ESCOLA POLITÉCNICA
DE PERNAMBUCO

Departamento de
Sistemas
Computacionais

Análise de Algoritmos para Segmentação de Textos de Imagens de Documentos Históricos

Trabalho de Conclusão de Curso

Engenharia da Computação

Juliane Cristina Botelho de Oliveira Lima
Orientador: Prof. Dr. Carlos Alexandre Barros de Mello
Co-orientador: Prof. Dr. Ángel Sanchez

Recife, novembro de 2006

UNIVERSIDADE
DE PERNAMBUCO

ESCOLA POLITÉCNICA
DE PERNAMBUCO

Departamento de
Sistemas
Computacionais

Análise de Algoritmos para Segmentação de Textos de Imagens de Documentos Históricos

Trabalho de Conclusão de Curso

Engenharia da Computação

Este Projeto é apresentado como requisito parcial para obtenção do diploma de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Juliane Cristina Botelho de Oliveira Lima
Orientador: Prof. Dr. Carlos Alexandre Barros de Mello
Co-orientador: Prof. Dr. Ángel Sanchez

Recife, novembro de 2006

UNIVERSIDADE
DE PERNAMBUCO

Juliane Cristina Botelho de Oliveira Lima

Análise de Algoritmos para Segmentação de Textos de Imagens de Documentos Históricos

Resumo

Segmentação de documentos é uma área crítica no processo de reconhecimento de caracteres. Muitos dos avanços na identificação de caracteres têm ocorrido devido a estudos na área de segmentação de caracteres numa imagem. Selecionar o método que melhor extraia as características necessárias é provavelmente um dos fatores mais importante para assegurar uma alta taxa de reconhecimento.

Quando se esta tratando documentos históricos às etapas pré-reconhecimento se tornam mais críticas devido ao processo de envelhecimento que o papel sofre com o decorrer dos anos. Problemas como manchas clareamento da tinta entre outros são aspectos freqüentes nesses textos, e que dificultam a localização dos caracteres numa imagem.

Este trabalho apresenta um estudo das fases de pré-reconhecimento de caracteres, dando ênfase à etapa de segmentação de documentos. A imagem é adquirida no padrão RGB e passa por diversas etapas antes de ser segmentada. Primeiramente ela é binarizada, depois são aplicados algoritmos que minimizem informações não relevantes, deixando a imagem adequada ao processo de segmentação.

A implementação dos algoritmos foi realizada utilizando o Matlab. Os algoritmos implementados foram aplicados a um acervo histórico que possui tanto documentos datilografados e manuscritos. E foi avaliada sua taxa de identificação de segmentos numa cena.

Abstract

Document Segmentation is a critical area in the process of character recognition. Many of the advances in the identification of characters have occurred because of studies in the area of segmentation of characters in an image. To select the method that better extracts the necessary characteristics is probably one of the most important factors to assure one high rate of recognition.

When you are treating historical documents, the step before recognization is more difficult due the aging process that the paper suffers, so the segmentation becomes more critical. Problems as spots, discolouration, desitegration of parts, rust from metalics clips among others are frequent aspects in these texts, and they make the localization of the characters in an image more difficult.

This work presents a study of the phases of before recognition of characters, giving emphasis to the stage of document segmentation. The image is acquired in RGB standard and passes for diverse stages before being segmented. First it is thresholded, later is applied algorithms that minimize inconsistent information, leaving the image to the segmentation process adequate.

The implementation of algorithms was made using the Matlab. The implemented algorithms had been applied to a historical documents that possesss as many typed handwritting. We evaluated its rate of identification of segments in a scene.

Sumário

Índice de Figuras

Índice de Tabelas

Tabela de Símbolos e Siglas

CRT - *Color Cathode-Rays Tubes.*
OCR - *Optical Character Recognition*.
dpi – dots per inch (pontos por polegada).
PDI – Processamento de Imagem

Agradecimentos

Agradeço a todos que contribuíram de forma direta ou indireta para o desenvolvimento deste trabalho, e em especial a:

• Meus pais, que me incentivaram, dando total apoio e confiança em todos os momentos, principalmente os mais difíceis e me deram força para buscar e realizar meus sonhos;

• Meus irmãos, primos e tios pelo apoio e pela disposição em ajudar em outras atividades para que eu tivesse mais tempo para me dedicar à produção deste trabalho, além de entender o domínio do computador durante algum (longo) tempo;

• Carlos Alexandre de Mello, que me confiou a incumbência de tratar de um tema tão instigante e por me orientar de forma brilhante, sempre atencioso e disposto a colaborar com o desenvolvimento do trabalho além da ajuda gratificante do professor Angel Sánchez;

• Maria Lencastre, por ter me dado à primeira oportunidade de realizar um trabalho de pesquisa e por não hesitar em me ajudar em todos os momentos de dúvida;

• Aos meus grandes amigos de turma: Emanuel Pessôa, Gabriel Falconieri, Renata Medeiros, Robson Alcântara e Tássia Lima e, uns estão comigo desde o começo da caminha e outros entraram no decorrer do curso, e a todos os outros não menos importantes, mas que não tenho condições de citar por questões de espaço, que compartilharam de momentos de alegria e momentos difíceis durante a graduação;

• A todos os meus amigos, Margarethe Barbosa e Talita Rampazzo e aos demais que de alguma forma contribuíram positivamente não só para este trabalho, mas também de qualquer outra forma nessa caminhada até aqui, dando-me apoio em momentos que fraquejei e que pouco me tentaram durante a produção deste trabalho.

Capítulo 1

Introdução

É conhecido que o papel, com o decorrer do tempo, sofre um processo de deterioração devido a seu material ser suscetível à ação de diversos elementos como umidade, sujeira, etc, bem como sua fragilidade. Esse fator é preocupante quando tratamos de documentos de importância histórica, já que uma grande quantidade de informação histórica valiosa possui seu conteúdo armazenado apenas de forma impressa. A necessidade de conservar essa informação impulsionou a busca pela digitalização de documentos como forma de preservar seu conteúdo, além de proporcionar uma maior divulgação desse material. Isso vale, principalmente, para acervos históricos, os quais apenas hoje em dia vêm sendo convertidos para meios digitais e armazenados em imagens. Técnicas de análise de documentos [1][2][3] estão sendo aplicadas no processamento dessas imagens, convertendo-as em textos editáveis que ocupam uma menor área de armazenamento, preservando o conteúdo. Por exemplo, a imagem de um documento que ocupa cerca de 4MB, pode vir a ocupar cerca de 80 Kb [2].

A conversão de documentos históricos para a sua representação eletrônica desprende um esforço muito maior do que a digitalização dos documentos contemporâneos, devido ao envelhecimento e a condições que aumentam a dificuldade no processo de análise de documentos, já que os métodos tradicionais assumem que o plano de fundo da imagem seja liso e que possua uma uniformidade na qualidade do conteúdo escrito. Além disso, documentos como cartas, escrituras mesmo estes sendo datilografados ou manuscritos não possuem um *layout* pré-definido o que dificulta a identificação da localização dos textos [4]

A análise das imagens de documentos consiste na interpretação dos objetos que compõem o documento, extraindo informações relevantes [2]. Essa análise envolve várias etapas de processamento que determinam a estrutura física do documento, o relacionamento entre os objetos que o compõem e, por fim, a recomposição do seu conteúdo [5][6][7]. Dentre as etapas podemos destacar a limiarização, o pré-processamento, a segmentação dos documentos , a segmentação do texto e a classificação do conteúdo encontrado. Esses processos encontram-se detalhados em [2].

O processo de limiarização [8] consiste em converter a imagem para preto-e-branco. No caso de documentos, pode-se entender o processo como a separação em duas classes, a do plano de fundo a dos objetos quem compõem a imagem.

Algumas imagens quando digitalizadas não possuem um enquadramento perfeito, como requer o processo de reconhecimento de caracteres. Parte do pré-processamento consiste em trabalhar na correção de erros sobre a orientação da imagem, identificando seu eixo de rotação e aplicando as correções necessárias. Várias técnicas podem ser utilizadas para a identificação da inclinação das imagens, como a transformada de Hough [3]. Além disso, nessa fase, é necessário tratar o nível de impureza encontrado nas imagens. É comum o aparecimento de manchas nos documentos e, quando esses são digitalizados, essa informação se mantém na imagem e, em geral, não é filtrada pelo processo de limiarização. É necessário então a aplicação de algoritmos que façam essa limpeza na imagem.

Como resultado desta fase, têm-se apenas os objetos que serão tratados, sem identificar o que cada um representa na cena, apenas excluindo dados que são irrelevantes. A fase a seguir é responsável pelo tratamento lógico [2] dos componentes da imagem, a segmentação de documentos. Ela consiste na implementação de algoritmos que identifiquem áreas gráficas e textuais, organizando-as em classes distintas. A fase de segmentação de texto trata a classe textual fazendo uma interpretação do conteúdo da imagem, identificando linhas, palavras e *ghyphs* (segmentos que representam prováveis caracteres). A etapa de classificação trata do reconhecimento de caracteres que irá identificar o conteúdo do texto, através de algoritmos de OCR (*Optical Character Recognition*).

1.1 Motivação

As pesquisas correntes em OCR estão tratando documentos que ainda não são manipulados perfeitamente pelos sistemas atualmente disponíveis no mercado. Problemas como a degradação dos documentos, a variedade na intensidade da escrita e a variação da caligrafia em documentos manuscritos, além do clareamento da tinta do texto são fortes campos de pesquisa. Para assegurar um alto desempenho no processo de reconhecimento, o desafio é localizar onde se encontram os caracteres na imagem. Para isso, a segmentação de documentos é o ponto chave, pois tem a função de identificar as regiões de interesse no processamento de documentos, que são as áreas que contém a texto.

Dada a grande quantidade de métodos de segmentação apresentados pela literatura, o desafio encontra-se em determinar qual o método que melhor identifique a localização dos caracteres. Este trabalho encontra-se em um projeto desenvolvido pela Universidade de Pernambuco em parceria com a Universidad Rey Juan Carlos (Espanha). Alguns trabalhos relacionados são [9][10][11].

1.2 Objetivo

O parâmetro de avaliação da qualidade dos resultados no reconhecimento de caracteres era padronizado pelo quão exato um algoritmo de OCR pudesse identificar um caractere [2]. Entretanto, as etapas executadas antes possuem um nível de influência muito decisivo nesse processo; quanto melhor identificado um segmento que represente um caractere, melhor será o grau de acerto do algoritmo. Isso ocorre principalmente em imagens de documentos históricos que além da dificuldade inerente ao processo de reconhecimento, possuem características que o agravam como: intemperismos sofrido pelo papel, o desgaste da tinta utilizada, e a grande quantidade de acervo manuscrito.

O objetivo deste trabalho é estudar algoritmos de segmentação de documentos, implementá-los e testar a sua eficiência quando aplicados a acervos de documentos históricos. A

etapa de segmentação de documentos é de fundamental importância para a extração da informação requerida pelo processo de reconhecimento de caracteres. Esse processo tem como finalidade definir o algoritmo que melhor separa os componentes textuais desses acervos. Um algoritmo de segmentação aplicado a uma imagem pode identificar bem objetos dentro de uma cena, enquanto que em outras imagens isso não ocorre, possibilitando inclusive, a distorção da imagem como a adição de ruídos. Como existe uma grande variedade de algoritmos de segmentação, foram priorizados os que são mais representativos nesta área.

Uma avaliação em termos de classificação e tempo de resposta do desempenho dos algoritmos não está no escopo deste trabalho.

Os algoritmos foram implementados em Matlab [12] por ser um sistema cujo elemento básico é uma matriz o que facilita trabalhar com problemas que envolvem formulações matriciais – observando que as imagens digitais são representadas mais comumente como matrizes. Além disso, o Matlab possui um alto desempenho para cálculos numéricos.

1.3 Descrição dos Capítulos

Esta monografia está organizada da seguinte forma:

- **Capítulo 1:** Apresenta uma breve introdução do que é o trabalho especificando a motivação e os objetivos.
- **Capítulo 2**: Descreve os conceitos básicos na área de processamento de imagens, que são de fundamental importância para o entendimento do desenvolvimento do projeto.
- **Capítulo 3:** Descreve as principais etapas do processamento digital de documentos.
- **Capitulo 4:** Apresenta alguns algoritmos de segmentação e as implementações desenvolvidas neste trabalho
- **Capitulo 5**: Estão dispostos os resultados obtidos.
- **Capítulo 6:** Apresenta as considerações finais do trabalho.

Além desses Capítulos, a monografia inclui ainda um Apêndice com os códigos implementados neste trabalho.

Capítulo 2

Conceitos Básicos

Apresentamos aqui os conceitos básicos envolvidos em processamento de imagens digitais que servem de base para o entendimento dos tópicos abordados neste trabalho.

2.1 Imagens Digitais

Uma imagem digital é a representação computacional de uma imagem no computador. É definida como uma função bi-dimensional não contínua que pode assumir valores em um intervalo pré-determinado que descreve a sua aparência. De acordo com [6], pode-se considerar que uma imagem é um retângulo composto por linhas no eixo X e colunas no eixo Y. A resolução dessa imagem identifica a quantidade de linhas pela quantidade de colunas (X x Y). O padrão de coordenadas não segue o adotado pela matemática dos eixos X e Y, convencionou-se o ponto *f(0,0)* como a margem superior esquerda e que o ponto *f(X-1, Y-1)* na margem inferior direita (sendo *f* a imagem). A Figura 1 apresenta essa representação. Essa função é definida como:

$$f(x, y) = \text{Cor no ponto } (x, y).$$
$$x \in [0,X] \text{ e } y \in [0,Y]$$

Cada coordenada da imagem é denominada pixel, e a quantidade de pixels determina a resolução de uma imagem. O valor de cada pixel depende da natureza da imagem, pois é o resultado da combinação de vários fatores como a intensidade da luz, a distância, a temperatura além do sistema de cores, podendo esse pixel assumir qualquer valor num intervalo pré-definido. Esse intervalo determina o limite do contraste e do brilho.

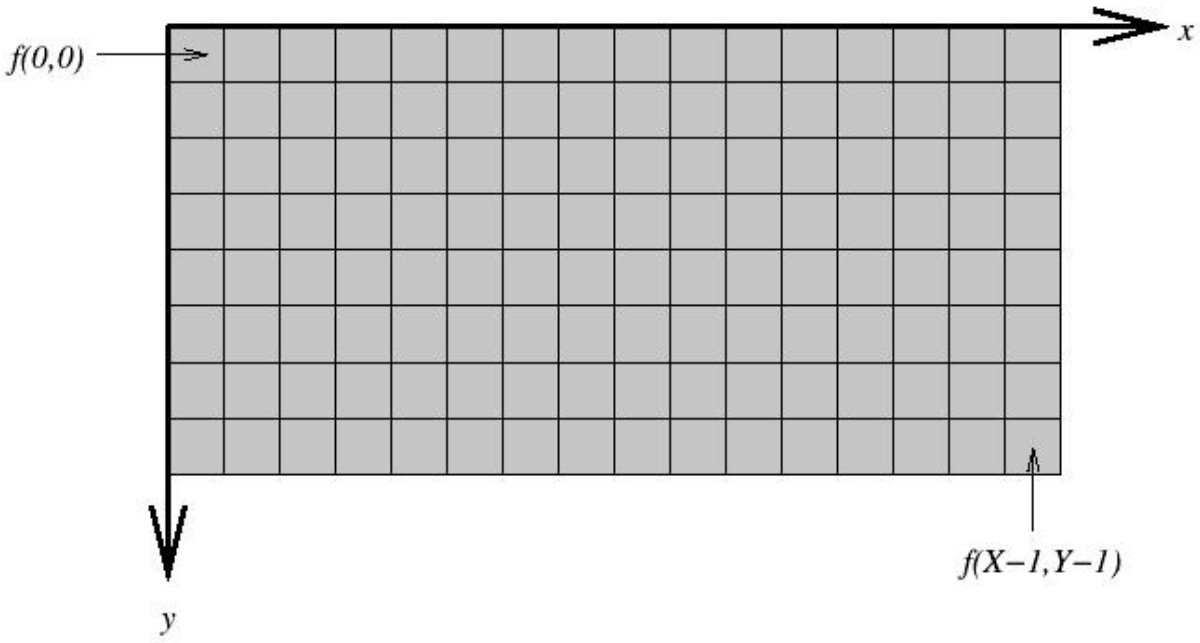


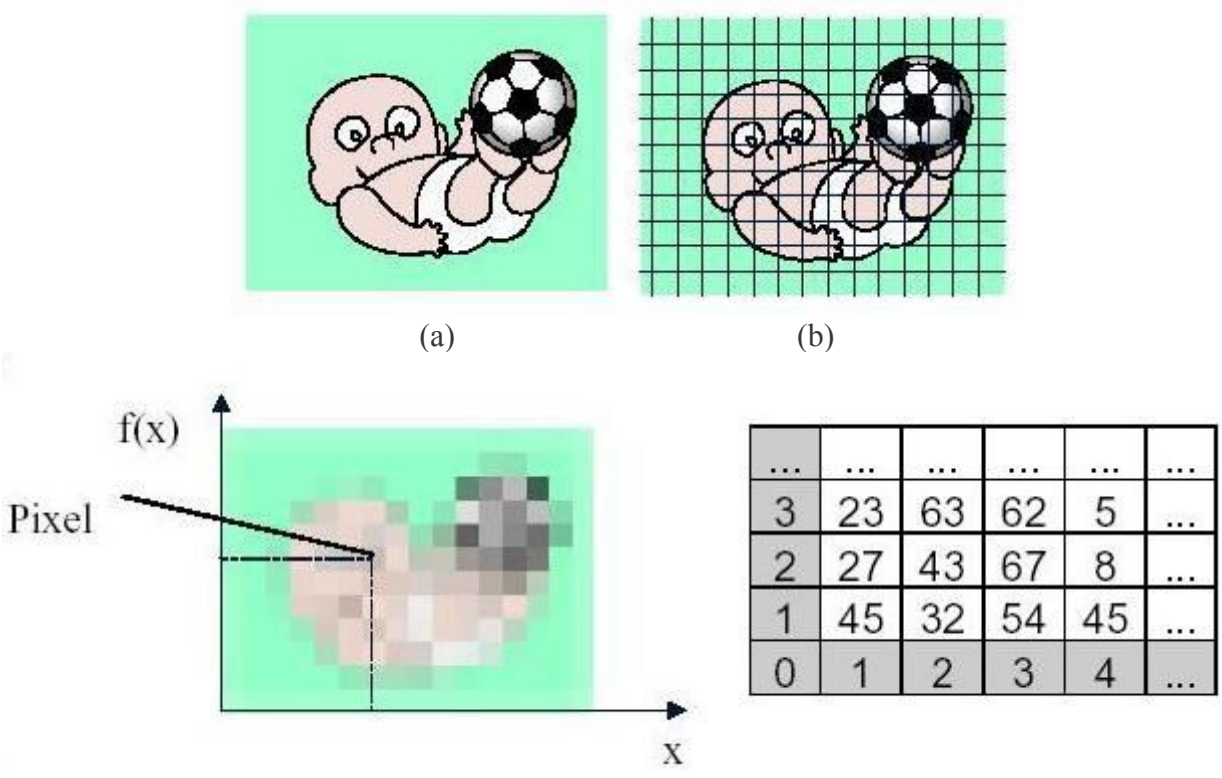
Figura 1. Imagem digital retangular com resolução de 16x8.

Uma imagem pode ser em tons de cinza ou colorida. A primeira é obtida captando apenas a intensidade da luz. O pixel de menor brilho é considerado o preto e o de maior brilho branco, e os demais tons de cinza são distribuídos proporcionalmente para cada pixel de acordo com o seu brilho. Uma imagem colorida é obtida avaliando a intensidade e a crominância da luz para identificar a medida da cor do pixel. Cada cor é definida como um vetor de componentes que identifica a intensidade de cada cor básica. Os sistemas mais comuns do espaço de cores são o RGB cujas cores básicas são o vermelho, o verde e o azul; o HSV que utiliza a matiz (*hue*), a saturação (*saturation*) e o valor (*value*) [5][6]. A Figura 2 apresenta um exemplo de imagens colorida e em tons de cinza.

Como foi visto, uma imagem digital é uma função discreta, enquanto que a imagem real é uma função contínua. Então, esta deve ser discretizada tanto espacialmente quanto em intensidade a fim de que o computador possa manipulá-la. O processo de discretização produz uma imagem discreta, a partir de um número finito de amostras colhidas da imagem contínua. A discretização ao longo dos eixos x e y é a espacial, definida como amostragem. No entanto, cada célula dessa estrutura também sofre um processo de discretização a fim de definir sua cor; esse processo é denominado quantização [13]. Ambos os processos são uniformes, ou seja, a imagem é amostrada em pontos igualmente espaçados distribuídos na forma de uma matriz que representa a imagem como na Figura 1. Cada elemento é o pixel que representa a cor para o ponto amostrado. A imagem digital resultante é a aproximação da imagem analógica a partir da qual ela foi obtida. Esse processo é mostrado na Figura 3.

(a) (b)

Figura 2. Exemplos de imagens: (a) apresenta uma imagem em tons de cinzas com resolução 256x256; (b) uma imagem colorida utilizando o sistema de cores RGB com resolução 256x256.



(a) (b)

(c) (d)

Figura 3. Exemplos do processo de Digitalização: (a) representa uma imagem analógica; (b) o processo de amostragem; (c) o processo de quantização e em (d) a codificação da imagem.

2.1.1 Vizinhança de um Pixel

Vizinhança [13] ou janela em torno de um pixel é o conjunto de pontos que ficam ao redor do primeiro. Por exemplo, uma janela de 3x3 em torno de um ponto P na posição (x,y) é composto pelos pontos descritos pela função:

$$f(x, y) = \begin{matrix} I(x-1, y-1) & I(x-1, y) & I(x-1, y+1) \\ I(x, y-1) & I(x, y) & I(x, y+1) \\ I(x+1, y-1) & I(x+1, y) & I(x+1, y+1) \end{matrix}$$

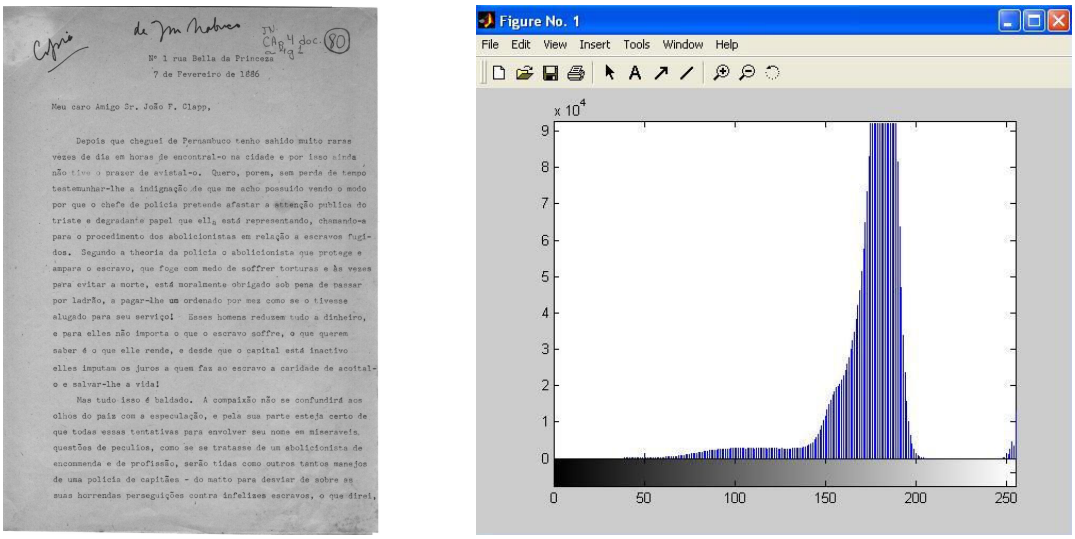
O tamanho da vizinhança varia de acordo com a necessidade da aplicação, podendo ser utilizada janelas maiores como 4x4, 5x5. Este conceito é importante já que vários algoritmos de processamento de imagem não utilizam apenas o ponto da imagem, mas também os pixels imediatamente próximos ao ponto em que o processamento está sendo feito.

