Resumo

Este trabalho apresenta um novo algoritmo de binarização de imagens de documentos históricos e estudo sobre binarização de imagens. Para tanto, é apresentado uma breve revisão de alguns algoritmos clássicos. Binarização é um estágio de pré-processamento que pode ser usado para classificar objetos em uma imagem como parte do *background* ou do *foreground*. No processamento de documentos, os pixels classificados como tinta são convertidos para preto e os pixels classificados como papel são convertidos para branco. Um total de 27 algoritmos foram estudados e estão apresentados neste trabalho. Um novo algoritmo de binarização é proposto. Ele é baseado na entropia da imagem com correção do ponto de corte por curvas ROC. Este algoritmo, no entanto, apresenta baixo desempenho, considerando a velocidade de processamento. Uma mudança foi efetuada, resultando em um algoritmo de velocidade satisfatória. Ambos estão mostrados neste trabalho. Os resultados são comparados com os algoritmos clássicos em termos qualitativos (por inspeção visual) e quantitativos (avaliando *precision*, *recall*, *accuracy* e *specificity*).

Abstract

This work presents a new binarization algorithm of historical document images and a study conducted on binarization. Therefore there are presented brief explanations of a number of classical algorithms. Binarization is a step in image pre-processing that can be used to classify objects as part of the backgroud or of the foreground. In document processing, pixels that are classified as ink are then converted to black while pixels classified as paper are converted to white. A total of 27 algorithms were studied and presented in this work. The new binarization algorithm is based on the image´s entropy and correction through ROC curves. However, this algorithm lacks in performance when considering processing time. A modification was made, which resulted in an algorithm of satisfactory speed. Both are shown in this work. The results are compared to the results of the classic algorithms in qualitative (visual inspection) and quantitative terms (evaluating precision, recall, accuracy, and specificity).

Sumário

Índice de Figuras

Índice de Tabelas

Tabela de Símbolos e Siglas

(Dispostos por ordem de aparição no texto)

CD – compact disc (disco compacto)
DVD – digital video disc (disco digital de vídeo)
HD – hard drive (disco rígido)
JPEG – Joint Photographic Experts Group (grupo conjunto especialista em fotografia)
dpi – dots per inch (pontos por polegada)
Kbytes – kilo byte
BMP – Windows bitmap (bitmap do Windows)
ASCII – American Standard Code for Information Interchange (código padrão Americano para intercâmbio de informação)
JCR – Journal of Cultural Heritage
LAT – Local Average Threshold (limiar médio local)
2D – duas dimensões
1D – uma dimensão
SIS – Simple Image Statistic (estatística simples de imagem)
ROC – Receiver Operating Characteristic (característica de operação de receptor)
PD – probability of detection (probabilidade de detecção)
PFA – probability of false alarm (probabilidade de alarme falso)
OCR – Optical Character Recognition (reconhecimento ótico de caracteres)
TP – true positive (positivo verdadeiro)
FP – false positive (falso positivo)
TN – true negative (negativo verdadeiro)
FN – false negative (falso negativo)
GIF – Graphics Interchange Format (formato para intercâmbio de gráficos)

Agradecimentos

Agradeço em primeiro lugar à Deus, a quem devo graças e que me deu a saúde e a força para chegar até aqui.

À minha família, em especial aos meus pais, por toda criação e educação que me deram e continuam a dar ainda hoje.

Ao professor Carlos Alexandre, não só pela orientação impecável neste trabalho, mas pela orientação que começou a mais de três anos com um projeto de iniciação científica. Agradeço também aos professores do DSC, que contribuíram ativamente para minha formação.

Agradeço à Sofia, pelo carinho, companheirismo, atenção e, claro, paciência durante todos esses anos e pelas revisões do texto.

Aos amigos da POLI que dividiram comigo dias e noites de estudos e projetos, que conseguiram trazer momentos de descontração e bom humor independente da situação. Agradeço à Marcela, Izaura, Pedro e todos da Máfia que me acompanharam desde o início da faculdade e à Leopoldo, Júlio e Thiago que depois também chegaram para ajudar.

Capítulo 1

Introdução

Papel continua sendo um dos mais importantes meios de distribuição e armazenamento de informação. No entanto, o papel é um objeto bastante frágil e suscetível ao envelhecimento. Ainda que hoje encontremos novos métodos de fabricação, produzindo papel de melhor qualidade, toda folha que utilizamos tem uma curta vida útil. Algumas dessas folhas de papel têm nelas conteúdo que precisa sobreviver por muito mais tempo que a vida útil do papel permite. Uma cópia de um documento pode ser feita, utilizando uma máquina copiadora ou traduzindo-o manualmente utilizando outra folha de papel. No entanto, em ambos os casos, continuaremos a ter problemas com o papel.

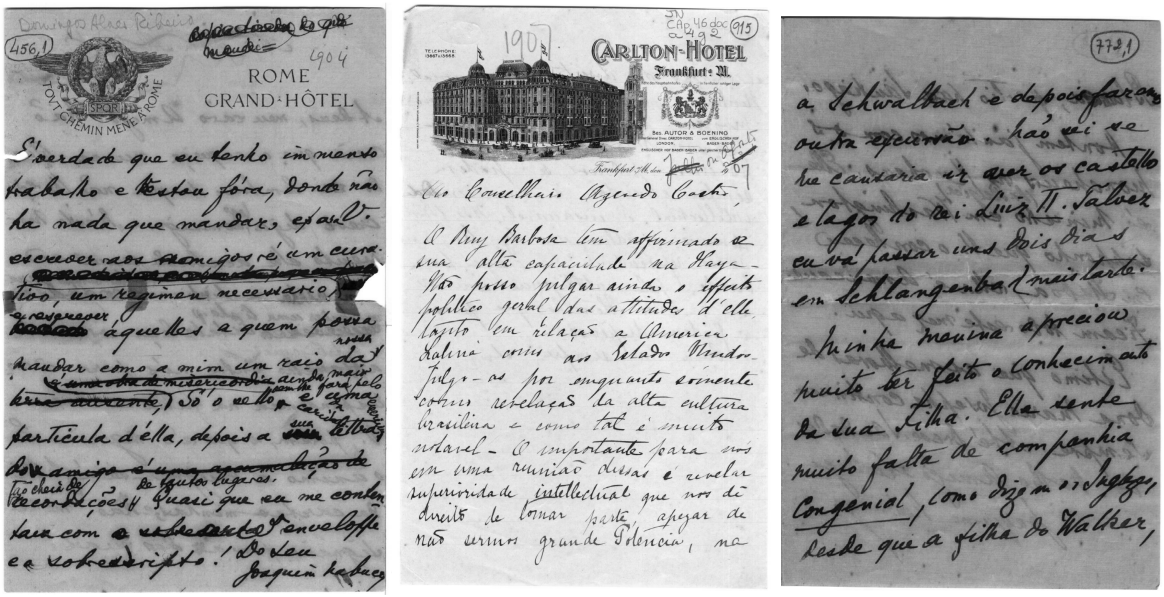
Com o avanço de sistemas computacionais, a solução para este problema veio com a geração de documentos eletrônicos que utilizam editores de texto. Para documentos já existentes, a digitalização é o processo mais apropriado, já que os documentos poderão ser convertidos para imagens digitais. Esse processo resolve vários dos problemas em relação à preservação dos documentos, no entanto, quando o objetivo é facilitar o acesso aos arquivos o problema persiste.

Para estudar nosso passado, entender tudo que acontece à nossa volta e conseqüentemente aprender com os erros dos que vieram antes de nós, precisamos de registros que relatem fielmente o que aconteceu. Encontramos estes em documentos em papel, sejam manuscritos ou datilografados. Esses documentos muitas vezes são antigos e de difícil acesso a todos, por isso, sua digitalização e disponibilização por meio digital (Internet, CDs, etc.) se torna de vital importância. Com a geração de documentos eletrônicos, a informação pode ser armazenada em diversos tipos de meios digitais. Esses meios digitais compreendem vários tipos de mídia, como CDs, DVDs e também tipos de armazenamento magnéticos como HDs e Zip Drives. Ao passarmos as informações para um meio digital, retardamos a perda ocorrida com a degradação causada pelo tempo e também diminuímos o espaço físico necessário para armazenar incontáveis livros, documentos, relatórios, etc. Também encontramos neste meio digital outra grande vantagem que é a da fácil indexação e procura dos documentos. Arquivos em um computador são localizados mais facilmente do que um pequeno documento em uma grande biblioteca ou acervo.

O acervo usado neste projeto é composto de documentos do final do século 19 e começo do século 20. Contém cartas, documentos e cartões postais de Joaquim Nabuco e está atualmente sob os cuidados da Fundação Joaquim Nabuco. Ele soma mais de 6,500 documentos (chegando a 30,000 páginas). Também foi usado, em menor escala, o acervo do Diário de Pernambuco, que contém páginas de jornal digitalizadas datando do século passado.

Joaquim Nabuco (1849-1910) foi uma das mais importantes personalidades da história do Brasil. Ele foi deputado, escritor, diplomata e uma das figuras chave na campanha abolicionista.

Ele também foi o embaixador brasileiro em Londres. A Figura 1 apresenta alguns documentos (em escala de cinza) do acervo. Esses documentos relatam um dos períodos mais importantes da história do Brasil, quando os escravos receberam sua liberdade na campanha abolicionista.



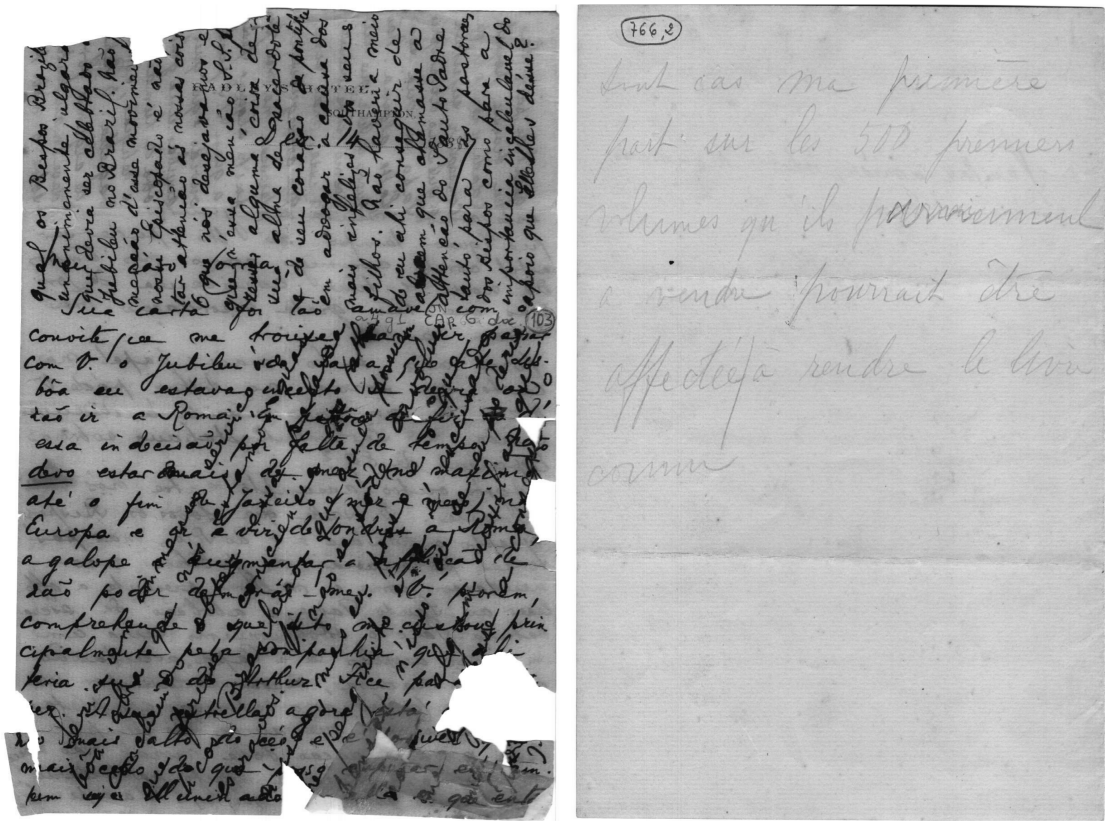
a) b) c)

Figura 1. Exemplos de documentos do acervo.

A coleção completa de documentos do acervo de Nabuco é constituída de documentos manuscritos e datilografados, cartas e cartões postais que apresentam vários desafios importantes para aplicações de processamento de imagens:

1) alguns documentos possuem várias palavras escritas em várias direções (Figura 2a);
2) outros documentos têm escrita de ambos os lados (isso produz um ruído chamado interferência frente-verso à medida que a tinta de um lado começa a passar para o outro lado do papel) como pode ser visto na Figura 1c.
3) existem marcas de fita adesiva no papel ou os documentos estão danificados (Figura 1a);
4) existem documentos que têm a tinta quase apagada (Figura 2b);
5) algumas cartas são datilografadas e manuscritas (Figura 1b);

A Figura 2 exemplifica algumas das características mencionadas acima. O pior problema é lidar com documentos com escrita em ambos os lados do papel. Geralmente, a tinta de um lado passa para o outro lado do papel, dificultando o reconhecimento automático do texto escrito.



a) b)

Figura 2. Algumas características do acervo: (a) documento com escrita em duas direções e com papel danificado e (b) documento com tinta esvanecida.

Quanto à digitalização, são encontrados documentos que não foram digitalizados da maneira mais aconselhável para preservar suas informações, muitas vezes sendo armazenados em um formato de imagem com grande quantidade de perda de informação. Um formato muito utilizado pelas suas propriedades de compressão é o JPEG [1]. Esse padrão utiliza algoritmos de compressão para diminuir o tamanho em disco de uma imagem, mas não sem suas desvantagens. O JPEG pode trazer perdas às imagens, perdas essas que podem ser escolhidas na hora da seleção do formato. As taxas de perda podem facilmente chegar a 40% em relação à imagem original (valor padrão em alguns softwares de tratamento de imagens), gerando assim imagens de baixa qualidade para algumas aplicações.

Um importante aspecto do trabalho com documentos histórico diz respeito justamente à digitalização do documento. Essa tarefa, muitas vezes banalizada por quem está digitalizando o acervo, é responsável por grande perda de qualidade e conseqüentemente de informações nos documentos, como citado anteriormente. Encontramos várias imagens, principalmente de um acervo da Assembléia Legislativa de Pernambuco, que foram mal digitalizadas, o que resultou imagens de péssima qualidade.

Alguns problemas, no entanto, são inerentes ao desgaste do papel. A Figura 3 mostra um documento bastante danificado pelo tempo. Em certos casos, não se imagina um meio de melhoria automática das imagens dos documentos dado o nível de destruição do mesmo.

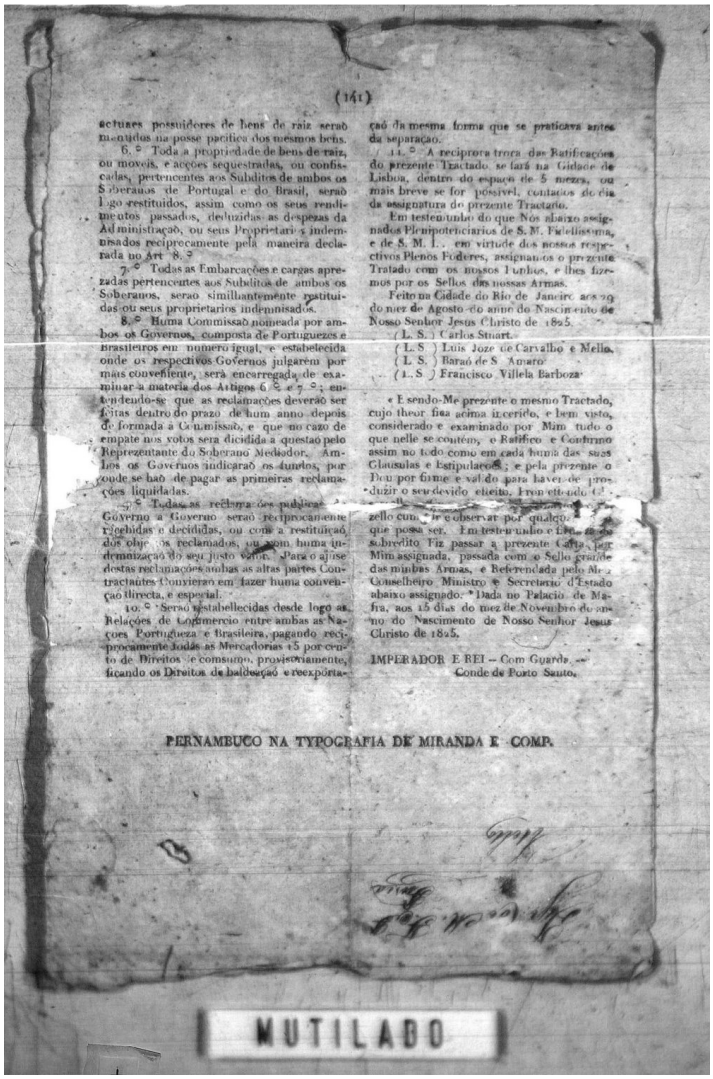


Figura 3. Imagem com má qualidade de digitalização (acervo do Diário de Pernambuco).

