ALGORITMOS ROBUSTOS PARA FILTRAGEM DISCRIMINATIVA

Carlo Marcello de Oliveira Siqueira

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

_____
Prof. Eduardo Antônio Barros da Silva, Ph.D.

_____
Prof. Alexandre Pimentel Mendonça., D.Sc.

_____
Prof. Sergio Lima Netto, Ph.D.

_____
Prof. Marco Antonio Grivet Mattoso Maia, Ph.D.

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2006

SIQUEIRA, CARLO MARCELLO DE OLIVEIRA

Algoritmos Robustos para Filtragem Discriminativa [Rio de Janeiro] 2006

VIII, 132 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia Elétrica, 2006)

Dissertação - Universidade Federal do Rio de Janeiro, COPPE

1. Reconhecimento de Padrão.

2. Filtragem Discriminativa.

3. Restauração de Impulso.

4. Algoritmos Robustos ao Ruído.

I. COPPE/UFRJ II. Título (série)

Agradecimentos

Agradeço primeiramente a Deus pela serenidade e saúde que foram necessárias para o término desta tese.

À toda minha família, minha mãe Oneide, meu pai Aluízio meus irmãos Jean e Junior e a todos os meus outros parentes, minhas avós, tios, tias e primos, que sempre torceram por mim e me deram ânimo ao longo do desenvolvimento desta Tese.

Aos meus orientadores Eduardo e Alexandre por todas as explicações, incentivos para a finalização deste trabalho.

A todos os meus amigos, do LPS, em especial ao Tadeu que quebrou vários galhos para mim; da faculdade; colégio; infância e tantos outros que participaram de minha vida nesses três anos de mestrado e que agüentaram meu mau humor, que conviveram com minhas faltas aos eventos sociais e que também me ajudaram me dando forças e incentivos nesta jornada.

Agradeço também aos meus amigos da Eletronuclear, que me incentivaram na conclusão deste Mestrado e que possibilitaram minhas tantas vindas ao Fundão, ou para assistir aula ou para minhas reuniões com o orientador.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ALGORITMOS ROBUSTOS PARA FILTRAGEM DISCRIMINATIVA

Carlo Marcello de Oliveira Siqueira

Março/2006

Orientadores: Eduardo Antonio Barros da Silva
Alexandre Pimentel Mendonça.

Programa: Engenharia Elétrica

Com o crescente estudo em processamento de imagens e visão computacional, a pesquisa em discriminação e reconhecimento de padrões em imagens tem se tornado cada vez mais importante, surgindo, desta forma, vários algoritmos e técnicas para os mais diversos padrões e aplicações. Com isso, iniciou-se o estudo em uma técnica bem recente e inovadora de reconhecimento de padrão: a Filtragem Discriminativa via Restauração de Impulso. Neste estudo, viu-se o grande potencial de um filtro discriminativo que detecta multipadrões, o filtro OU.

A construção deste filtro inclui a reunião de um conjunto de imagens, que participa do seu treinamento, e que o habilita a discriminar as letras que o criou. A partir disto, decidiu-se criar um algoritmo que deixasse este filtro robusto ao ruído e que detectasse qualquer tipo de fonte para uma determinada letra. Para isto, seria preciso agrupar um conjunto de imagens que proveria tal robustez ao filtro. A formação deste conjunto de imagens surgiu, primeiramente, da idéia de representar uma letra a partir de uma função de probabilidade (PF), onde estariam representadas as suas mais variadas formas. A partir desta PF, extrair-se-iam as imagens que treinariam o filtro, e esta é principal característica que diferencia os métodos que serão apresentados.

Visto isso, a principal contribuição desta dissertação é então apresentar os vários algoritmos criados e avaliá-los através de testes com várias letras, em imagens com e sem ruído. Mostrando desta forma o grande potencial deste método para o reconhecimento de padrões de imagens na presença de ruído.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ROBUST ALGORITHMS FOR DISCRIMINATIVE FILTERING

Carlo Marcello de Oliveira Siqueira

March/2006

Advisors: Eduardo Antonio Barros da Silva

Alexandre Pimentel Mendonça.

Department: Electrical Engineering

With the increasing study in image processing and computer vision, the research in pattern discrimination and recognition in images has become more important, appearing, therefore, many algorithms and techniques for the most diverse patterns and applications. So it was started a study about a recent techinique in pattern recognition: the Discriminative Filtering by Impulse Restoration. In this study, it was seen the great potencial about a discriminative filter to detect multipatterns, the OR filter.

