		Renaudin *et al.*		Alhajj e Elnagar	
			x	x		
					x	x
					x	x
						
					x	x
						
						
			x	x	x	x
					x	x
			x	x		
					x	x
			x	x		

A partir da análise da Tabela 2, pode-se notar que nenhuma das técnicas obteve bons resultados, ou seja, segmentação correta em mais da metade das situações. Em muitos casos, houve traços que não constituem possibilidade alguma de serem reconhecidos como dígitos, além de alguns casos que provavelmente gerariam erros no classificador, por parecer com outro dígito que não o original.

Detalhando mais sobre a técnica Alhajj e Elnagar, podem-se observar alguns pontos. Primeiramente, em boa parte dos casos, a coluna de segmentação não dividiu a imagem corretamente. Além de que, a divisão em uma coluna específica apresenta o problema de não se adaptar à inclinação da escrita.

Segundo, não houve resposta para a metade das imagens testadas. Isso inclusive pode ser visto pelo fato de haver vinte e quatro imagens de resultado, quando deveriam existir quarenta e oito imagens de resultado já que há vinte e quatro imagens de teste compostas por dois dígitos.

Em nove dos casos em que o algoritmo Alhajj e Elnagar não apresentou uma resposta, uma situação não prevista no trabalho original ocorreu. Mais detalhadamente, se relaciona a quando um dos agentes envolvidos na segmentação não consegue encontrar um vale ou morro, dependendo do agente em questão. Nesse caso, considerou-se que a profundidade ou altura é igual a zero, e que o ponto que funciona como centro de um vale, seria o centro da imagem. Assim, quando se tem um ponto característico na mesma coluna que o centro da imagem, a confidência do agente – Equação (7) – resulta em uma indeterminação matemática, do tipo “%”. Essa situação não foi abordada no trabalho original, e foi até surpreendente já que no artigo [6], Alhajj e Elnagar comentam que, quando não se encontra vales, a tendência é que o outro agente vença a negociação. Porém, analisando de outra forma, no caso citado, o ponto característico está no mesmo eixo vertical do centro do vale, fato que normalmente conduziria a confidência desse agente a seu valor máximo, a unidade. Dessa maneira, fica evidente a confusão entre os conceitos, fato que motivou a reportar esses casos como falta de resposta do algoritmo.

Nos outros três casos, caminhos que não constituíam vales foram considerados como tal e, então, houve inconsistências nos cálculos, fato que gerou a falta de resposta do algoritmo. Relata-se que esse problema pode ser específico da implementação realizada.

Segundo consta no trabalho de Alhajj e Elnagar, o processo de reconstrução resolve vários casos de traços partidos, ou elementos que não fazem parte de um lado da imagem, porém do outro. No entanto, pontos não foram bem especificados no algoritmo, como por exemplo, a diferenciação entre um traço isolado (que ele

consegue recuperar) e um dígito ‘1’ (o qual permanece onde está). Isso reforça a justificativa da não-implementação desse mecanismo.

Em relação ao algoritmo Renaudin *et al.*, houve cinco casos em que o algoritmo não apresentou resposta alguma. Isso provavelmente está relacionado aos grafos construídos a partir da sobre-segmentação, os quais podem não ter permitido a construção dos segmentos. Não se conseguiu identificar a causa exata desse problema. Além disso, houve alguns casos em que a imagem foi dividida horizontalmente, como se existisse um dígito na parte superior e outro na inferior, em vez do correto o qual é um dígito à esquerda e o outro à direita.

Relacionado ao algoritmo *Drop-Fall*, pode-se identificar um ponto positivo o qual é o fato de sempre haver resposta para uma entrada, nesse mérito específico não importando se a resposta é correta ou não. No entanto, observando-se a segmentação em si, nota-se que em alguns casos, pequenos traços foram considerados como dígitos. Esse fato ocorreu devido ao ponto de início adotado e na sequência, à estratégia de continuar a trajetória se não há nenhum *pixel* ativo como obstáculo. Ressalta-se mesmo assim que o comportamento do *Drop-Fall* foi o esperado.

Não se fez uma análise quanto ao desempenho dos algoritmos em termos de velocidade de processamento. Isso se deveu principalmente a dois motivos: (i) não se tem os dados referentes ao algoritmo Renaudin *et al.*, dado que se utilizou os resultados já prontos gerados pelo próprio Christophe Renaudin; (ii) os outros dois algoritmos estão programados em linguagens de programação diferentes – o *Drop-Fall* em Java, e o Alhajj e Elnagar em MATLAB.