A digitalização de um documento encontra ainda outro entrave: o espaço de armazenamento necessário em disco. Por exemplo, uma página em papel A4 digitalizada, utilizando uma resolução para digitalização de 200 dpi e 256 tons de cinza, gera uma imagem de cerca de 1.700x2.400 pixels que ocupa 4.113 Kbytes, se armazenada no formato padrão do sistema Windows, o BMP. Isso torna o armazenamento da imagem muito custoso do ponto de vista computacional. Uma solução para isto é a extração do texto da imagem e a subseqüente geração de um documento de texto simples, que ocuparia menos de 100 Kbytes para a mesma imagem citada acima. Visto que o que procuramos realmente nesses documentos é a informação contida neles, os estudos são focalizados em maneiras de extrair o texto destes documentos de maneira que as informações não sejam perdidas, podendo transpor o que foi escrito à mão para caracteres ASCII (*American Standard Code for Information Interchange* – padrão computacional de caracteres), gerando um documento de texto. Tal processo pode ser visto de forma resumida na Figura 4. Após ser digitalizada, a imagem passa por um tratamento para segmentação, filtragem e binarização. Em seguida, algoritmos de reconhecimento óptico de caracteres atuam nessa imagem para gerar o arquivo de saída em formato de texto.

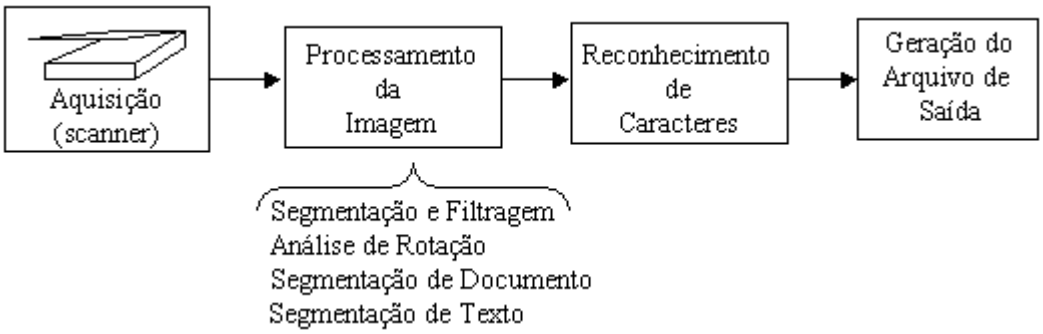


Figura 4. Processo de digitalização até a geração de um documento de texto.

Dessas etapas, uma das mais importantes é a binarização ou limiarização [2]. Esse é um estágio de pré-processamento que pode ser usado para classificar objetos em uma imagem. No processamento de documentos, os pixels classificados como tinta são convertidos para preto e os pixels classificados como papel são convertidos para branco. Para obter uma melhor classificação, um valor correto para o limiar deve ser definido. O valor é considerado correto, se todas as informações essenciais contidas na imagem forem preservadas. Para imagens de documentos isto significa que todo o texto está presente na imagem final (chamada de imagem binarizada).

No Capítulo 2, apresentamos um estudo sobre diversos algoritmos de binarização, foco deste trabalho. Em seguida, uma nova proposta de algoritmo de binarização para documentos históricos é apresentada. Esse novo algoritmo foi publicado em abril de 2008 no periódico internacional "Journal of Cultural Heritage" (índice JCR 0,768). O autor deste projeto final de curso é co-autor dessa publicação. O Capítulo 4 apresenta os resultados alcançados pela nova proposta e os compara com os algoritmos clássicos e o Capítulo 5 conclui esta monografia.

Capítulo 2

Algoritmos de Binarizacão

Vários algoritmos de binarização conhecidos podem ser encontrados na literatura como nível médio de cinza [2], two peaks [2], seleção iterativa [3], Brink [4], Kittler e Illingworth [5], Fisher [6], Otsu [7], C Means [8], Huang [9], Yager [10] e Ye-Danielsson [11]. Fazemos uma breve revisão de alguns desses algoritmos neste Capítulo, servindo como um *survey* para o tema.

Interesse particular é dado a algoritmos baseados em entropia. Entropia é a medida do conteúdo de informação. Na teoria da informação, assumindo que, se existem *n* símbolos possíveis, *s*, que ocorrem com probabilidade *p(s)*, a entropia associada com a fonte S de símbolos é dada por:

$$H(S) = -\sum_{i=0}^{n} p(s_i)\log(p(s_i)) \qquad \text{(Eq. 1)}$$

medida em bits/símbolos. Embora uma base logarítmica não seja definida por Shannon [12], Kapur [13] e Kullback [14] analisam que mudanças na base não afetam o conceito de entropia como foi explorado em termos de binarização em [15]. Seis algoritmos de segmentação baseados em entropia são bem conhecidos para limiarização: Pun [16], Kapur *et al* [17], Johannsen [18], Li-Lee [19], Wu-Lu [20] e Renyi [21].

2.1 Medidas de distância de Ali-Silvey

A técnica de Ali-Silvey [22], também conhecida como medidas de distância de Ali-Silvey, está sendo usada para estender o uso da entropia relativa, como o método proposto por Chang *et al,* também conhecido como o número de Kullback-Leibler. O método proposto por Chang *et al* usa matrizes de co-ocorrência [2] e uma função critério baseada em entropia relativa. Uma descrição de matrizes de co-ocorrência pode ser encontrada na descrição do algoritmo que leva esse mesmo nome. Entropia relativa faz parte de uma classe mais abrangente de medidas de distância, chamada de medidas de distância de Ali-Silvey, conhecidas também como J-Divergente.

É descrito agora, em termos gerais, como é obtido o ponto de corte a partir do método de Ali-Silvey e suas medidas de distância.

As funções critério usam fórmulas para definir a probabilidade de limiarização de uma imagem *p'*. A expressão geral para a medida de distância, uma vez determinada duas distribuições de probabilidade é

$$D(p.p') = f\left[E_o\left(C(L)\right)\right]$$

onde *p* e *p'* são distribuições no mesmo espaço e *D(p.p')* é a distância entre as duas. Analisando o restante da equação, temos *f* que é uma função crescente, E_o é a expectativa em respeito a *p*, *C* é uma função convexa e *L* é a razão da semelhança.

Existem várias medidas de distância nessa classe, algumas estão mostradas abaixo:

1) Entropia Relativa ou Número de Kullback-Leibler: $I = E_o[-ln(L)]$

2) J-Divergente: $J = E_o[(L-1)ln(L)]$

3) Distância de Bhattacharrya: $B = -ln\left(E_0\left[\sqrt{L}\right]\right)$

4) Distância de Matsusita: $M = \left(E_0\left[\left(\sqrt{L}-1\right)^2\right]\right)^{\frac{1}{2}}$

5) Distância de Chernoff: $Ch = -ln(E_0[L^\alpha])$, *onde* $0 =< \alpha =< 1$

Chang *et al* utilizou entropias relativas como um método de limiarização baseado na idéia de que pequenos pedaços de informação seriam perdidos entre a imagem original e a imagem segmentada. Portanto, minimizando a distância em relação a *t*, o ponto de corte ideal *t** seria encontrado. Essa é uma forma de encontrar o ponto de corte utilizando qualquer membro da classe de medidas de distância de Ali-Silvey. Acima, mostramos cinco medidas como exemplo. Em geral, o ponto de corte *t** pode ser encontrado por:

$$t^* = ArgMin(D(t))._{0 =< t =< l-1}$$

onde *D* é qualquer uma das medidas de distância de Ali-Silvey.

2.2 Maximização de Brink

O método de Brink [4] consiste basicamente em encontrar dois valores de limiar para uma dada imagem. Esses valores são obtidos usando o método de maximização de Brink. Quando os dois valores de ponto de corte são obtidos, pixels com valores menores que o primeiro ponto de corte são convertidos para o preto e os pixels com valores maiores que o segundo ponto de corte são convertidos para o branco. Ainda nos resta um terceiro grupo de pixels: aqueles que têm valores entre os dois pontos de corte. Um método de vizinhança determina se esses pixels são convertidos para preto ou branco.

O algoritmo de Brink utiliza duas variáveis, T_1 e T_2 que são determinadas pelo critério de correlação de Brink, que segue abaixo:

$$P_{xy}(T_1, T_2) = \frac{E_{xy}(T1, T2) - E_x E_y(T1, T2)}{\left(V_x V_y(T1, T2)\right)^2}$$

onde:

$$E_{xy}(T1, T2) = \sum_{g=0}^{T_1} agp_g + \sum_{g=T_1+1}^{T_2} bgp_g + \sum_{g=T_2+1}^{n} cgp_g$$

$$E_x = \sum_{g=0}^{n} gp_g \qquad E_y = \sum_{g=0}^{T_1} ap_g + \sum_{g=T_1+1}^{T_2} bp_g + \sum_{g=T_2+1}^{n} cp_g$$

$$E_{xx} = \sum_{g=0}^{n} g^2 p_g \qquad E_{yy} = \sum_{g=0}^{T_1} a^2 p_g + \sum_{g=T_1+1}^{T_2} b^2 p_g + \sum_{g=T_2+1}^{n} c^2 p_g$$

$$V_x = E_{xx} - [E_x]^2 \qquad V_y(T1,T2) = E_{yy}(T1,T2) - \left[E_y(T1,T2)\right]^2$$

T_1 e T_2 são obtidos quando o valor para $P_{xy}(T_1, T_2)$ é máximo. Para obter esses valores, todas as possibilidades para T_1 e T_2 devem ser testadas e o P_{xy} correspondente a cada par calculado. Após a verificação de todos os pixels, os valores finais de T_1 e T_2 são encontrados. Como mencionado anteriormente, os pixels são convertidos para preto ou para branco, dependendo de onde seus valores estão localizados em relação a T_1 e T_2. Caso o pixel esteja na região entre os valores, seu valor final é calculado usando um método de vizinhança. Uma região de 25x25 é utilizada para fazer esse cálculo. Caso exista algum pixel do terceiro grupo, com valor acima de T_2, nesta região de 25x25, então o pixel em análise (localizado entre as regiões) é convertido para branco.

2.3 Operador Laplaciano: Método de Chehikian

O método de Chehikian [23] apresenta uma proposta diferente pelo fato de ser adaptativo, processando a imagem não como um todo, mas quebrando-a em janelas menores. Um valor de corte pode ser obtido a partir do histograma da imagem, que é um gráfico em duas dimensões que mostra a distribuição de cores na imagem. O ponto de corte pode facilmente ser encontrado quando o histograma da imagem apresenta distribuições de pixels concentradas nas extremidades, mais próximos do preto e do branco. No caso de duas distribuições distintas, existem dois picos e o vale entre os picos é o local onde pode ser encontrado o ponto de corte. No entanto, imagens de documentos históricos podem não ter essa distribuição ideal, fazendo com que seus histogramas não tenham a distribuição otimizada como discutido acima. Os histogramas podem ter distribuições bastante variadas, o que inviabiliza o método de ponto de corte global. O método de Laplace, um operador de segunda derivada, intensifica regiões de fronteira, facilitando a identificação de objeto e plano de fundo. Com essa intensificação de regiões fronteiriças, um histograma aceitável para uso no método global pode ser montado. O ruído em uma imagem não pode ser desprezado, já que o ruído, quando analisado isoladamente, apresenta um alto contraste em relação aos seus pixels vizinhos, da mesma forma como objeto e fundo apresentariam. Um filtro para tratamento de ruído é aplicado para que esse método não intensifique o ruído da imagem juntamente com as fronteiras.

Esse método utiliza uma máscara 3x3 correspondente a um filtro passa-alta Laplaciano [24] que é aplicada a todos os pixels:

L =

0	1	0
1	-4	1
0	1	0

A máscara é então convoluída sobre a imagem [2].

Após a filtragem, a imagem pode ser binarizada. Como existem valores de operador para cada pixel, seus níveis de branco ($B[i,j]$) e de preto ($N[i,j]$) podem ser encontrados. O valor do operador para cada pixel é chamado de $\delta[i, j]$, o cálculo dos níveis da imagem é feita de forma recursiva utilizando as seguintes regras:

$B[i,j] = A[i,j]$, se $\delta[i, j]<0$

$B[i,j] = B[i,j-1]$, caso contrário

$N[i,j] = A[i,j]$, se $\delta[i, j]>0$
$N[i,j] = N[i,j-1]$, caso contrário

Esses níveis são então usados para calcular o valor de corte final *t:*

$$t = \frac{N[i,j] + B[i,j]}{2}$$

2.4 Limiarização Fuzzy C Means

O algoritmo de limiarização Fuzzy C Means [8] constrói valores iniciais para dois níveis de cinza, μ_o e μ_b. Esses valores, na verdade, representam a pertinência de que esses níveis sejam encontrados no objeto ou no plano de fundo. O cálculo destes valores é feito através das seguintes fórmulas:

$$\mu_0(t) = \frac{\sum_{j=0}^{t} j.h(j)}{\sum_{j=0}^{t} h(j)} \qquad \mu_b(t) = \frac{\sum_{j=t+1}^{255} j.h(j)}{\sum_{j=t+1}^{255} h(j)}$$

Onde:
j é o nível de cinza e *h(j)* é o número de níveis de cinza que o tom *j* já tem na imagem. Como lidamos com pertinencia, esses valores não precisam variar dentro de um intervalo fixo. Assim, uma manipulação dos valores é feita para que eles se encaixem na faixa de 0 a 1.A manipulação é descrita pelas duas seguintes equações:

$$\bar{\mu_0}(t) = \frac{\mu_0(t)}{\mu_0(t)+\mu_b(t)} \qquad \bar{\mu_b}(t) = \frac{\mu_b(t)}{\mu_0(t)+\mu_b(t)}$$

Após o cálculo desses valores, feito através de fórmulas mostradas acima, o algoritmo entra em um laço para encontrar um valor médio entre as duas pertinências para cada pixel, executando as duas equações abaixo. A primeira descreve a obtenção de um valor de cinza médio do objeto e do fundo e a segunda e terceira equações descrevem as atualizações de μ_o e μ_b .

$$v_i = \frac{\sum_{j=0}^{255} j.h(j).\mu_i(j)^{\tau}}{\sum_{j=0}^{255} h(j).\mu_i(j)^{\tau}},\ i=1,2$$

$$\bar{\mu_0}(j) = \frac{1}{1+\left[\frac{d(j,v_0)}{d(j,v_b)}\right]^{2/(\tau-1)}} \qquad \bar{\mu_b}(t) = 1 - \bar{\mu_0}(j)$$

onde:
v_i é o valor médio de tom de cinza do objeto ou fundo, α é o *fuzziness* devendo ser maior que 1 e $d(j, v_i)$ é a distância entre o nível de cinza e o nível médio correspondente.

Quando a diferença entre os dois valores não for mais significante, o valor do ponto de corte pode ser encontrado calculando a média entre as duas pertinências. Nomeando as pertinências de v_1 e v_2 e o ponto de corte como *t*, temos:

$$Tf = \frac{v_0 + v_b}{2}$$

2.5 Cseke

Esse algoritmo, proposto por Istvan Cseke [25], é originalmente utilizado na contagem de células brancas sangüíneas. Imagens de células sangüíneas têm células brancas e vermelhas misturadas e o número de células brancas é importante para o diagnóstico de algumas doenças. O método se inicia encontrando a posição aproximada das células brancas e removendo da imagem componentes como citoplasma, fundo e células vermelhas. A determinação da posição aproximada das células brancas pode ser feita pelo histograma dado os diferentes tons de cores nos núcleos das células.

A seção de limiarização do processo utiliza um método proposto por Otsu [7], que encontra dois pontos de corte *T1 e T2*, que são selecionados pela maximização da variância interclasse entre as regiões escuras, cinzas e claras. O processo de maximização pode ser reduzido utilizando a seguinte fórmula:

$$E(T1,T2) = \frac{m^2(0,T1)}{n(0,T1)} + \frac{m^2(T1,T2)}{n(T1,T2)} + \frac{m^2(T2,L)}{n(T2,L)}$$

onde *L* representa os tons de cinza e as funções *m()* e *n()* estão mostradas abaixo

$$m(x,y) = \sum_{i=x}^{y-1} i.H[i] \qquad n(x,y) = \sum_{i=x}^{y-1} H[i] \quad ,y>x$$

onde *H[i]* representa o histograma da sub-imagem que está sendo submetida ao processo de limiarização. A função *E* usa valores máximos onde as fórmulas abaixo são satisfeitas

$$\frac{m(0,T1)}{n(0,T1)} + \frac{m(T1,T2)}{n(T1,T2)} = 2.T1$$

$$\frac{m(T1,T2)}{n(T1,T2)} + \frac{m(T2,L)}{n(T2,L)} = 2.T2$$

2.6 Ye-Danielsson e Ridler-Calvard

Nesta Seção, são apresentados dois algoritmos distintos. A razão para a inclusão de dois algoritmos em uma Seção é o fato de que o método de Ye-Danielsson [11] utiliza como seu valor de ponto de corte inicial o valor do ponto de corte final da técnica de Ridler-Calvard [26].