2.1.2 Histograma

O histograma [6] de uma imagem é uma função que define o número de ocorrências de cada nível de cinza, ou de ocorrência de cada cor em uma imagem colorida. Ele representa a distribuição dos valores dos pixels. Então, dado um tom de cinza ou uma cor, a função é a relação entre a quantidade de vezes que essa cor aparece pela quantidade de pixels da imagem. A função é definida como:

$$f(c) = \frac{Nc}{N}$$

Em que, *c* é a cor ou o tom de cinza, *Nc* é a quantidade de vezes que *c* aparece na imagem e *N* é a quantidade de pixels. O histograma permite avaliar a distribuição dos níveis de cinza, ou das cores numa imagem, verificando características como determinar se uma imagem é colorida, escura ou clara. A Figura 4 é um exemplo de um histograma de uma imagem em tons de cinza..



(a) (b)

Figura 4. Imagem de um documento em tons de cinza em (a), e em (b) o histograma da imagem gerado pelo Matlab, aplicando a função *imhist*.

2.2 Fundamentos de Cores

Segundo [6] a utilização de cores no processamento de imagens ocorre principalmente devido a capacidade de identificação simples de objetos e a extração de formas numa cena. Além de os seres humanos poderem distinguir uma infinidade de cores entre milhares de máscaras, facilitando uma análise visual da imagem. O processamento de cores é dividido em duas áreas: uma que trata uma infinidade de cores através dos dispositivos de captura/ geração como um senso, uma televisão, já que estes podem representar uma infinidade de cores; a outra trata de pseudocores, ou seja, atribui uma cor particular a uma intensidade monocromática.

Uma imagem é o resultado de estímulos luminosos [6] associados a um suporte bidimensional que corresponde à superfície (curva) da retina do olho humano. A luz é uma onda eletromagnética cuja intensidade da radiação visível em função do comprimento de onda encontra-se em entre 350 e 780 nanômetros (nm), ver Figura 5. A luz possui a propriedade da dualidade, comportando-se ora como partícula ora como onda eletromagnética.

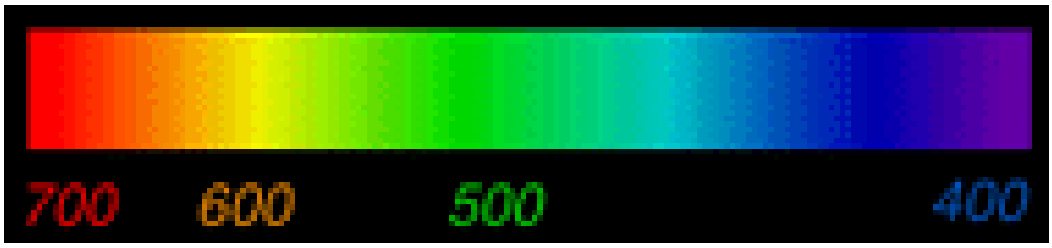


Figura 5. Faixas visíveis do espectro.

A luz chega aos olhos humanos de três formas: através de uma fonte luminosa como o Sol, ou uma lâmpada; através do reflexo da luz sobre os objetos ou através da luz transmitida através de um objeto translúcido. Em cada situação, o olho pode identificar cores diferentes. A cor é a interpretação da emissão da luz pelo sistema visual de acordo com o comprimento de onda. Segundo Isaac Newton, [5][6]a luz do Sol é uma mescla heterogênea de raios (cores) diferentes. Ele provou sua teoria através do efeito da refração, aplicando um raio de luz solar sobre um prisma. Nesse experimento o raio se dividiu em várias cores como mostrado na Figura 6.

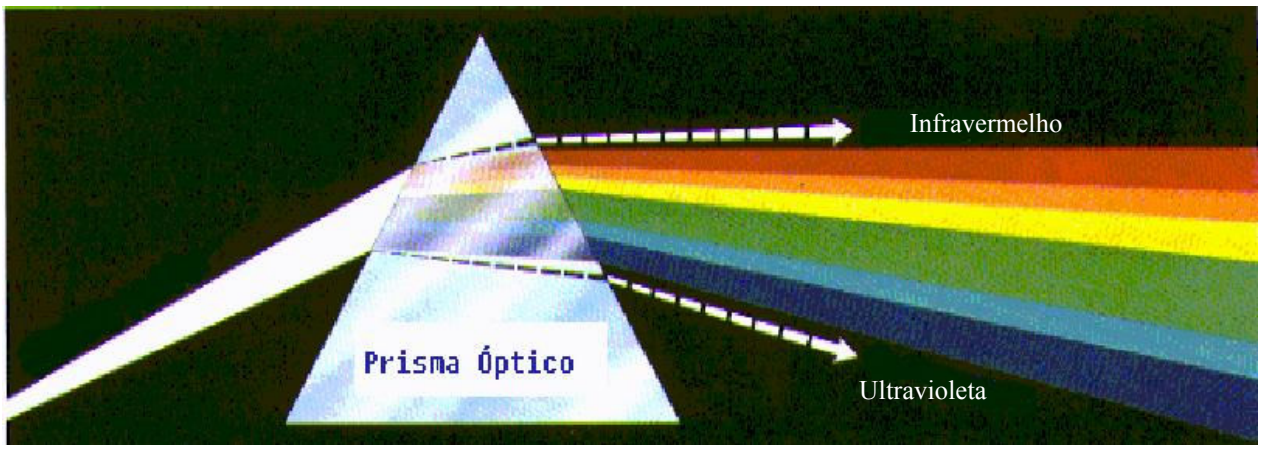


Figura 6. Experimento de Issac Newton que provou a sua teoria de que a luz solar é a mescla heterogênea das cores.

Existem dois tipos de células receptoras no olho com sensibilidades diferentes: os bastonetes os cones [5][6]. O primeiro é responsável por captar a intensidade da luz, independente da cor, a

Figura 7 mostra o gráfico da luminosidade. Enquanto os cones são responsáveis por captar a intensidade das ondas que representam as cores básicas. De acordo com o tamanho das ondas, os cones define a intensidade de cada cor, em que as ondas curtos representa a cor é azul, as ondas medias a cor verde e as ondas longas o vermelho, a Figura 8 mostra o gráfico do espectro das cores captadas pelo olho. O processo de percepção ocorre da seguinte maneira: a retina do olho capta a freqüência da luz e combina essa informação definindo a luminância e a crominância. Luminância [5][6] define a intensidade da luz, ou seja, corresponde à percepção do brilho irradiado pelos emissores ou pela luminosidade refletida por um objeto enquanto que a crominância [5][6] contém a tonalidade cromática associada ao comprimento de onda e a saturação que determina se a cor é forte ou não. O sinal luminoso é então enviado para cérebro como impulsos nervosos[5] .

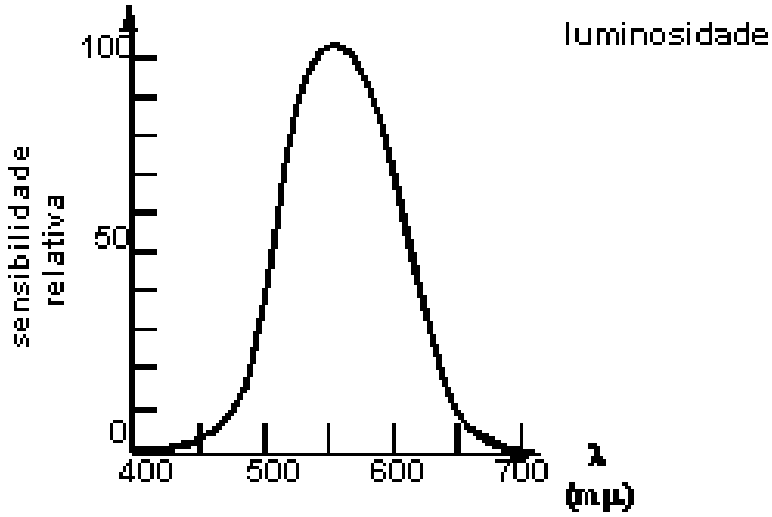


Figura 7. Gráfico que representa a luminosidade captada pelos bastonetes.

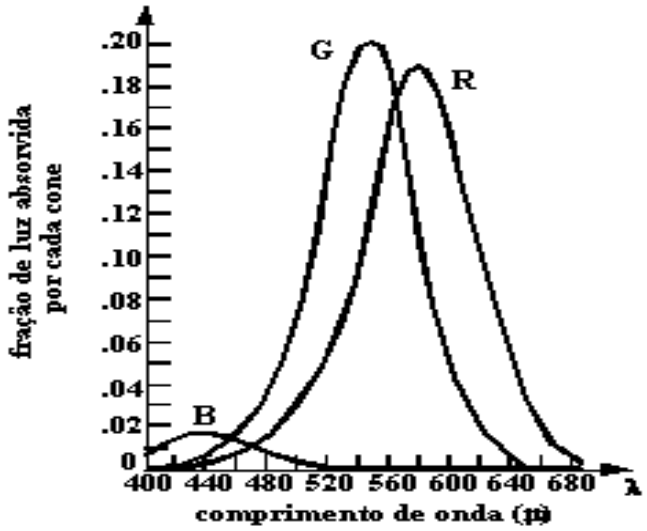


Figura 8. Gráfico que representa a noção de cores captada pelos cones.

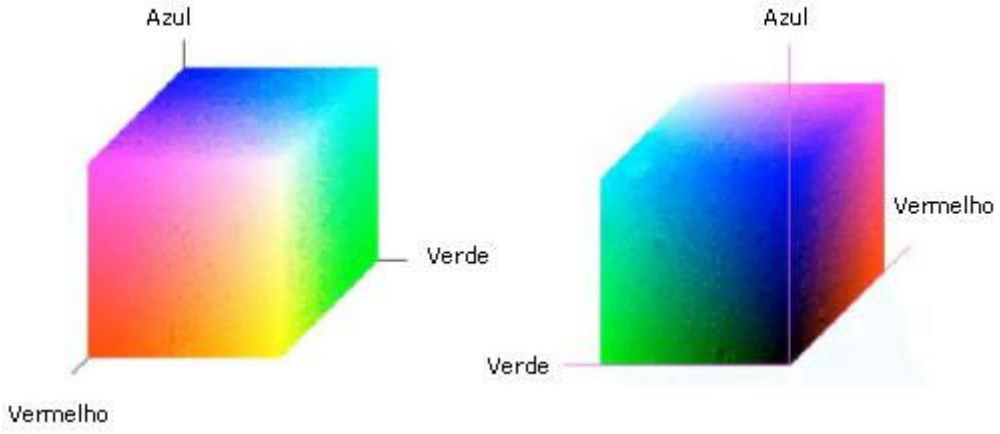
Como foi visto, cada pixel representa a cor em um ponto da imagem e o conjunto desses pixels compõe a imagem. Os modelos de cores especificam as cores conforme algum padrão. A especificação das cores em um padrão, ou modelo, é efetuada através de um sistema de

coordenadas de três dimensões, e através de um subespaço nesse sistema no qual cada uma das cores possíveis é representada por um único ponto conforme descrito em [6].

Um modelo de cores é incapaz de representar todas as cores visíveis, por ser um subconjunto do espaço de cores de todas as combinações possíveis de um conjunto de cores básicas. Esse subconjunto de cores representáveis pelo modelo recebe o nome de gamute de cores [1][6]. Diferentes modelos de cores foram criados, visando a atender aos objetivos de diferentes dispositivos ou aplicações. Cada dispositivo de captura ou exibição de cores possui seu próprio gamute de cores, de acordo com um modelo de cores específico.

O padrão de cores mais comum é o RGB que foi criado para tubos de raios catódicos coloridos (*color Cathode-Rays Tubes* - CRT), utilizados em monitores de vídeo. Cada uma das cores é representada de acordo com os valores de seus respectivos componentes espectrais, ou seja, os componentes espectrais de uma cor correspondem a valores para o vermelho, para o verde e para o azul [6].

A representação das cores baseia-se em um sistema de coordenadas cartesianas tridimensional. O subespaço de cores é formado pelas combinações das três cores básicas utilizadas pelo RGB, como no cubo apresentado na Figura 9.



(a) (b)

Figura 9. Modelo de cores RGB: (a) apresenta uma visão do ponto (1, 1,1) e em (b) do ponto (0 0,0).

2.2.1 Mudanças no Espaço de Cor

Uma cena real possui uma infinidade de cores, mas quando essa cena é digitalizada vários tons são perdidos devido à quantidade limitada de tons que um dispositivo pode representar. Como foi visto na Seção 2.2, alguns atributos são utilizados para classificar as cores e para qualificá-las: tonalidade, saturação e luminosidade ou brilho. Eles são muito úteis também para combinar as cores entre as diversas formas de uso: *scanner*, monitor, impressora e outros.

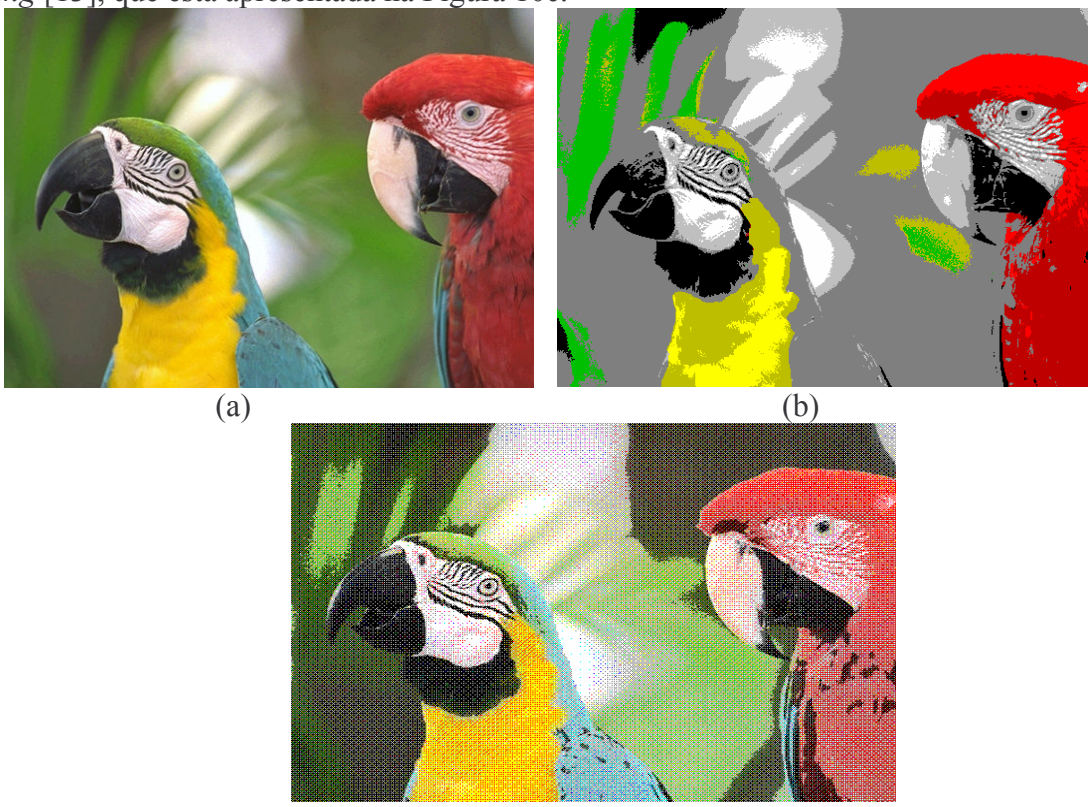
A profundidade de cor [5] de uma imagem define o número de cores possíveis, que pode ser comparada com uma imagem de um artista que usa uma paleta com determinada variedade de cores. Uma imagem digital pode ser criada ou digitalizada com diferentes paletas de cores, que vão de duas cores (1 bit, por exemplo: branco ou preto) até milhões de cores (24 bits), conforme Tabela 1.

Tabela 1. Número de *bits* e quantidade de cores.

Número de *bits*	Quantidade de cores
1	2
8	256
13	65536
24	16,7 milhões
32	4 bilhões

Cada dispositivo possui uma resolução e uma capacidade de geração e/ ou exibição de um certo número de cores. Portanto, é necessário adaptar as características da imagem aos recursos disponíveis no dispositivo utilizado. O processo de quantização [5][6] consiste em reduzir a quantidade de intensidades de cores. Como no novo padrão existe uma quantidade inferior de cores disponíveis o sistema altera o valor da cor para uma cor mais próxima. Nesse processo de redução ocorre à perda da informação, e vários problemas podem ser adicionados à imagem como falsas bordas e a perda de textura.

Várias técnicas são desenvolvidas para minimizar as falhas [13] introduzidas pelo processo de redução da precisão das cores ver Figura 10(a) e (b). Uma técnica é o processo de *dithering* [13], que está apresentada na Figura 10c.



(a) (b)

(c)

Figura 10. Exemplo de recorte de cores: (a) Imagem original com 4331, (b) imagem com 16 cores e (c) a imagem com 16 cores geradas pelo processo de *dithering*.

2.2.2 *Dithering*

Como fora visto, quando ocorre o recorte de cores, a perda de informação produz um erro nos contornos dos objetos que compões a imagem por não possuir a mesma quantidade de cores nos dois gamute. *Dithering* [13] é uma técnica que foi criada para compensar as cores que faltam. Ela consiste em misturar as cores disponíveis de forma a simular as cores que faltam, gerando imagens mais realistas com diferentes paletas de cores. Então, pontos pretos, mais ou menos agrupados em um fundo branco aparentam formar diferentes tons de cinza. Isso ocorre devido à forte correlação entre a cor de um pixel e a cor dos seus vizinhos. Então é necessário um processo de filtragem para minimizar a percepção dos contornos de quantização. Em certos casos, mesmo utilizando-se bons algoritmos de quantização fica difícil não perceber os contornos de quantização.

Dithering se fundamenta numa característica da visão humana que o sistema visual integra os estímulos luminosos recebidos dentro de um ângulo, então é possível perceber intensidades que não necessariamente existem na imagem e sim o resultado da integração dada pela média das intensidades de cor nas vizinhanças de cada elemento da imagem [13].

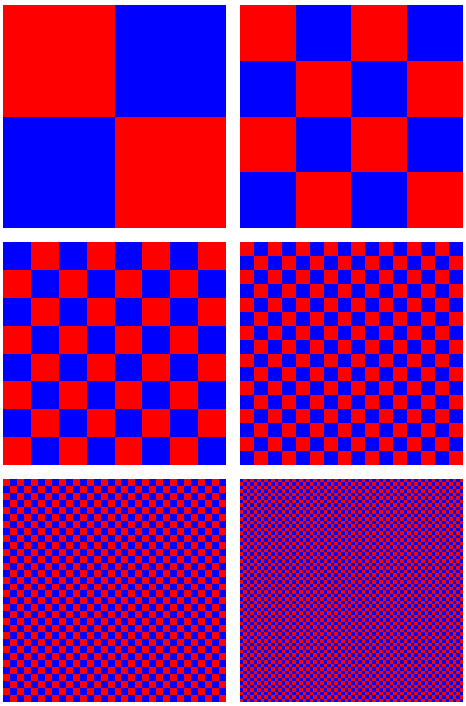


Figura 11. Justaposição de pixels utilizando duas cores para dar a ilusão de uma terceira.

2.3 Processamento Digital de Imagens

Processamento de imagens (PDI) é definido como o processo de manipulação, modificação e análise em que a entrada é uma imagem e a saída também é imagem [13]. Ou seja, PDI trabalha sobre uma matriz numérica que representa a imagem original, aplicando funções e/ ou técnicas a fim de gerar uma nova imagem diferente da original. A estação de processamento de imagens deve fornecer três ferramentas que possibilitem um meio de digitalização, de visualização e de manipulação. PDI tem como propulsor a necessidade de melhorar a qualidade das informações de uma imagem e o desenvolvimento de dispositivos com percepção e extração do conteúdo da imagem automaticamente, sendo capaz de processar dados a partir de uma cena [6]. Uma das

primeiras aplicações nos anos 20 foi o melhoramento das imagens de jornal transmitidas via cabo submarino entre Londres e Nova York, além da introdução do sistema do cabo Bartlane que reduziu o tempo de transmissão de imagens de duas semanas para três horas, utilizando equipamento dedicado. As imagens tinham apenas cinco níveis de intensidade, como mostra a Figura 12.

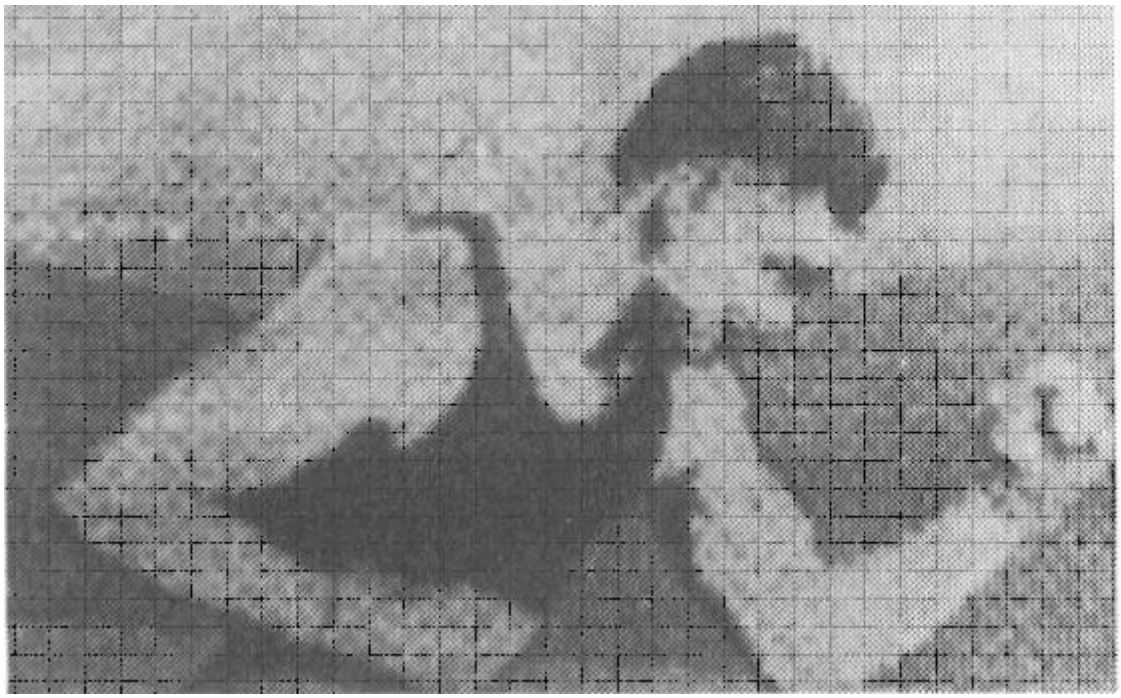
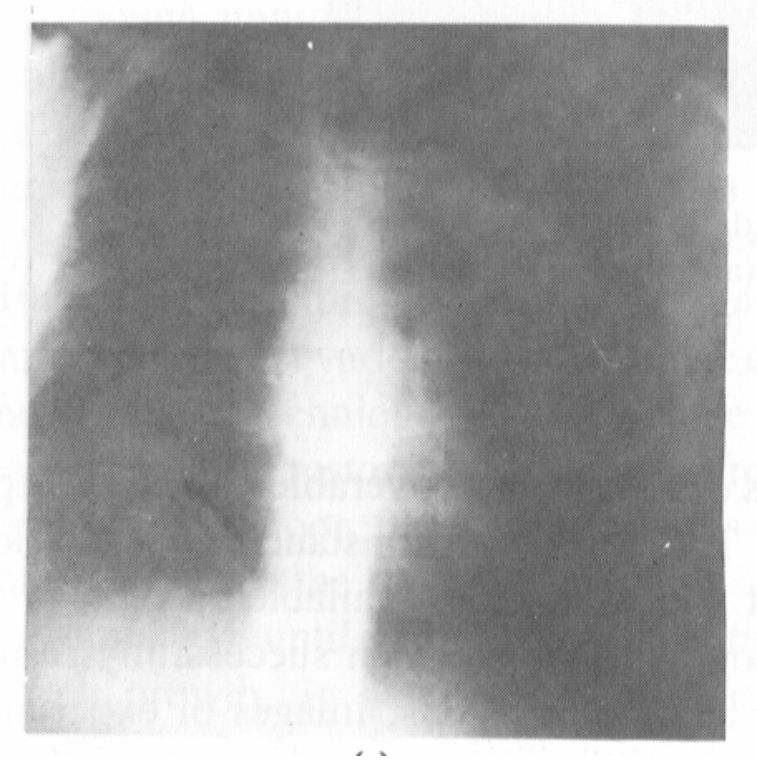
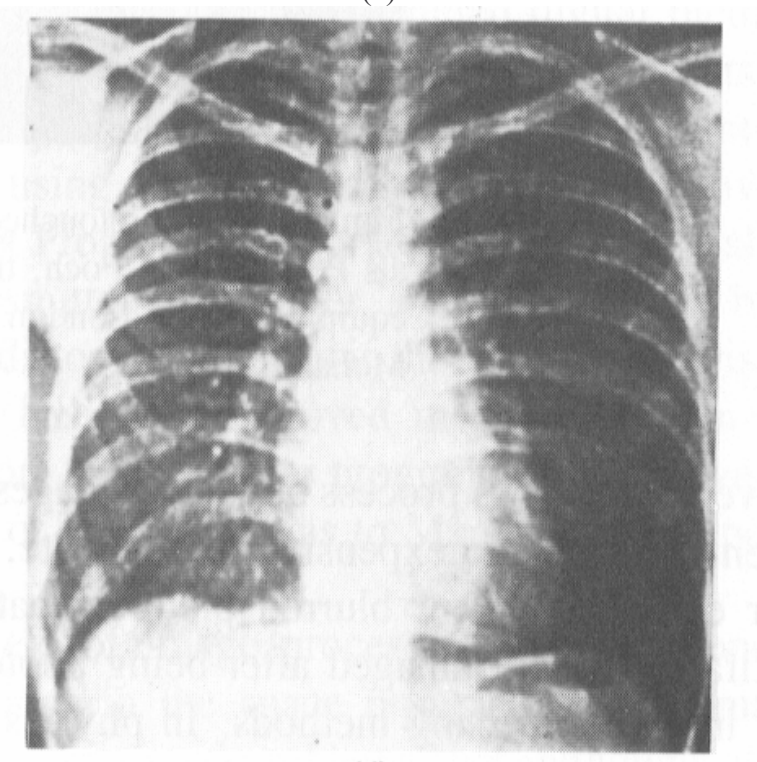


Figura 12. Imagem digital de 1921 utilizando uma impressora de telégrafo com 5 tons de cinza.