Quanto à complexidade computacional [30], certamente o Renaudin *et al*. é o mais custoso, visto que a complexidade envolvida no processo de geração dos traços candidatos, a partir das áreas regulares é exponencial, $O(2^n)$. Para os outros dois algoritmos, a complexidade está mais relacionada ao tamanho da imagem, ou dos dígitos que as constituem, indicando uma complexidade linear, $O(n)$, em relação a isso. Lembra-se nesse ponto, que para qualquer algoritmo envolvendo Processamento de Imagens, no cômputo geral, a complexidade computacional no mínimo é quadrática, $O(n^2)$, devido ao fato de uma imagem ser representada como uma matriz [12].

Capítulo 5

Conclusões

O processamento de imagens de documentos se constitui em uma tarefa de grande importância para a sociedade. Esse fato se deve ao valor agregado a certos documentos, como os documentos históricos, os quais possuem alto valor cultural, além de necessitarem serem preservados e desejavelmente divulgados. Isso também se aplica a cheques bancários, os quais envolvem valor financeiro diretamente, além de cartas e outros documentos que podem apresentar informações relevantes, sejam de interesse pessoal, governamental, etc.

Nesse contexto, a transposição de documentos para o meio digital se torna um importante modo de se preservar e divulgar a cultura de uma sociedade. Para a divulgação dos diversos acervos, a transformação das imagens geradas para arquivos texto se mostra como alternativa viável para a realização dessa tarefa. As ferramentas de OCR se prestam a gerar os arquivos texto a partir das imagens dos documentos. Isso se mostra ainda mais evidente dado que é impraticável realizar esse processo de forma manual para a grande quantidade de documentos existentes e gerados a cada dia.

A segmentação de caracteres é uma etapa muito importante no processamento automático de documentos, consequentemente em uma aplicação de OCR. Isso se deve ao fato que erros nessa etapa acarretam insucesso ao reconhecimento de caracteres mesmo que se utilize um bom classificador.

5.1 Contribuições

Este trabalho se propôs a estudar técnicas de segmentação de dígitos, contribuindo para os estudos e pesquisas realizadas na área nos últimos anos. Mostrou-se a importância dessa tarefa no processamento de imagens de documentos, o estado da arte da área, no sentido em que se estudaram técnicas bastante recentes. Estudou-se em mais detalhes o algoritmo Alhajj e Elnagar [6], inclusive implementando-o em MATLAB. Aplicou-se esse algoritmo a um conjunto de vinte e

quatro imagens, a fim de se verificar seu desempenho, e também compará-lo com outras duas técnicas estudadas no trabalho – *Drop-Fall* [9] e Renaudin *et al.* [7], sendo essa uma das maiores contribuições do trabalho. Através de melhoramentos no trabalho, especificamente sobre os experimentos para que os mesmos tenham relevância estatística, pode-se estender e contribuir cientificamente a partir da comparação realizada.

Verificou-se que para o conjunto de imagens utilizado, nenhuma das técnicas obteve resultados visualmente satisfatórios: houve vários casos de traços que não seriam reconhecidos como dígitos, casos que gerariam confusão a um classificador, além de casos de rejeição por parte dos algoritmos. Ressalta-se que se utilizaram casos complexos de segmentação, pois para casos simples (conexão por um ponto ou caracteres já separados) há técnicas que resolvem o problema sem maiores dificuldades.

Ainda mais, encontrou-se uma situação não prevista no algoritmo Alhajj e Elnagar. Em vários exemplos, chegou-se a uma indeterminação matemática no cálculo da confidência dos agentes; fato que *a priori*, impossibilita a geração do resultado. Esta também se configura como contribuição importante deste trabalho. Também se contribuiu em relação ao método Alhajj e Elnagar, com uma especificação mais detalhada do processo para encontrar vales, além de uma definição das equações empregadas no algoritmo de forma mais natural aos trabalhos realizados em Processamento de Imagens.

Além disso, este trabalho também contribuiu com a publicação de um artigo científico no ACM Symposium on Document Engineering 2008 (ACM DocEng'08) [4], congresso internacional classificado como Qualis A pela CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), realizado na cidade de São Paulo.

5.2 Trabalhos Futuros

Segmentação de caracteres é uma área de pesquisa em constante desenvolvimento. Como trabalho futuro, primeiramente pode-se tentar aplicar os segmentos obtidos a um classificador, a fim de se ter um melhor panorama sobre o desempenho dos métodos estudados.

Pode-se também fazer uma análise da base de imagens utilizada, para verificar se algum método consegue bons resultados, pois é possível que as imagens sejam bastante complexas para as técnicas já existentes. Isso inclusive engloba também estudo de mais algoritmos de segmentação de dígitos. Além disso, podem-se aplicar os algoritmos a uma base de imagens natural, ou seja, amostras de casos de conexão e sobreposição entre caracteres de documentos escritos, e não gerados artificialmente como foi feito neste trabalho para validar os resultados obtidos.

Um trabalho futuro, como aprimoramento deste, é realizar experimentos com maior relevância estatística a fim de se consolidar as afirmações geradas a partir dos experimentos.