Na técnica de Rilder-Calvard, o histograma *h(i)* da imagem é considerado como sendo um vetor de 256 posições, com cada posição contendo o número de pixels com aquele tom de cinza *i*. O algoritmo tem seu funcionamento baseado nas seguintes equações:

$$At = \sum_{i=0}^{t} h(i) \qquad Bt = \sum_{i=0}^{t} h(i)$$

$$An = \sum_{i=0}^{255} h(i) \qquad Bn = \sum_{i=0}^{255} h(i)$$

$t = \frac{Bn}{An}$ (ponto de corte inicial)

$\mu_t = \frac{B_t}{A_t}$ $\quad\quad$ $v_t = \frac{B_n - B_t}{A_n - A_t}$

As variáveis *A, B, C, u* e *v* são calculadas usando o histograma da imagem e os valores dos níveis de cinza. O valor de ponto de corte é dado por:

$$t = \frac{\mu_t + v_t}{2}$$

O valor final é encontrado recalculando cada variável em cada iteração até que o valor do ponto de corte convirja, ou seja, duas iterações consecutivas sejam iguais.

Após discutir o algoritmo de Ridler-Calvard, a técnica de Ye-Danielsson pode ser explicada. Para este algoritmo, além das equações mostradas anteriormente, devemos também considerar:

$$C_t = \sum_{i=0}^{t} i^2 . h(i)$$

$$C_n = \sum_{i=0}^{255} i^2 . h(i)$$

A partir do limiar *t* encontrado em Ridler-Calvard, usado agora como limiar inicial, podemos aplicar as seguintes fórmulas descritas por Ye-Danielsson:

$p_t = \frac{A_t}{A_n}$ $\quad$ $q_t = \frac{A_n - A_t}{A_n}$ $\quad$ $\sigma_t^2 = \frac{C_t}{A_t} - \mu_t^2$ $\quad$ $\tau_t^2 = \frac{C_n - C_t}{A_n - A_t} - v_t^2$

Em seguida, aplicamos estes valores a uma equação de segundo grau:

$$x^2 \left\{\frac{1}{\sigma_t^2} - \frac{1}{\tau_t^2}\right\} - 2x \left\{\frac{\mu_t}{\sigma_t^2} - \frac{v_t}{\tau_t^2}\right\} + \left\{\frac{\mu_t^2}{\sigma_t^2} - \frac{v_t^2}{\tau_t^2} + \log\left(\frac{\sigma_t^2 . q_t^2}{\tau_t^2 . p_t^2}\right)\right\}$$

Com a equação de segundo grau em mãos, temos seus componentes w_o, w_1 e w_2. Estes valores são usados para calcular o limiar, que será a raiz positiva da seguinte fórmula:

$$t_{final} = \frac{\omega_1 + \sqrt{\omega_1^2 - \omega_0 . \omega_2}}{\omega_0}$$

Os cálculos são feitos repetidamente até que o valor do ponto de corte convirja ou até que uma raiz imaginária seja encontrada em um dos seus cálculos.

2.7 Limiarização Fuzzy de Huang

O algoritmo de Huang [9] usa uma função *E(t)* que é aplicada a todos os valores de *t*. Quando o menor valor é encontrado, esse é tomado como sendo o ponto de corte. Aplicando a imagens com 256 tons de cinza, *E(t)* pode ser expresso como:

$$E(t) = \frac{1}{n.m} \sum_{g=0}^{255} Hf\left(u_x\left(g\right)\right).h(g)$$

onde:
g = nível de cinza
h(g) = nível do histograma para um dado nível de cinza
n = número de linhas na imagem
m = número de colunas na imagem

Ux e *Hf* são dois valores encontrados pelas seguintes fórmulas:

$$u_x = \begin{cases} \frac{1}{1+\left|g-\mu_o(t)\right|/C} & se\ g \le t \\ \frac{1}{1+\left|g-\mu_1(t)\right|/C} & se\ g \ge t \end{cases} \qquad Hf(x) = -x.\log(x) - (1-x).\log(1-x)$$

$$\mu_0(t) = \frac{\sum_{g=0}^{t} g.h(g)}{\sum_{g=0}^{t} h(g)}$$

$$\mu_1(t) = \frac{\sum_{g=t+1}^{255} g.h(g)}{\sum_{g=t+1}^{255} h(g)}$$

onde *C* é constante(*C*=255), a diferença entre o maior e menor nível de cinza.

2.8 Fisher

O algoritmo de Fisher [6] consiste em encontrar valores de ponto de corte entre classes de níveis de cinza. Em um histograma normalizado, uma imagem com objetos escuros em um plano de fundo claro terá duas distribuições de cinza distintas, ou duas classes de cinza distintas. Na técnica de Fisher, os valores de ponto de corte entre as classes *C1* e *C2* são encontrados pela minimização da soma das inércias associadas às diferentes classes. A inércia *W(P)* para as duas classes é descrita na seguinte fórmula:

$$W(P) = \sum_{k \in C_1} h(k).\left(k - G(C_1)\right)^2 + \sum_{k \in C_2} h(k).\left(k - G(C_1)\right)^2$$

onde *k* é o nível de cinza atual e *h(k)* a quantidade de pixels do nível de cinza *k*. $G(C_n)$ representa o centro de gravidade da classe C_n.

No caso particular de segmentação de imagens de documentos históricos, nosso foco, um ponto de corte *t* pode ser encontrado após otimizar o lado direito da equação da inércia mostrada acima e maximizar a fórmula resultante. A otimização de *W(P)* é mostrada a seguir:

$$J(P) = \frac{\left(\sum_{k \in C_1} k.h(k)\right)^2}{\sum_{k \in C_1} h(k)} - \frac{\left(\sum_{k \in C_2} k.h(k)\right)^2}{\sum_{k \in C_2} h(k)}$$

2.9 Fisher LAT (*Local Average Threshold* – Limiar médio local)

Esse outro método de Fisher, conhecido como Fisher LAT [27], propõe o uso de um histograma bi-dimensional da imagem. Um histograma de duas dimensões apresenta mais informações que um histograma de apenas uma dimensão. Dessa forma, são extraídas mais informações da imagem, o que possibilita uma melhor saída com mais informações para cálculo. Esse histograma de duas dimensões proposto por Fisher contém o valor do pixel (usado em histogramas normais) e o valor médio dos seus vizinhos. Com esses dois valores para cada pixel da imagem, um histograma 2D pode ser desenhado. Ao aplicar a técnica de Fisher a esse histograma, um histograma médio local é encontrado. O algoritmo pode ser explicado nos seguintes passos:

1. – Calcular os valores médios locais da imagem com uso da seguinte equação, onde *N* é a quantidade de pixels.

$$g(x,y) = \frac{1}{n^2} \sum_{s=-n/2}^{n/2} \sum_{t=-n/2}^{n/2} f(x+s, y+t)$$

2. – Desenhar o histograma médio local $\bar{h}(.)$, onde temos a ocorrência média local de um nível de cinza, representada por r_j:

$$\bar{h}(j) = \{r_j | 0 \leq j < L\}$$

onde *L* representa a quantidade de níveis de cinza.

3. – Escolher um algoritmo de limiarização que utiliza uma abordagem 1D normal como Otsu [7] ou Kapur [13].
4. – Separar a imagem em duas classes a partir de um nível de cinza local *T,* onde *f(t)* é o LAT:

$$f_T(x,y) = \begin{cases} b_0, & se\ g(x,y) < T \\ b_1, & se\ g(x,y) \geq T \end{cases}$$

2.10 Limiarização Fuzzy Através de Estimativas de Densidades Normais

O algoritmo de Densidades Normais [28] utiliza as distribuições de níveis de cinza para encontrar seu ponto de corte. Como mencionado anteriormente, imagens podem ter seus histogramas bem distribuídos onde podem ser encontradas duas classes facilmente distinguíveis, objeto e plano de fundo. O método então procede calculando o ponto ótimo entre as duas classes, este ponto será o ponto de corte. O algoritmo pode ser descrito da seguinte forma:

1 – Inicialização com a descrição limiar de $\mu 0$ e μb de modo que satisfaçam as seguintes equações:

$$\mu_i(x_j) \in [0,1] \qquad 0 < \sum_{j=1}^{n} \mu_i(x_j) < n \qquad \sum_{i=1}^{c} \mu_i(x_j) = 1.0$$

2 – Calcular os valores medianos para as duas classes citadas usando:

$$v_i = \frac{\sum_{j=0}^{L-1} h_j j \mu_i(j)^T}{\sum_{j=0}^{L-1} h_j \mu_i(j)^T}, i = 1,2,$$

3 – Calcular os seguintes componentes:

$$\beta_i = \frac{\sum_{i=0}^{L-1} \mu_i^T(j) h_j}{\sum_{i=0}^{L-1} \mu_0^T(j) h_j + \sum_{i=0}^{L-1} \mu_B^T(j) h_j}$$

$$\sigma_i = \frac{\sum_{i=0}^{L-1} \mu_i^T(j) h_j (j - v_i)^2}{\sum_{i=0}^{L-1} \mu_i^T(j) h_j}$$

4 – Atualização dos valores de $\mu 0$ e μb usando:

$$\mu_0(j) = \frac{1}{1+\left[\frac{d(j,v_0)}{d(j,v_b)}\right]^{2/(T-1)}} \qquad \mu_b(j) = 1 - \mu_0(j)$$

$$d(j, v_i) = \frac{1}{2}\left(\frac{j - v_i}{\sigma_i}\right)^2 + log\sigma_i - log\beta_i$$

5 – Repetir a partir do passo 2 até que não existam mudanças entre $\mu 0$ e μb.

2.11 Limiarização Fuzzy de Yager

Baseando a análise em lógica fuzzy, pode ser dito que um pixel tem um grau de pertinência em relação a uma de duas categorias, nesse caso, objeto ou plano de fundo. A técnica de Yager [10] se baseia no cálculo de duas medidas de pertinência de níveis de cinza, para objeto e para plano de fundo. O pixel pertence a objeto ou a plano de fundo dependendo do seu valor de nível de cinza e a classe este chega mais perto. Essas pertinências para objeto e para fundo são medidas através do nível fuzzy da imagem. Denominaremos μ_0 como o nível de cinza para o fundo e μ_1 como sendo o nível de cinza para os objetos. O cálculo feito para verificar o nível de pertinência de um pixel em relação a uma das duas classe é feito da seguinte maneira:

$$u_x(g) = \begin{cases} \dfrac{1}{1 + |g - \mu_o(t)|/C} & se\ g \le t \\ \dfrac{1}{1 + |g - \mu_1(t)|/C} & se\ g \ge t \end{cases}$$

onde *C* é uma constante, a diferença entre o valor máximo e o valor mínimo de nível de cinza, e *t* é o valor do limiar.

Agora com as pertinências em mãos, devemos fazer a medida do *fuzziness*, que deve ter um valor máximo de 1. Uma maneira para fazer isso pode ser baseada na entropia de Shannon [12], segue o cálculo do *fuzziness:*

$$E(t) = \frac{1}{M.N} \sum_g Hf\left(\mu_x(g)\right).h(g)$$

Após a medição do *fuzziness*, é necessário uma medida estimada dos valores de μ_0 e μ_1, que dependem do limiar. O limiar escolhido deverá ser aquele que gere o menor valor de *fuzziness*.

$$\mu_0(t) = \frac{\sum_{g=0}^{t} g.h(g)}{\sum_{g=0}^{t} h(g)}$$

$$\mu_1(t) = \frac{\sum_{g=t+1}^{254} g.h(g)}{\sum_{g=t+1}^{254} h(g)}$$

Outra forma de medir o *fuzziness* também pode ser usada, que é uma forma proposta por Yager. Nela, Yager propõe o uso de um plano A onde não existem elementos comuns entre o plano A e seu complemento. O *fuzziness* então é a medida do nível em que A e seu complemento não são distintos. Segue a medida proposta por Yager para níveis *g*:

$$Dp(t) = \left[\sum_g |\mu_x(g) - \mu_{\bar{x}}(g)|^p\right]^{1/p}$$

Aqui, *p* é a distância medida(p=2 para corresponder à distância Euclidiana) e $\mu_{\bar{x}}(g) = 1 - \mu_x(g)$.

2.12 Kittler-Yan

Kittler [5] usa a seguinte função critério para selecionar um ponto de corte otimizado para uma dada função de histograma *h(z):*

$$J_{KI}(t) = \sum_{z=0}^{L-1} h(z) c_{KI}(z,t) = \sum_{z=0}^{t} h(z)\, c_{KI}^{(1)}(z,t) + \sum_{z=t+1}^{L-1} h(z)\, c_{KI}^{(2)}(z,t)$$

onde:

- *z* é o nível de cinza para um pixel
- *L* é o numero de níveis de cinza na imagem e 0 =< *z* =< *L*-1
- *H(z)* é o histograma normalizado dos níveis de cinza na imagem, onde:

$$\sum_{z=0}^{L-1} h(z) = 1$$

- *T* é o valor de limiarização como uma variável na função critério. É necessário achar um valor otimizado para *t* que maximize ou minimize a função critério.
- C_{KI} pode ser visto como duas partes da função valor:

$$C_{KI}(z,t) = \begin{cases} C_{KI}^{(1)}(z,t), se\ z \le t \\ C_{KI}^{(2)}(z,t), se\ z > t \end{cases}$$

2.13 Limiarização Iterativa de Lam-Leung

O método de Lam-Leung [29] usa seleção iterativa para encontrar o ponto de corte que será usado por toda imagem.

Segue uma descrição de seleção iterativa:

1- Inicializa.
2- Seleciona um valor aleatório para *t.*
3- Segmenta a imagem usando *t.*
4- Estima parâmetros para a imagem segmentada.
5- Determina um novo ponto de corte *t* pelos novos parâmetros.
6- Verifica se um novo *t'* foi encontrado, se sim continua, se não volta ao passo (2).
7- Verifica se *t=t´*. Assim sendo, continua. Sendo diferente, *t* recebe o valor de *t`* e retorna para (3).
8- Fim.

Os parâmetros e as estimativas usadas no processo podem variar dependendo de cada implementação de seleção iterativa. No caso do algoritmo de Lam-Leung, o novo ponto de corte será encontrado utilizando a seguinte equação:

$$f(x) = \alpha f(x) + (1-\alpha) f_1(x), onde\ 0 \le \alpha \le 1$$

Outra equação que define seleção iterativa deve ser incluída, usada no método de Lam-Leung como uma extensão da primeira equação aqui mostrada:

$$f_i(x) = N(x; \mu_i, \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} exp\left[-\frac{(x-\mu_i)^2}{2\sigma_i}\right], com \begin{cases} x = 0,1,2,\dots,N-1 \\ i = 0,1 \end{cases}$$

No caso do método de Lam-Leung, os parâmetros são estimados da seguinte maneira: α = 0, μ_i e σ_i são o valor médio e a variação dos pixels que são classificados como C_1 na imagem segmentada. Os pixels que satisfazem à primeira equação mostrada, são considerados como classe C_0 e os que não satisfazem como classe C_1.

2.14 Limiarização por Entropia de Li-Lee

O algoritmo de Li-Lee [19] é mais um método que utiliza entropia. Li-Lee usa entropia mínima cruzada, e encontra o ponto de corte pela minimização da entropia cruzada entre a imagem e sua versão segmentada.

Para entender este algoritmo, é necessário compreender o conceito de entropia máxima e, a partir dessa, o conceito de entropia mínima. O princípio de entropia máxima permite a escolha da solução que gera a maior entropia. Foi demonstrado por experimentos que distribuições com maior entropia têm maior multiplicidade, dessa forma tendo uma maior probabilidade de serem observadas.

Entropia cruzada mede a distância teórica entre duas distribuições *P={p1, p2, p3,…, pn}* e *Q={q1,q2,q3,…,qn}* por:

$$D(Q,P) = \sum_{k=1}^{N} q_k \log_2 \frac{q_k}{p_k}$$

O método de entropia mínima cruzada pode ser visto como uma extensão do método da entropia máxima cruzada, alocando valores estimados iniciais para todos p_i quando nenhuma informação esta disponível.