The construction of this filter includes the formation of a set of letters, that participates of its training. So, this filter will be able to discriminate all these patterns of letters that created it. After that, it was decided to create an algorithm to become this filter noise robust, and that detects any type of morphology that a letter could have. For this, it was needed to group a set of images to provide such robustness to the fitler. The formation of this set of images surges, first, from the idea to represent a letter by a probability function (PF), where would be shown the many types that a letter could have. The images, that participated on the filter training, would be extracted from this PF, and this is the mainly characteristic, that becomes the algorithms (that will be presented) differents.

So the mainly contribution of this dissertation is to present the algorithms bred and evaluate them, through many tests with differents letters, in images with and without noise. Showing of this form the great potential of this method for the image pattern recognition in the noise presence.

Sumário

Capítulo 1
Introdução

1.1 Motivação

A constante busca por tornar um computador ou uma máquina num sistema cada vez mais inteligente, que possa pensar e agir como um ser humano, é que faz com que alguns ramos da ciência se aprofundem em pesquisas como por exemplo: a robótica simulando partes do corpo humano, o reconhecimento de sinais de voz, os sinais de imagem simulando a visão humana, a inteligência artificial que permite a simulação de sentidos como se está frio ou quente, se é bonito ou feio, se é bom ou ruim, dentre outras tantas áreas da ciência.

Com este avanço, a discriminação e o reconhecimento de padrões em imagens se tornam cada vez mais importantes, surgindo desta forma vários algoritmos e técnicas diferentes para as mais diversas formas de padrões [1-3].

As aplicações de reconhecimento de padrões têm-se expandido em várias áreas de interesse, principalmente em visão computacional, onde o principal objetivo é retirar de imagens informações que sirvam para a aplicação a que se propõe como, por exemplo: na automação de processos em indústrias, no controle de qualidade na produção de manufaturas; na engenharia biomédica, com a classificação de padrões em imagens geradas por ultra-som ou raios-X; na indústria de vigilância, com o reconhecimento de impressões digitais, face e íris [4-5]; na indústria militar, no uso de mísseis tele-guiados que buscam alvos pré-definidos; no reconhecimento de caracteres em placas de carros ou em documentos [6]. Enfim, o reconhecimento e a classificação de padrões está em constante evolução na busca por soluções que tragam novos e melhores resultados.

Desta forma, o objetivo desta tese é apresentar um estudo de reconhecimento de padrões de imagem via filtragem discriminativa e restauração de impulso, já iniciado por Mendonça [7], apresentando novas características e extensões que visam a dar ao método uma maior robustez ao ruído.

1.2 Desenvolvimento da tese

O reconhecimento e a discriminação de padrões de imagens utilizando filtragem discriminativa (em duas dimensões) é uma técnica ainda bastante nova e que teve seu estudo aprofundado por Mendonça [7]. No entanto, suas aplicações em uma dimensão já têm sido pesquisadas e utilizadas desde o início da década de 90 pelo Prof Ben-Aire e colaboradores [8-10].

A principal característica da filtragem discriminativa é obter um operador que, ao ser convoluído com um determinado padrão, gere um sinal com energia concentrada em numa das amostras do sinal de saída. Com isso, como base de todo o problema enfocado por essa tese, o Capítulo 2 mostra como foi formulada a filtragem discriminativa bidimensional desenvolvida em [7]. Depois disso, é também apresentada uma técnica muito parecida com a filtragem discriminativa, que é a detecção de padrões via restauração de impulso, onde seu objetivo é encontrar também um operador linear que, ao ser convoluído por um padrão gere um impulso no sinal de saída na posição em que o padrão procurado estiver localizado. O interessante desta técnica é que sua solução é analítica, não requerendo nenhum processo de otimização.

No final deste capítulo, será apresentada uma aplicação da filtragem discriminativa via restauração de impulso para mais de um padrão de imagem, que será um filtro, onde uma de suas características é que ele é criado por um conjunto de imagens, que são exatamente as imagens que se quer discriminar ao longo do experimento. Quando ele é convoluído com o gabarito que se quer reconhecer, gera então uma saída com alta concentração de energia em uma de suas amostras e baixa energia nas demais. Ou seja, imagine um banco de imagens com as letras “A”, “B” e “C”, em fonte Arial. Ao se tentar reconhecer estas letras em uma frase em um texto *scaneado* ou em uma foto, o filtro ao ser convoluído com esta imagem gerará um sinal com impulsos nas posições onde estas letras estiverem presentes. Este filtro é chamado de filtro “OU” [7].

A principal contribuição desta tese é de criar um filtro OU que servisse para reconhecer cada uma das letras do alfabeto em suas mais variadas fontes e formas e que fosse robusto ao ruído. Para se fazer a sua detecção, seria criado um filtro OU para cada letra, mas para isso se deve primeiro formalizar um método que gere um banco de imagens que represente os vários tipos de fontes para uma letra mesmo com ruído.