Com o aprimoramento computacional (aquisição, armazenagem, capacidade de processamento, comunicação e exibição) e o desenvolvimento de novos algoritmos, o processamento de imagem entrou no dia-a-dia de outras áreas e acabou se tornando algo transparente ao usuário [6]. Um exemplo da utilização do processamento de imagem é na área médica em que estão sendo estudadas formas de extrair informações relevantes como um tumor a partir de uma radiografia, ou simplesmente melhorar a qualidade da imagem aplicando filtros e contrastes (ver Figura 13).



(a)



(b)

Figura 13. Raio-x de tórax. Em (a) encontra-se a imagem original e em (b) os contornos são melhorados pela aplicação do contrate.

2.3.1 Filtragem

Os filtros são utilizados para otimizar a visualização da imagem, seja pela redução de ruídos ou pela minimização de pequenos detalhes que não fazem parte dos objetos que compõem a imagem [6]. A grande maioria das técnicas para melhorar ou realçar imagens são baseadas em operações espaciais realizadas sobre os vizinhos mais próximos dos pixels da imagem original. Os filtros são utilizados de acordo com a natureza da aplicação. O uso de máscaras ou janelas de observação para o processamento de imagens é chamado de filtragem em que as máscaras são os filtros.

Técnicas de filtragem podem também ser utilizada para melhorar imagens muitas vezes com o objetivo de retirar ruídos introduzidos no processo de digitalização ou mesmo devidos ao processo de preparação das amostras. De um modo geral, uma forma simples para a remoção de ruído é analisar os pixels vizinhos para todos os pontos de uma imagem e a seguir substituir pelo nível médio da vizinhança. Em muitas aplicações deseja-se realçar não a imagem como um todo, mas apenas os contornos dos objetos, ou seja, o que se deseja é um operador que possa filtrar informações que estejam prejudicando a boa definição dos contornos e realizar algum tipo de operação que possa salientar os objetos ou suas bordas.

As operações com filtros são divididas em duas classes: a filtragem no domínio da freqüência [6] e a filtragem no domínio espacial [6]. A primeira utiliza transformações locais baseadas no princípio da análise de Fourier [3][6]. Ou seja, consiste em manipular a componente de amplitude da imagem que é o resultado da aplicação da transformada de Fourier sobre a imagem. O resultado é a aplicação da transformada inversa de Fourier utilizando a amplitude modificada. É bastante útil para redução de ruído expressivos em uma imagem.

O uso de filtros no domínio espacial tem como conseqüência a variação no valor de um pixel da cena original, em que o valor varia de acordo com a vizinhança dos pixels que compõem a máscara em função de pesos atribuídos a cada elemento da máscara, podendo ser um valor positivo, negativo ou nulo. Como os pesos podem assumir valores diferentes, ponderando a influência de cada vizinho. Através da combinação dos valores de entrada aos pesos, se promoverá um maior ou menor realce da cena, segundo as direções de interesse. A esse processo dá-se o nome de convolução [6][13]. A máscara de pesos é definida como kernel de convolução. A operação ocorre da seguinte forma: aplica-se a máscara com centro na posição (x, y), e substitui o valor do pixel nesta posição pelo resultado do somatório da multiplicação de cada elemento da vizinhança pelo valor associado do kernel. É necessário que o resultado seja normalizado para que o valor resultante encontre-se na paleta de cores que está sendo utilizada. Esse processo é apresentado pela Figura 14.

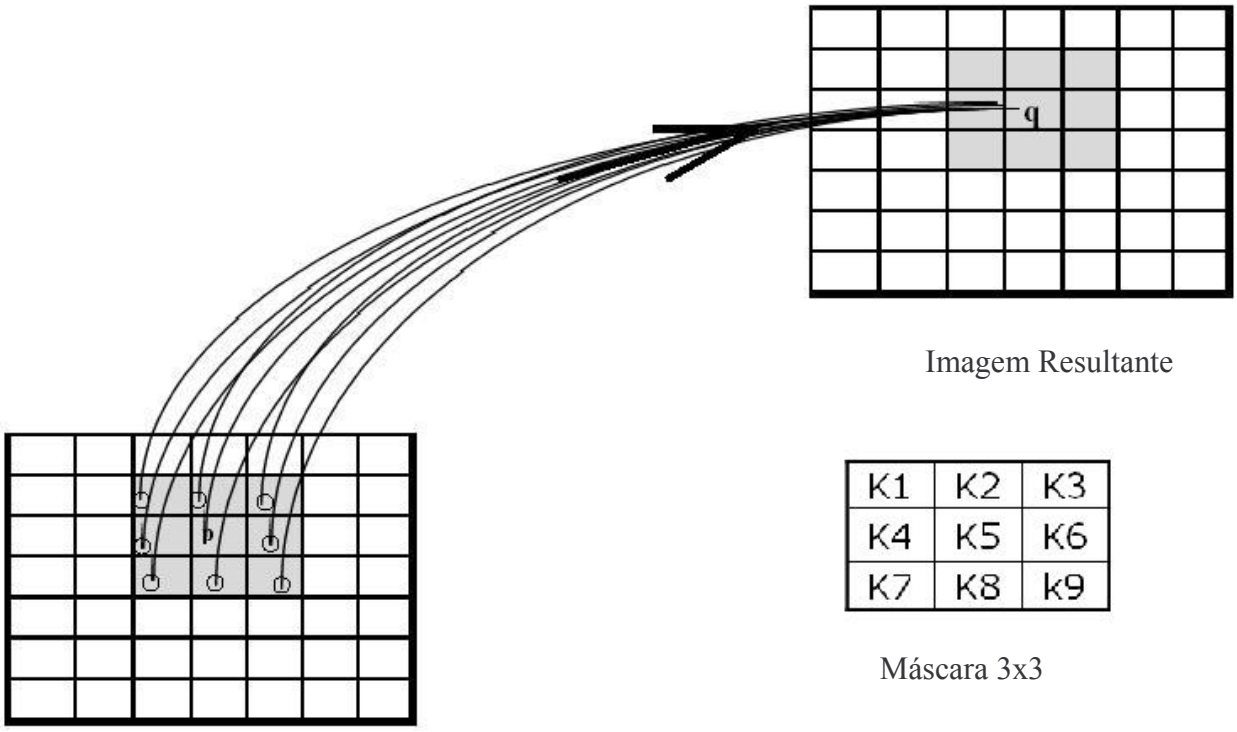


Figura 14. Aplicação de uma máscara sobre a imagem utilizando a convolução.

Filtros espaciais consistem em realçar de altas, médias ou baixas freqüências (tonalidades) [13] de variação do tom da imagem em detrimento das demais. Eles podem ser classificados como passa-baixa e em passa-alta.

O filtro passa-baixa [6] não afeta os pontos ou pixels de baixa freqüência, de tal forma que, após a filtragem, estes pontos permanecem como antes na imagem original. Já os componentes de alta freqüência são atenuados ou eliminados, por exemplo, no caso de pontos randomicamente distribuídos em imagens digitalizadas. Estes pontos são considerados de alta freqüência uma vez que os valores dos pixels vizinhos variam com muita rapidez em relação ao ponto ruidoso. O efeito visual é o de suavização da imagem e apresentam o efeito de borramento já que os pontos pertencentes ao contorno são considerados ruídos em relação ao vizinho. Esse filtro é conhecido como filtros de média, pois é o resultado da média dos pontos vizinhos. A Figura 15 apresenta um exemplo de um filtro passa-baixa. A aplicação do filtro passa-baixa é exemplificada na Figura 16 (b).

$$1/9 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Figura 15. Exemplo de mascara 3x3 de um filtro passa-baixa. O 1/9 é o termo de normalização.

(a) (b)

(c)

Figura 16. Exemplo da aplicação do filtro passa-baixa e passa-alta. A imagem (a) apresenta a figura original e em (b) o resultado da aplicação do filtro passa-baixa e em (c) o resultado da aplicação do filtro passa-alta.

Em contrapartida, a filtragem passa-alta [6] não afeta os pontos de alta freqüência, e sim atenuam ou até mesmo elimina os componentes de baixa freqüência. O resultado principal dos filtros passa-alta é evidenciar as regiões cujas variações nas intensidades dos pixels sejam bruscas, ou seja, realçando os contornos dos objetos que compõem a cena, tornando as transições entre regiões diferentes mais nítidas. São utilizados para realçar certas características da cena como bordas, linhas curvas ou manchas. A apresenta Figura 17 um exemplo de um filtro passa-alta. A aplicação do filtro passa-alta é exemplificada na Figura 16 (c).

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

Figura 17. Exemplo de máscara 3x3 de um filtro passa-alta.

Capítulo 3

Processamento de Documentos

Atualmente, documentos, de uma maneira geral, encontram-se armazenados em sua forma impressa. O papel, com o tempo, passa pelo processo de deterioração devido a fatores climáticos ou até mesmo à forma com que se encontram armazenados. O processo de digitalização vem sendo utilizado como uma alternativa para conservar o conteúdo desses documentos, devido à alta capacidade de armazenamento e a facilidade de cópia de conteúdo. O processamento de documentos (ou a análise da imagem do documento) [1][2][7] consiste em analisar o conteúdo da imagem, interpretando a informação contida nela a fim de possibilitar a conversão do documento em um arquivo de texto editável, mantendo as características inerentes ao documento como figuras, selos, diagramação, dentre outras. Várias técnicas estão sendo utilizadas para a extração de informação [20][21][22] dos documentos impressos, mas ainda é necessário muito trabalho para tratar esse problema.

O processamento de documentos combina o conhecimento da linguagem escrita [1][2], da formatação do documento [7][20][21], o processamento da imagem e sistemas de reconhecimento de caracteres [1][2] para tratar o conteúdo literário do documento. Vários aspectos são abordados na análise do documento, destacando o tratamento de caracteres manuscritos e impressos por máquinas de datilografar. De acordo com [2], o objetivo do processamento de documento é o reconhecimento dos componentes textuais e gráficos do documento agrupando-os em classes distintas e assim dando um tratamento direcionado a cada classe, utilizando ferramentas adequadas para texto e gráficos. O tratamento textual [7][21] do documento consiste na segmentação do documento que identifica as partes lógicas que compõem o texto como parágrafos, linhas, palavras e *glyphs* – fortes candidatos a ser um caractere – e o reconhecimento dos caracteres utilizando uma ferramenta de OCR (*Optical Character Recognition*). O tratamento gráfico [2] consiste em tratar símbolos não textuais no documento, identificando como linhas retas, curvas, figuras, selos e etc. Este trabalho não utiliza a abordagem voltada para a interpretação do conteúdo gráfico, apenas o textual.

Para assegurar a qualidade da análise da imagem dos documentos, pode-se organizar esse processo duas etapas: uma na qual será abordada a estrutura física dos documentos, sendo responsável pela utilização de técnicas para aumentar a qualidade da imagem em nível de pixels [2][22]; e a outra é o tratamento lógico [2][21] do documento que trata seus componentes e como esses estão relacionados. Na primeira etapa, podemos destacar o processo de limiarização, o pré-processamento; e, na segunda, a segmentação dos documentos, a segmentação do texto e a classificação do conteúdo encontrado, conforme ilustrado na Figura 18.

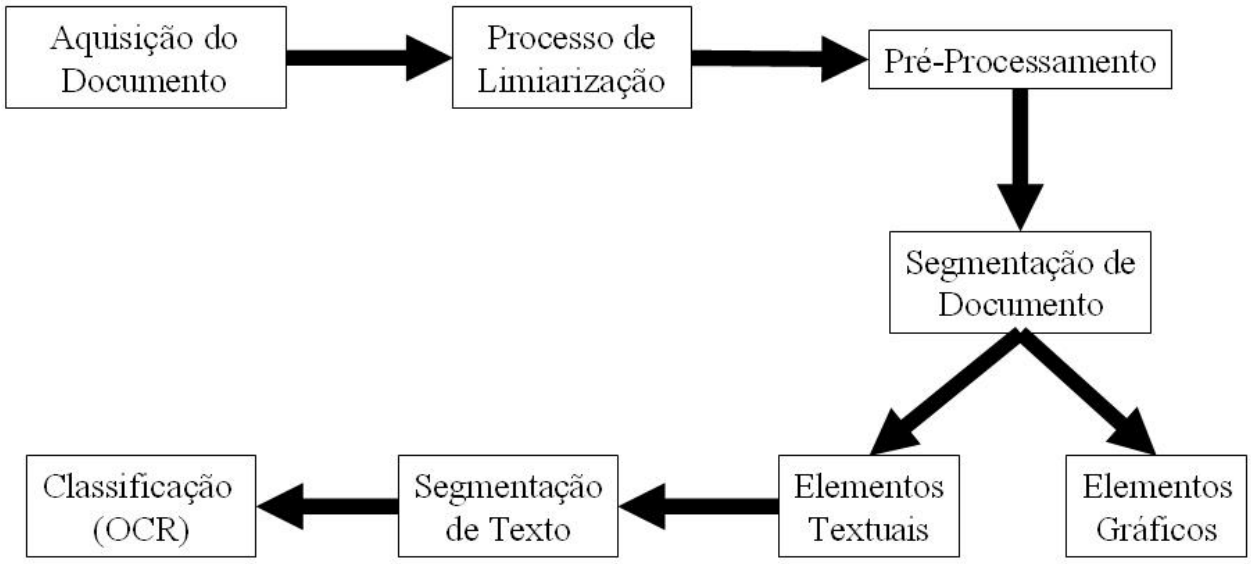


Figura 18. Ilustração das etapas do processo da análise de documentos históricos.

A primeira etapa é de fundamental importância para que haja um desempenho satisfatório nas demais etapas. Ela consiste em realçar [22] as características do documento, minimizando os defeitos encontrados em suas imagens. Dentre esses problemas, podem-se destacar: manchas nos documentos; a não uniformidade das cores; enfraquecimento da tinta da impressão; e problemas adicionados pelo processo de digitalização como sombras na imagem devido à falta de iluminação no processo de aquisição. Nessa etapa, pode-se destacar os processos [2][5] de captura do documento, limiarização e o pré-processamento. A imagem do documento é digitalizada, utilizando dispositivos especializados. São capturadas as cores naturais da imagem, adquirindo o máximo de características possível. A seguir, a imagem é binarizada [7], passando por algoritmos de limiarização com a finalidade convertê-la em apenas dois tons, preto-e-branco. Essa imagem é pré-processada utilizando algoritmos para a redução de informações não pertinentes na imagem como ruído, a correção da orientação do documento dentre outras.

O resultado da etapa anterior serve como entrada para a próxima etapa. A segmentação dos documentos recebe a imagem pré-processada e subdivide os componentes da imagem em classes distintas de acordo com as características do conteúdo do documento. São geradas duas classes, a textual e a gráfica [2][7].Os elementos textuais são tratados pelo processamento de texto para identificar quais os possíveis caracteres que compõem a cena. Cada caractere é individualizado em um bloco e cada bloco é submetido a uma ferramenta de classificação que identifica qual o caractere representado pela *glyph* reconhecido. Esse dado é armazenado em um arquivo de texto que permite que o conteúdo do texto seja editado e ocupe uma menor área de armazenamento.

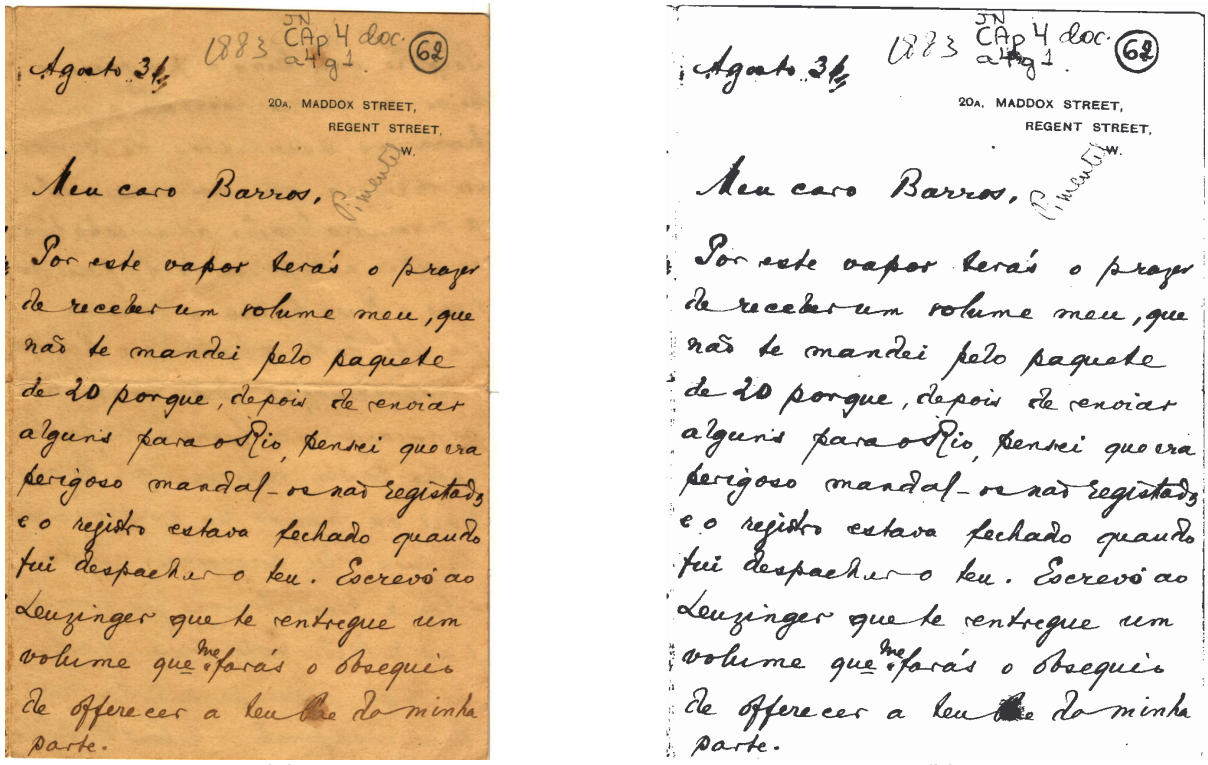
3.1 Pré-Processamento

Uma imagem digital é definida como uma matriz bi-dimensional onde o conteúdo armazenado em cada célula da matriz representa a intensidade da cor ou o tom de cinza nesse ponto, o pixel. Um documento é convertido para a sua forma digital através do processo de digitalização. O desempenho no processamento de imagens depende da qualidade da imagem original e do processo de digitalização. Manchas, buracos, clareamento da tinta são problemas encontrados nos documentos devido ao processo de deterioração que o papel sofre com o tempo. Esse defeitos não

são eliminados no processo de digitalização. Outros problemas podem ser adicionados caso o processo de aquisição da imagem não seja cuidadoso [7]: como ruídos devido a impurezas no dispositivo de captura da imagens; a falta de luminosidade nesse processo pode gerar sombras; o limite da resolução [22] do dispositivo pode não capturar informações relevantes do documento; e a rotação do documento. O pré-processamento consiste em trabalhar a imagem para minimizar esses defeitos. Nesse processo, são considerados algoritmos de limiarização, de correção da orientação do documento e de eliminação de ruído.

3.1.1 Limiarização

Quando se utiliza uma imagem em tons de cinza, o primeiro passo no processamento é a binarização que consiste em converter a imagem para preto-e-branco. Caso a imagem seja colorida, ela deve ser convertida em tons de cinza e o resultado binarizado. Pode-se definir esse processo como a separação dos objetos em uma cena do *background* da imagem, trabalhando apenas com as informações que constituem o documento[1][2]. Isso ocorre através da definição de um limiar, definindo que os valores dos pixels que estiverem à cima desse limiar pertencem ao *background* e seus valores são substituídos pelo valor do pixel branco; e os que estiverem abaixo são armazenados como preto, representando o conteúdo do documento. A Figura 19 mostra um exemplo de binarização. Esse processo se torna difícil quando temos uma imagem de baixo contraste, aproximando a cor da tinta da cor do papel.



(a) (b)

Figura 19. Exemplo de binarização de um manuscrito colorido. (a) Mostra a imagem original e (b) a imagem binarizada.

Existem duas abordagens na binarização [2]: a global, e a local. A primeira utiliza um limiar de binarização para todos os pixel da imagem, sendo esse único para separar o fundo dos componentes do documento. Vários métodos são utilizados para determinar um ponto ótimo para a limiarização global da imagem como a utilização de um histograma que gere um gráfico com a

quantidade de pixels de cada cor contida na imagem. Se a quantidade de pixels de fundo for distinta da quantidade do conteúdo, o histograma de cores gerado possuirá dois picos: um correspondente ao pixel de fundo e outro correspondente ao pixel do texto. O valor ótimo de limiarização é definido pelo pixel que possuir menor quantidade entre os picos [5].

A segunda abordagem é a local [2]. Ela consiste na avaliação de regiões da imagem para determinar qual o melhor valor para o seu limiar. A região é definida por uma janela que percorre toda a imagem, identificando qual o valor ótimo para cada uma.

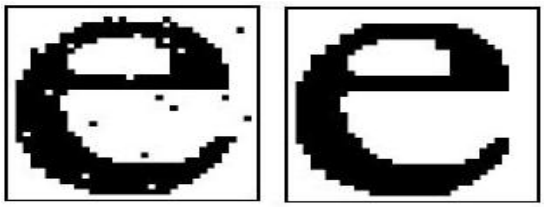
Além desses, existem ainda vários métodos de limiarização de uma imagem. Independente do método selecionado, não existe um que obtenha um resultado perfeito para todo tipo de imagem. A qualidade do desempenho [8] de um algoritmo de binarização depende da qualidade da imagem que está sendo utilizada. Por isso, é necessário avaliar que técnica melhor se adapta ao tipo de imagem que está sendo manipulada, de forma que o resultado possua a menor quantidade de erro.