O algoritmo roda de uma forma que é considerado um processo de reconstrução da distribuição da imagem. A imagem segmentada *g(x, y)* será construída da seguinte maneira:

$$g(x,y) = \begin{cases} \mu 1, se\ f(x,y) < t \\ \mu 2, se\ f(x,y) \geq t \end{cases}$$

A imagem segmentada *g(x, y)* será determinada somente pela função *f(x, y)* que terá as variáveis desconhecidas *μ1, μ2* e *t*. Uma função critério deve ser usada para determinar os melhores valores para estas variáveis de modo que possam melhor satisfazer a *f*. Neste método, a função critério usada é a entropia cruzada. A função será associada com as funções mostradas acima, achando o valor de limiar *t* e a imagem final *G(x, y)*.

$$\mu_1(t) = \frac{\sum_{fi<t} fi}{N_1} \qquad \mu_1(t) = \frac{\sum_{fi\geq t} fi}{N_2} \qquad \partial(t) = \sum_{fi<t} f_i log\left(\frac{f_i}{\mu_1(t)}\right) + \sum_{fi\geq t} f_i log\left(\frac{f_i}{\mu_2(t)}\right)$$

O limiar é dado por:

$$t_0 = min(\partial(t))$$

2.15 Lloyd

A técnica de Lloyd [30] é considerada como sendo uma variação do algoritmo de Ridler-Calvard [26], com uma mudança na equação final que calcula o ponto de corte. A equação para o cálculo do ponto de corte de Lloyd é:

$$t = \frac{1}{2}(\bar{z_1} + \bar{z_2}) + \frac{\sigma^2}{(\bar{z_1} + \bar{z_2})}\log\left(\frac{n_2}{n_1}\right)$$

onde $\bar{z_1}$ e $\bar{z_2}$ são os valores das médias dos níveis de cinza, σ é a variância e n_2 e n_1 são a quantidade de pixels.

O método de Lloyd inclui probabilidades de distribuições de cinza e valores de vizinhos em seus cálculos. O algoritmo pode ser resumido nos seguintes passos:

1. Um valor inicial de limiar *t* é selecionado, que é encontrado através do cálculo do valor médio de pixel da imagem.
2. A imagem é segmentada em duas regiões, II(1) e II(2), usando o valor de ponto de corte encontrado em 1.
   - Pixels que têm valores baixos de níveis de cinza são alinhados com II(1) e o restante com II(2).
3. Valores médios de níveis de cinza ($\bar{z_1}$ e $\bar{z_2}$) são calculados para ambas as regiões, estes são então associados com o número de pixels n_2 e n_1 junto com as regiões II(1) e II(2), respectivamente.
4. O novo ponto de limiar é calculado usando os novos valores médios de nível de cinza e números de pixels, utilizando a mesma equação mostrada anteriormente:

$$t = \frac{1}{2}(\bar{z_1} + \bar{z_2}) + \frac{\sigma^2}{(\bar{z_1} + \bar{z_2})}\log\left(\frac{n_2}{n_1}\right)$$

5. Repetir os passos 2) 3) e 4) até que não ocorra grandes variações nos valores.

2.16 Limiarização Fuzzy Baseado na Distância de Mahalanobis

O algoritmo de Mahalanobis [28] usa limiarização fuzzy, um conceito já explorado nos algoritmos anteriores. Essa técnica estabelece dois valores U_0 e U_i para os dois níveis que irão fazer a separação dos níveis de cinza da imagem. Como mostrado anteriormente, estes dois valores representam a o grau de pertinência de um pixel em relação a uma das classes, objeto ou fundo.

$$\mu_0(t) = \frac{\sum_{j=0}^{t} j.h(j)}{\sum_{j=0}^{t} h(j)} \qquad \mu_b(t) = \frac{\sum_{j=t+1}^{255} j.h(j)}{\sum_{j=t+1}^{255} h(j)}$$

onde:
j é o nível de cinza e *h(j)* é o número de níveis de cinza que o tom *j* já tem na imagem.

O algoritmo inicia como qualquer outro que use limiarização por lógica fuzzy, obtendo os dois valores iniciais U_0 e U_i baseado na técnica fuzzy. O método em questão atualiza esses valores usando a distância de Mahalanobis, que é calculada usando o somatório de U_0 e U_i,. Como esse valor de somatório não se encaixa em um intervalo fixo (0 e 1), fazemos uma manipulação algébrica:

$$\overline{\mu_0}(t) = \frac{\mu_0(t)}{\mu_0(t)+\mu_b(t)} \qquad \overline{\mu_b}(t) = \frac{\mu_b(t)}{\mu_0(t)+\mu_b(t)}$$

O limiar final é obtido quando a atualização das duas classes de pertinência não trás mais mudanças significativas em relação aos valores que estão sendo atualizados. As seguintes equações são utilizadas para calcular os parâmetros e fazer as atualizações.

$$v_i = \frac{\sum_{j=0}^{255} j.h(j).\mu_i(j)^\tau}{\sum_{j=0}^{255} h(j).\mu_i(j)^\tau}, i = 1,2$$

Essa equação é utilizada para fazer o cálculo de v_b e v_0, que são os valores médios de nível de cinza do fundo e do objeto. A próxima equação é utilizada para fazer a atulização mencionada anteriormente. Os valores a serem atualizados são σ_0 e σ_b.

$$\sigma_i^2 = \frac{\sum_{j=0}^{L-1} \mu_i^\tau(j)h(j)(j-v_i)^2}{\sum_{j=0}^{L-1} \mu_i^\tau(j)h(j)}, \sigma > 1$$

A distância de Mahalanobis é calculada da seguinte forma:

$$d(j, v_i) = \frac{|j-v_i|}{\sigma_i}$$

As equações mostradas a seguir então fazem as atualizações de U_0 e U_i, usando a distância de Mahalanobis.

$$\overline{\mu_0}(j) = \frac{1}{1+\left[\frac{d(j,v_0)}{d(j,v_b)}\right]^{2/(\tau-1)}} \qquad \overline{\mu_b}(t) = 1 - \overline{\mu_0}(j)$$

Para as equações mostradas acima temos que v_i é o valor médio de tom de cinza do objeto ou fundo, α é o *fuzziness* devendo ser maior que 1 e $d(j, v_i)$ é a distância de Mahalanobis entre o nível de cinza e o nível médio correspondente.

Quando a diferença entre os valores de U_0 e U_i, não for mais significante, o valor do ponto de corte pode ser encontrado calculando a média entre as duas pertinências.

$$Tf = \frac{v_0+v_b}{2}$$

2.17 Matriz de Co-ocorrência

Assim como um histograma de uma imagem, a matriz de co-ocorrência [31] é uma forma de representar uma imagem e os valores dos seus pixels. Uma matriz de co-ocorrência irá prover uma visão especial dos valores encontrados em uma imagem. A matriz contém informações sobre a posição dos pixels em relação aos outros pixels da imagem. A imagem pode então ser percorrida verticalmente, horizontalmente ou diagonalmente à procura de pixels que diferem de

um valor *n* e que estão separados a uma distância *d*. Para exemplificar, se uma célula de posição (0,1) de uma matriz de co-ocorrência tiver um valor 10, isto quer dizer que existem 10 ocorrências na imagem em que os pixels com valores 0 e 1 estão separados de uma distância *d*. Se a imagem for analisada em todas as direções, uma grande quantidade de dados será gerada, o que não é ideal por diminuir a eficiência. A informação encontrada pode ser agrupada em grupos menores.

O algoritmo calcula o número de pixels na imagem separados de uma distância *d* em quatro direções: 0(direita), π/2(para cima), π(esquerda) e 3π/2(para baixo). Quando todas as direções são calculadas, as quatro matrizes (uma para cada direção) são somadas, o que irá gerar uma matriz de co-ocorrência final para a imagem.

Tomando *t* como o limiar, esse irá dividir a matriz em quatro seções. O valor de *t* pode ser escolhido de várias maneiras, uma delas sendo a Medida Condicional de Probabilidade, que escolhe um limiar que irá minimizar a soma dos dois últimos quadrantes da matriz de co-ocorrência (os quadrantes são obtidos como mencionado acima, o limiar irá dividir a matriz em quatro seções).

2.18 Limiarização por Entropia de Pun

O algoritmo de Pun [16] é mais um que faz uso da entropia da imagem, que é definida como sendo a medida de conteúdo informação na imagem. Pode então ser assumido que existem *x* possíveis símbolos e que o símbolo *i* irá ocorrer com uma probabilidade *p(y)*. A entropia para uma dada fonte de símbolos é então:

$$H(X) = -\sum_{i=0}^{255} p(i)\log(p(i)) \qquad \text{(Eq. 1)}$$

onde a entropia é medida em bit/símbolos.

A entropia associada com pixels pretos que foram submetidos a um limiar *t* é mostrada como sendo

$$H_b = -\sum_{i=0}^{t} p(i)\log(p(i))$$

A entropia para pixels brancos é a mesma:

$$H_w = -\sum_{i=t+1}^{255} p(i)\log(p(i))$$

O algoritmo encontra um limiar *t* que maximize a função $H = H_b + H_w$. Que seria maximizar

$$f(t) = \frac{H_t}{H_T}\frac{logP_t}{log(max\{p_0,p_1,\ldots,p_t\})} + \left[1 - \frac{H_t}{H_T}\right]\frac{log(1-P_t)}{log(max\{p_{t+1},p_{t+2},\ldots,p_{255}\})}$$

onde H_t é a entropia de pixels pretos com limiar *t*, H_T é a entropia total e P_t é a probabilidade de que um determinado pixel terá um valor menor ou igual à *t*.

$$H_t = -\sum_{i=0}^{255} p_i logp_i \qquad H_T = -\sum_{i=0}^{255} p_i logp_i \qquad P_t = \sum_{i=0}^{t} p_i$$

2.19 Limiarização por Entropia de Johanssen

Esse método, conhecido como Método da entropia de Johanssen [18], é baseado na entropia da imagem e seu objetivo principal é dividir uma imagem de tons de cinza em duas partes, obtendo uma imagem binária e minimizando as interdependências entre os níveis de cinza. Entropia é a medida de conteúdo de informação. A função da entropia para um dado símbolo *x* pode ser vista como na Equação 1.

Uma imagem pode ser considerada uma fonte de símbolos. O algoritmo é baseado no cálculo das entropias para preto *Sb(t)* e para branco *Sw(t)*. Isso acontece porque o método divide os níveis de cinza da imagem em dois níveis de cinza, preto e branco, criando uma imagem binária. O ponto de corte ideal para esta técnica será o valor de *t* que dará o valor mínimo para a soma *Sb(t)* + *Sw(t)*, onde:

$$S_w(t) = \log\left(\sum_{i=t+1}^{255} p_i\right) + \left(1 \Big/ \sum_{i=t+1}^{255} p_i\right)\left[E(p_t) + E\left(\sum_{i=t+1}^{255} p_i\right)\right]$$

e

$$S_w(t) = \log\left(\sum_{i=0}^{t} p_i\right) + \left(1 \Big/ \sum_{i=0}^{t} p_i\right)\left[E(p_t) + E\left(\sum_{i=0}^{t-1} p_i\right)\right]$$

onde *E(x)=-xlog(x)* e *t* é o valor de corte.

2.20 Limiarização por Entropia de Renyi

Esse método, conhecido como método da Entropia de Renyi [21], é um método global automático que é uma extensão de outras duas técnicas conhecidas como Kapur [17] e Chang [32]. Sua meta é otimizar uma função critério. A entropia de Renyi usa duas distribuições de probabilidade (objeto e fundo do objeto), derivadas das distribuições de cinza originais da imagem e inclui os métodos da soma da entropia máxima e correlação entrópica. Dado que *po, p1, p2,..., p255* são probabilidades de níveis de cinza da imagem, duas probabilidades são derivadas delas, *A1* e *A2*, para objeto e fundo do objeto, respectivamente. Associando a Entropia de Renyi com essas distribuições probabilísticas, é encontrada a entropia para cada distribuição, H_{A1} e H_{A2}. A Entropia de Renyi está mostrada abaixo:

$$H_T^{\alpha} = \frac{1}{1-\alpha} ln \sum_{k=0}^{255} \left(p_k{}^{\alpha}\right)$$

Para as distribuições individuais temos então:

$$H_{A_1}^{\alpha} = \frac{1}{1-\alpha} ln \sum_{k=0}^{t} \left(\frac{p_i}{p(A_1)}\right)^{\alpha} \qquad H_{A_2}^{\alpha} = \frac{1}{1-\alpha} ln \sum_{k=t+1}^{255} \left(\frac{p_i}{p(A_2)}\right)^{\alpha}$$

O valor de ponto de corte para este método é encontrado maximizando a soma de H_{A1} e H_{A2}, da seguinte forma:

$$t^*(\alpha) = ArgMax\left\{H_{A_1}^{\alpha}(t) + H_{A_2}^{\alpha}(t)\right\}$$

O limiar *t* é uma função de α, em termos práticos gera três valores, *t1, t2* e *t3*. O limiar final otimizado pode ser encontrado aplicando estes três valores na seguinte equação:

$$t_C^* = t_{[1]}\left[p(t_{[1]}) + \frac{1}{4}\omega\beta_1\right] + \frac{1}{4}t_{[2]}\omega\beta_2 + t_{[3]}\left[1 - p(t_{[3]}) + \frac{1}{4}\omega\beta_3\right]$$

O limiar final *t* será sempre encontrado entre t_1 e t_3.

2.21 SIS (*Simple Image Statistic*)

O algoritmo de SIS [33] depende somente da imagem em si, não fazendo uso do histograma como várias outras técnicas fazem. Basicamente, o algoritmo consiste na comparação de imagem original com duas máscaras de gradiente.

Esse método assume que uma imagem *I* é uma representação imperfeita de um objeto e seu plano de fundo. O objeto ideal seria composto de pixels de um nível de cinza "a" e o plano de fundo composto de um nível de cinza "b".

É admitido que *e(i, j)* é o máximo absoluto das máscaras de gradiente *s* e *t* que são aplicadas à imagem *I(i,j).* Tem-se assim:

$$S = \begin{array}{|c|}\hline 1 \\ \hline \times \\ \hline -1 \\ \hline \end{array} \qquad\qquad T = \begin{array}{|c|c|c|}\hline -1 & \times & 1 \\ \hline \end{array}$$

onde *x* é o operador *don´t care*, que quer dizer que esta posição da máscara não será levada em conta nos cálculos, podendo assumir qualquer valor, já que não tem relevância.

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{m}|a(i,j).e(i,j)|}{\sum_{i=1}^{n}\sum_{j=1}^{m}e(i,j)} = \frac{a+b}{2}$$

A seção da direita da equação é o ponto médio entre os níveis de cinza do objeto e do plano de fundo. O limiar então é o ponto médio. Dessa forma, pode ser demonstrado que o algoritmo SIS calcula o limiar usando:

$$T = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}|a(i,j).e(i,j)|}{\sum_{i=1}^{n}\sum_{j=1}^{m}e(i,j)}$$

2.22 Limiarização Iterativa de Thrussel

A técnica de Thrussel [34] usa seleção iterativa para selecionar o limiar. Como visto anteriormente, seleção iterativa irá constantemente atualizar o limiar usando informações da imagem até que não exista mais mudança no valor do limiar, chegando assim ao ponto de corte final.

O valor do limiar inicial é metade da escala de cinza em uso. Informações sobre a imagem são coletadas, em sua maior parte estatísticas acerca das regiões pretas e brancas. A média da escala de cinza para pixels encontrados abaixo do limiar é T_b e para os acima, T_o. O novo limiar pode então ser calculado por

$$T_k = \frac{T_b + T_0}{2}$$

O processo é repetido até que não ocorram mais mudanças entre o limiar atual e o que está sendo calculado.

Esse algoritmo apresenta bom desempenho quando implementado em hardware, podendo também ser implementado em software, mas com pequenas mudanças. Quando implementado em software, o uso do histograma da imagem se faz necessário.

$$T_k = \frac{\sum_{i=0}^{T_{k-1}} i.h(i)}{2.\sum_{i=0}^{T_{k-1}} h(i)} + \frac{\sum_{T_{k-1}+1}^{255} i.h(i)}{2.\sum_{T_{k-1}+1}^{255} h(i)}$$

A equação mostrada acima é usada para implementação via software. As duas metades da equação são iguais, mudando somente os limites dos somatórios.

2.23 Limiarização por Entropia de Wu-Lu

O método de WuLu [20] é mais um algoritmo de limiarização por entropia. Similar ao que é feito no algoritmo de Pun, a entropia da imagem pode ser dividida em duas seções diferentes, objeto e fundo, cada um com sua entropia distinta, $H_o(t)$ e $H_b(t)$.