Outro trabalho futuro consiste em se generalizar o algoritmo Alhajj e Elnagar, propondo alguma solução para a situação não prevista encontrada. Isso pode incluir alguma heurística ou então a utilização de outro método de segmentação quando este caso ocorrer.

Ainda mais, é possível se aplicar as técnicas estudadas, as quais se prestam a separação de dígitos, em caracteres tanto para avaliar o desempenho, como possibilitar a generalização delas para o caso de caracteres, ou ainda proporcionar bases para o desenvolvimento de novos algoritmos de segmentação de caracteres.

Bibliografia

[1] PARKER, J. R. "*Algorithms for Image Processing and Computer Vision*". John Wiley and Sons, 1ª ed., 1997.

[2] WITTEN, I. H., MOFFAT, A., "*Managing Gigabytes - Compressing and Indexing Documents and Images*". Van Nostrand Reinhold, 1994.

[3] NEVES, R. F. P., MELLO, C. A. B., SILVA, M. S., BEZERRA, B. L. D. "*A New Technique to Threshold the Courtesy Amount of Brazilian Bank Checks*". In: 15th International Conference on Systems, Signals and Image Processing (15th IWSSIP), Bratislava, Eslováquia, junho, 2008. Proceedings of the 15th International Conference on Systems, Signals and Image Processing, IEEE Computer Society Press, 2008, v. 1, p. 93-96.

[4] MELLO, C. A. B., ROE, E., LACERDA, E. B., "*Segmentation of Overlapping Cursive Handwritten Digits*". In: ACM Symposium on Document Engineering 2008 (ACM DocEng'08), São Paulo, Brasil, setembro, 2008. Proceedings of the Eight ACM Symposium on Document Engineering, v. 1, p. 271-274.

[5] RUSSEL, S., NORVIG, P., "*Inteligência Artificial*", Campus, 4ª ed., 2004.

[6] ALHAJJ, R., ELNAGAR, A., "*A Novel Approach to Separate Handwritten Connected Digits*". In: Seventh International Conference on Document Analysis and Recognition (ICDAR2003), Edimburgo, Escócia, agosto, 2003. Proceedings of the Seventh International Conference on Document Analysis and Recognition, IEEE Computer Society Press, 2003, v.2., p. 1198-1202.

[7] RENAUDIN, C., RICQUEBOURG, Y., CAMILLERAP, J., "*A General Method of Segmentation-Recognition Collaboration Applied to Pairs of Touching and Overlapping Symbols*". In: Ninth International Conference on Document Analysis and Recognition (ICDAR2007), Curitiba, Brasil, setembro, 2007. Proceedings of the Ninth International Conference on Document Analysis and Recognition, IEEE Computer Society Press, 2007, v.1, p. 659-663.

[8] CASEY, R. G., LECOLINET, E. "*A Survey of Methods and Strategies in Character Segmentation*". IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 18, n. 7, 1996, p. 690-706.

[9] CONGEDO, G., DIMAURO, G., IMPEDOVO, S., PIRLO, G., "*Segmentation of Numeric Strings*". In: Third International Conference on Document Analysis and Recognition (ICDAR1995). Proceedings of the Third International Conference on Document Analysis and Recognition, IEEE Computer Society Press, 1995, v. 1, p. 1038-1041.

[10] MELLO, C. A. B., SCHULER, L. A. "*Thresholding Images of Historical Documents Using a Tsallis-Entropy Based Algorithm*". Journal of Software, v. 3, n. 6, 2008, p. 29-36.

[11] SEZGIN, M., SANKUR, B. "*A Survey* over Image Thresholding Techniques and Quantitative Performance Evaluation". Journal of Electronic Imaging, v. 13, n. 1, 2004, p. 146-165.

[12] GOMES, J., VELHO, L. "*Computação Gráfica: Imagem*". Sociedade Brasileira de Matemática, 2ª ed., 2002.

[13] KOERICH, A. L., LEE, L. L. "*Processamento Automático de Cheques Bancários: Armazenamento e Recuperação de Informações*". Revista

Controle e Automação, Sociedade Brasileira de Automática, v. 12, n. 1, 2001, p.52-63.

[14] MELLO, C. A. B., OLIVEIRA, A. L. I., SANCHEZ, A. "*Historical Document Image Binarization*". In: Third International Conference on Computer Vision Theory and Applications (VISAPP 2008), Funchal, Portugal, janeiro, 2008. Proceedings of the Third International Conference on Computer Vision Theory and Applications, INSTICC Press, 2008, v. 1, p.108-113.

[15] MELLO, C. A. B, SANCHEZ, A., OLIVEIRA, A. L. I., LOPES, A. "*An Efficient Gray-Level Thresholding Algorithm for Historic Document Images"*. Journal of Cultural Heritage, v. 9, 2008, p. 109-116.