3.1.2 Redução de Ruído

É comum o aparecimento de manchas nos documentos e, quando esses são digitalizados, essa informação se mantém na imagem e, em geral, não é filtrada pelo processo de binarização. Então é necessário tratar o nível de impureza, os ruídos, encontrados nas imagens através da aplicação de algoritmos que façam uma limpeza na imagem [7][22].

Ruído pode ser entendido como informações que não fazem parte do conteúdo do documento como mostrado na Figura 20 (a). Os ruídos afetam a imagem, podendo impossibilitar a identificação do caractere tratado, além de gerar uma imagem de baixa qualidade. O ruído sal-e-pimenta [22] é o mais comum; ele é constituído de um pixel preto envolto por pixels brancos – a pimenta – ou o inverso – o sal.

Várias soluções são utilizadas para tratar esse problema. Segundo [5] uma solução pode ser no processo de digitalização da imagem, em que são capturadas vários exemplos dos documentos e as informações persistentes na maioria das amostras são as validadas. Essa solução não é tão viável devido à necessidade de várias amostras o que não ocorre com freqüência, além do tempo de processamento para tratar todas as imagens.



(a) (b)

Figura 20. Exemplo de imagem com ruído sal-e-pimenta (a), e o resultado após a aplicação de um algoritmo de redução de ruído.

Então, foram desenvolvidos algoritmos para minimizar a quantidade de ruídos em imagens binarizadas. O processo de redução de ruído é denominado *fill* [5][22]. O filtro da vizinhança é bastante utilizado para essa finalidade, como o mostrado na Figura 20. Ele percorre os vizinhos conectados a um pixel numa área pré-definida por uma matriz quadrada, convertendo a cor pixel relevante para a cor que aparece em maior quantidade nessa área. Outros filtros bastante utilizados são os morfológicos cujas operações mais importantes são a erosão e dilatação [6][22]. A primeira consiste em reduzir a quantidade de pixels pretos, afinando a silhueta dos componentes de cada texto, ver Figura 21. Enquanto que a segunda consiste na expansão dos

pixels pretos, alargando a silhueta das palavras. Isso ocorre através de um elemento estruturante em que na dilatação é resultado da união de todos os pontos da imagem tal que o elemento estruturante intercepte os pixels coloridos da imagem [2][5]. Considerando as imagens binarizadas que são as abordadas nesse processo, um exemplo seria atribuir a cor de todos os vizinhos ao ponto de referência. Então, na dilatação se um ponto faz parte do fundo e tem um vizinho que faz parte do objeto, então o primeiro também faz parte do objeto, enquanto que na erosão, se um ponto faz parte do objeto e tem um vizinho que faz parte do fundo então o primeiro deve fazer parte do fundo.

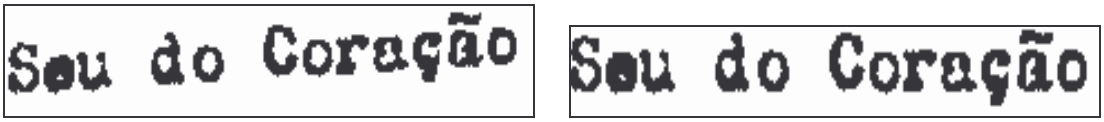
(a) (b) (c)

Figura 21. Exemplo de aplicação de filtros morfológicos utilizando as funções do Matlab: em (a) apresenta a imagem original, em (b) a aplicação da função *imdilate* e ao resultado foi aplicada a função *imerode* (c).

É importante utilizar métodos para reduzir a quantidade de ruído para minimizar a quantidade de erros nas demais etapas do processamento do documento, principalmente no processo de segmentação e de reconhecimento de caracteres.

3.1.3 Orientação da Imagem

Algumas imagens textuais (imagens de documentos) quando digitalizadas não possuem um enquadramento perfeito, como requer o processo de reconhecimento de caracteres. Devido a essa possibilidade de rotação na imagem, é necessário tratar a inclinação em que a imagem encontra-se. A orientação de um documento é o ângulo de inclinação predominante nas linhas do texto do documento como mostrado na Figura 22. Os algoritmos de orientação identificam o ângulo de rotação, permitindo que seja aplicada uma rotação contrária para correção do problema.



(a) (b)

Figura 22. Exemplo de imagem rotacionada: (a) apresenta a imagem rotacionada e (b) a versão corrigida.

De acordo com [3], várias abordagens foram desenvolvidas para determinar o ângulo de rotação de documentos [21][22]. Uma delas estima o ângulo de rotação pela obtenção de uma linha vertical e reta através da margem esquerda do texto. Outra avalia os intervalos que ocorrem entre as linhas horizontais e a partir deles mensuram o ângulo que faça esses intervalos serem o mais claro e reto. A abordagem mais utilizada é a transformada de Hough [3] que, aplicada a uma imagem, pode identificar um conjunto de pixels que pertencem a uma mesma linha reta ou segmentos que formam curvas. A transformada mapeia as coordenadas da linhas no espaço de Hough utilizando senóides, e define o ângulo de orientação como o ponto de maior conversão das linhas.

A transformada de Hough é aplicável quando se possui informações precisas sobre a forma da curva, mas não de sua posição. A idéia transformada é aplicar numa imagem de forma que todos os pontos pertencentes a uma mesma curva, sejam mapeados em num único ponto de um novo espaço de parametrização. Uma reta pode ser parametricamente definida tanto em coordenadas cartezianas como em coordenadas polares. A transformada de hough utiliza a segunda formulação. Então, dada uma imagem I(x,y), a equação da reta é definida como

$$\rho = x \cos \varsigma + y \operatorname{sen} \varsigma$$

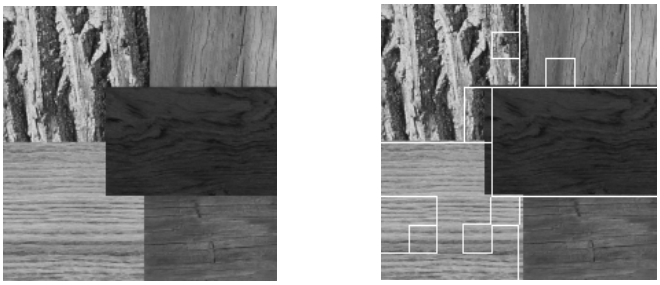
em que ρ e ς são constantes

Para a detecção de uma reta, os passos são ser organizados da seguinte maneira: para cada ponto da imagem I(x,y), são definidas todas as retas de parâmetros ρ e ς que resolvam a equação polar que passam pelo ponto I(x,y). O espaço de parametrização, definido pelo parâmetros ρ e ς, é discretizado em uma tabela de acumulação. Cada elemento da tabela corresponde a um intervalo de ρ e de ς. O valos final de ρ e ς é obtido pela informação do gradiente. O problema desse método é que ele demanda de muito processamento para a sua execução.

Assim que for definido o ângulo de rotação da imagem, o próximo passo é corrigi-lo, fazendo uma rotação no sentido inverso [3][22]. Algo que deve ser levado em consideração nesse procedimento é que componentes conectados na imagem original podem ficar fragmentados ao fim do rotacionamento da imagem, tornando a imagem inadequada para processamento posterior. Uma solução para esse problema é à medida que rotacionar a imagem, calcular a novas coordenadas do pixel e fazer uma estimativa da posição original do pixel a fim de que a imagem seja o mais coerente possível.

3.2 Segmentação de Documentos

A fase seguinte trata da segmentação de documentos. Segmentação [6][13] é o processo de identificação de objetos que possuam características semelhantes em uma imagem. Segmentar uma imagem consiste em particioná-la em regiões de pixels relevantes para uma dad aplicação. Por exemplo, na Figura 23 as diferentes texturas são separadas [5][6]. Dado cada objeto como sendo parte de uma mesma textura, o processo seguinte (dependendo da aplicação) poderia ser a classificação de cada uma dessas texturas.



(a) (b)

Figura 23. Exemplo de segmentação de texturas em uma imagem: (a) mosaico de texturas e (b) regiões identificadas como havendo similaridades.

A segmentação documentos é uma das principais etapas na maioria das aplicações e representa um dos maiores desafios em processamento de imagens. A principal razão dessa dificuldade está na falta de informação sobre os objetos nas imagens [5]. A segmentação consiste de duas tarefas básicas: identificação e delineamento. A identificação indica a posição aproximada do objeto de trabalho na cena e o delineamento extrai sua extensão na imagem [13]. Seres humanos executam a primeira tarefa com relativa facilidade, mas o computador é capaz de

executar a segunda com muito mais precisão do que os seres humanos. A dificuldade da máquina na identificação de objetos se deve à falta de uma descrição global dos objetos na forma de um modelo matemático [2].

Quando se trata de processamento de documentos, o desafio é identificar a posição de cada possível caractere, e ter como resultado uma sub-imagem que possua o *glyphs* sem os pixels do *background* [29]. Como a segmentação é o ponto focal deste trabalho em termos de documentos, ela será tratada em detalhes em um Capítulo específico sobre o assunto.

3.3 Classificação

O processo de classificação consiste em receber um segmento que represente um caractere e identificá-lo como uma letra do alfabeto cujo classificador foi treinado, montando cadeias de letras que representam as palavras que compõem o texto que será armazenado em um arquivo textual que possa ser editado. De acordo com [2], o reconhecimento do caractere deve ocorrer de forma análoga a processada pelos seres humanos, então se um indivíduo consegue facilmente identificar o *glyph* como sendo um 'b', o sistema deve responder o mesmo. Caso a identificação de um segmento seja ambígua, como na Figura 24 que o caractere central pode ser reconhecido como um 'o' ou um 'e', o sistema deve identificar o conjunto de possibilidades de resposta e associar a cada uma, um intervalo de confiança [1][2].

Figura 24. O segundo fragmento, quando submetido a uma ferramenta de reconhecimento de caractere, deve identificar a possibilidade do caractere ser um 'e' ou um 'o'.

O reconhecimento dos caracteres depende das características da fonte que está sendo utilizada. Quando são avaliados caracteres manuscritos, o processo se torna mais complexo devido à diversidade que um caractere pode ser escrito por um mesmo indivíduo. Isso pode ocorrer pela variação da caneta utilizada pela pessoa, a sua habilidade de escrita, o seu estado psicológico, o espaço disponível para a escrita, entre outras. O reconhecimento pode ser organizado em três categorias: estática, sintática ou estrutural e a baseada em redes neurais.

O reconhecimento estático [2] consiste em utilizar uma larga quantidade de amostras que compõem o máximo de possibilidades de ocorrência de um caractere. Dessa forma, quando um *glyph* for submetido ao classificador, esse poderá checar todas as possibilidades e identificar a que mais se aproxima e realizar o reconhecimento. Uma das técnicas mais utilizadas é a de "casamento" de padrões em que são utilizadas comparações para determinar as similaridades entre duas entidades do mesmo tipo, sendo o padrão a ser reconhecido comparado com os armazenados.

A abordagem estrutural [2] consiste em capturar regras que descrevam o relacionamento de um padrão, identificando primitivas e operadores que compõem o caractere. Os atributos representam a topologia e o desenho dos caracteres. Uma abordagem utilizada para analisar os atributos de caracteres é o processo de esqueletização que consiste no processo de afinamento das letras, procurando obter uma réplica da imagem, conservando a estrutura original da imagem.

A última categoria é a baseada em redes neurais que utiliza a habilidade de generalização através do treinamento [2].

Acordo com [2], o processo de reconhecimento de caracteres pode ser organizado em três fases:

1. A extração de características;
2. Classificação propriamente dita
3. Re-segmentação

A extração de características é o método utilizado para capturar características inerentes ao estilo da fonte que está sendo utilizada. É definida como a fase de treinamento, em que o sistema é submetido ao padrão dos caracteres e assim, se tornar apto a reconhecer determinada fonte ou até mesmo uma caligrafia. No alfabeto latino, existem 128 diferentes caracteres que necessitam ser identificado. É catalogado o conjunto de possibilidades que cada caractere pode assumir em um vetor.

A próxima fase é a responsável pelo reconhecimento propriamente. Neste caso, o sistema recebe o *glyph* e identifica as características desse segmento, e compará-as com as que foram treinadas pelo classificador e define um intervalo de probabilidade para o qual o *ghyph* possa ser reconhecido. No melhor cenário, é identificado corretamente o caractere. Caso haja dúvidas qual o caractere representado, a saída será o conjunto dos possíveis caracteres e a sua probabilidade [1][2]. Nesse caso são utilizadas técnicas que avaliam o contexto para identificar qual a opção mais plausível do caractere. Um exemplo seria a utilização de uma árvore de decisão [1][2] que avalia as possibilidades de composição da palavra e de acordo com um dicionário preestabelecido, verifica se a palavra formada é valida. Na pior hipótese, o *glyph* não é identificado, então o sistema junta os segmentos vizinhos, e re-segmenta essa área da imagem [2] para identificar novos segmentos, que serão submetidos ao classificador para a tentativa de identificação do caractere.

Capítulo 4

Segmentação de Documentos

Segmentação de imagem é o processo de análise a fim de identificar regiões semelhantes e separar os objetos que as compõem, levando em consideração atributos similares como textura, iluminação, bordas entre outros. O Capítulo anterior deu uma visão geral do que é o processo de segmentação em documentos históricos. A segmentação de texto é responsável pela localização dos caracteres e do realce das entidades textuais, ou seja, consiste em dar um tratamento lógico ao conteúdo do documento, decompondo a imagem em blocos que representam os parágrafos, tabelas, dentre outros. Segundo [25], existem três estratégias puras no processo de segmentação, e a mistura dessas gerou varias abordagens híbridas. São elas:

1. A clássica, em que os segmentos são identificados baseados pelas suas características; como resultado têm-se os *glyphs*;
2. A segmentação baseada no reconhecimento, que identifica os segmentos de acordo com o resultado de uma pesquisa numa base de dados já existente;
3. O método holístico que tenta reconhecer uma palavra como um todo, para não ter a necessidade de segmentá-la em caracteres.

A estratégia utilizada neste trabalho é a clássica em que o texto é decomposto em sub-imagens, usando características gerais, independente do conteúdo de cada sub-imagem, e deixando o processo de classificação do conteúdo da imagem para ferramentas de reconhecimento de caracteres. O desafio, neste caso, é conseguir chegar a uma unidade que represente um *glyph*, e que o nível de acerto da ferramenta de OCR seja maximizada. O passo mais simples para segmentar os caracteres é pela utilização dos espaços em brancos entre os caracteres. Vários algoritmos de segmentação foram desenvolvidos para essa finalidade [2][5][21], dentre os quais podem ser destacados o de análise de projeção, *break cost*.

4.1 Análise de Projeção

A análise de projeção [21][25] consiste em projetar a quantidade de pixels preto em uma linha ou numa coluna da imagem num histograma, dependendo da orientação da projeção. A Figura 25 mostra um exemplo de um histograma resultante da projeção horizontal e vertical. Ao examinar o histograma da projeção horizontal, percebe-se a existência de pontos que não existem pixels pretos. Esses pontos são interpretados como o espaço entre as linhas do texto. Analogamente, a projeção vertical pode identificar a quebra entre palavras das frases e dependendo de como

estiverem distribuídas às letras que formam cada palavra, identificar os caracteres. Este algoritmo é bastante útil quando utilizamos documento cujo conteúdo foi impresso por uma máquina. Quando tratamos de documentos manuscritos, no entanto, esse processo gera resultados menos precisos.

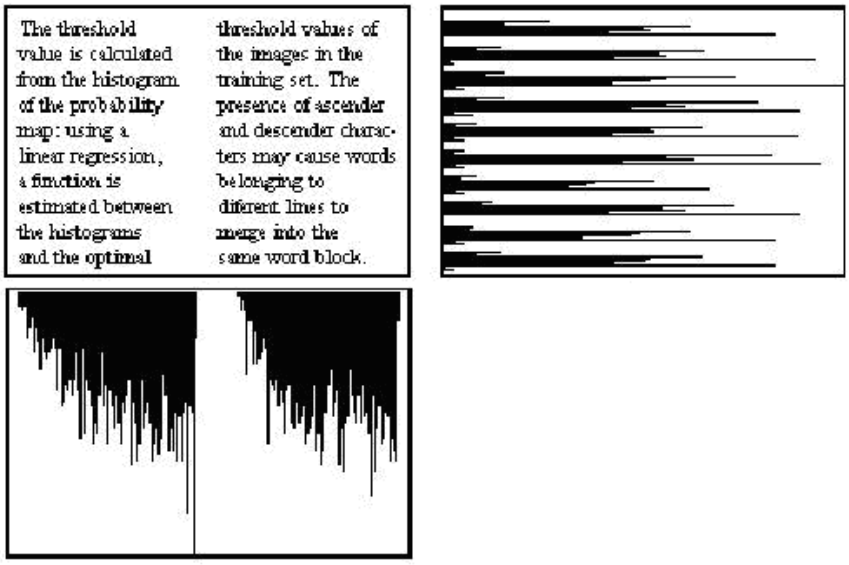


Figura 25. Exemplo de *Projection Profile*. A figura central é uma amostra de uma imagem de texto. A direita encontra-se a projeção horizontal e em baixo a projeção vertical.

4.2 Ponto de Quebra

Break Cost, ou ponto de quebra [5] é uma técnica semelhante à de análise de projeção vertical, que leva em consideração duas colunas adjacentes. Então, para definir o ponto de quebra do segmento, é avaliada a vizinhança da coluna tratada. O valor da quebra é definido como o somatório dos pixels pretos de uma coluna que possuam pixels pretos na mesma linha nas colunas anteriores e posteriores. Colunas cujo custo é pequeno são fortes candidatos a serem o ponto de quebra dos segmentos. De fato, a projeção vertical não define nenhum parâmetro que identifique caracteres conectados. Utilizar a técnica de *break cost* auxiliar na identificação dessa quebra, mesmo assim, ruídos não descartados no pré-processamento podem afetar o resultado.

4.3 Segmentação usando o Algoritmo de *Watershed*

O algoritmo de *watershed* é uma poderosa estratégia para o processo de segmentação. Os conceitos de topografia e de hidrografia são bastante úteis para o desenvolvimento dessa metodologia. Ela é largamente utilizada em diferentes domínios como na biomedicina e visão computacional como uma ferramenta eficiente para segmentação [27]. O algoritmo segmenta a imagem baseado em regiões, sendo sua metodologia baseada na geografia, em que a imagem é tratada como uma superfície topográfica, representando uma elevação do modelo de terreno. Ele simula o escoar da água para as áreas mais baixar, provocando o enchimento progressivo do terreno e demarca as linhas de divisão de água, ou seja, os máximos locais; os segmentos que dividem duas regiões [26].

A representação da imagem passa a ser tri-dimensional, considerando além das coordenadas espaciais x no plano horizontal e y no plano vertical, a terceira coordenada é o nível da intensidade luminosa, ou seja, os tons de cinza de cada pixel, provendo uma idéia de profundidade na imagem. Nesse caso, os pixels de alta amplitude são considerados os cumes e os pixels de baixa amplitude, são considerados os pontos de vale [25][27]. Então, se uma gota de água cair em um ponto da superfície, ela deve escoar para uma altitude mais baixa até alcançar o mínimo local. A acumulação da água na vizinhança de um mínimo local é definida como *basin*, ou bacias de *catchment*. Todos os pontos que drenam em uma bacia comum fazem parte de uma região denominada *watershed* [27]. Conforme a superfície vai sendo preenchida, os pontos correspondentes a uma bacia recebem um rótulo único para diferenciar as diversas bacias e os pontos correspondentes ao encontro de duas bacias recebem um nome especial *dams* [26] que são as linhas de partição, ver Figura 26.

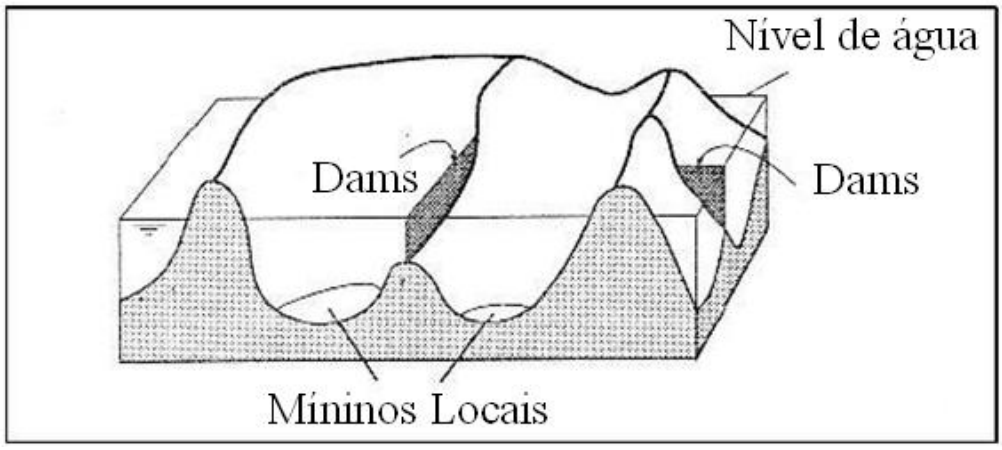


Figura 26. Representação da imagem como uma superfície.

De acordo com a representação "topográfica" da imagem, são considerados três elementos básicos: os pontos da imagem que pertencem a um mínimo local; pontos intermediários, ou de escoamento que pertencem a uma mesma bacia, e pontos de máximo local que são as linhas divisórias que limitam e representam as bordas dos objetos na imagem.

O algoritmo funciona da seguinte maneira: primeiro, definem-se os pontos de mínimo locais, então são abertas tubulações para que a água possa inundar essas regiões. Enquanto a água não passe para outra região, ou seja, encontre um ponto de máxima local, ela continua inundando. Suponha que a imagem (superfície) tenha sido imersa até certo ponto h, então todos os pontos que possuem nível de cinza inferior a h foram rotulados. Como os pontos estão rotulados de acordo com o seu vizinho, têm-se acesso a todos os elementos que possuem níveis de cinza h+1. O processo segue até encontrar um ponto de máximo local em que será construída uma barragem. O algoritmo pára quando for encontrado um ponto em que todos os pontos de máximos locais terão se transformado em barragem. Existem várias abordagens que tratam à transformada de *watershed*. Para mais detalhes veja [27][28].

As regiões produzidas pela aplicação da segmentação a um documento, utilizando a transformada de *watershed* correspondem aos traços contínuos da escrita de uma imagem, como apresentado na Figura 27.