De acordo com os métodos descritos por Kapur, Sahoo e Wong, a segmentação ótima tem seu limiar dado pelo nível de cinza que maximizaria a soma das entropias do objeto e do fundo. No entanto, WuLu aplica o algoritmo a imagens de ultra-som, que são imagens com baixo contraste e que, normalmente, têm a informação mesclada com o fundo e com ruído da imagem. Para resolver esses problemas, WuLu propôs uma abordagem que minimiza a diferença entre as entropias do objeto e do fundo.

$$t = ArgMin|\, H_o\,(t) - H_b\,(t)|$$

onde:

$$H_0(t) = -\sum_{i=0}^{t} \frac{p_i}{P_T} ln\left(\frac{p_i}{P_T}\right) \qquad H_b(t) = -\sum_{i=0}^{L-1} \frac{p_i}{1-P_T} ln\left(\frac{p_i}{1-P_T}\right) \qquad P_t = \sum_{i=0}^{t} p_i$$

2.24 TholdH

TholdH é um método de limiarização por entropia proposto por Mello [15]. O conceito de entropia foi explorado nas discussões de outros algoritmos acima, mas será mostrada novamente para um melhor entendimento. Se uma imagem for considerada como uma fonte de símbolos, então existem *n* possíveis símbolos *s* que têm uma probabilidade *p(s)* de ocorrerem. A entropia é então calculada como como na Equação 1.

Pun mostra que o limiar *t* é tal que maximiza a soma da entropia dos pixels pretos, H_b, e a soma da entropia dos pixels brancos, H_w.

$$H_b = -\sum_{i=0}^{t} p(i)\log(p(i))$$

$$H_w = -\sum_{i=t+1}^{255} p(i)\log(p(i))$$

$$H = H_b + H_w$$

O algoritmo propõe uma troca na base logarítmica da entropia. O algoritmo procura pela cor mais freqüente na imagem e utiliza este valor como seu limiar inicial. Como este método foi concebido para documentos históricos, é tomado como regra que a cor mais freqüente pertence ao plano de fundo, ou papel. Estudos da Xerox mostraram que, na maioria dos documentos escritos, a tinta corresponde a aproximadamente 10% dos pixels na imagem, o restante pertence ao papel. O valor de limiar inicial é usado para avaliar as entropias dos pixels brancos e pretos. Usando um acervo de 500 documentos, os autores desse algoritmo determinaram empiricamente que os valores de entropia ficam situados na faixa entre 0,2 e 0,4 para uma base logarítmica de 256 (que corresponde ao número máximo de níveis de cinza presentes em uma imagem). Como ainda existe uma grande probabilidade de interseção entre as classe quando separando tinta do papel, a entropia é decomposta em duas projeções H_w e H_b. Constantes *mw* e *mb* foram usadas para aumentar ou diminuir a influência de cada projeção no valor final do limiar. Essas constantes foram determinadas empiricamente.

- Se $H =< 0,25$, então $mw=2$ e $mb =3$
- Se $0,25 =< H =< 0,30$, então $mw=1$ e $mb=2,6$
- Se $H>=0,3$, então $mw= mb = 1$

O valor final do limiar, *t*, pode então ser encontrado como:

$$t = mw.Hw + mb.Hb$$

2.25 ThROC e ThROC2

Esses dois algoritmos estão agrupados em decorrência de suas semelhanças. Eles são derivados do *TholdH,* discutido anteriormente. Assim como o *TholdH,* esses dois métodos foram propostos por Mello em [35]. Os dois algoritmos usam o mesmo conceito de entropia, a mesma mudança de base e o uso de constantes para controlar a influência das projeções. *Tholdh* estava mostrando resultados fracos quando aplicado a imagens com interferência frente-verso. Para corrigir esse problema, o valor de limiar agora sofre correções baseadas em curvas ROC.

Receiver Operating Characteristic, ou ROC, surgiu da Teoria da Detecção de Sinais. Uma curva ROC mostra a relação entre a probabilidade de detecção (*probability of detection* - PD) e a probabilidade de falso alarme (*probability of false alarm* - PFA) para diferentes valores de limiar para uma dada imagem. Essa é uma maneira de classificar os limiares e de mostrar sua taxa de detecção, neste caso a meta é detectar tinta. A probabilidade de detecção é a probabilidade de detectar um pixel de tinta quando este é de fato tinta. A probabilidade de falso alarme é a probabilidade de que o pixel seja detectado como tinta quando na verdade ele não é.

O *ThROC* inicia da mesma forma que o *TholdH*, quando o valor de limiar *t* é determinado baseado na cor mais freqüente da imagem, uma curva ROC será construída para avaliar o quão acertado é o limiar. O limiar inicial é usado para construir uma matriz binária com as mesmas dimensões da imagem original. Cada célula da matriz é colocada como verdadeiro se seu pixel correspondente na imagem original for igual a *th.* Isso permite a construção da curva PD vs. PFA,

ou curva ROC. Para imagens de documentos, como mencionado anteriormente, valores de tinta representam aproximadamente 10% do total de pixels. Para esses documentos, a curva ROC tem a forma de uma função degrau. A curva ROC correta deve crescer para 1 quando os valores de PD forem próximos a 0,9. O valor inicial do limiar é então traçado nessa curva e deve estar na vizinhança do valor de PD de 0,9 para ser o limiar ótimo, senão esse valor é incrementado ou decrementado até que atinja o valor ótimo na curva ROC.

ThROC2 é uma variação do *ThROC* com uma pequena diferença somente no processo de correção. Foi observado que o *ThROC* não tem boa velocidade de processamento. Para melhorar a velocidade de processamento, uma mudança foi feita na correção do limiar por curvas ROC. Inicialmente, a correção era feita de um valor unitário por vez, independente de quão distante o limiar inicial estava do ponto procurado na curva ROC. Nessa segunda versão do algoritmo, o valor do limiar é incrementado levando em consideração sua distância para chegar ao ponto ideal. Se estiver longe, o incremento terá um valor maior, economizando tempo de processamento e melhorando o desempenho do algoritmo.

Maiores detalhes sobre o *ThROC* são tratados no Capítulo 3 desta monografia.

Capítulo 3

Novo Algoritmo de Binarizacão

Neste Capítulo, apresentamos um novo algoritmo de binarização para imagens de documentos históricos. Baseado no *ThROC*, esse novo algoritmo apresenta um grande ganho em termos de desempenho.

3.1 Algoritmo de Limiarização Baseado na Entropia

Um algoritmo de limiarização foi proposto por Mello *et al.*, denominado *ThROC* [35], e discutido na Seção 2.25. Apresentamos um breve resumo nesta Seção.

Similar ao algoritmo de Pun, o processo de limiarização é iniciado avaliando *Hb* e *Hw*. Essas variáveis são calculadas quebrando a entropia, *H*, em duas partes, tendo como ponto de corte a cor mais freqüente na imagem. Como estamos trabalhando com documentos, é normal aceitar que a cor mais freqüente pertença ao fundo da imagem, já que a maior parte de um documento é papel e não tinta. A base logarítmica é tomada como o produto da altura e da largura da imagem (*i.e.*, sua área). Como dito anteriormente, essa mudança de base não afeta a definição de entropia.

Hb e *Hw* podem ser vistos como projeções ortogonais do vetor entropia e *H,* a qual pode ser expressa como: $H = Hb + Hw$.

O valor do limiar é obtido através da entropia da imagem, mas mudando-o através das contribuições de *Hb* e *Hw*. Isso é feito pela definição de duas constantes multiplicativas *mw* e *mb*. Essas constantes estão relacionadas com a classe do documento. Na classe 1 encontramos documentos que possuem pouco texto ou texto muito claro; a classe 2 agrupa documentos com muitas áreas escuras; a classe 3 são os outros casos sem características específicas.

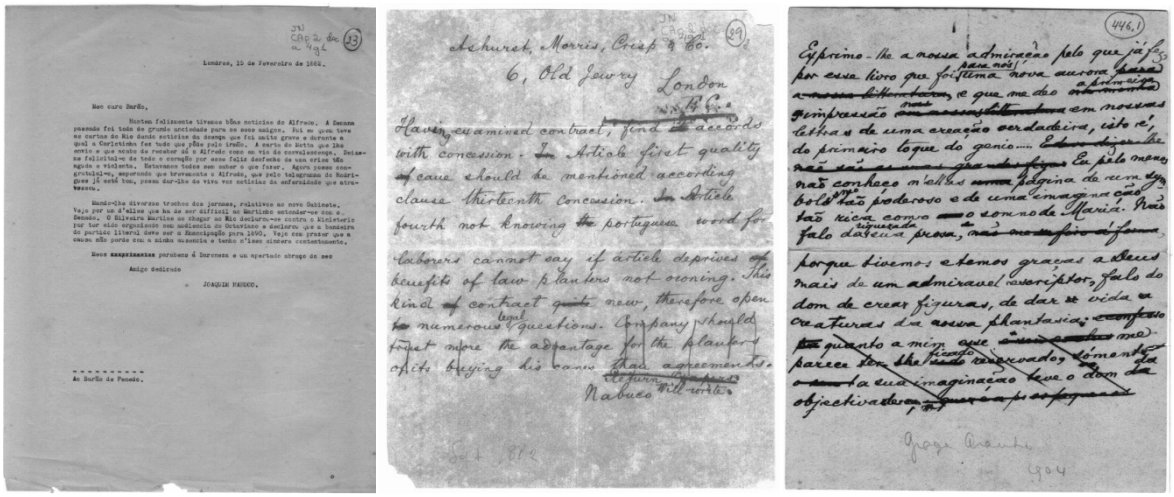
Para cada uma das classes temos:

- Classe 1: $H \leq 0.25$, então $mw = 2$ e $mb = 3$;
- Classe 2: $H \geq 0.30$, então $mw = mb = 1$;
- Classe 3: $0.25 < H < 0.30$, então $mw = 1$ e $mb = 2.6$.

O valor do limiar, *th*, é definido por:

$$th = mw * Hw + mb * Hb$$

A Figura 5 mostra um exemplo de imagem para cada classe citada acima.



a) b) c)

Figura 5. Exemplos de imagens das classes descritas em ThROC: (a) classe 1, (b) classe 2 e (c) classe 3.

A imagem é percorrida e cada pixel *i* com cor *graylevel[i]* se torna branco se:

$$(graylevel[i]/256) \geq th$$

caso contrário, é convertido para preto. Os valores das constantes *mw* e *mb* foram definidos de forma heurística, analisando a qualidade da imagem por inspeção visual e pela taxa de acerto de ferramentas de OCR (*Optical Character Recognition*) para documentos datilografados. Para os casos da classe 2, a entropia em si serve como valor de recorte (porque *th* = *Hw* + *Hb* = *h*). Para as outras duas classes, um aumento no valor da entropia é aplicado aumentando a importância de *Hw* ou *Hb*. Em ambos os casos, a multiplicação pelas constantes aumenta o valor de *th* quando comparado somente com o valor da entropia.

O valor de limiar definido pelo algoritmo baseado na entropia não é sempre o melhor valor. Para ajustar este valor, é usada uma curva ROC (*Receiver Operating Characteristic*) [36]. Esta técnica normalmente é utilizada em análises médicas onde os testes podem gerar respostas verdadeiros positivos (*true positives* – TP), falsos positivos (*false positives* – FP), verdadeiros negativos (*true negatives* – TN) ou falsos negativos (*false negatives* – FN). No caso específico de documentos históricos, TP representa a probabilidade que um pixel de tinta seja classificado verdadeiramente como tal. FP representa os falsos positivos, um pixel do plano de fundo é classificado pelo algoritmo como sendo tinta. Analogamente a TP e FP temos TN e FN. Uma curva ROC mostra a relação entre a probabilidade de detecção (PD) e a probabilidade de falso alarme (PFA) para diferentes valores de corte. Um limiar de detecção é variado sistematicamente para examinar o desempenho do modelo para diferentes limiares. A variação de limiares produz diferentes classificadores com diferentes valores de PD e PFA. Ao utilizar os valores de PD e PFA para diferentes valores de limiar, pode-se chegar a uma curva ROC. A Figura 6 mostra uma curva ROC típica.

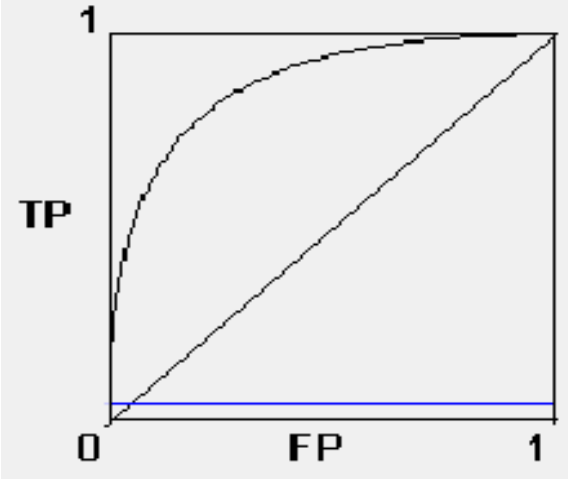


Figura 6. Exemplo de curva ROC.

Em *ThROC* [35], esse valor de limiar é ajustado usando curvas ROC. O valor de corte inicial *th* é usado para definir uma matriz binária *M* do mesmo tamanho da imagem de entrada. Cada célula da matriz recebe o parâmetro *verdadeiro* se o pixel correspondente na imagem de entrada *IM* é igual a *th*. Isso produz a curva de Probabilidade de Detecção (PD) versus Probabilidade de Falso Alarme (PFA), que corresponde à curva ROC, de acordo com o Algoritmo 1.

Algoritmo 1: Cálculo dos vetores PD e PFA.

n1 ← o número de elementos igual a th na imagem IM
n0 ← o número de elementos que são diferentes de *th* na imagem
for t = 0 **to** 255
    PD(t) ← $\Sigma_{i=1..Nrow} \Sigma_{j=1..Ncol}$ (IM(i,j)>t **and** M(i,j))/n_1;
    PFA(t) ← $\Sigma_{i=1..Nrow} \Sigma_{j=1..Ncol}$ (IM(i,j)>t **and** ¬M(i,j))/n_0;
end for

Para o caso de documentos históricos, a curva ROC definida por esse algoritmo é uma função degrau que tem seus valores máximos iguais a 1 para ambos os eixos. Diferentes pontos de cortes iniciais definem curvas ROCs diferentes. A Figura 7 ilustra um exemplo do comportamento da curva.

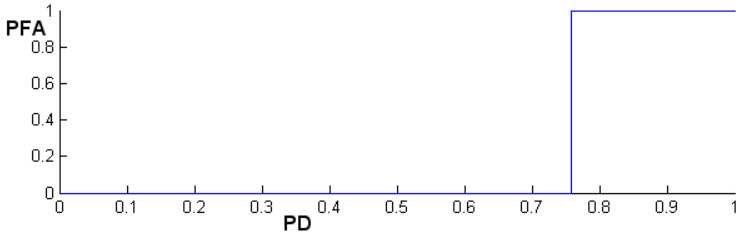


Figura 7. Comportamento da curva ROC dado o algoritmo usado aplicado a documentos históricos.

Foi observado que a porcentagem de tinta corresponde a aproximadamente 10% do total de um documento. Embora esse valor não seja o mesmo para todos os documentos, ele é aceito

para a maior parte dos documentos no acervo. O valor correto da curva ROC deverá então crescer para 1 quando PD tiver um valor por volta de 0,9. Para obter isso, diferentes valores de *th* devem ser usados. Isso cria matrizes *M* diferentes gerando novas curvas PD versus PFA. Se a curva cresce para 1 com o valor de PD menor que 0,9, então o valor inicial deve ser decrementado de uma unidade; caso contrário, deve aumentar uma unidade. O ponto onde a curva PD cresce para 1 é chamado de *PD_point*. Esse método com curvas ROC produziu imagens de melhor qualidade que o método anterior.

3.2 Variação Para Melhoria de Desempenho do Algoritmo

Uma desvantagem do algoritmo discutido na Seção anterior é seu alto consumo de tempo para processamento. Como o sistema tem de analisar cada imagem gerada por cada novo valor de corte, o processamento pode durar vários minutos até chegar ao limiar ideal. Para aumentar esse desempenho, uma estratégia é proposta: tentamos aproximar mais o valor de corte inicial do valor final. Quanto mais perto o valor inicial estiver do valor final, menos tempo será necessário para atingir o valor correto. Para isto, utilizamos um novo algoritmo [37] para definir o valor de corte inicial. Ao invés de usar o algoritmo baseado na entropia, propomos utilizar o algoritmo de porcentagem de preto. Esse é um algoritmo bastante simples que trabalha unicamente com o histograma da imagem, o que o torna muito eficiente. A Figura 8 mostra um histograma de uma imagem de um documento histórico.