[16] BASTOS FILHO, C. J. A., MELLO, C. A. B., ANDRADE, J. D., FALCÃO, D. M. A., LIMA, M. P., SANTOS, W. P., OLIVEIRA, A. L. I. "*Thresholding Images of Historical Documents with Back-to-Front Interference Based on Color Quantization by Genetic Algorithms*". In: 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007), Patras, Grécia, outubro, 2007. Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence, v. 1, 2007, p. 488-491.

[17] MELLO, C. A. B. "*An Algorithm for Foreground-Background Separation in Low Quality Patrimonial Document Images*". In: 12th Iberoamerican Congress on Pattern Recognition (CIARP 2007), Valparaiso, Chile, novembro, 2007. Lecture Notes in Computer Science, Springer, 2007, p. 911-920.

[18] MELLO, C. A. B., BEZERRA, B., ZANCHETTIN, C., MACARIO, V. "*An Efficient Algorithm for Thresholding of Brazilian Bank Checks*", In: Ninth International Conference on Document Analysis and Recognition (ICDAR2007), Curitiba, Brasil, setembro, 2007. Proceedings of the Ninth International Conference on Document Analysis and Recognition, IEEE Computer Society Press, 2007, v.1, p. 193-197.

[19] GONZALEZ, R., WOODS, R. "*Digital Image Processing*". Addison-Wesley, 2002.

[20] O'GORMAN, KASTURI, R. "*Document Image Analysis*". IEEE Computer Society Press Executive Brief Series, 1997.

[21] MELLO, C. A. B. "Filtragem, Compressão e Síntese de Imagens de Documentos Históricos". Tese de Doutorado, Centro de Informática, UFPE, 2002.

[22] ALVES, V. M. O. "Segmentação de Linhas em Imagens de Documentos Históricos". Trabalho de Conclusão de Curso, Departamento de Sistemas e Computação, UPE, 2007.

[23] SHAPIRO, L. G., STOCKMAN, G. C. "*Computer Vision*". Prentice-Hall, 2001.

[24] SILVA JÚNIOR, E. R. "*Investigação de Técnicas de Extração e Seleção de Características e Classificadores Aplicados ao Problema de Classificação de Dígitos Manuscritos de Imagens de Documentos Históricos*". Trabalho de Conclusão de Curso, Departamento de Sistemas e Computação, UPE, 2007.

[25] OLIVEIRA, A. L. I., MELLO, C. A. B., SILVA JÚNIOR, E. R., ALVES, V. M. O. "*Optical Digit Recognition for Images of Handwritten Historical Documents*", In: Simpósio Brasileiro de Redes Neurais (SBRN'06), Ribeirão Preto, Brasil, outubro, 2006. Proceedings of Ninth Brazilian Symposium on Neural Networks, 2006, v. 9, p. 29.

[26] HAYKIN, S. "*Redes Neurais: Princípios e Prática*". Bookman, 2ª ed., 2001.

[27] SAYRE, K. M. "*Machine Recognition of Handwritten Words: a Project Report*". Pattern Recognition, v. 5, n. 3, 1973, p. 213-228.

[28] SHRIDAR, M., BADREDLIN, A. "*Recognition of Isolated and Simply Connected Handwritten Numerals*". Pattern Recognition, v. 19, n. 1, 1986, p. 1-12.

[29] PETTIER, J. C., CAMILLERAP, J. "*Script Representation by a Generalized Skeleton*". In: Second International Conference on Document Analysis and Recognition (ICDAR1993), Tsukuba, Japão, outubro, 1993. Proceedings of the Second International Conference on Document Analysis and Recognition, IEEE Computer Society Press, 1993, v. 1, p. 850-853.

[30] MANBER, U. "*Introduction to Algorithms: A Creative Approach*", Addison-Wesley, 1989.

[31] KHOTAZAND, A., HONG, Y. H., "*Invariant Image Recognition by Zernike Moments*". IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 12, n. 5, 1990, p. 489-497.

[32] LAM, L., LEE, S. W., SUEN, C. Y. "*Thinning Methodologies: a Survey*". IEEE Transactions on Pattern Analysis and Machine Intelligence, v.14, n.9, 1992, p. 869-885.

[33] HANSELMAN, D., LITTLEFIELD, B. "*MATLAB 6: Curso Completo*". Prentice-Hall, 2003.

[34] MUGHAL, K. A., RASMUSSEN, R. "*A Programmer's Guide to Java Certification"*. Addison-Wesley, 1ª ed., 2000.

[35] LECUN, Y., CORTES, C. "*The MNIST Database of Handwritten Digits*". Disponível em: http://yann.lecun.com/exdb/mnist/. Acesso em: 30 mai. 2009.