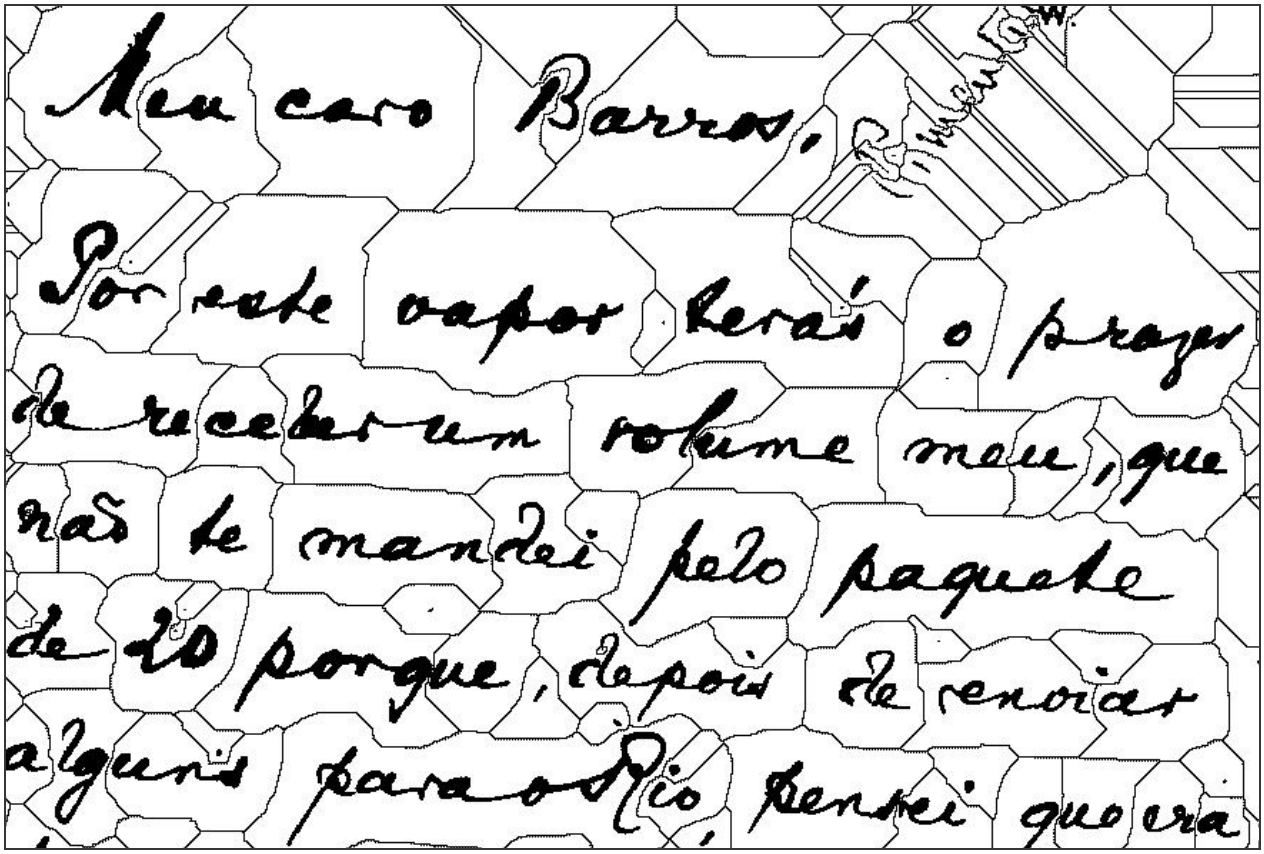


Figura 27. Documento segmentado pelo algoritmo de *watershed*.

4.4 Falhas no processo de segmentação

Idealmente, o processo de segmentação tem como saída um segmento que representa um caractere. Como foi visto, os algoritmos de segmentação possuem uma limitação na identificação de uma falha no processo de identificação de pontos de corte que sejam condizentes com os caracteres. As falhas [2] mais comuns no processo de segmentação são as seguintes:

1. Identificação se um seguimento corresponde a um caractere ou a múltiplos caracteres e/ou representa o fragmento de um caractere. Isso é abordado pela Figura 28 na qual a palavra foi segmentada utilizando a análise de projeção vertical. A Figura 28 identifica a ocorrência de caracteres conectados e segmentos que correspondem a fragmentos de caracteres.

Figura 28. Exemplo de imagem com fragmentos de caracteres, como o primeiro e segundo segmentos e caracteres conectados, como o terceiro, o quarto e o quinto segmento.

2. Verificação se o segmento corresponde a um caractere ou se ele é parte de outro caractere. Um exemplo deste problema ocorre quando a letra u é segmentada em dois fragmentos,

podendo ser reconhecida com dois i's seguidos, e/ ou ser reconhecido como 'u'. Ver Figura 29.

Figura 29. Exemplo de imagem com fragmentos de caracteres que podem ser identificados erradamente. Neste caso temos a segmentação da letra u em dois *glyphs*.

3. Conflitos entre o que faz parte do conteúdo do texto ou é apenas ruído. Neste caso o problema é definir se um determinado pixel faz parte de um caractere ou se ele compõe algum defeito na imagem, ou seja, se um ponto é o pingo da letra i, ou apenas ruído, se um traço é um acento ou um risco que não faz parte do documento. A Figura 30 mostra um exemplo.

(a) (b)

Figura 30. Nesse caso, o terceiro fragmento da imagem (a) representa um apóstrofo. O sistema deve reconhecê-lo como parte constituinte do documento, enquanto que na imagem (b) o segmento representa apenas ruído, devendo ser ignorado.

4. Não selecionar uma área que faça parte do texto. Esse é a falha mais grave já que estará ocorrendo perda do conteúdo do documento, gerando arquivos inconsistentes.

Uma solução para os três primeiros problemas é a utilização de técnicas de classificação que identificam o caractere pela avaliação do contexto, sendo o processo de reconhecimento e de classificação interativo.

4.5 O método Proposto

Foram implementados duas abordagens para a segmentação de documentos. A primeira foi baseada no processo de segmentação de imagem utilizando a análise de projeção. O acervo utilizado possui tanto documentos manuscritos quanto datilografados. Os resultados foram satisfatórios quando aplicados aos documentos datilografados, enquanto que não foi tão bom em documentos manuscritos.

A segunda abordagem utilizou como base a segmentação morfológica, baseada na transformada de *watershed*. Foi visto que o resultado foi satisfatório em fragmentos de documentos manuscritos, mas se tornou inviável devido ao longo tempo de processamento para o documento completo. A terceira abordagem foi unir a análise de projeção para identificar blocos de textos e linhas e nesta imagem utilizou a transformada de *watershed* para separar os caracteres.

Os métodos implementados foram organizados em uma seqüência de três etapas distintas. O processo de binarização que converte as imagens para preto-e-branco. Seguido pelo pré-

processamento da imagem para compensar possíveis problemas que não foram filtrados pelo primeiro processo. E, por fim, a aplicação dos algoritmos de segmentação implementados. As duas primeiras etapas são comuns às abordagens de segmentação utilizada.

4.5.1 Características do Acervo Utilizado

A base de dados utilizada neste projeto foi obtida a partir da digitalização de documentos históricos datado do século XIX. Por possuir mais de 100 anos, esse acervo sofreu bastante com o processo de envelhecimento e pelo manuseio errado durante vários anos. O papel encontra-se escurecido, regiões desintegradas, sujeiras, manchas, marcas de clipes metálicos, dentre outros, como mostra a Figura 31.

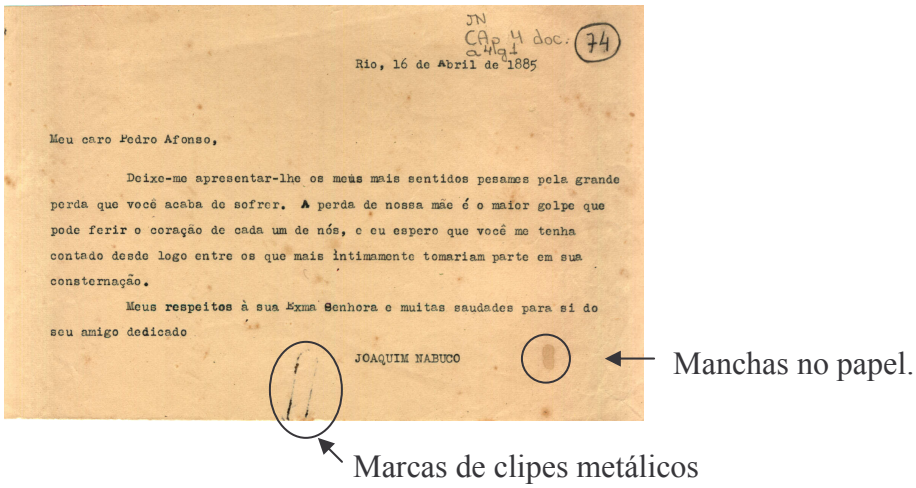
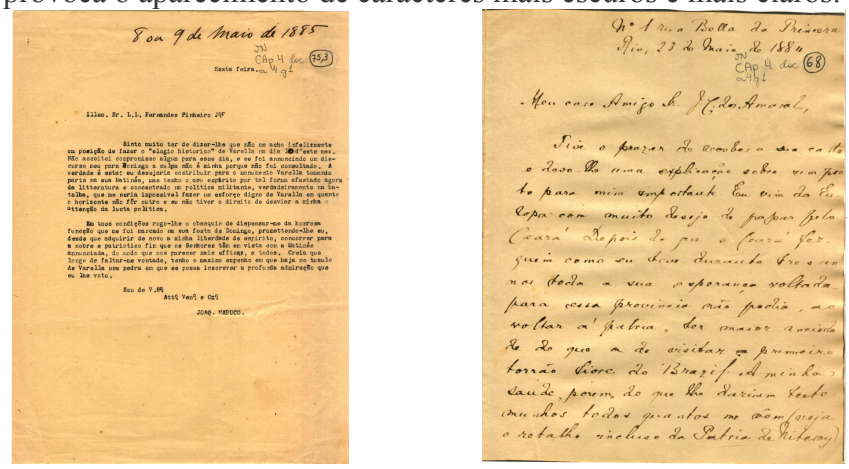


Figura 31. Exemplo de problemas nos documentos.

O acervo é composto por cartas datilografadas e manuscritas, a Figura 32 apresenta algumas amostras. Além dos problemas já citados, documentos manuscritos possuem características adicionais como a pressão com que cada letra é escrita pelo autor, criando letras com intensidades diferentes. Isso provoca o aparecimento de caracteres mais escuros e mais claros.



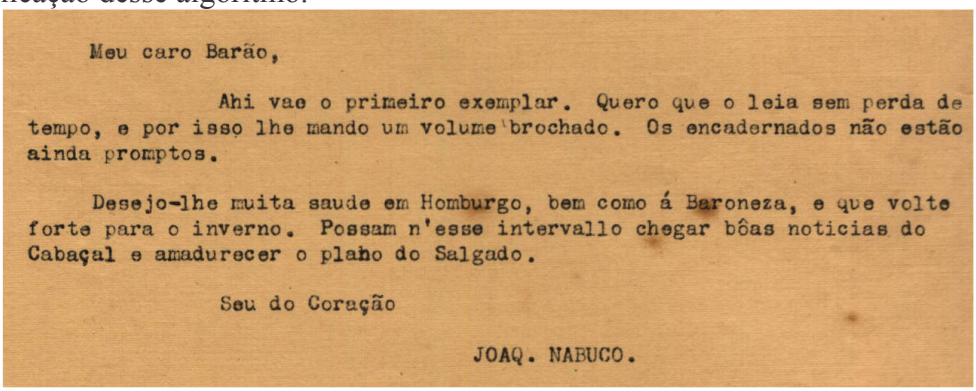
(a) (b)

Figura 32. Exemplo de documento datilografado (a) e de documento manuscrito (b).

4.5.2 Binarização dos documentos

As imagens do acervo encontram-se digitalizadas com 200 dpi no padrão RGB. Documentos históricos possuem basicamente um objeto em cena que são os caracteres. Nesse caso, é mais importante a análise da forma do que a intensidade dos pixels, então, é necessária a utilização de técnicas para reduzir a quantidade de dados a serem tratados. A binarização consiste em converter uma imagem para apenas dois tons, preto-e-branco, em que o preto corresponde aos objetos da cena enquanto que o branco corresponde ao *background*. Basicamente, os pixels que têm um valor inferior que um limiar são convertidos para preto e os que estão acima são convertidos para branco.

Existem vários métodos de binarização, mas não existe um que seja ótimo para todas as aplicações. Como as imagens encontram-se coloridas, foi aplicada a função do Matlab que converte uma imagem colorida para tons de cinza e em seguida o algoritmo utilizado foi o proposto em [30], desenvolvido para ser aplicado especificamente com imagens de documentos históricos, em especial ao acervo utilizado. O algoritmo percorre a imagem procurando pela cor que aparece com mais freqüência e determina que essa deve ser a cor do papel do documento. Define esse valor como o limiar inicial e avalia a entropia do histograma em consideração à resolução da imagem. O algoritmo pode gerar imagens em tons de cinza ou em preto-e-branco. Neste trabalho foi utilizado para gerar imagens do segundo tipo. A Figura 33 apresenta um exemplo da aplicação desse algoritmo.


Meu caro Barão,

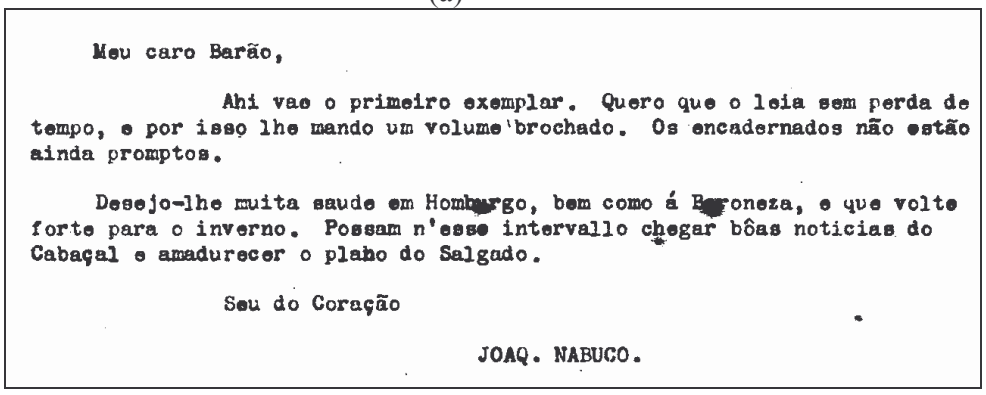
Ahi vae o primeiro exemplar. Quero que o leia sem perda de tempo, e por isso lhe mando um volume brochado. Os encadernados não estão ainda promptos.

Desejo-lhe muita saude em Homburgo, bem como á Baroneza, e que volte forte para o inverno. Possam n'esse intervallo chegar bôas noticias do Cabaçal e amadurecer o plano do Salgado.

Seu do Coração

JOAQ. NABUCO.

(a)


Meu caro Barão,

Ahi vae o primeiro exemplar. Quero que o leia sem perda de tempo, e por isso lhe mando um volume brochado. Os encadernados não estão ainda promptos.

Desejo-lhe muita saude em Homburgo, bem como á Baroneza, e que volte forte para o inverno. Possam n'esse intervallo chegar bôas noticias do Cabaçal e amadurecer o plano do Salgado.

Seu do Coração

JOAQ. NABUCO.

(b)

Figura 33. Fragmento de um documento original em (a) e em (b)o resultado da aplicação do algoritmo proposto por [30].

4.5.3 Pré-Processamento

Essa etapa tem como objetivo a correção da imagem e a preparação da mesma para facilitar o processo de segmentação. Consiste em remover informações desnecessárias, como ruídos e falhas, além de avaliar se o texto encontra-se inclinado e corrigir este problema [22].

4.5.3.1 Eliminação de Ruídos

Um problema proveniente do mau uso dos documentos históricos e da aquisição de imagens são informações inconsistentes no documento e podem afetar os resultados do processamento, então devem ser eliminados. Com essa finalidade, foi implementado o filtro da vizinhança que se baseia na forte correlação de um pixel com os seus vizinhos. Foi definida uma máscara 3x3, e avaliado o pixel central. Então para cada conjunto de pixels que compõem a máscara, se não houver nenhum pixel da mesma cor que o do centro, esse é considerado ruído e alterado para a cor do conjunto [2]. A Figura 34 apresenta a execução desse algoritmo.

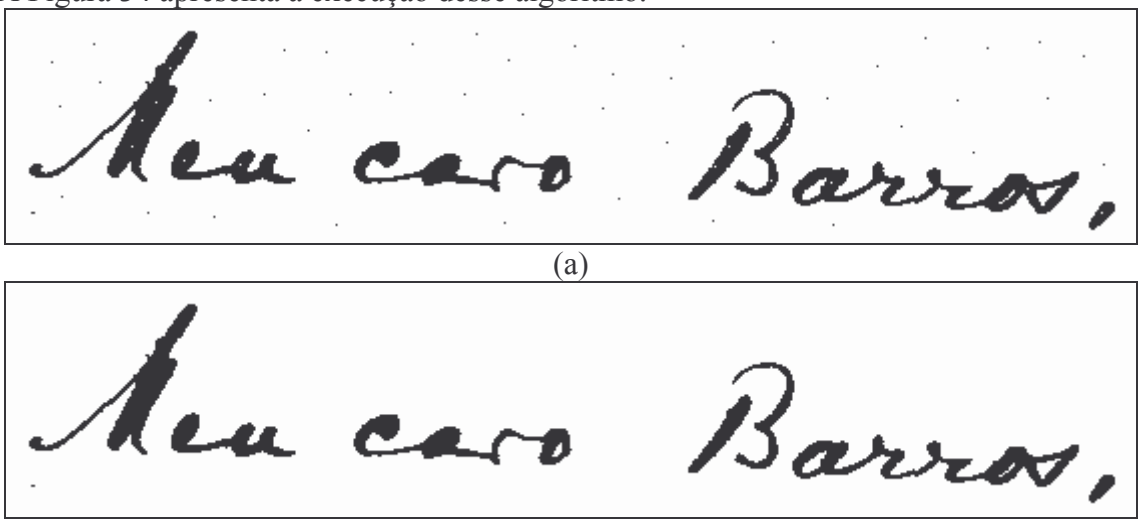


Figura 34. Em (a) apresenta uma imagem cujo ruído foi adicionado através de um algoritmo para geração aleatória de ruídos e em (b) o resultado da aplicação do algoritmo implementado.

4.5.3.2 Correção de Inclinação

Um problema muito comum é a inclinação da imagem. Isso pode ter ocorrido devido ao processo de aquisição. Como o acervo trata de documentos manuscritos e datilografados, esse problema pode, também, ser proveniente da forma de escrita do autor em documentos manuscritos e/ ou no processo de edições nos documentos datilografados na quebra de linha.

Considerando que um documento deve possuir o ângulo de inclinação de zero grau, deve-se determinar a orientação do documento de acordo com os dados que o compõem. Para isso, foi implementada a transformada de Hough para a determinação do ângulo de orientação das linhas do documento [3]. A transformada de Hough mapeia cada ponto de uma linha da imagem original em um único ponto. Os pontos são codificados para se ter à localização das linhas. A imagem de destaque é transformada, então, no espaço de Hough. Nesse ponto, cada valor é acumulado em um ponto que representa os segmentos de linha. No espaço de Hough o cume, mostrado na Figura 35, aponta para o ângulo ao qual o número maior de linhas diretas deve ser ajustado pelos pixels originais, e esse é o ângulo de inclinação.

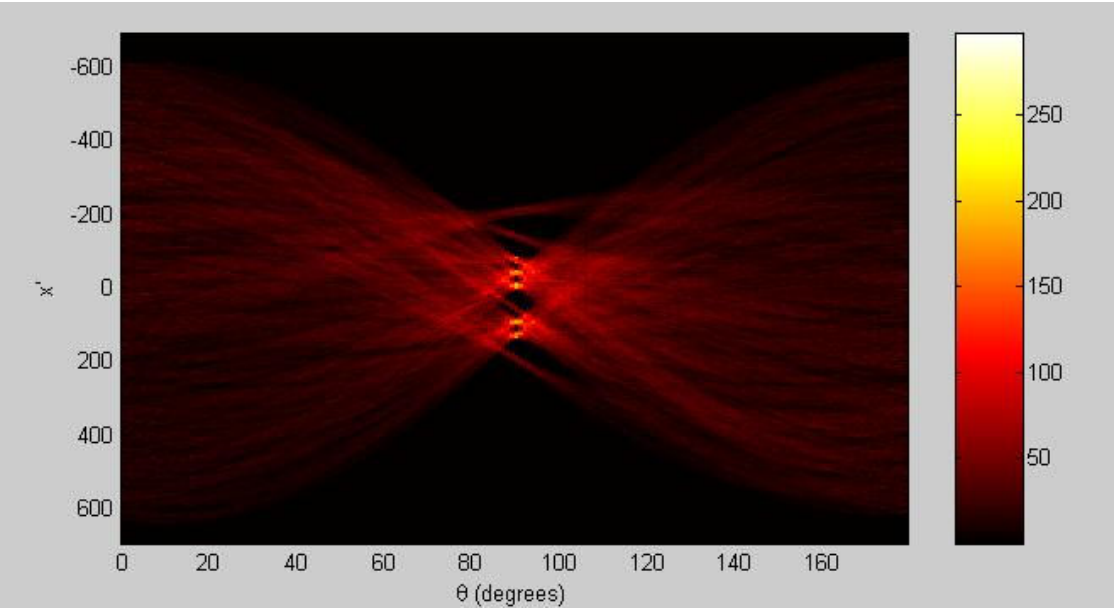
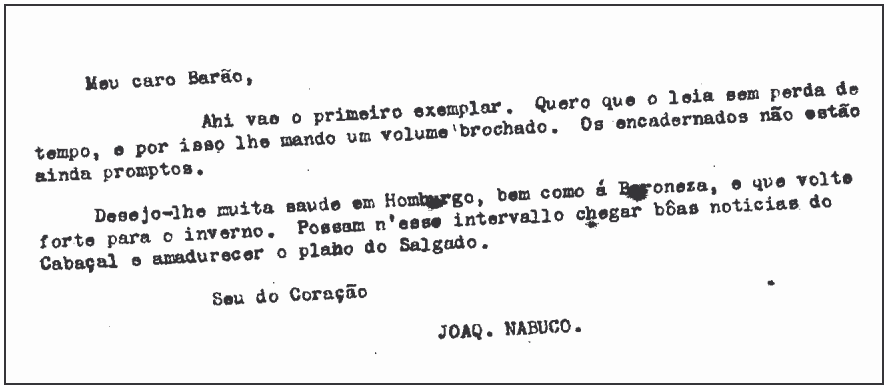


Figura 35. O cume representa o ângulo de inclinação do documento.

De posse do ângulo, a imagem é rotacionada utilizando a função *imrotate* do Matlab. A Figura 36 apresenta o exemplo da aplicação do algoritmo implementado. Alguns algoritmos estão sendo desenvolvidos para tratar além da inclinação do texto, a inclinação dos caracteres quando estes são escritos à mão livre. Para mais detalhes veja [32].



Meu caro Barão,

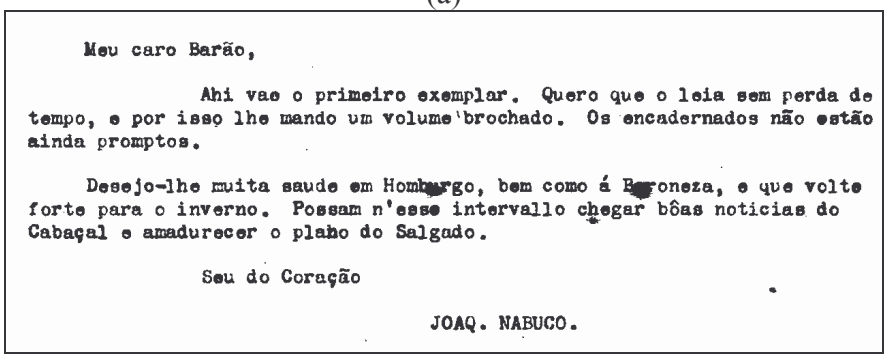
Ahi vae o primeiro exemplar. Quero que o leia sem perda de tempo, e por isso lhe mando um volume brochado. Os encadernados não estão ainda promptos.

Desejo-lhe muita saude em Homburgo, bem como á Baroneza, e que volte forte para o inverno. Possam n'esse intervallo chegar bôas noticias do Cabaçal e amadurecer o plano do Salgado.

Seu do Coração

JOAQ. NABUCO.

(a)



Meu caro Barão,

Ahi vae o primeiro exemplar. Quero que o leia sem perda de tempo, e por isso lhe mando um volume brochado. Os encadernados não estão ainda promptos.

Desejo-lhe muita saude em Homburgo, bem como á Baroneza, e que volte forte para o inverno. Possam n'esse intervallo chegar bôas noticias do Cabaçal e amadurecer o plano do Salgado.

Seu do Coração

JOAQ. NABUCO.