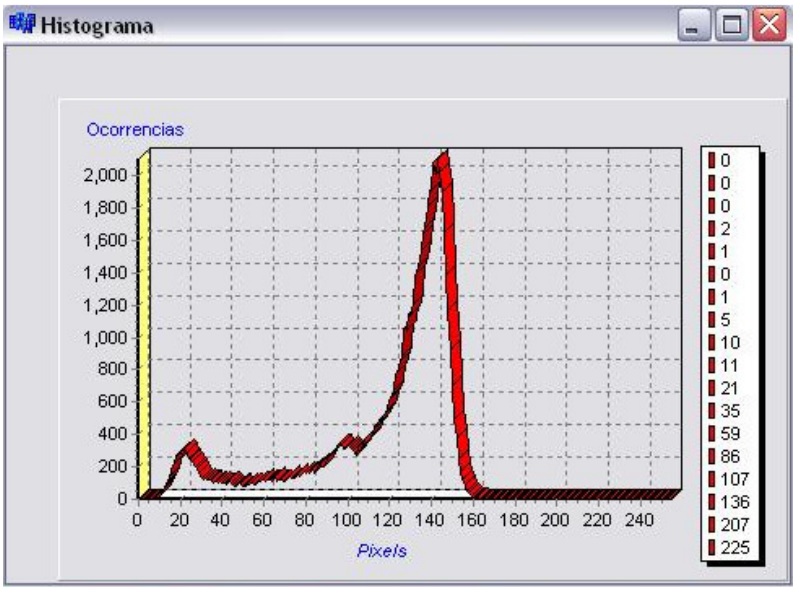


Figura 8. Histograma de uma imagem de documento histórico.

Como explicado anteriormente, consideramos que um documento é composto 10% de tinta e os 90% restantes pertencem ao papel. Um valor de corte é procurado que gere uma imagem com esta porcentagem de pixels pretos. Este seria o valor de corte inicial.

Um ponto de corte define que todas as cores abaixo dele são convertidas para preto. Assim, o algoritmo procura um ponto que converta para preto não mais que 10% dos pixels da

imagem. Essa procura acontece linearmente, a partir to primeiro tom da imagem. No nosso caso, o tom zero.

O uso da porcentagem de preto aproximou o valor de corte inicial do valor de corte final. Isso melhorou ainda mais o processamento do algoritmo. O tempo para processar um documento inteiro caiu significativamente. O tempo de processamento do algoritmo caiu em média para 98,7% do tempo do algoritmo original.

A correção por curvas ROC ainda se faz necessário, visto que o valor de corte inicial é corrigido de uma maneira mais apropriada. No entanto, agora o limiar converge para seu valor final mais rapidamente que antes. É importante ressaltar que essas mudanças só afetam o tempo de processamento do algoritmo original, já que os valores de corte e conseqüentemente as imagens finais são os mesmos de antes.

Ambas as versões do algoritmo foram implementadas no MatLab e estão disponíveis em: *http://www.dsc.upe.br/~recpad/site_hist/thresholding*

Capítulo 4

Resultados

A diminuição do tempo de processamento é facilmente medida. Para um conjunto de 500 imagens de teste, o processo de limiarização levou em média 3,2 segundos por imagem usando o novo algoritmo de porcentagem de preto com correção por curvas ROC implementado em C++ pelo mesmo programador e sem otimização de código. As imagens têm tamanho médio de 1.140 x 1.800 pixels. O algoritmo também gerou imagens de melhor qualidade que outros algoritmos de binarização testados.

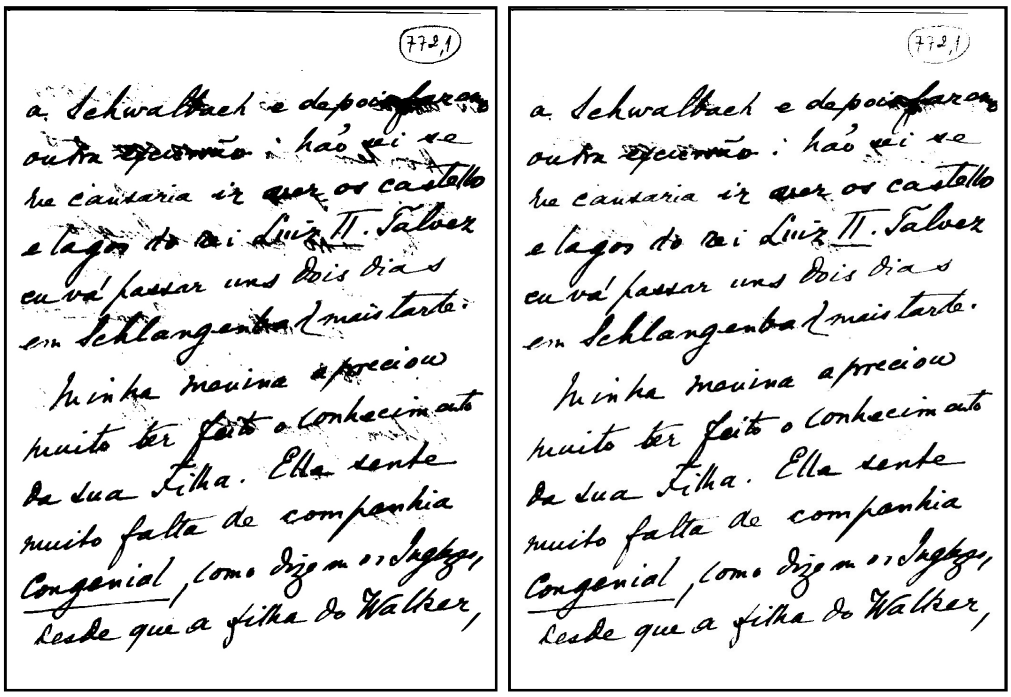
As imagens binarizadas são armazenadas em formato GIF (próprio para imagens binárias) e têm tamanho médio de 40 KBytes.

As Figuras 9 e 10 apresentam duas amostras da aplicação do algoritmo, o valor de corte final e o tempo de processamento. A Figura 9 apresenta o resultado da aplicação do algoritmo aplicado na imagem apresentada na Figura 1c. A imagem da Figura 1c é um dos documentos que têm interferência frente-verso. As Figuras 9 e 10 apresentam uma versão ideal para esses documentos. O uso dessas imagens está explicado no texto.

Para avaliar quantitativamente o desempenho do algoritmo, imagens ideais foram geradas manualmente. Essas são imagens de dois níveis de cor, com o valor de corte definido por inspeção visual. Essas imagens ideais são usadas para comparação com imagens produzidas por outros algoritmos de binarização. Avaliamos verdadeiros positivos (TP - o número de pixels corretamente classificados como tinta), verdadeiros negativos (TN – número de pixels corretamente classificados como papel), falsos positivos (FP – número de pixels que fazem parte do papel, mas são classificados erradamente como tinta) e falsos negativos (FN – número de elementos de tinta classificados como papel). Com esses valores, as seguintes métricas podem ser usadas [36]:

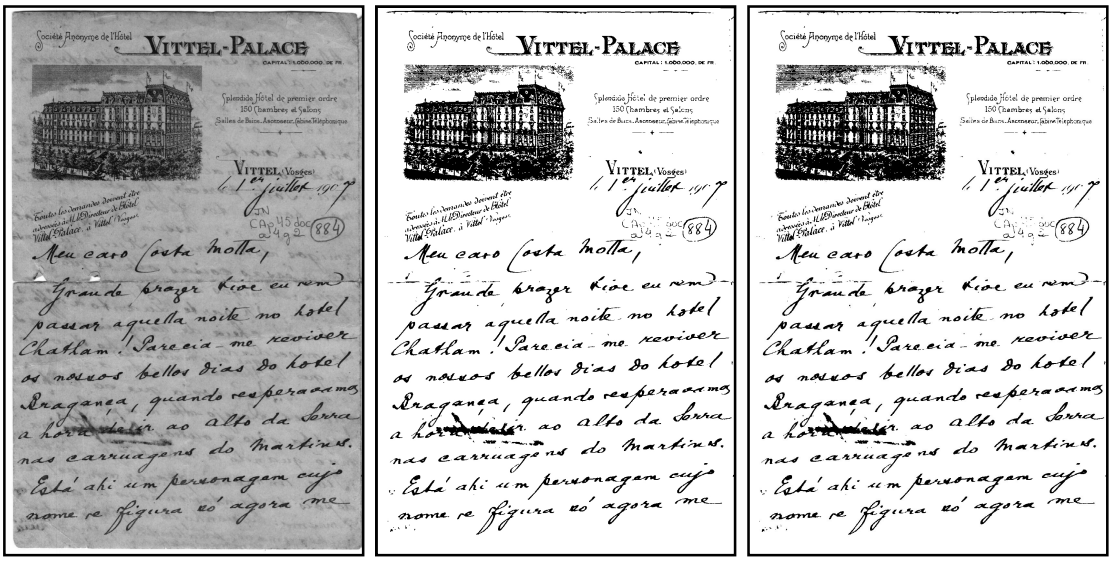
$$\text{Precision} = TP/(TP + FP)$$
$$\text{Recall} = \text{Sensitivity} = TP/P;$$
$$\text{Accuracy} = (TP + TN)/(P + N);$$
$$\text{Specificity} = TN/N$$

onde $P = TP + FN$; $N = FP + TN$.



a) b)

Figura 9. a) Versão binarizada de uma amostra de documento com interferência frente-verso gerado pelo novo algoritmo usando porcentagem de preto e curvas ROC (th =68; tempo de processamento = 2,2s, imagem com 1.042 x 1.364 pixels). b) A imagem binária ideal para este documento gerada manualmente para propósito de comparação.



a) b) c)

Figura 10. a) Amostra de documento e b) sua versão binarizada gerada pelo novo algoritmo (th=89, tempo de processamento=4,3s, imagem com 1.106 x 1.684 pixels). c) A imagem ideal para este documento.

As Tabelas 1 e 2 apresentam os valores de *precision, recall, accuracy* e *specificity* para as imagens mostradas nas Figuras 1a e 1c após serem limpas e comparadas com suas versões geradas pelo algoritmo proposto. O melhor algoritmo deve atingir os valores dessas quatro métricas mais próximos de 1 ao mesmo tempo. Não estão incluídos TholdH e ThROC pois estes

apresentam o mesmo ponto de corte final que o novo algoritmo, a diferença sendo a velocidade de processamento.

Tabela 1. Métricas para a avaliação do desempenho de algoritmos de binarização aplicados na Figura 1a.

Algoritmo	Figura 1a			
	Precision	*Recall*	*Accuracy*	*Specifitiicy*
Brink	1,2e-5	1	0,8281	0,8281
Fisher	0,936	0,866	0,9641	0,9865
C Means	0,9918	0,543	0,8552	0,9979
Huang	0,9167	0,89	0,9661	0,9826
Yager	0,9518	0,837	0,9598	0,9897
Seleção Iterativa	1	0,735	0,9379	1
Johannsen	1	0,18	0,2151	1
Kapur	1	0,697	0,9254	1
Kittler	1,2e-5	1	0,8281	0,8281
Li-Lee	0,7934	0,945	0,9566	0,9585
Tom de Cinza Médio	0,977	0,742	0,9377	0,9949
Otsu	1	0,807	0,9588	1
Porcentagem de Preto	0,5916	0,994	0,9292	0,9218
Pun	1	0,301	0,6013	1
Renyi	0,9832	0,690	0,9213	0,9962
Two Peaks	0,9948	0,216	0,3788	0,9957
Wu-Lu	0,9709	0,776	0,9468	0,9936
Ye-Danielsson	0,9226	0,883	0,9658	0,9838
Novo Algoritmo	**0,6292**	**0,986**	**0,9348**	**0,9284**

Baseado nas Tabela 1 e 2, o novo algoritmo é o que atingiu os mais altos valores para todas as quatro métricas analisando ambas as imagens. Seu valor de *precision* diminuiu para a Figura 1c na medida que alguns pixels de fundo foram considerados como sendo tinta, como pode ser visto na Figura 9b. Precisamos também avaliar o valor de *precision* equivalente a 1 em alguns algoritmos. Pode ser percebido que quando uma imagem é completamente preta (como as geradas pelos algoritmos de Brink ou Johannsen) a sua comparação com a imagem limpa de referência resulta que todos os pixels pretos foram classificados como tal. Para a real análise do desempenho do algoritmo precisamos verificar as outras métricas. O novo algoritmo tem uma melhor resposta para *recall* e *accuracy* do que qualquer outro algoritmo. O algoritmo de Kittler se aproximou do desempenho do nosso algoritmo proposto nessas duas métricas, mas sua medida de *precision* é zero; na verdade o algoritmo de Kittler produziu uma imagem completamente branca que classifica corretamente todos os elementos do papel, mas erra todos os pixels relativos à tinta.

Tabela 2. Métricas para a avaliação do desempenho de algoritmos de binarização aplicados na Figura 1c.

Algoritmo	Figura 1c			
	Precision	*Recall*	*Accuracy*	*Specifiticy*
Brink	1	0,096	0,1190	1
Fisher	1	0,507	0,9088	1
C Means	1	0,533	0,9178	1
Huang	1	0,507	0,9088	1
Yager	1	0,096	0,1204	1
Seleção Iterativa	1	0,498	0,9056	1
Johannsen	1	0,096	0,1191	1
Kapur	1	0,466	0,8925	1
Kittler	3e-5	0,906	0,9063	1
Li-Lee	1	0,096	0,1173	1
Tom de Cinza Médio	1	0,257	0,7291	1
Otsu	1	0,507	0,9088	1
Porcentagem de Preto	1	0,771	0,9721	1
Pun	1	0,166	0,5281	1
Renyi	1	0,153	0,4798	1
Two Peaks	1	0,135	0,3994	1
Wu-Lu	1	0,641	0,9475	1
Ye-Danielsson	1	0,498	0,9056	1
Novo Algoritmo	**0,9027**	**0,831**	**0,9736**	**0,9898**

A Tabela 3 faz uma comparação entre os tempos de processamento do novo algoritmo por porcentagem de preto e correção por curvas ROC em relação ao algoritmo anterior, o *ThROC2*, baseado em entropia e correção por curvas ROC. Foram analisadas cinco imagens, cada uma com um tamanho médio de 1000 x 1700 pixels. A figura 11 apresenta exemplos das imagens utilizadas nesta comparação de tempo de processamento.

Tabela 3. Avaliação de tempo de processamento entre o novo algoritmo e *ThROC2*.

Imagem	**Tempo do Novo Algoritmo**	Tempo de *ThROC2*	Tempo do novo algoritmo em relação a *ThROC2*
1	6,3930s	149,4200s	4,28%
2	9,1490s	40,5540s	22,56%
3	9,4040s	341,9670s	2,75%
4	9,3140s	32,0550s	29,06%
5	9,3240s	201,9570s	4,62%

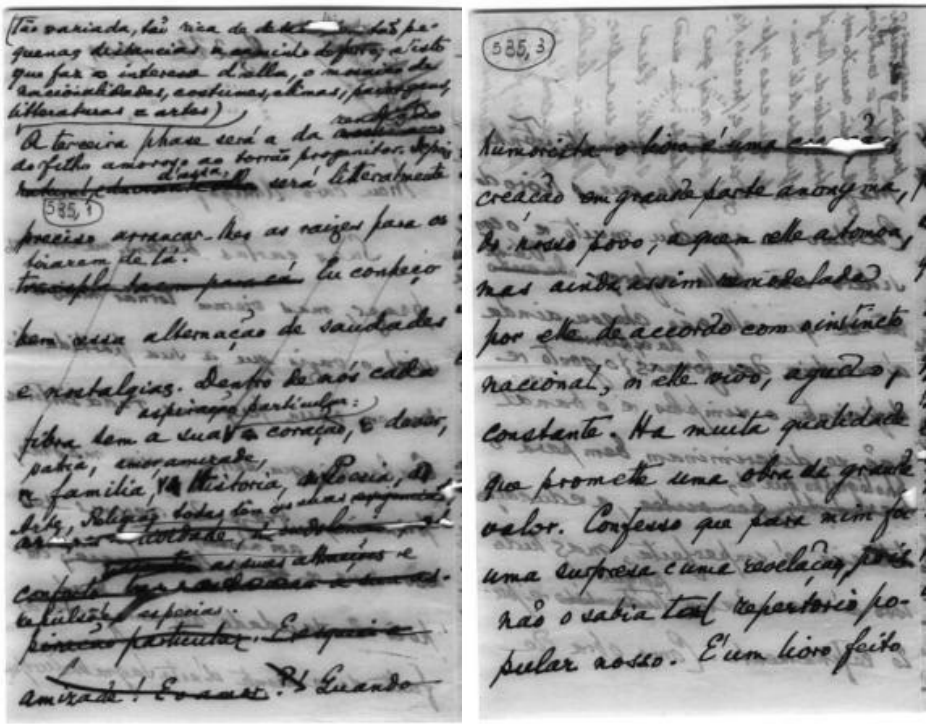


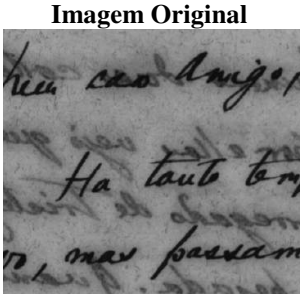
Figura 11. Exemplos das imagens usadas no teste comparativo de tempo entre o novo algoritmo e o *ThROC2*. Cada imagem tem 1098 x 1708 pixels.