(b)

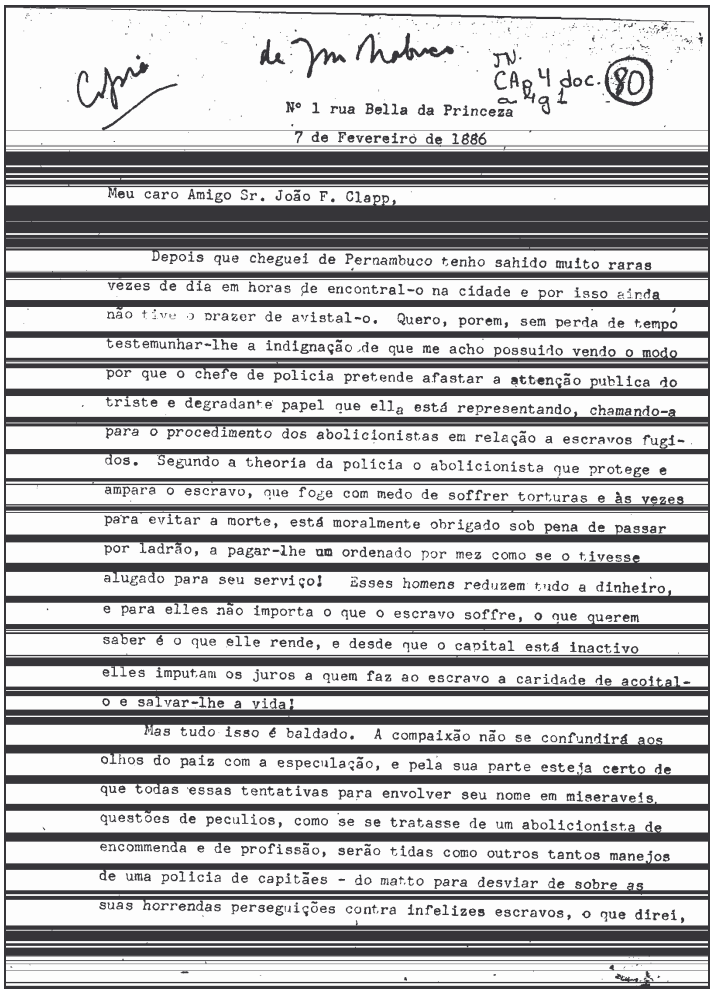
Figura 36. Em (a) apresenta a imagem de um documento rotacionada, e em (b) o resultado da aplicação da função *imrotate* do Matlab com o ângulo determinado pelo algoritmo que calcula a inclinação da imagem.

4.5.4 Abordagens Propostas

Este trabalho propõe a identificação de um algoritmo que melhor segmente documentos históricos. Foram utilizadas três estratégias, uma utilizando a análise de projeção, bastante utilizada pela academia e as outras duas forma utilizam a segmentação utilizando *watershed*, uma sobre a imagem binarizada e a segunda sobre a imagem além de binarizada dilatada. Estas duas ultimas estratégias são propostas deste trabalho.

4.5.4.1 Segmentação utilizando Análise de Projeção

A primeira estratégia utilizada foi à implementação da análise de projeção dos documentos. O algoritmo implementado tem como parâmetros o nome da imagem, a extensão da imagem, e a definição de dois índices, uma para a projeção vertical e um para a projeção horizontal. O algoritmo verifica se a imagem encontra-se em tons de preto-e-branco. Caso não esteja, são executados os passos descritos nas Seções 4.5.1, 4.5.2 e 4.5.3 . Então, a imagem é percorrida, avaliando a quantidade de pixels preto em cada linha. As linhas que possuírem uma quantidade de pixel preto inferior ao limiar definido pelo índice horizontal é a linha é convertido para preto, caso contrário não é realizada nenhuma operação. O histograma de projeção representa uma estrutura que armazena a quantidade de pixels pretos que compõem a imagem. O resultado da execução dessa etapa do algoritmo é uma imagem segmentada em linhas, como mostrada pelas Figura 37 e Figura 38.



Nº 1 rua Bella da Princeza

7 de Fevereiro de 1886

Meu caro Amigo Sr. João F. Clapp,

Depois que cheguei de Pernambuco tenho sahido muito raras vezes de dia em horas de encontral-o na cidade e por isso ainda não tive o prazer de avistal-o. Quero, porem, sem perda de tempo testemunhar-lhe a indignação de que me acho possuido vendo o modo por que o chefe de policia pretende afastar a attenção publica do triste e degradante papel que ella está representando, chamando-a para o procedimento dos abolicionistas em relação a escravos fugidos. Segundo a theoria da policia o abolicionista que protege e ampara o escravo, que foge com medo de soffrer torturas e às vezes para evitar a morte, está moralmente obrigado sob pena de passar por ladrão, a pagar-lhe um ordenado por mez como se o tivesse alugado para seu serviço! Esses homens reduzem tudo a dinheiro, e para elles não importa o que o escravo soffre, o que querem saber é o que elle rende, e desde que o capital está inactivo elles imputam os juros a quem faz ao escravo a caridade de acoital-o e salvar-lhe a vida!

Mas tudo isso é baldado. A compaixão não se confundirá aos olhos do paiz com a especulação, e pela sua parte esteja certo de que todas essas tentativas para envolver seu nome em miseraveis questões de peculios, como se se tratasse de um abolicionista de encommenda e de profissão, serão tidas como outros tantos manejos de uma policia de capitães - do matto para desviar de sobre as suas horrendas perseguições contra infelizes escravos, o que direi,

Figura 37. Resultado da aplicação da projeção horizontal em documento datilografado.

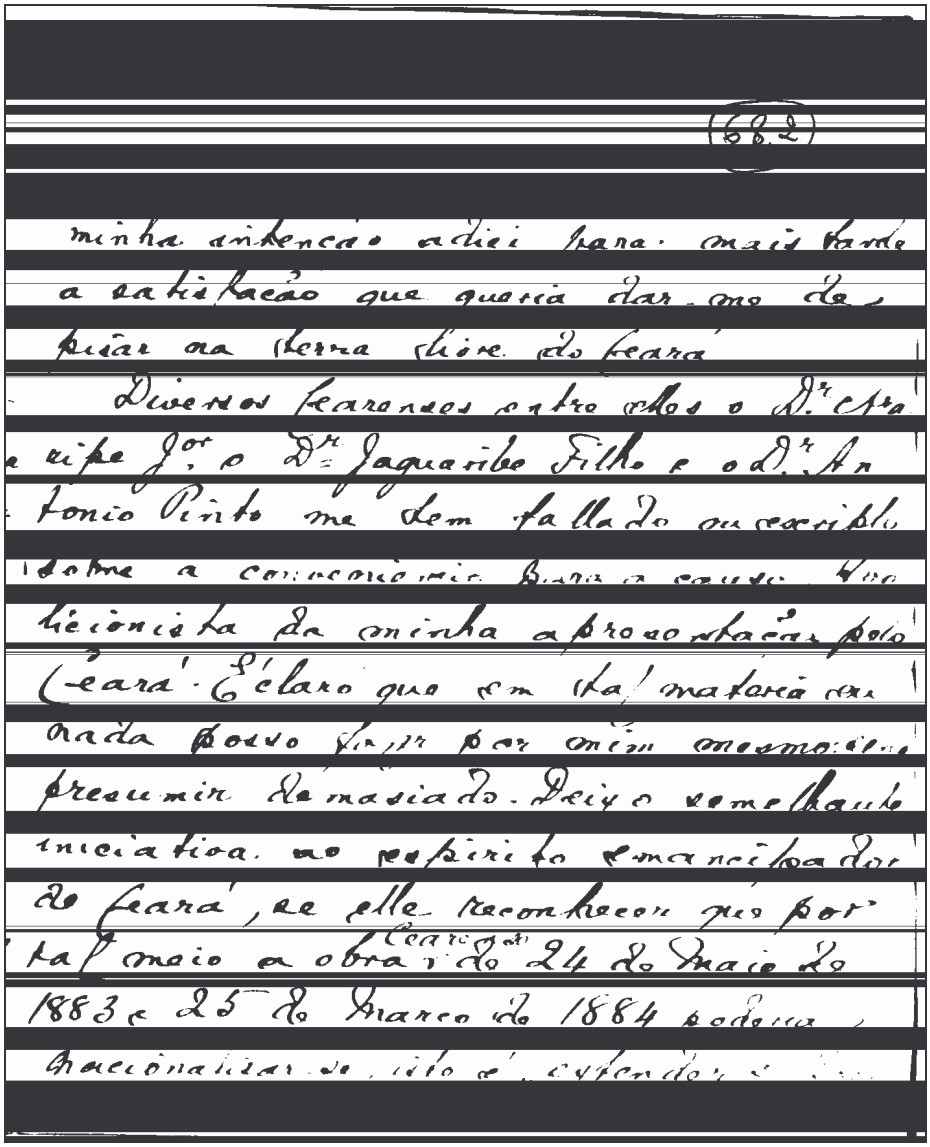


Figura 38. Resultado da aplicação da projeção horizontal em documento manuscrito.

A próxima etapa é a segmentação de cada linha identificada pela etapa anterior. O algoritmo é aplicado a cada intervalo identificado, de forma análoga, avalia a quantidade de pixels pretos em cada coluna. Caso a quantidade seja inferior à quantidade definida pelo índice vertical, a coluna é convertida para preto. O resultado deste processo é apresentado nas 0 e 40.

Os valores do índice de corte da projeção horizontal e vertical foram obtidos de forma experimental. O algoritmo foi executado com valores entre 0 e 50 escolhidos aleatoriamente. O valor ideal foi identificado com uma avaliação visual do resultado.

Essa abordagem foi satisfatória na identificação de caracteres em documentos datilografados como pode ser vista na 0, pois a máquina de datilografar insere espaçamento entre os caracteres. Por outro lado, em documentos manuscritos não houve o mesmo nível de satisfação, como apresentado na 0.

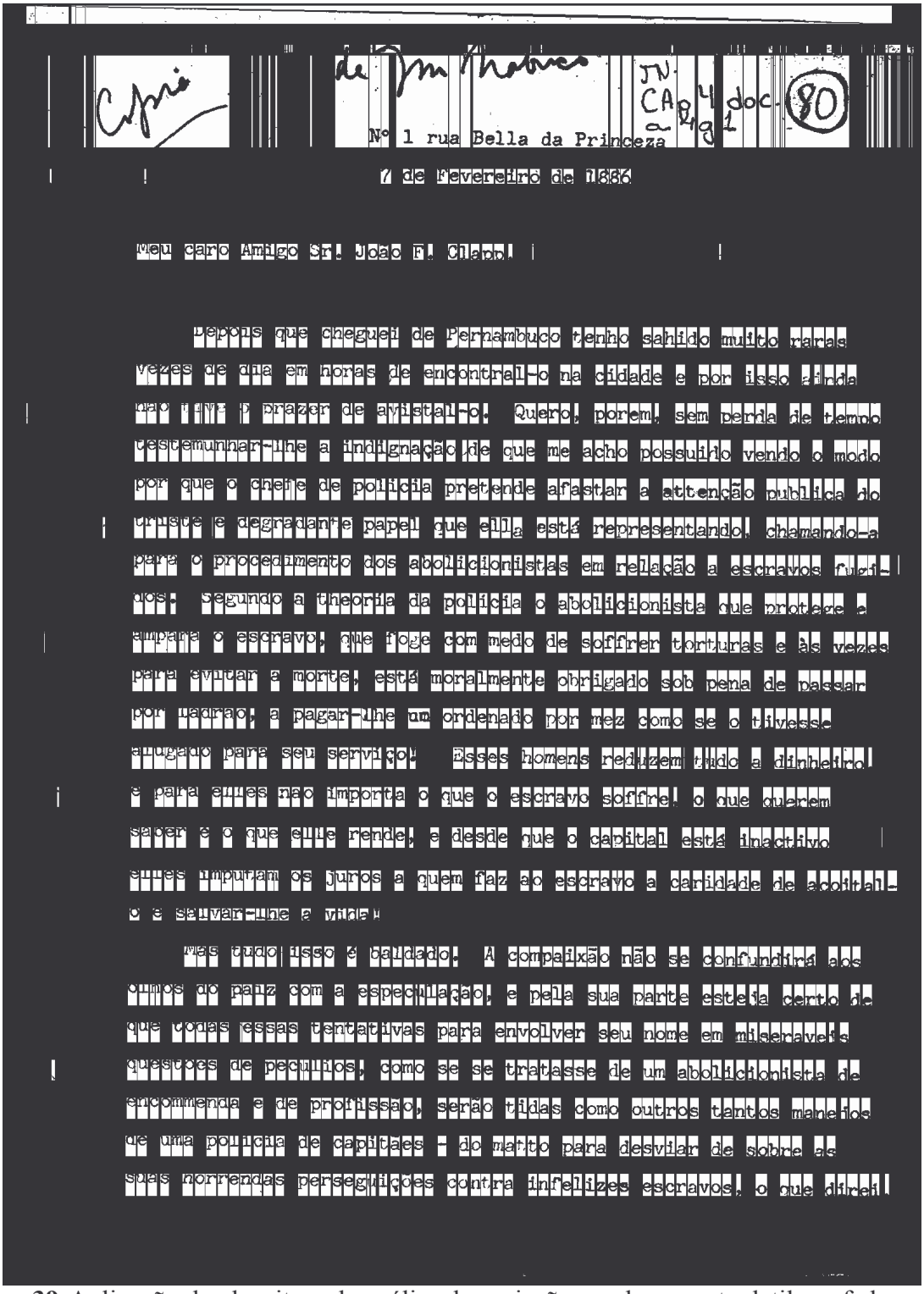


Figura 39. Aplicação do algoritmo de análise de projeção em documento datilografado.

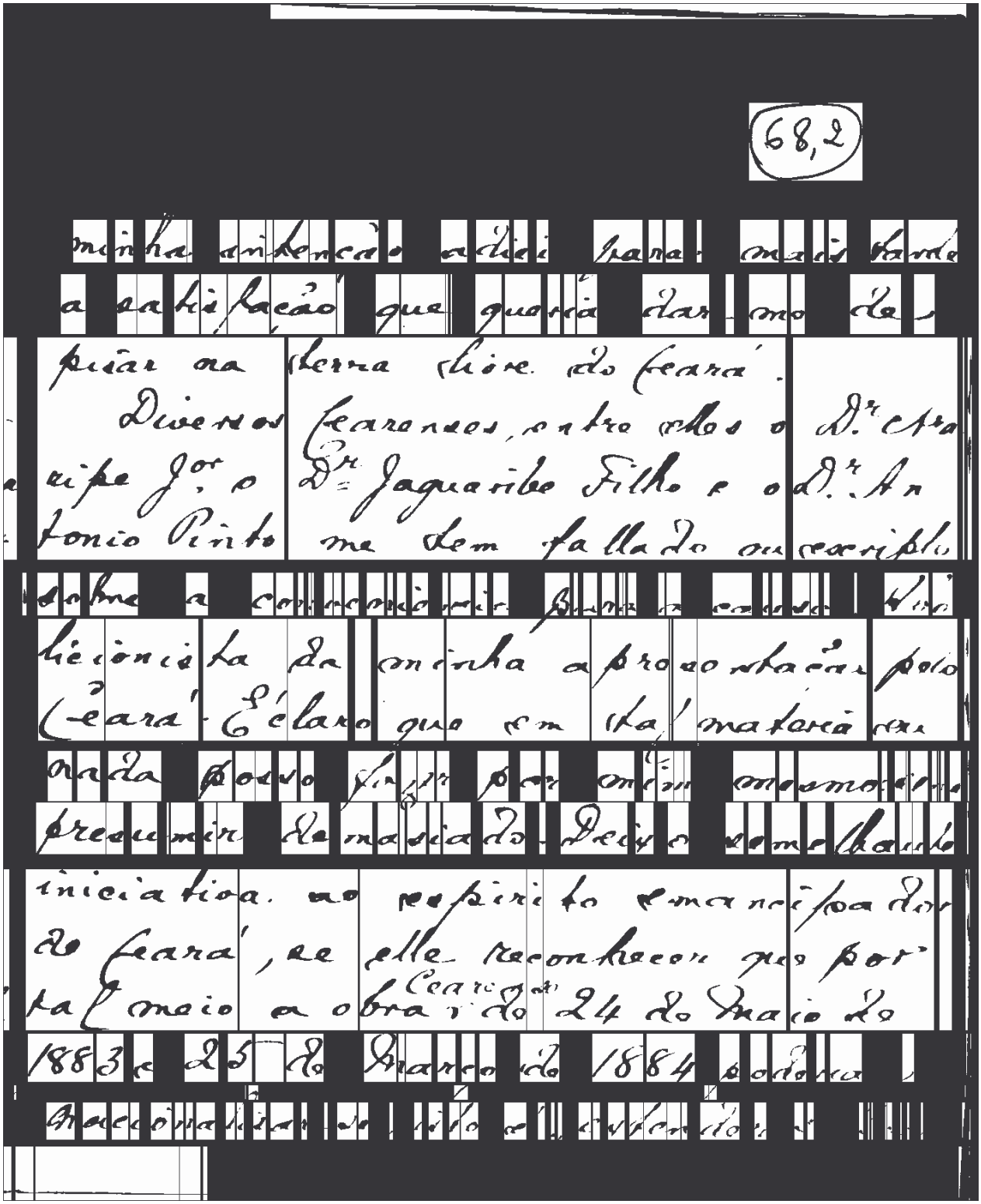
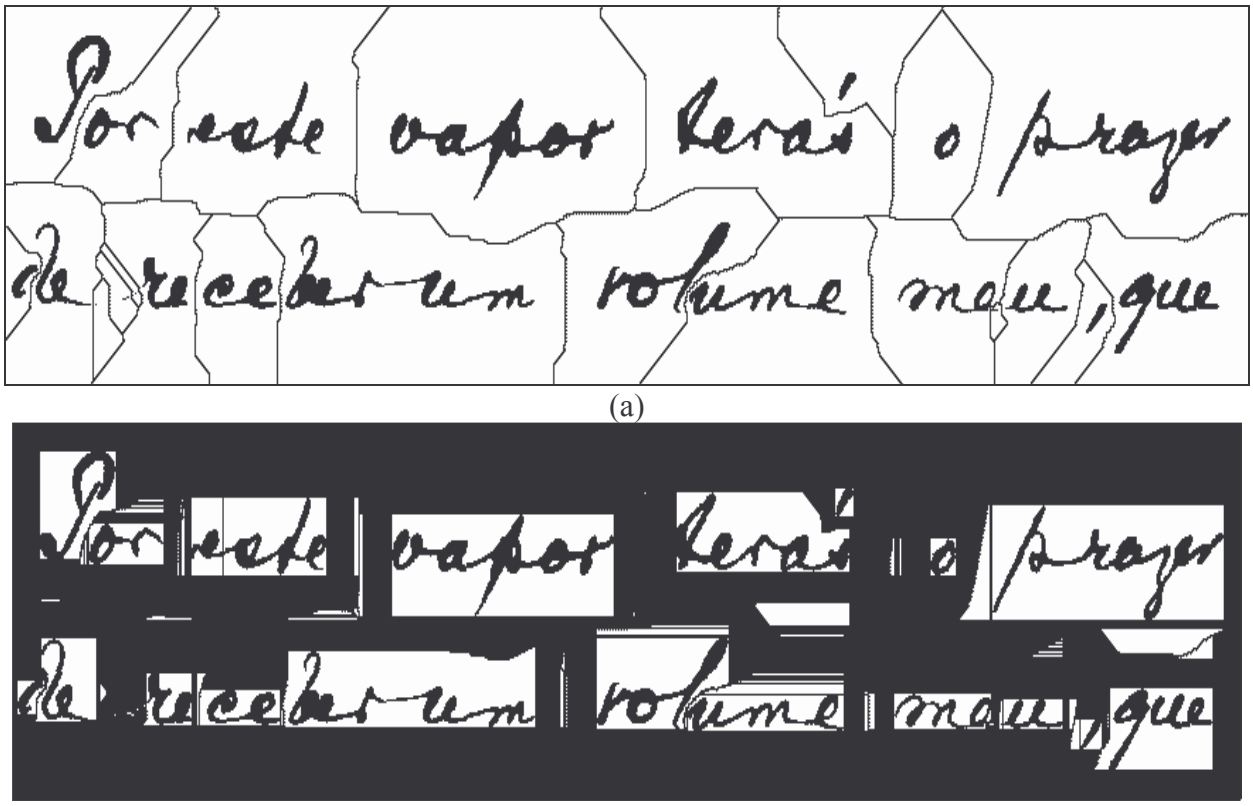


Figura 40. Aplicação do algoritmo de análise de projeção em documento manuscrito.

4.5.4.2 Segmentação utilizando *Watershed*

Para melhorar os resultados em documentos datilografados foi utilizada a segmentação de *watershed*. Nessa abordagem, o documento é submetido à função *watershed*. Esse algoritmo cria uma máscara que rotula as linhas divisórias das regiões como '1' e, para cada segmento, é rotulado com um valor que varia até a quantidade de segmentos criados. Essa máscara é

sobreposta na imagem original e dá origem à Figura 41(a). O próximo passo é avaliar cada segmento para identificar os dados que não são pertinentes ao caractere. Foi implementado da seguinte forma: para cada linha do segmento, não possuir pixels pretos a linha é convertida para preto. O resultado deste algoritmo é apresentado na Figura 41(b). A seqüência da execução do algoritmo é apresentada na Figura 42.



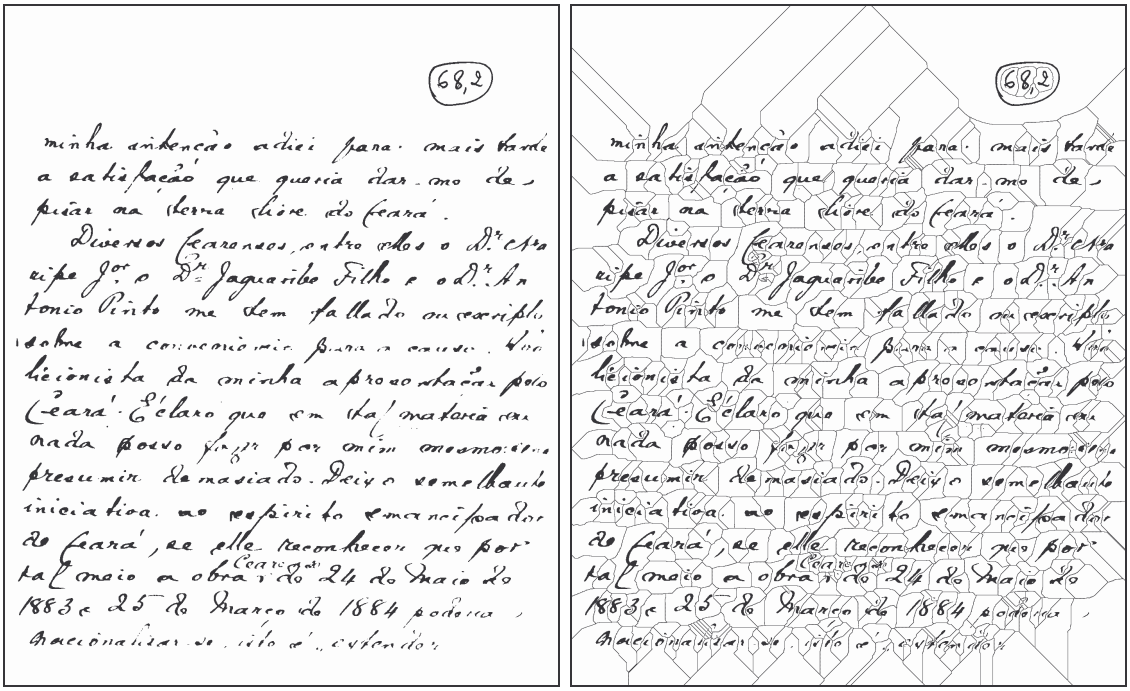
(a)

(b)

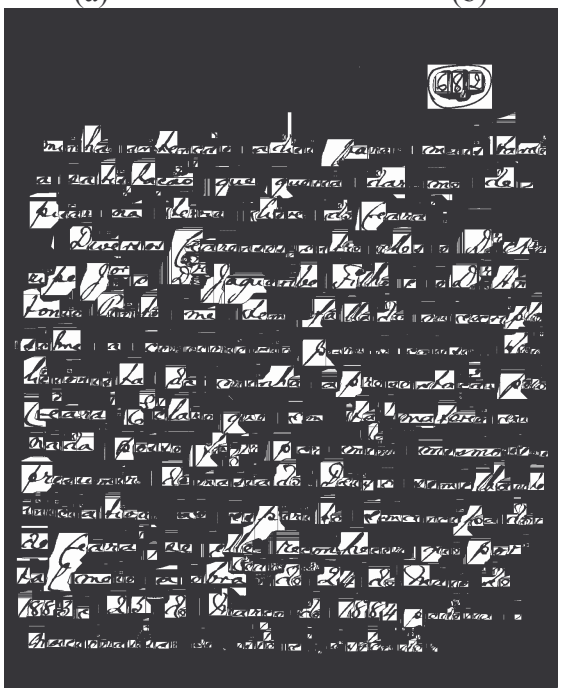
Figura 41. Em (a) apresenta o resultado da segmentação, aplicando apenas a transforma de *watershed* em um fragmento de documento manuscrito, e em (b) o algoritmo implementado.

A segunda estratégia teve um bom resultado para fragmentos de documentos manuscritos. No entanto, pela quantidade de segmentos criados pelo algoritmo de *watershed*, o tempo de processamento de um documento inteiro é longo. Para solucionar esse problema foi implementada uma estratégia que uniu as citadas acima.

O documento foi primeiramente segmentado utilizando a projeção horizontal que quebra o documento em linhas. A imagem resultante é então submetida ao algoritmo implementado utilizando a transformada de *watershed*. Isso foi feito com a finalidade de identificar uma menor quantidade de segmentos. No entanto, essa abordagem não foi tão satisfatória. Além de o tempo de processamento não ter diminuído, a imagem resultante não apresentou uma segmentação tão boa quanto à segunda estratégia. A Figura 43 apresenta o resultado. Essa estratégia foi descartada.



(a) (b)



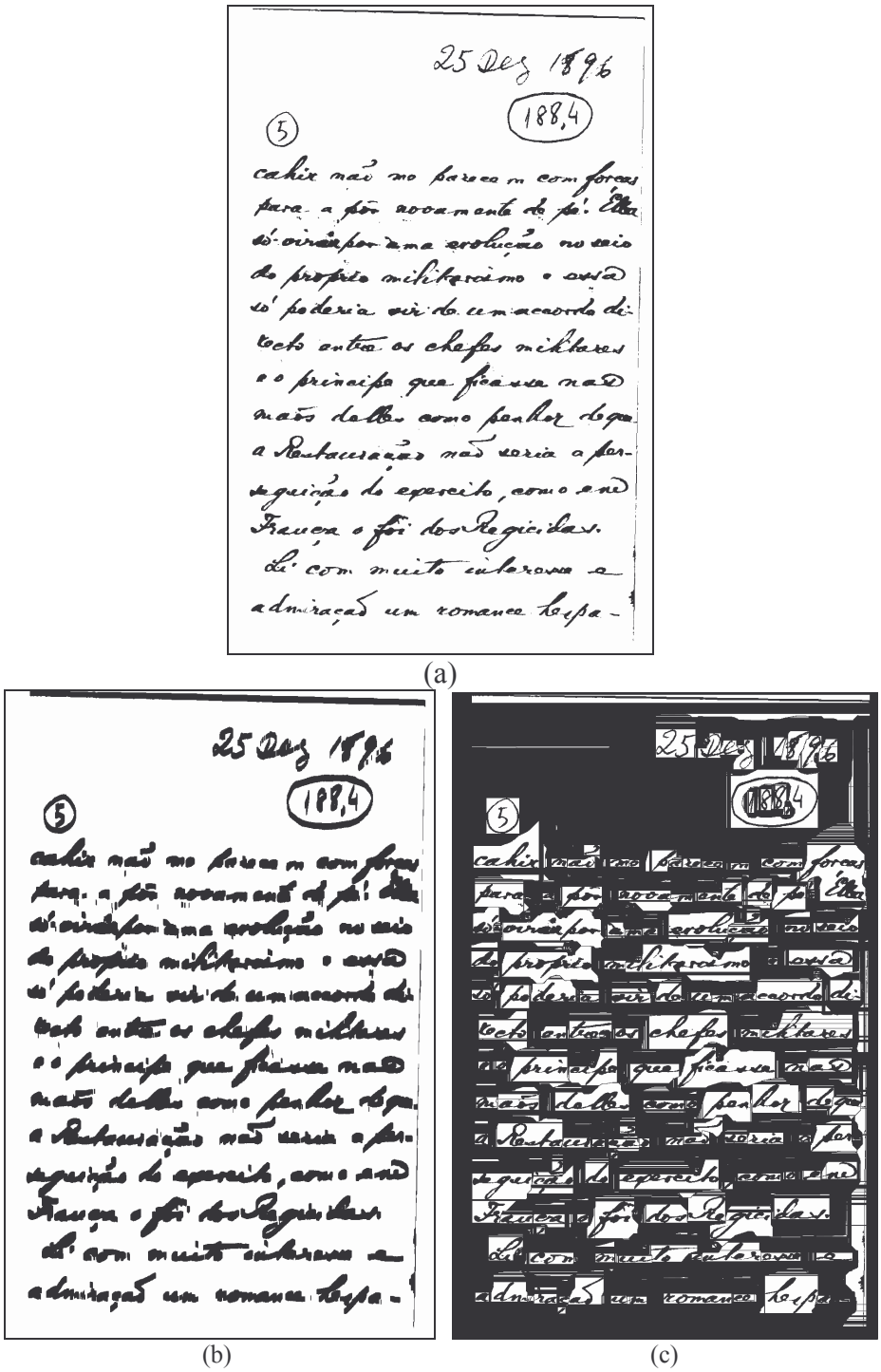
(c)

Figura 42. Em (a) apresenta a imagem original binarizada, em (b) aplicando apenas a transforma de *watershed*, e em (c) o resultado do algoritmo implementado.

Figura 43. Apresenta o resultado da execução algoritmo implementado pela terceira estratégia, sem dilatação, na imagem apresentada em Figura 42(a).

4.5.4.3 Segmentação utilizando *Watershed* em imagens dilatadas

A terceira estratégia foi submeter a imagem a um algoritmo morfológico que altere o contorno da imagem. O escolhedo foi o de dilatação que ao ser aplçicado juntaos caracteres não conectados e assim, gera uma imagem com uma menor quantidade de segmentos, Figura 54. O tempo de execução reduzia a aproximadamente a dois terços do algoritmo anterior, mas continua longo em relação ao da análise de projeção.



(a) (b) (c)

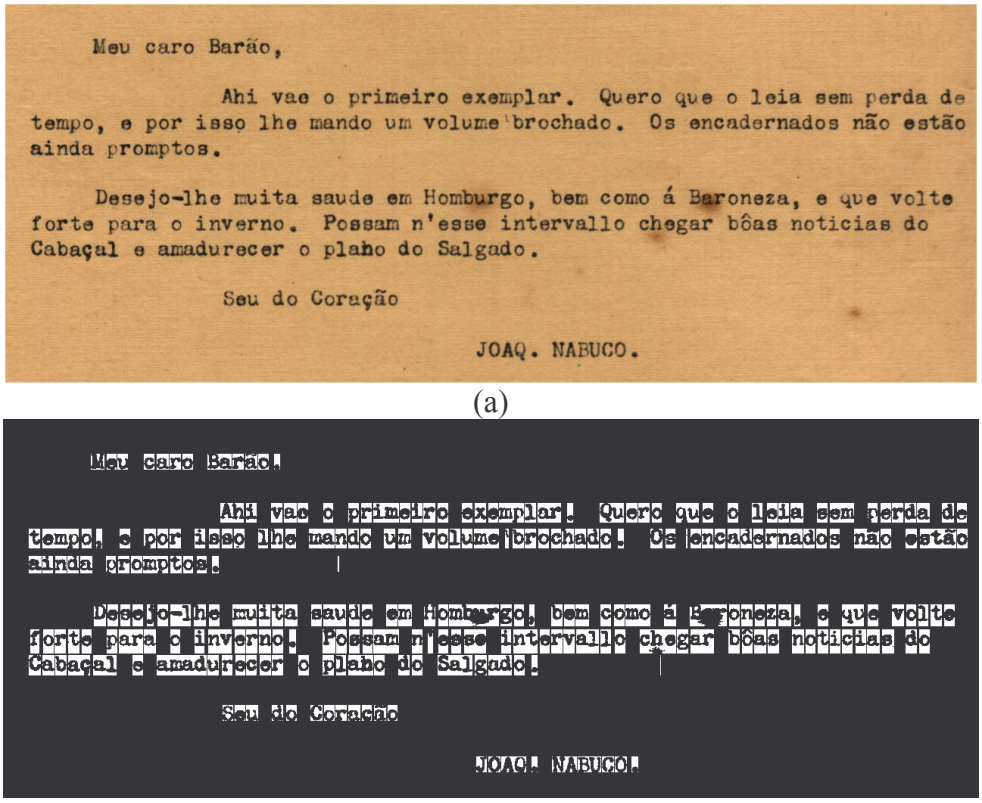
Figura 44. Apresenta em (a) uma imagem binarizada de um documento manuscrito. Em (b) a imagem dilatada e em (c) o resultado da execução algoritmo implementado utilizando a transformada de *watershed* na imagem dilatada.

Capítulo 5

Experimentos

O Capítulo 4 apresentou o proejto que foi implementado e alguns resultados quando aplicados ao acervo de teste. Este Capítulo apresenta resultados mais detalhados do processo de segmentação de documentos históricos implementado. Os testes foram realizados em um acervo que possui tanto documentos manuscritos quanto datilografados. Esses documentos foram produzidos por um mesmo autor, então não houve muita variação em relação ao tipo de fonte. Como foram utilizados documentos manuscritos e datilografados, não foi identificada uma técnica que fosse ótima para ambos os tipos. Foram definidos alguns parâmetros para avaliar o nível de acerto dos algoritmos implementados. Foram adotados seis aspectos a serem avaliados. Os algoritmos descritos no Capítulo 4 foram implementados e aplicados ao acervo utilizado. A Figura 45(a) mostra um exemplo de um documento datilografado e a Figura 46(a) mostra um exemplo de um documento manuscrito. Ambas as figuras apresentam também o resultado da segmentação dos documentos Figura 45(b) e Figura 46(b). As implementações e os experimentos foram realizados utilizando o Matlab.

Os algoritmos foram aplicados depois de rodar as mesmas etapas de pré-processamento para fazer uma avaliação justa das implementações. Como o acervo encontra-se como imagem colorida no padrão RGB, foi aplicada uma função do Matlab que converte a imagem para tons de cinza. À imagem resultante foi aplicado o algoritmo de binarização proposto em [30]. Um filtro para redução de ruído sal-e-pimenta é aplicado em seguida, para minimizar informações irrelevantes. Por fim, foi aplicado algoritmo para correção de inclinação da imagem. A seqüência de execução das etapas de pré-processamentos está mostrada na Figura 47.



Meu caro Barão,

Ahi vae o primeiro exemplar. Quero que o leia sem perda de tempo, e por isso lhe mando um volume brochado. Os encadernados não estão ainda promptos.

Desejo-lhe muita saude em Homburgo, bem como á Baroneza, e que volte forte para o inverno. Possam n'esse intervallo chegar bôas noticias do Cabaçal e amadurecer o plano do Salgado.

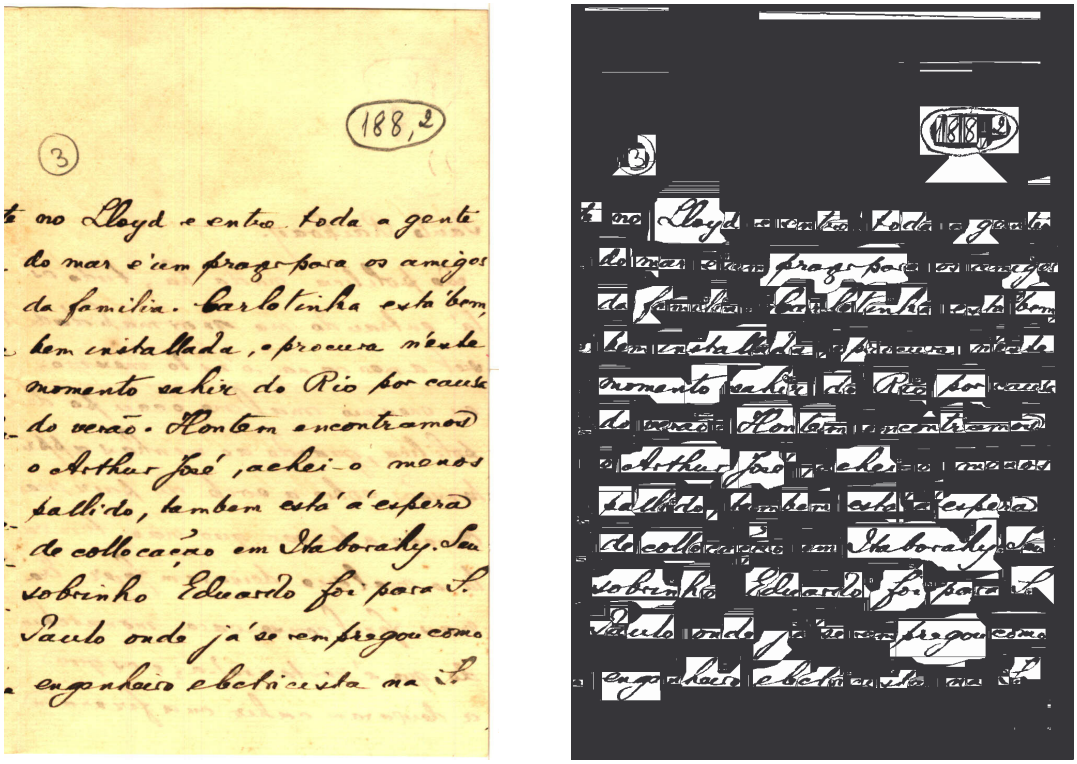
Seu do Coração

JOAQ. NABUCO.

(a)

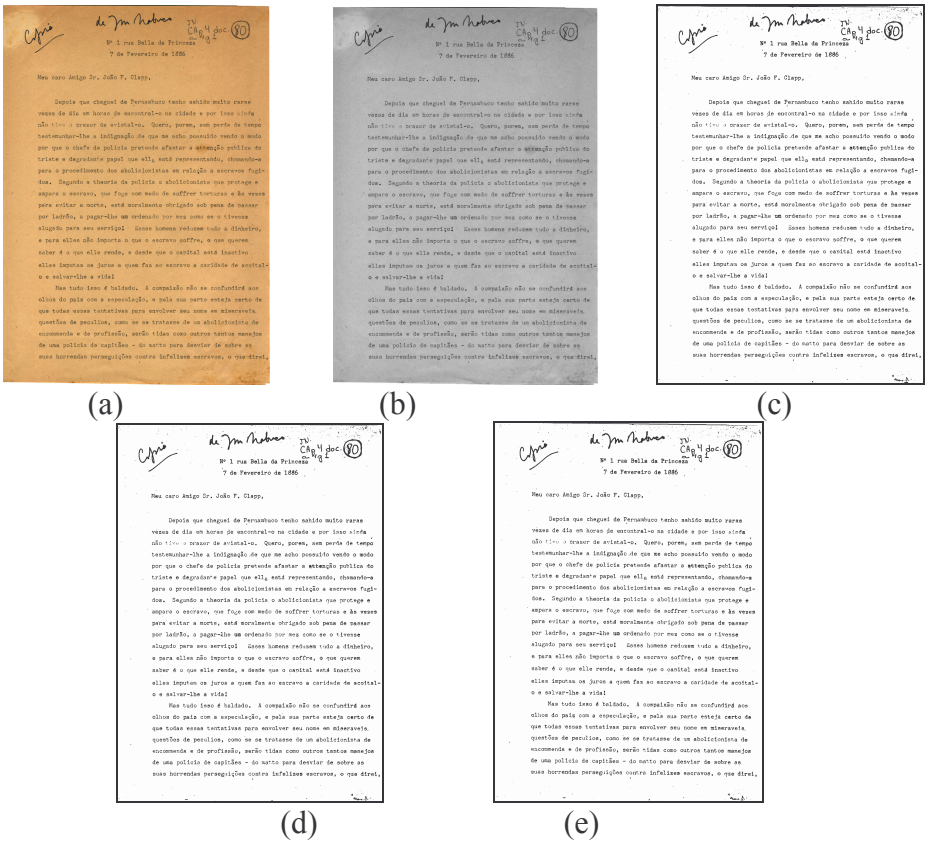
(b)

Figura 45. A imagem original do documento datilografado está apresentada em (a) e a segmentação desta imagem esta em (b).



(a) (b)

Figura 46. A imagem original do documento manuscrito está apresentada em (a) e a segmentação desta imagem está em (b).



(a) (b) (c)

(d) (e)

Figura 47. Etapas aplicadas a cada imagem antes de serem aplicados os algoritmos de segmentação. Em (a) o documento original, em (b) o documento em tons de cinza, em (c) o resultado da binarização, em (d) a aplicação do algoritmo para redução de ruído e em (e) a imagem encontra-se com rotacionada.

5.1 Aspectos Avaliados

Foram listadas 10 imagens de documentos com documentos manuscritos e datilografados para a avaliação dos resultados utilizando os algoritmos implementados. Foram avaliados quatro aspectos em cada documento segmentado: a quantidade de blocos, a quantidade de linhas, a quantidade palavras e a quantidade caracteres. Essa avaliação foi feita de forma subjetiva e aproximada pela interpretação manual da imagem resultante. Para cada aspecto avaliado foi considerada a quantidade de segmentos extras encontrados na imagem. O primeiro a ser considerado foi à quantidade de blocos de texto. Foi verificado se o algoritmo segmentou um bloco uma área que não possui texto (bloco extra). A Figura 48 exemplifica o que foi considerado um bloco extra. O próximo aspecto tratado foi em relação às linhas segmentadas pelo algoritmo. Foi verificado se ocorreu à segmentação de linhas extras, ou seja, que não possui texto, Figura 48. Se duas linhas foram segmentadas como apenas uma (união de linhas), Figura 49, e o inverso, se uma linha foi segmentada como duas linhas, Figura 50. Outro aspecto abordado foi em relação às palavras segmentadas. Analogamente, verificou-se a ocorrência de palavras extras – regiões com a dimensão de uma palavra – Figura 51 , a união de palavras, Figura 53 e a quebra de uma palavra em duas ou mais. No caso da quebra de palavras, foi considerada uma linha de divisão entre as palavras superior à média, ver Figura 52. Por fim, foram avaliadas as mesmas características em relação à segmentação dos caracteres, caracteres extras, Figura 51, conectados

Figura 52 e separados, Figura 54. Para cada aspecto foi contabilizada a quantidade de ocorrências em cada documento avaliando.

A Tabela 2 apresenta a avaliação do algoritmo de análise de projeção, apresentando os dados encontrados. A Tabela 3 apresenta a avaliação do algoritmo utilizando a transformada de *watershed*, mostrando os dados encontrados. A Tabela 4 apresenta o resultado do algoritmo utilizando a transformada de *watershed* na imagem dilatada.

Figura 48. Exemplo de um bloco extra na imagem, representada pela área superior da imagem. Este bloco é composto por duas linhas extras. Imagem segmentada utilizando a transformada de watershed.

Figura 49. Exemplo de duas linhas segmentadas como apenas 1. Imagem segmentada utilizando a análise de projeção.

Figura 50. Exemplo de uma linha segmentada com duas. .Imagem segmentada utilizando a análise de projeção.

Figura 51. Exemplo de um segmento extra na imagem, considerado os dois segmentos no canto esquerdo da imagem. Ele é composto por dois segmentos que foram considerados dois caracteres extras. Imagem segmentada utilizando a transformada de watershed com dilatação.

Figura 52. Exemplo de palavra quebrada em três: promovido foi segmentado em ‘p’, ‘rom’ e ‘ovido’. O último segmento é um exemplo de caracteres conectados. Imagem segmentada utilizando a análise de projeção.

Figura 53. Exemplo de palavras unidas. Imagem segmentada utilizando a transformada de watershed com dilatação.

Figura 54. Exemplo de quebra de caractere. Imagem segmentada utilizando a análise de projeção.

Tabela 2. Resultados da análise de projeção.

Nome da Imagem	Aspectos Considerados	Extra	Quebra	Junção
d0068-3	Blocos	1	-	-
	Linhas	1	0	2
	Palavras	4	15	7
	Caracteres	20	3	65
d0080-1	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	4	0	0
	Caracteres	10	8	9
d0087	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	1	1	2
	Caracteres	23	0	70
d0092	Blocos	2	-	-
	Linhas	2	0	0
	Palavras	15	0	0
	Caracteres	30	32	41
d0113	Blocos	1	-	-
	Linhas	1	0	2
	Palavras	2	0	0
	Caracteres	3	0	36
d0188-3	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	3	3	5
	Caracteres	3	5	53
d0188-4	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	2	1	2
	Caracteres	0	2	65
d0188-5	Blocos	1	-	-
	Linhas	1	1	0
	Palavras	2	4	1
	Caracteres	0	1	71
d0188-6	Blocos	1	-	-
	Linhas	1	0	2
	Palavras	1	1	6
	Caracteres	1	3	70
d0188-7	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	5	2	3
	Caracteres	9	1	44

Tabela 3. Resultados do algoritmo da transformada de *watershed.*

Nome da Imagem	Aspectos Considerados	Extra	Quebra	Junção
d0119-1	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	5	6	0
	Caracteres	10	22	35
d0119-2	Blocos	2	-	-
	Linhas	3	0	0
	Palavras	28	0	10
	Caracteres	13	30	66
d0120-7	Blocos	1	0	0
	Linhas	2	1	4
	Palavras	3	3	4
	Caracteres	14	5	108
d0158-3	Blocos	1	-	-
	Linhas	1	0	2
	Palavras	2	1	3
	Caracteres	3	8	50
d0158-4	Blocos	2	-	-
	Linhas	1	0	0
	Palavras	6	1	22
	Caracteres	2	10	83
d0188-3	Blocos	1	-	-
	Linhas	1	0	0
	Palavras	2	4	4
	Caracteres	3	14	49
d0188-4	Blocos	2	-	-
	Linhas	3	0	0
	Palavras	7	4	3
	Caracteres	30	3	59
d0188-5	Blocos	1	-	-
	Linhas	2	0	0
	Palavras	4	1	2
	Caracteres	2	11	74
d0188-6	Blocos	2	-	-
	Linhas	2	2	2
	Palavras	5	6	5
	Caracteres	4	4	73
d0188-7	Blocos	2	-	-
	Linhas	3	0	0
	Palavras	8	3	0
	Caracteres	15	3	57

Tabela 4. Resultados do algoritmo da transformada de *watershed* na imagem dilatada.

Nome da Imagem	Aspectos Considerados	Extra	Quebra	Junção
d0119-1	Blocos	1	-	-
	Linhas	1	1	2
	Palavras	4	8	2
	Caracteres	15	16	35
d0119-2	Blocos	2	-	-
	Linhas	2	3	3
	Palavras	10	0	23
	Caracteres	16	10	54
d0120-7	Blocos	1	-	-
	Linhas	2	2	0
	Palavras	4	1	1
	Caracteres	8	2	50
d0158-3	Blocos	1	-	-
	Linhas	1	1	3
	Palavras	3	1	4
	Caracteres	30	5	89
d0158-4	Blocos	1	-	-
	Linhas	1	1	3
	Palavras	3	3	25
	Caracteres	12	4	83
d0188-3	Blocos	1	-	-
	Linhas	2	0	0
	Palavras	3	0	1
	Caracteres	16	8	69
d0188-4	Blocos	2	-	-
	Linhas	2	0	0
	Palavras	5	0	0
	Caracteres	22	0	42
d0188-5	Blocos	1	-	-
	Linhas	2	1	2
	Palavras	4	0	5
	Caracteres	5	2	80
d0188-6	Blocos	2	-	-
	Linhas	3	0	0
	Palavras	6	2	2
	Caracteres	12	0	44
d0189	Blocos	2	-	-
	Linhas	2	0	2
	Palavras	3	5	3
	Caracteres	7	8	60

Como apresentado nas tabelas, foi observado que em todos os documentos, houve a adição de um bloco extra, em todas as implementações. Isso se deve às características do acervo. Não foi avaliada a ocorrência de blocos conectados nem quebra de um bloco em dois, por isso o

campo destes dados está composto por '-'. Observou-se que o problema maior encontra-se na conexão de caracteres que nas três abordagens tiveram uma taxa superior as demais. Foi visto que o algoritmo que utiliza a transformada de *watershed* segmenta melhor os caracteres do que a que utiliza a transformada sobre a imagem dilatada, enquanto que esta segmenta melhor a as palavras. A tabela 5 mostra os resultados de uma mesma imagem aplicada ao algoritmo utilizando *watershed* e, aplicando o mesmo algoritmo sobre a imagem dilatada. Percebe-se que em relação aos blocos ambos os algoritmos se comportaram da mesma forma, apresentado dois blocos de textos extras. Em relação a quantidade de linhas, o segundo algoritmo portou-se melhor na identificação de linhas extras enquanto que nos demais aspectos o primeiro apresentou menores taxas de erro. Em relação a quantidade de palavras, o algoritmo utilizando *watershed* sobre a imagem segmentada conseguiu uma quantidade inferior de palavras extras, a quantidade de quebra foi a mesma, por outro lado a quantidade de junção de palavras teve uma taxa bem maior. O problema maior na aplicação dos dois algoritmos foi na segmentação de caracteres. A abordagem utilizando *watershed* sobre a imagem apresentou uma taxa de erro maior que outra abordagem exceto em relação a identificação de caracteres extras.