Baseado na Tabela 3, podemos observar que o novo algoritmo diminui consideravelmente o tempo de processamento em relação a *ThROC2*. Em alguns casos, levando menos que 5% do tempo que *ThROC2* levou para processar uma mesma imagem. Nesses testes, os algoritmos foram implementados em Matlab pelo mesmo programador e sem otimização de código.

Apresentamos na Figura 12 imagens resultantes do processamento dos algoritmos apresentados anteriormente. Foram escolhidas duas imagens para os processamentos: a primeira é uma imagem manuscrita com interferência frente-verso; a segunda imagem é um documento datilografado com uma boa qualidade. É possível observar uma variação significativa quanto aos resultados. Alguns algoritmos apresentam imagens binarizadas de boa qualidade, filtrando bem as interferências. Outros algoritmos não têm um desempenho bom. Alguns geram imagens pretas, como Kittler-Yan e Fisher-LAT. Outros geram imagens brancas, como Brink. As variações ocorrem devido aos pontos de cortes diferentes encontrados por cada algoritmo. Alguns dos algoritmos em questão têm uma aplicação bem específica, como o Cseke que foi desenvolvido para imagens de células sanguíneas, o que pode explicar o rendimento menor destes algoritmos para a aplicação de documentos históricos.

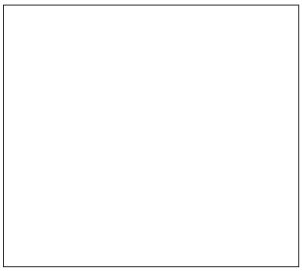
Imagem 1

Imagem Original



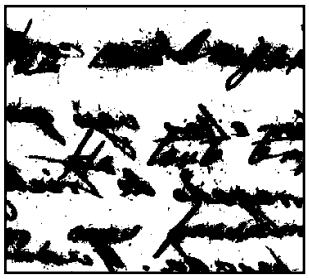
Ali-Silvey (J-Divergente)

Brink

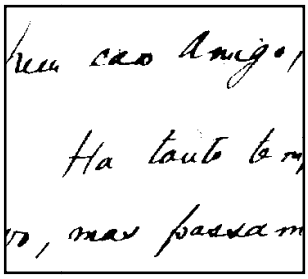


Chehikian

C-Means



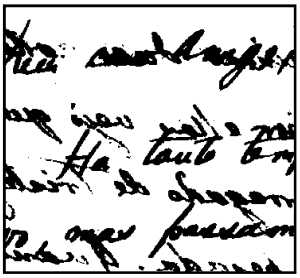
Cseke



Ye-Danielsson

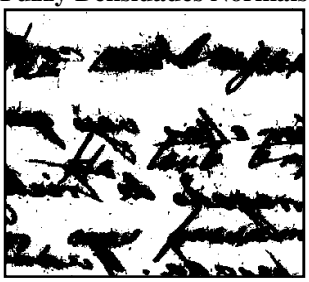
Fuzzy Huang

Fisher

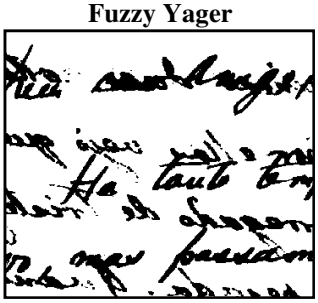


Fisher-Lat

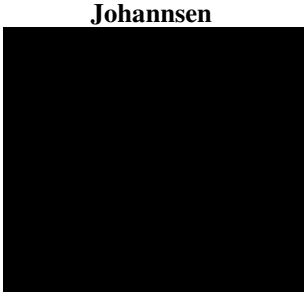
Fuzzy Densidades Normais



Fuzzy Yager

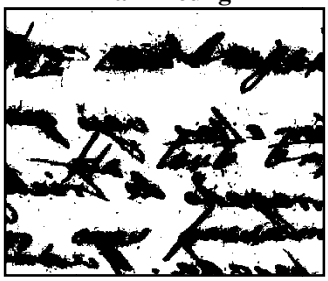


Johannsen

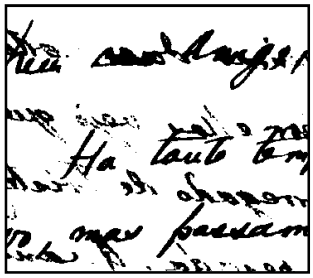


Kittler-Yan

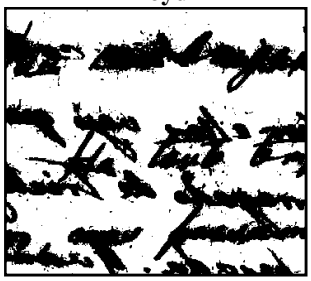
Lam-Leung



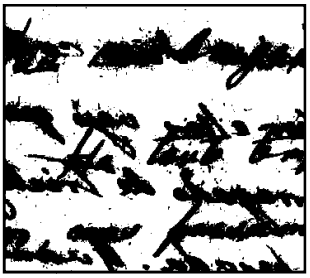
Li-Lee



Lloyd

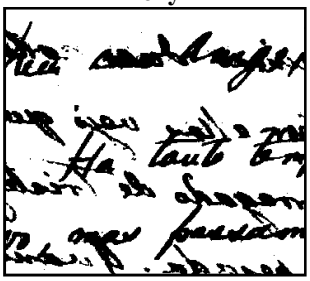


Mahalanobis

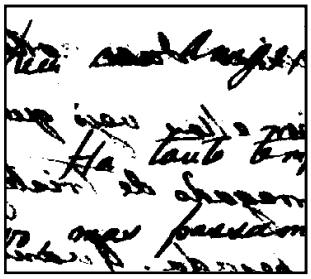


Pun

Renyi

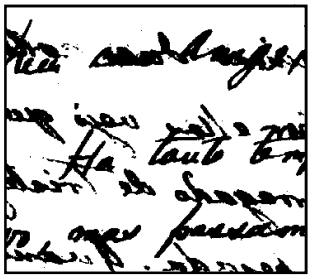


Ridler-Calvard

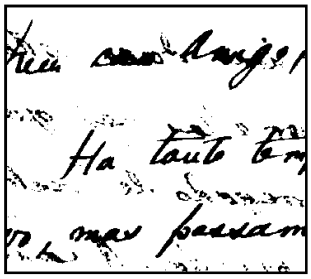


SIS

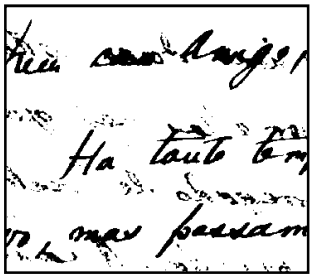
TholdH



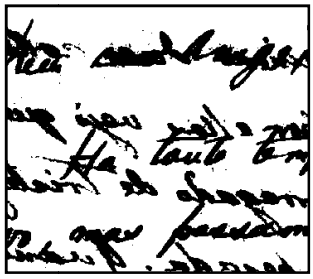
ThROC



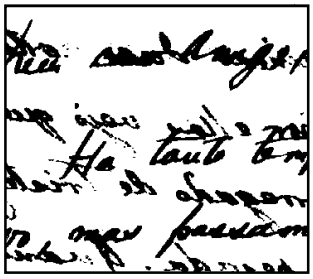
ThROC2



ThRussel



Wu-Lu



Co-Ocorrência

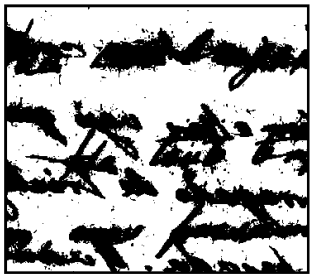
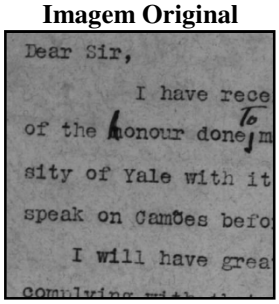


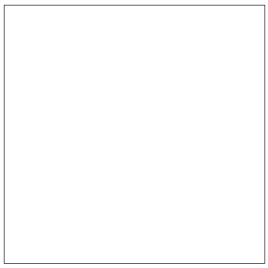
Imagem 2

Imagem Original

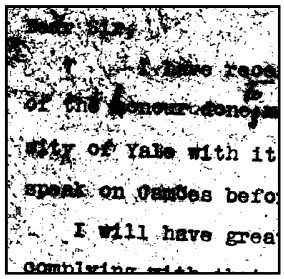


Ali-Silvey (J-Divergente)

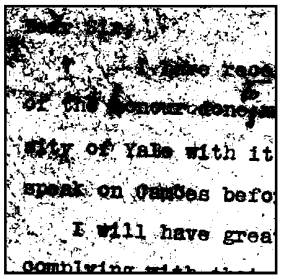
Brink



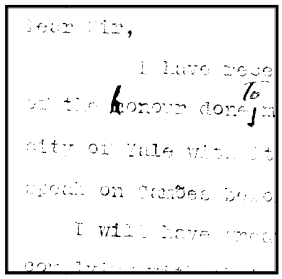
Chehikian



C-Means

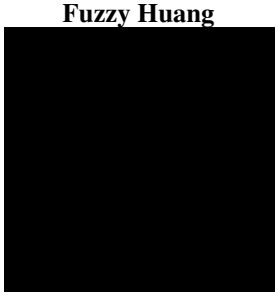


Cseke

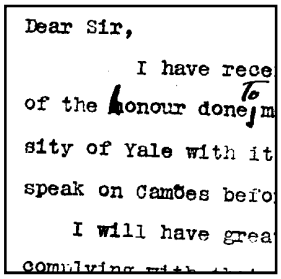


Ye-Danielsson

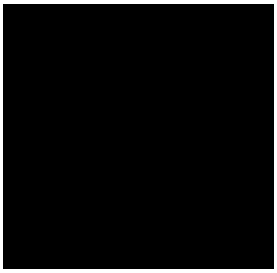
Fuzzy Huang



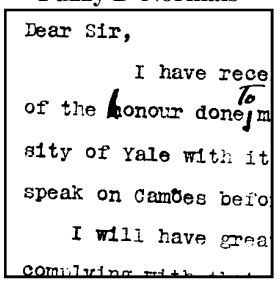
Fisher



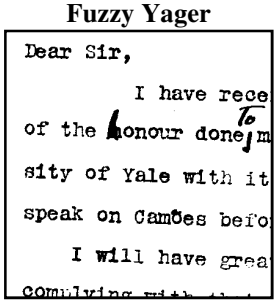
Fisher-Lat



Fuzzy D-Normais



Fuzzy Yager



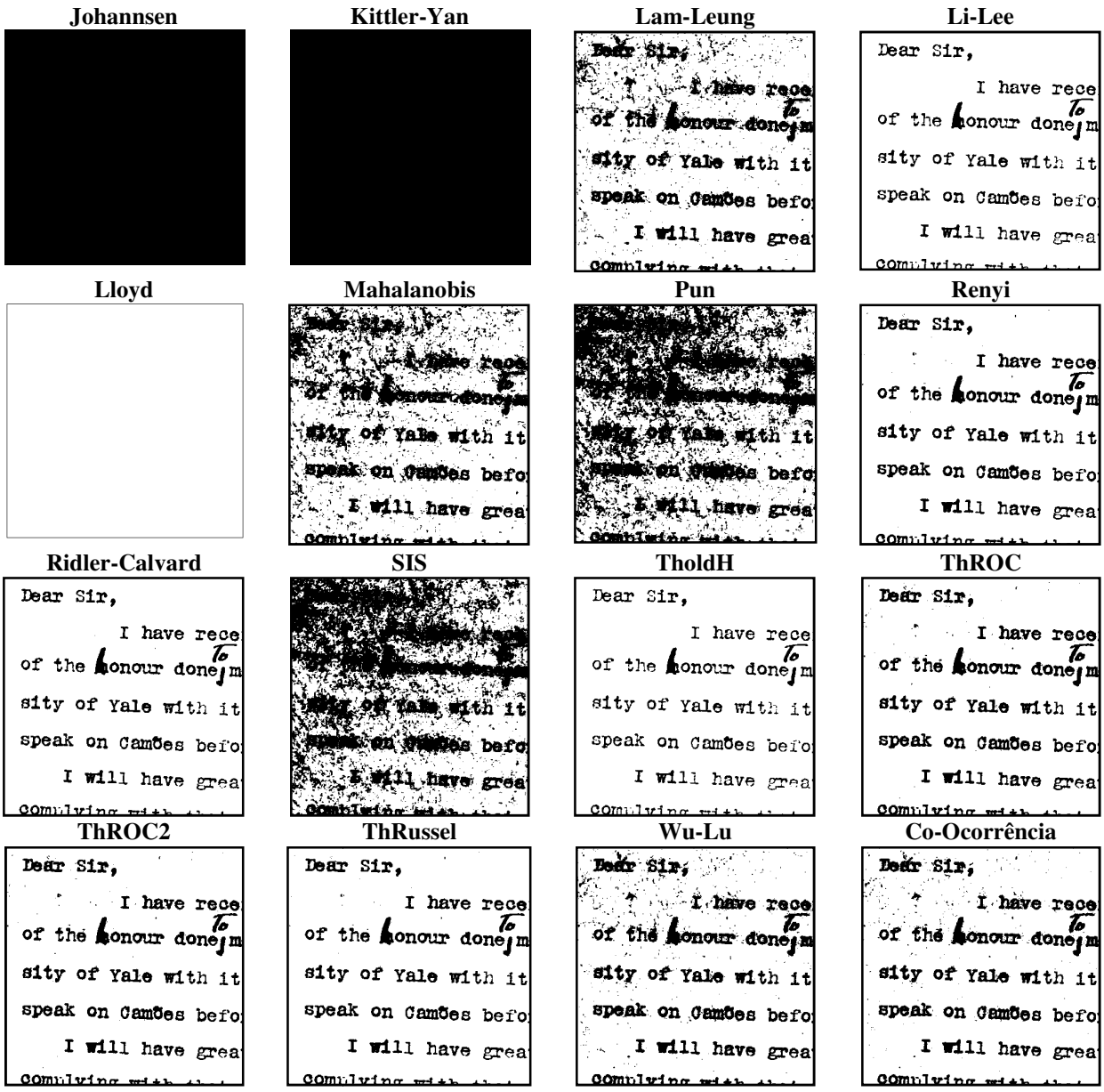


Figura 12. Exemplos de binarização de imagens de documentos históricos por algoritmos clássicos.

Capítulo 5

Conclusões

Este trabalho apresenta alguns avanços no processo de limiarização de documentos históricos. O algoritmo desenvolvido foi aplicado a um conjunto de documentos do final do século 19 e começo do século 20. Os documentos fazem parte do acervo do projeto PROHIST [38].

Para poder facilitar a divulgação dos documentos, uma conversão das imagens de uma base em tons de cinza para preto e branco gera imagens de menor tamanho (em KBytes). Para obter essas imagens, um algoritmo de binarização foi proposto para gerar imagens de duas cores (preto e branco) dos documentos históricos. Nesse formato, as imagens podem ser facilmente divulgadas já que seu tamanho fica em volta de 40 KBytes. Para gerar imagens monocromáticas de qualidade um algoritmo foi proposto em [35] baseado em entropia e curvas ROC. Essa técnica tinha a desvantagem de ter um tempo de processamento alto das imagens de alta resolução usadas em aplicações de documentos históricos e sua velocidade foi aprimorada como apresentado. A variação proposta fez o algoritmo ter grandes ganhos de velocidade em comparação ao original. Também verificamos que as imagens testadas têm melhor qualidade do que as geradas por várias outras técnicas clássicas de binarização. Aliando a melhor qualidade das imagens geradas com um incremento no tempo de processamento do algoritmo, temos agora um método rápido para gerar imagens monocromáticas de qualidade.

Além de permitir a fácil divulgação dos documentos, pelo seu menor tamanho em KBytes, as imagens binarizadas também podem ser utilizadas para gerar documentos de texto. Dado a qualidade das imagens, ferramentas do tipo OCR podem ser utilizadas para gerar esses documentos de texto. Como toda ferramenta, o OCR tem suas limitações, os documentos a ele submetidos teriam de ser os datilografados, visto que para documentos manuscritos o seu aproveitamento cai bastante. Todavia, a quantidade de documentos datilografados é considerável, ainda mais quando levamos em conta outros acervos como de jornais. Temos uma publicação diária desde o início do século XIX, o que gera um montante de grandes proporções. Disponibilizar esses textos ou até transpô-los para documentos de texto seria uma tarefa árdua, mas que o método proposto vem a sanar grande parte do problema. Geramos as imagens monocromáticas de qualidade em pouco tempo, podendo ser tomado o passo seguinte, a utilização de ferramentas OCR, ou a divulgação direta dos documentos em imagens binárias.