Em geral, as duas abordagens apresentam taxas de erros bastante próximas. Uma vantagem de utilizar a abordagem sobre a imagem dilatada é o tempo de processamento que fica reduzido para cerca de dois terços em relação ao tempo de processamento da abordagem sobre a imagem sem alteração. A imagem apresentada na Tabela 5 passou 3h51m53s para processar utilizando a abordagem utilizando *watershed*. Por outro lado, a mesma imagem passou 2h59m04s com o algoritmo utilizando a imagem dilatada.

Tabela 5. Resultados ddas abordagens que utilizam o algoritmo de *watershed*.

Imagem d0119-2	Aspectos Considerados	Extra	Quebra	Junção
Algorimo de *watershed*	Blocos	2	-	-
	Linhas	3	0	0
	Palavras	28	0	10
	Caracteres	13	30	66
Algorimo de *watershed* sobre a imagem dilatada	Blocos	2	-	-
	Linhas	2	3	3
	Palavras	10	0	23
	Caracteres	16	10	54

Capítulo 6

Conclusão

A preocupação na preservação de informações históricas e a divulgação desse material motivaram várias pesquisas nesta área. Os seres humanos possuem a habilidade de extrair da imagem os objetos relevantes facilmente pela sua visão, no entanto, a modelagem desse processo para ser automatizado pelo computador não é tão simples [33]. A grande dificuldade na segmentação encontra-se na variação de informação nas imagens que estão sendo tratadas, sem uma definição de antemão das características do objeto desejado. A segmentação de documentos é uma tarefa de fundamental importância no processo de análise de documentos. Tem como finalidade localizar os caracteres dentro da imagem a fim de que ele seja posteriormente reconhecido por uma ferramenta de classificação. É visto que a maioria dos erros no reconhecimento de caracteres ocorre devido às falhas na etapa de segmentação.

Este trabalho dá uma visão geral do processamento de documentos dando ênfase no processo de segmentação de documentos históricos. Foram implementadas três abordagens para identificação de texto em documentos históricos. Foi utilizado um acervo datado do século XX que possui tanto documentos datilografados quanto manuscritos. A esse acervo foram aplicadas técnicas de pré-processamento para remover dados irrelevantes como o algoritmo de binarização definido em [30] que gera a imagem em preto-e-branco. Outro processo aplicado às imagens foi o alinhamento das linhas do documento em relação à pauta do papel utilizando a transformada de Hough [3].

O processo de segmentação utilizado em uma imagem varia de acordo com o tipo de objeto que se deseja manipular, e, além disso, foi verificado que também depende da natureza da imagem. Então, em documentos datilografados a abordagem utilizando análise de projeção se portou melhor, enquanto em documentos manuscrito a abordagem utilizando a transformada de *watershed* obteve melhor desempenho. Como resultado, tem-se uma imagem com o texto segmentado.

6.1 Trabalhos Futuros

Segmentação de documentos é uma área que está em constante desenvolvimento. Os algoritmos implementados neste trabalho são suficientemente simples que permitem sua expansão. Uma sugestão é a utilização de uma linguagem de programação que proporcione maior flexibilidade e desempenho, como C ou C++. Outra abordagem é aplicar aos acervos de diferentes autores para verificar seu desempenho em diferentes caligrafias e estilo de texto.

Outra sugestão de trabalho futuro consiste em desenvolver a abordagem que utiliza a transformada de watershed para segmentação, utilizando uma arquitetura paralela para reduzir o tempo de execução do algoritmo. Outra abordagem é aplicar a análise de projeção para identificar as linhas de um documento e gerar imagens pra cada linha. A cada imagem criada aplicar o algoritmo de algoritmo de segmentação utilizando a abordagem de *watershed.*

Outro trabalho futuro consiste em implementar algoritmos voltados para a caligrafia do autor do acervo utilizado como a identificação da inclinação das lembras manuscritas, além de uma ferramenta de reconhecimento de caracteres específica para a sua caligrafia.

Bibliografia

[1] PARKER, J. R. *Algorithms for Image Processing and Computer Vision.* Ed. John Wiley and Sons, Inc. 1995.

[2] O'GORMAN, L. E KASTURI, R. *Document Image Analysis*. Califórnia, IEEE Computer Society Press, 1995.

[3] WITTEN, I. H. E MOFFAT, A. *Managing Gigabytes Compressing And Indexing Documents And Images*. San Francisco, Ed. Morgan Kaufmann, 1999, 3ª ed.

[4] ANTONACOPOULOS, A. E KARATZAS, D. *Semantics-Based Content Extraction in Typewritten Historical Documents*. Proc. da Conferência Internacional de Análise e Reconhecimento, ICDAR, Seul, Coréia., Agosto, 2005.

[5] PRATT, W. K. *Digital Image Processing*. Ed.Addison Wesley, 3ª ed, Nova York, 2001.

[6] GONZALEZ, R. E WOODS, R. *Digital Image Processing.* Massachusetts, Ed. Addison-Wesley, 2002.

[7] KASTURI, R., O'GORMAN, L. E GOVINDARAJU, V. *Document Image Analysis: A Primer.* Sadhana, Document Processing, , 2002.

[8] SEEMANN, T. *Digital Image Processing using Local Segmentation*. Tese de Doutorado, Escola de Ciências da Computação e Engenharia de Software. Faculdade de Tecnologia da Informação, Universidade da Monash, Austrália. Abril, 2002.

[9] MELLO, C. A. B. ; OLIVEIRA, A. L. I. de ; SANCHEZ, A. *Image Thresholding of Historical Documents: Application to the Joaquim Nabuco's File*. In: Digital Cultural Heritage Conference - Eva Vienna, 2006, Viena. Proceedings of the 1st EVA 2006 Vienna Conference. Vienna : Austrian Computer Society, 2006. p. 115-122.

[10] MELLO, C. A. B. ; COSTA, A. H. *Image Thresholding of Historical Documents Using Entropy and ROC Curves*. Lecture Notes in Computer Science, v. 3773, p. 905-916, 2005.

[11] MELLO, C. A. B. ; CAVALCANTI, C. S. entre outros. *Geração Automática de Imagens de Documentos Históricos*. Revista Engenharia Ciência Tecnologia, Espírito Santo, v. 7, n. 3, p. 11-16, 2004.

[12] The MathWorks, disponível em: http://www.mathworks.com/. Último acesso em: 20/06/2006.

[13] Tutorial disponível na internet via WWW. URL: http://hanyfarid.org/tutorials/fip.pdf. Último acesso em 17/11/2006.

[14] BOAVENTURA, R. A. *Analise da utilização de filtros lineares e não-lineares na recuperação de imagens degradadas*. Dissertação de Mestrado, UNICAMP, Brasil. 2004.

[15] DE ANDRADE, M.C., BERTRAND, G. , EVERAT, J.C. e ARAÚJO, A. de A. *Segmentation of ceramical micrographies by flooding simulation: A catchment basins merging algorithm*. Proc. do IX Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens - SIBGRAPI, pp 355-356, Brasil, 1996.

[16] MANMATHA, R. *A scale space approach for automatically segmenting words from historical handwritten documents*, IEEE Transactions on Pattern Recognition and Image Analysis, vol. 27(8), pp 1212-1225, agosto, 2005.

[17] SILVA JR., N. I. *Um Sistema de Compressão de Imagens Aplicado a Documentos Históricos*. Dissertação de Mestrado. UFMG, Brasil. 1993.

[18] SIMÕES, A. S. *Segmentação de imagens por classificação de cores: uma abordagem neural*. Dissertação de Mestrado, Universidade de São Paulo, USP, Brasil. 2002.

[19] MARCOLINO, A., RAMOS, V., RAMALHO, e CALDAS, M. J. *Line and Word Matching in Old Documents*, Proc. do SIARP'2000 - 5th Simpósio Ibero-Americano em Reconhecimento de Padrões. pp. 123-135, Lisboa, Portugal, 11-13 Setembro. 2000.

[20] ROCHA, J. E PAVLIDIS, T. *A Shape Analysis Model with Applications to a Character Recognition System*, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.16 n.4, p.393-404, Abril 1994.

[21] MAO, S., ROSENFELD, A. E KANUNGO, T. *Document Structure Analysis Algorithms: a Literature Survey*", Proceedings of SPIE/IS&T, vol. 5010, pp. 197-207, 2003.

[22] BRITTO JR, A. S., FREITAS, C. O. A., JUSTINO, E., BORGES, D. L., FACON, J., BORTOLOZZI, F., SABOURIN, R. *Técnicas em Processamento e Análise de Documentos Manuscritos*. Revista de Informática Teórica e Aplicada. Porto Alegre - UFRGS:, v.8, n.2, p.47 - 68, 2001.

[23] ANDRADE, A. F., BOTELHO, M. F. e CENTENO, J. A. S. *Classificação de Imagem de alta resolução Integrando Variáveis Espectrais e Forma Utilizando Redes Neurais Artificiais*. In: XI SBSR - Simpósio Brasileiro De Sensoriamento Remoto, Belo Horizonte. 2003.

[24] CAMPOS, G.L. *Uma técnica para alinhamento de imagens de documentos antigos*. Anais do XII Congresso da Sociedade Brasileira de Computação. pp.215 – 224, 2002.

[25] CASEY, R. e LECOLINET, E. *A Survey of Methods in Strategies in Character Segmentation.* IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, pp. 690-706, 1996.

[26] MOGA, A.N, CRAMARIUC, B. e GABBOUJ, M. *An Efficient Watershed Segmentation Algorithm Suitable for Parallel Implementation*. Proc. IEEE Int'l Conf. Image Processing, vol. 2, pp. 101-104, Washington, D.C., Outubro. 1995.

[27] ROERDINK, J. B. e MEIJSTER. A. *The watershed transform: Definitions, algorithms and parallelization strategies.* Fundamenta Informaticae 41, pp. 187–228, 2001.

[28] BEUCHER, S., BILODEAU, M. E YU, X. *Road segmentation by watersheds algorithms*. Proc. of the Pro-art vision group PROMETHEUS workshop, Sophia-Antipolis, França, Abril, 1990.

[29] ANTONACOPOULOS, A. E CASTILLA, C. C. *Flexible Text Recovery from Degraded Typewritten Historical Documents*. ICPR (2) 2006: 1062-1065

[30] MELLO, C and LINS, R. *Image Segmentation of Historical Documents*, Visual 2000,Agosto, 2000, México.

[31] KENNARD, D. J. E BARRETT, W. A. *Separating Lines of Text in Free-Form Handwritten Historical Documents* Proceedings of the Second International Conference on Document Image Analysis for Libraries (DIAL'06) 2006

[32] DONG, J., PONSON, D., KRZYZAK, A., e SUEN, C. Y. *Cursive word skew/slant corrections based on Random transform*. Proceedings of the 8th International Conference on Document Analysis and Recognition (ICDAR2005), Seul, Coréia., Agosto, 2005.

[33] ANTONACOPOULOS, A. E KARATZAS, D. *Semantics-Based Content Extraction in Typewritten Historical Documents*. Proceedings of the 8th International Conference on Document Analysis and Recognition (ICDAR2005), Seul, Coréia., Agosto, 2005.

Apêndice A

Algoritmos implementados

Neste apêndice são apresentados os códigos dos algoritmos implementados neste trabalho. Um tutorial sobre o Matlab pode ser encontrado em [12].

- **Abordagem utilizando a analise de projeção**

```
%Implementaçao do Projection Profile
function projection(x, ext, idx_hor,idx_vert)

%Verifica se a imagem esta binarizada; caso esteja, prossegue com a
%execuçao do projection profile. Caso nao esteja, a imagem passara pelo
%processo de para converter em tons de cinzas, binarizaçao, angulo de rotaçao
e
teste = imagemRGB(x, ext);

if (teste == 0)
    nome = [x '.' ext];
    Im = imread(nome);
else
    func(x, ext);
    nome = [x '_tc_binarizada_rr_rt3.bmp'];
    Im = imread(nome);
end

imshow(nome);
tam = size(Im);
Im_proj = Im;

aux = 0;

%Funçao que calcula a quantidade de pretos em cada linha da imagem e
%armazena em um vetor (mat)
for i=1:tam(1)
   for j=1:tam(2)
       if (Im(i,j)==0)
           aux = aux + 1;
       end
   end
   mat(i) = aux;
```

```
   aux = 0;
end
%mat
%idx_hor = 0;


%Funcao que traça as linhas pretas nas linhas que possuem uma quantidade de
%pixels preto abaixo de um valor idx_hor
for a=1:tam(1)
    if (mat(a)<=idx_hor) %verifica a quantidade de pixels pretos em uma linha
        for b=1:tam(2)
            Im_proj(a,b) = 0;
        end
        mat(a) = tam(2);
    end
end

p = -1;
q = 0;

%Funcao que calcula os intervalos que possuem letras. Armazena nos indices
%impares o inicio da area que possui letra e nos indices impares o fim
%desta area.

for a=1:(tam(1)-1)
    if (a==1 && mat(a) < tam(2))
        p = p + 2;
        mat(p) = a;
    else
           if (mat(a)==tam(2) && mat(a+1)<tam(2))
                p = p + 2;
                mat2(p) = a+1;
            else
                if (mat(a)<tam(2) && mat(a+1)==tam(2))
                    q = q + 2;
                    mat2(q) = a;
                else
                    if(a == tam(1)-1 && mat(a+1)<tam(2))
                        q = q + 2;
                        mat2(q) = a;
                    end
                end
        end
    end
end

erro(mat2,'horizontal');

%mat2
tam2 = size(mat2);
auxiliar = 0;
projecao = [];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
%Funçao que calcula a projeçao vertical em cada intervalo que possui letras
for c = 1:2: tam2(2)

    % Funcao que calcula a quantidade de pretos em uma coluna e armazena em
    % um vetor
```

```
    for i = 1:tam(2)
        for j = mat2(c):mat2(c+1)
            if(Im(j,i) == 0)
                auxiliar = auxiliar + 1;
            end
        end
        vetor(i) = auxiliar;
        auxiliar = 0;
    end
     erro(mat2,'vertical');

% vetor
%Funcao que preenche as colunas que separam as letras

  for a=1:tam(2)
      if (vetor(a)<=idx_vert)
          for b=mat2(c):mat2(c+1)
              Im_proj(b,a) = 0;
          end
          vetor(a) = mat2(c+1);
      end
    end

    k = -1;
    r = 0;
    tam_vetor = size(vetor);
    %tratar as bordas
    for (i=1:tam_vetor(2)-1)
        if (vetor(i) == mat2(c+1) &&  vetor(i+1) < mat2(c+1))
            k = k+2;
            projecao (k,1) = mat(c);
            projecao (k,2) = i+1;
            projecao (k,3) = mat(c+1);
            projecao (k,4) = i+1;
        else
            if (vetor(i) < mat2(c+1) && vetor(i+1) == mat2(c+1))
                r =r + 2;
                projecao (r,1) = mat(c);
                projecao (r,2) = i;
                projecao (r,3) = mat(c+1);
                projecao (r,4) = i;
            end
        end
    end
end

erro(projecao, 'projecao');

figure, imshow(Im_proj);
nomeproj = [x '_proj.bmp'];
imwrite (Im_proj, nomeproj, 'bmp');
nomeproj = [nomeproj '.mat'];
save nomeproj.mat,projecao;
```

- **Abordagem utilizando a transformada de watershed**

```
function water(x, ext)
```

```
nome = [x '.' ext];
Im = imread(nome);
tam = size(Im);
Im_Nova = Im;

L = watershed(Im,26);

%imshow(L);

vetor = [];

%montando o vetor com as areas contruidas pelo watershed
for i = 1:tam(1)
    for j=1:tam(2)
        if(L(i,j)==0)
        else
            vetor =  pertence(vetor,(L(i,j)));
        end
    end
end

tamVetor = size(vetor);

for i=1:tam(1)
    for j=1:tam(2)
        if (L(i,j) == 0)
            Im_Nova(i,j) = 0;
        end
    end
end

figure, imshow(Im_Nova);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%vetor
figure, imshow(Im);
b = false;
contador = 0;
contadorBranco = 0;

for v = 1:tamVetor(2)
    wat = vetor(1,v);
    for i=1:tam(1)
        %tratando a linha que pertence a um divisor de agua
        for j=1:tam(2)
            if (L(i,j) == wat) & not(j == tam(2))
                    if not(b)
                        b = true;
                        posicao = j;%armazena a primeira posicao do elemento
                    end %fim do if que verifica se ha encontrou algum elemento
daquela represa no

                    contador = contador+1;

                    if Im(i,j) == 255 | Im(i,j) == 1
                        contadorBranco = contadorBranco + 1;
                    end
```

```
               else % else do if (L(i,j) == wat)
                    if ((b) & (contador == contadorBranco) & not(contador ==
0))
                        b = false;
                        contador = 0;
                        contadorbranco = 0;
                        for(k = posicao:(j-1))
                            Im_Nova(i,k) = 0;
                        end % end do for que pinta
                    else
                        if (j == tam(2)) & (contador == contadorBranco) &
not(contador == 0)
                            b = false
                            contador = 0;
                            contadorbranco = 0;
                            for(k = posicao:(j))
                            Im_Nova(i,k) = 0;
                            end % end do for que pinta
                        end
                    end %fim do if (b) & (contador == contadorBranco)
            end %Fim do if (L(i,j) == wat)
        end %fim do for da variavel j
        b = false;
        contador = 0;
        contadorBranco = 0;
    end %for da variavel i
end %For da variavel v


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for v = 1:tamVetor(2)
    wat = vetor(1,v);
    for i=1:tam(2)
        %tratando a linha que pertence a um divisor de agua
        for j=1:tam(1)
            if (L(j,i) == wat) & not(j == tam(1))
                    if not(b)
                        b = true;
                        posicao = j;%armazena a primeira posicao do elemento
                    end %fim do if que verifica se ha encontrou algum elemento
daquela represa no

                    contador = contador+1;

                    if Im(j,i) == 255 | Im(j,i) == 1
                        contadorBranco = contadorBranco + 1;
                    end

               else % else do if (L(i,j) == wat)
                    if ((b) & (contador == contadorBranco) & not(contador ==
0))
                        b = false;
                        contador = 0;
                        contadorbranco = 0;
                        for(k = posicao:(j-1))
                            Im_Nova(k,i) = 0;
                        end % end do for que pinta
                    else
```

```
                        if (j == tam(1)) & (contador == contadorBranco) &
not(contador == 0)
                            b = false
                            contador = 0;
                            contadorbranco = 0;
                            for(k = posicao:(j))
                            Im_Nova(k,i) = 0;
                            end % end do for que pinta
                        end
                    end %fim do if (b) & (contador == contadorBranco)
            end %Fim do if (L(i,j) == wat)
        end %fim do for da variavel j
        b = false;
        contador = 0;
        contadorBranco = 0;
    end %for da variavel i
end %For da variavel v



figure, imshow(Im_Nova);

nomews = [x '_water2.bmp'];
imwrite (Im_Nova, nomews, 'bmp');
```

- **Abordagem utilizando a transformada de watershed na imagem dilatada**

```
function water(x, ext)

nome = [x '.' ext];
Im = imread(nome);
tam = size(Im);
Im_Nova = Im;

Im = inverte(Im);
se = strel('line',11,90);
Imdil = imdilate(Im,se);
Imdil = inverte(Imdil);
Im = inverte(Im);

L = watershed(Imdil,26);

%imshow(L);

vetor = [];

%montando o vetor com as areas contruidas pelo watershed
for i = 1:tam(1)
    for j=1:tam(2)
        if(L(i,j)==0)
        else
            vetor =  pertence(vetor,(L(i,j)));
        end
    end
end
```

```
tamVetor = size(vetor)

for i=1:tam(1)
    for j=1:tam(2)
        if (L(i,j) == 0)
            Im_Nova(i,j) = 0;
        end
    end
end

figure, imshow(Im_Nova);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%vetor
figure, imshow(Im);
b = false;
contador = 0;
contadorBranco = 0;

for v = 1:tamVetor(2)
    wat = vetor(1,v);
    for i=1:tam(1)
        %tratando a linha que pertence a um divisor de agua
        for j=1:tam(2)
            if (L(i,j) == wat) & not(j == tam(2))
                    if not(b)
                        b = true;
                        posicao = j;%armazena a primeira posicao do elemento
                    end %fim do if que verifica se ha encontrou algum elemento
daquela represa no

                    contador = contador+1;

                    if Im(i,j) == 255 | Im(i,j) == 1
                        contadorBranco = contadorBranco + 1;
                    end

               else % else do if (L(i,j) == wat)
                    if ((b) & (contador == contadorBranco) & not(contador ==
0))
                        b = false;
                        contador = 0;
                        contadorbranco = 0;
                        for(k = posicao:(j-1))
                            Im_Nova(i,k) = 0;
                        end % end do for que pinta
                    else
                        if (j == tam(2)) & (contador == contadorBranco) &
not(contador == 0)
                            b = false
                            contador = 0;
                            contadorbranco = 0;
                            for(k = posicao:(j))
                            Im_Nova(i,k) = 0;
                            end % end do for que pinta
                        end
                    end %fim do if (b) & (contador == contadorBranco)
            end %Fim do if (L(i,j) == wat)
```

```
        end %fim do for da variavel j
        b = false;
        contador = 0;
        contadorBranco = 0;
    end %for da variavel i
end %For da variavel v


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for v = 1:tamVetor(2)
    wat = vetor(1,v);
    for i=1:tam(2)
        %tratando a linha que pertence a um divisor de agua
        for j=1:tam(1)
            if (L(j,i) == wat) & not(j == tam(1))
                    if not(b)
                        b = true;
                        posicao = j;%armazena a primeira posicao do elemento
                    end %fim do if que verifica se ha encontrou algum elemento
daquela represa no

                    contador = contador+1;

                    if Im(j,i) == 255 | Im(j,i) == 1
                        contadorBranco = contadorBranco + 1;
                    end

               else % else do if (L(i,j) == wat)
                    if ((b) & (contador == contadorBranco) & not(contador ==
0))
                        b = false;
                        contador = 0;
                        contadorbranco = 0;
                        for(k = posicao:(j-1))
                            Im_Nova(k,i) = 0;
                        end % end do for que pinta
                    else
                        if (j == tam(1)) & (contador == contadorBranco) &
not(contador == 0)
                            b = false
                            contador = 0;
                            contadorbranco = 0;
                            for(k = posicao:(j))
                            Im_Nova(k,i) = 0;
                            end % end do for que pinta
                        end
                    end %fim do if (b) & (contador == contadorBranco)
            end %Fim do if (L(i,j) == wat)
        end %fim do for da variavel j
        b = false;
        contador = 0;
        contadorBranco = 0;
    end %for da variavel i
end %For da variavel v



figure, imshow(Im_Nova);
```

```
nomews = [x '_waterDil.bmp'];
imwrite (Im_Nova, nomews, 'bmp');
```