Outros ajustes do valor de corte estão sendo desenvolvidos utilizando algoritmos genéticos para tentar produzir imagens em tons de cinza com mais de duas cores para melhorar a qualidade visual das imagens.

Como citado anteriormente, as imagens do Diário de Pernambuco (outra base do acervo) são de má qualidade quando considerando o aspecto da digitalização. Para essas imagens verificamos que a qualidade das saídas geradas pelos algoritmos caiu bastante. Não encontramos um algoritmo que obteve desempenho satisfatório para um grupo de imagens do Diário. Tivemos, no entanto, alguns com bom desempenho para uma imagem, mas que não mantinham a qualidade nas imagens subseqüentes. Isso ocorreu devida a má qualidade de digitalização. Ao analisar as imagens, observamos que todas têm uma grossa borda ao redor do documento, borda essa que não faz parte do documento. Essa borda é considerada pelos cálculos de cada algoritmo como

sendo uma parte válida da imagem, dessa forma influindo diretamente no cálculo do ponto de corte final e assim contribuindo para a queda de qualidade para as imagens geradas. Para essas imagens, visto que o problema está na forma em que foram digitalizadas, um pré-processamento se faz necessário. Esse pré-processamento faria a remoção das bordas, gerando então somente o documento para ser processado pelo algoritmo de binarização. A remoção das bordas nessas imagens está ainda em fase de estudo e implementação, com a possibilidade de duas abordagens. Uma sendo a segmentação da imagem em janelas menores, dessa forma descartando as janelas exteriores que contêm as bordas. A segunda abordagem seria a extração das linhas e colunas mais exteriores da imagem que contêm um somatório mais elevado de preto, o que caracterizaria a borda escura.

A pesquisa aqui descrita foi publicada no *Journal of Cultural Heritage*, no artigo "*An Efficient Gray-Level Thresholding Algorithm for Historic Document Imagens*" [37], na edição Abril-Junho de 2008, com participação do autor desta monografia.

Bibliografia

[1] K. Sayood. Introduction to Data Compression, Ed. Morgan Kauffman, 1996, p.480.

[2] J.R. Parker. Algorithms for Image Processing and Computer Vision, John Wiley and Sons, 1997.

[3] T.W. Ridler, S. Calvard. Picture Thresholding Using an Iterative Selection Method, IEEE Transactions on Systems, Man and Cybernetics, 1978, Vol.SMC-8, 8, p. 630-632.

[4] S.W. Katz, A.D. Brink. Segmentation of Chromosome Images, IEEE COMSIG93, 1993, p. 85-90.

[5] J. Kittler, J. Illingworth. Minimum Error Thresholding, Pattern Recognition, 1986, 19, p. 41-47.

[6] M.S. Chang *et al.* Improved binarization algorithm for document image by histogram and edge detection, International Conference on Document Analysis and Recognition, Canadá, 1995, p.636-639.

[7] N. Otsu. A threshold selection method from gray-level histogram, Transactions on System, Man, and Cybernetics, 1978, vol 8, p. 62-66.

[8] C. Jawahar, P. Biswas, K.Ray. Investigations on Fuzzy Thresholding Based on Fuzzy Clustering, Pattern Recognition, 1997, vol 30, no 10, p. 1605-1613.

[9] L.K.Huang, M.J.Wang. Image Thresholding by Minimizing the Measures of Fuzziness, Pattern Recognition, 1995, vol 28, no 1, p. 41-51, Jan.

[10] R.R.Yager. On the Measures of Fuzziness and Negation.Part.1: Membership in the Unit Interval, International Journal of General Systems, 1979, Vol. 5, p. 221-229.

[11] C.A.Glasbye. An Analysis of Histogram-Based Thresholding Algorithm, Graphical Models and Image Processing, 1993, vol. 55, no. 6, p. 532-537.

[12] C.Shannon. A Mathematical Theory of Communication, in: Bell System Technology Journal, 1948, vol 27, p. 370-423, 623-656.

[13] J.N.Kapur. Measures of Information and their Applications, John Wiley and Sons, 1994.

[14] S.Kullback. Information Theory and Statistics.Dover Publications, Inc.1997.

[15] C.A.B.Mello. A New Entropy and Logarithmic Based Binarization Algorithm for Grayscale Images, Proceedings of IASTED International Conference on Signal and Image Processing 2004, 2004, Havaí, EUA, p. 418-423.

[16] T.Pun. Entropic Thresholding, A New Approach, Computer Graphics and Image Processing, 1981, vol. 16, p. 210-239.

[17] J.N.Kapur *et al*. A New Method for Gray-Level Picture Thresholding using the Entropy of the Histogram, Computer Vision, Graphics and Image Processing, 1985, Vol 29, no 3, p. 273-285.

[18] G.Johannsen, J.Bille. A Threshold Selection Method using Information Measures, Proceedings of 6th International Conference on Pattern Recognition, Munique, Alemanha, 1982, p.140-143.

[19] C.H.Li, C.K.Lee. Minimum Cross Entropy Thresholding, in: Pattern Recognition, 1993, vol. 26, no 4, p. 616-626.

[20] L.Wu, S.Ma, H.Lu. An Effective Entropic thresholding for Ultrasonic Images, International Conference on Pattern Recognition, 1998, p.1552-1554.

[21] P.Sahoo *et al.* Threshold Selection using Renyi's Entropy, Pattern Recognition, 1997, vol. 30, no 1, p. 71-84.

[22] L.C.Ramac, P.K.Varshney. Image Thresholding Based On Ali-Silvey Distance Measures, Pattern Recognition, 1997,Vol.30. no.7.pp.1161-1174.

[23] J.P.Cocquerez, S.Philipp. Analyse d´Images: filtrage et segmentation, Editor Masson, Paris, 1995, pp. 256-257.

[24] J.Gomes, L.Velho. Computação Gráfica: Imagem. IMPA-SBM, 1994.

[25] I.Cseke. A fast Segmentation Scheme for White Blood Cell Images, IEEE, Pattern Recognition,1992, pp 530-533.

[26] C.A.Glasbye. An Analysis of Histogram-Based Thresholding Algorithm, CVGIP: Graphical Models and Image Processing, 1993, v.55, No 6, pp 532-537.

[27] L.Li, J. Gong, W. Chen. Gray-level Image thresholding based on Fisher linear projection of two-dimensional histogram, Pattern Recognition, 1997, Vol 30, No 5, pp 743-749.

[28] C.V.Jawahar. P.K.Biswas, K.Ray. Investigations on Fuzzy Thresholding Based On Fuzzy Clustering, Pattern Recognition, 1997, Vol. 30, No. 10, pp. 1605-1613.

[29] C.K.Leung, F.K.Lam. Performance Analysis for a Class of Iterative Image Thresholding Algorithms, Pattern Recognition, 1996, vol 29, no. 9, pp. 1523-1530.

[30] K.V.Mardia, T.J. Hainsworth. Spatial Thresholding Method for Image Segmentatio, IEEE, Transactions on Pattern Analysis and Machine Intelligence, 1988, Vol .10, Nº 06, pp 919-927.

[31] B. Chanda, D.D. Majunder. A note on the use of the graylevel co-ocorrence matrix in threshold selection, Signal Processing, 1988, Vol 15, No.2, pp 149-167.

[32] C.-I.Chang, K. Chen, J. Wang and M.L.G. Althouse. A Relative Entropy-Based Approach to Image Thresholding, Pattern Recognition, 1994, 27, (9), pp. 1275-1289.

[33] Gerhard, Ritter, and Joseph, Handbook of Computer Vision Algorthms in Image Algebra, Florida, CRC Press, 1996.

[34] H.J. Thrussel. Picture Threshold using an Iterative Selection Method, IEEE Transactions on Systems, Man and Cybernetics, 1979, v..SMC 9, no.5, pp.311-325.

[35] C.A.B.Mello, A.Sanchez, A.L.I.Oliveira. Image Thresholding of Historical Documents: Application to the Joaquim Nabuco's File, Proceedings of the 1st EVA 2006 Vienna Conference, Viena, Áustria, 2006, p. 115-122.

[36] N.A.McMillan, C.D.Creelman. Detection Theory. LEA Publishing, 2005.

[37] C.A.B Mello *et al*. An Efficient Gray-Level Thresholding Algorithm for Historic Document Images. Journal of Cultural Heritage, 2008, Vol 9, pp 109-116, DOI: (10.10 16/j.culher.2007.09.004).

[38] PROHIST – Image Processing of Historical Documents. Disponível em: <http://www.dsc.upe.br/~recpad/prohist/>. Último acesso em 23 de maio de 2008.

Apêndice A

Implementações

Apresentamos aqui o programa Binarizador desenvolvido na pesquisa descrita. O programa foi desenvolvido pelo autor desta monografia em conjunto com o orientador e outros dois alunos de Iniciação Científica. Esse programa foi desenvolvido utilizando a linguagem C/C++ e a ferramenta da Borland, o C++ Builder. No programa, encontramos 32 algoritmos distintos de binarização. Esses são algoritmos conhecidos, propostos por diversos pesquisadores e que freqüentemente aparecem como tema de estudo quando se estuda segmentação de imagens. Desses algoritmos implementados no Binarizador, 27 foram discutidos e explicados no Capítulo 2. O programa recebe uma imagem em tons de cinza que o usuário deseja binarizar e permite que o algoritmo seja escolhido. Após a escolha do algoritmo, o programa binariza a imagem e retorna a imagem monocromática. O usuário pode escolher entre salvar a imagem binarizada ou somente visualizá-la, tendo também a opção de selecionar outros algoritmos para poder comparar diferentes resultados.

Os algoritmos incluídos no programa Binarizador são:

- Ali-Silvey
- Brink
- Chehikian
- CMeans
- Cseke
- Ye-Danielsson
- Fuzzy Huang
- Fisher LAT
- Fisher
- Fuzzy Distâncias Normais
- Fuzzy Yager
- Iteractive Selection
- Johannsen
- Kittler-Yan
- Lam-Leung
- Li-Lee
- Lloyd
- Mahalanobis
- Median Cut

- Co-ocorrência
- Pun
- Renyi
- Ridler-Calvard
- SIS
- Thrussel
- Wu-Lu
- Porcentagem de Preto
- Kapur
- TholdH
- ThROC
- ThROC2
- Two Peaks

As imagens a seguir mostram o Binarizador em sua tela inicial e também o seu funcionamento, exibindo uma imagem de um documento original e suas versões binarizadas.

Figura 13. Tela inicial do programa binarizador.

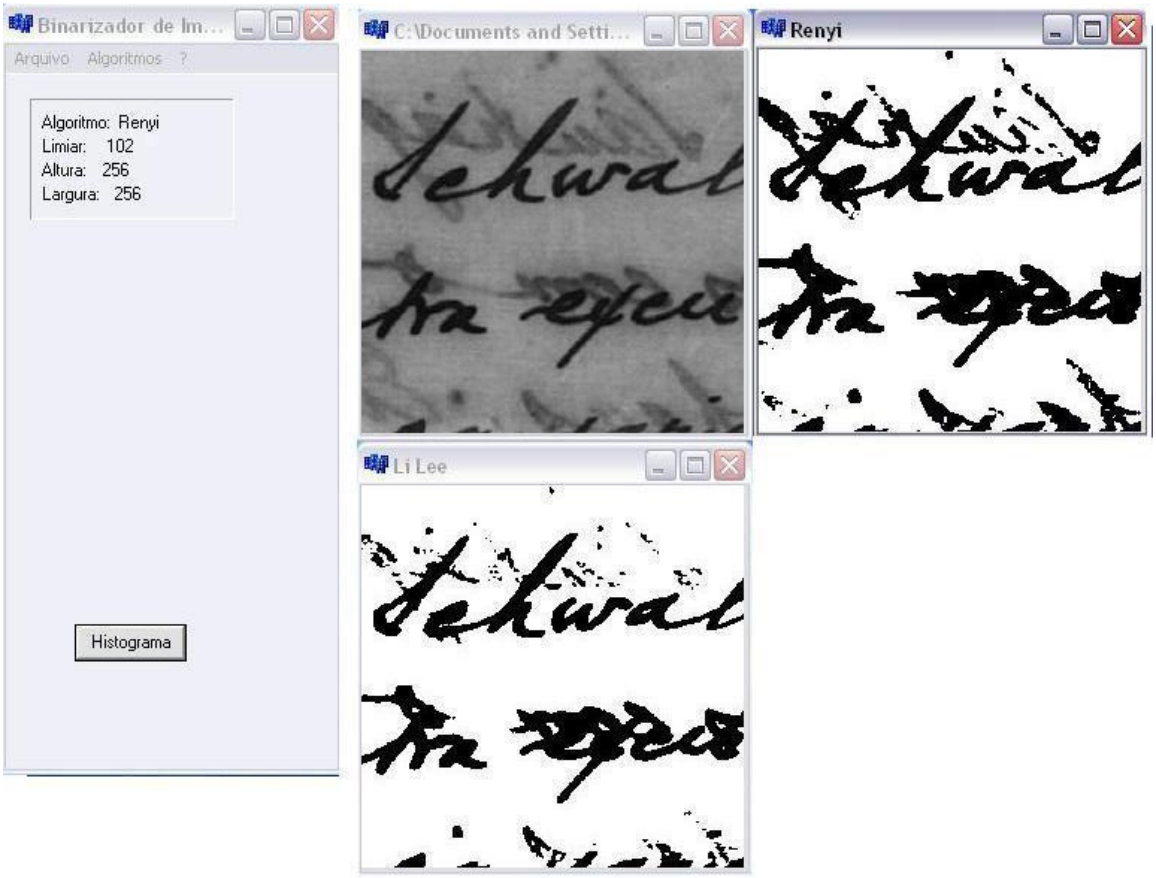


Figura 14. Exibição de imagens no programa binarizador.

Além das funcionalidades de exibir as imagens, binarizá-las e exibir os resultados, o programa também tem a capacidade de exibir o histograma das imagens que o usuário escolheu para serem binarizadas. A Figura a seguir ilustra essa funcionalidade.

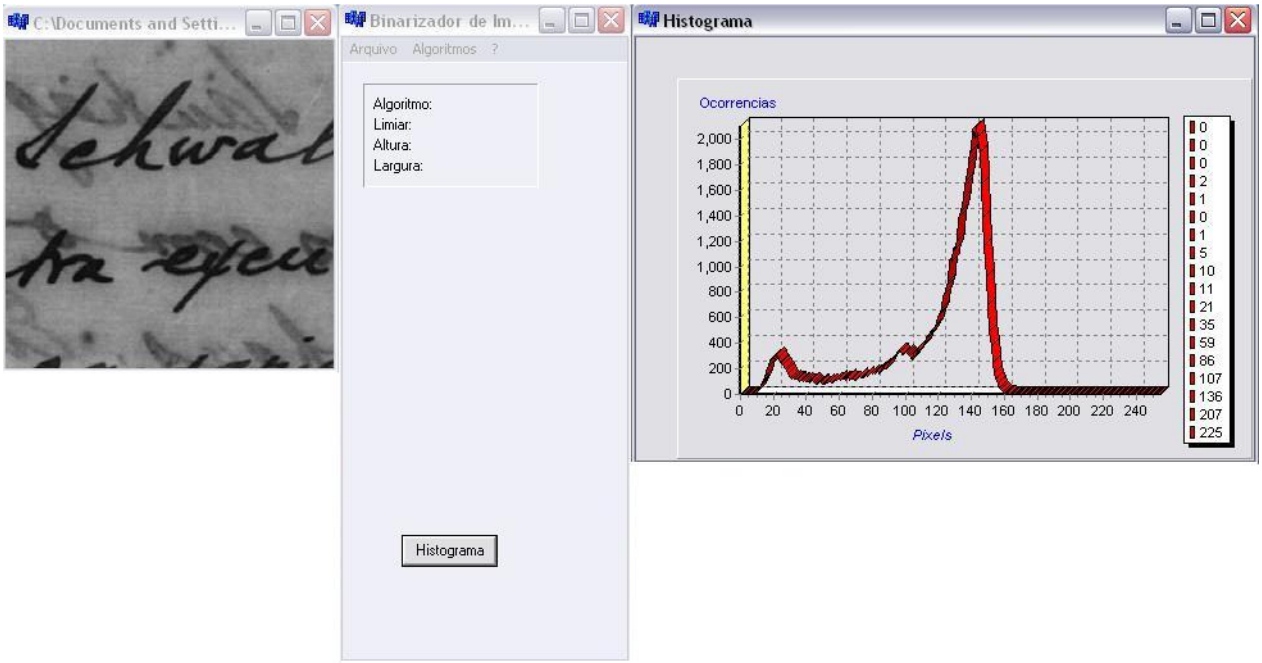


Figura 15. Exibição do histograma da imagem.