Desta forma, imaginou-se uma forma de representar todas as posições de pixels possíveis que, ao se combinar, poderiam representar uma determinada letra, com ruído ou não. Daí se pensou em criar uma função de probabilidade (PF), onde cada padrão de letra teria uma determinada freqüência associada e são estas fontes que compões o filtro "OU". A partir destas idéias, vários métodos para a criação da PF foram criados. O Capítulo 3 mostra cada um desses métodos até o mais satisfatório.

Após as apresentações dos métodos de criação da PF, o Capítulo 4 mostra alguns testes utilizando imagens com a Lena, imagens com ruídos binários e outros tipos de ruídos, onde serão discriminados vários tipos de letras. Enfim, as conclusões desta tese e propostas para trabalhos futuros são apresentadas no Capítulo 5.

Capítulo 2

Métodos para discriminação de padrões

A detecção e reconhecimento de padrões são de grande importância em um sistema de visão. Neste capítulo são vistas duas técnicas que servem de base para todo o desenvolvimento desta tese. Primeiro é visto o chamado Filtro Discriminativo que, ao ser convoluído com um padrão de entrada, gera um sinal de saída do filtro com a energia concentrada em uma de suas amostras. A segunda técnica, a restauração de impulso, é na verdade um caso mais geral da filtragem discriminativa, só que o projeto do filtro é diferente. A teoria de restauração do impulso diz que uma imagem é composta por um padrão numa posição desconhecida somado de um ruído aditivo (restante da imagem). O objetivo é encontrar uma imagem resultante da filtragem que possua impulsos somente nas posições onde existam os padrões procurados. A terceira parte deste capítulo consiste em mostrar uma aplicação de filtragem utilizando técnicas de restauração de impulso. É mostrado um filtro que reconhece os padrões que participaram do seu treinamento, excluindo os outros que não participaram. É o chamado Filtro "OU".

2.1 Filtros discriminativos

A filtragem discriminativa, como já foi dito, é um estudo relativamente novo. No início dos anos 90, Ben-Aire e sua equipe iniciaram um estudo sobre novos detectores de padrão, chamados de EXM (Expansion Matching) [8-10], onde sua principal aplicação é detectar algumas características como junções, bordas, etc.

2.1.1 Filtros discriminativos em uma dimensão

O modo mais comum para se detectar um padrão é projetar uma imagem em um subespaço vetorial, que é gerado por alguns gabaritos que são necessários para determinada aplicação, o que requer um grande esforço computacional. Ou seja, estimar uma imagem multidimensional $\hat{u}(x,y,\ldots)$, a partir da imagem original $u(x,y,\ldots)$ numa

base de gabaritos $\{\phi_i\}, i = 1,\ldots,t$. Assim desta forma o problema central é encontrar os coeficientes que aproximem $\hat{u}(x,y,\ldots)$ de $u(x,y,\ldots)$ com o menor erro quadrático possível,

$$\hat{u}(x,y,\ldots) = \sum_{i=1}^{t} c_i \phi_i(x,y,\ldots) \tag{2.1}$$

Mas o princípio da ortogonalidade diz que o menor erro é obtido projetando-se $u(x,y,\ldots)$ no subespaço gerado por $\{\phi_i\}$, tornando assim o erro perpendicular a estimativa. Desta forma deve-se garantir que,

$$< \hat{u} - u, \phi_i >= 0 \quad para\ todo\ \ i \in [1,t] \tag{2.2}$$

Substituindo a equação 2.1 na 2.2, obtém-se o conjunto de equações 2.3 que pode ser representado na forma matricial como mostram as equações 2.4 e 2.5:

$$< \phi_i - \phi_1 > c_1 + < \phi_i - \phi_2 > c_2 + \ldots + < \phi_i - \phi_t > c_t = < u, \phi_i >, \quad i - 1,2,\ldots,t \tag{2.3}$$

$$\begin{bmatrix} < \phi_1,\phi_1 > & < \phi_1,\phi_2 > & \cdots & < \phi_1,\phi_t > \\ < \phi_2,\phi_1 > & < \phi_2,\phi_2 > & \cdots & < \phi_2,\phi_t > \\ \cdots & \cdots & \cdots & \cdots \\ < \phi_t,\phi_1 > & < \phi_t,\phi_2 > & \cdots & < \phi_t,\phi_t > \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_t \end{bmatrix} = \begin{bmatrix} < u,\phi_1 > \\ < u,\phi_2 > \\ \cdots \\ < u,\phi_t > \end{bmatrix} \tag{2.4}$$

$$\mathrm{R}_{\phi\phi}\mathrm{C} = < \Phi, \Phi^{\mathrm{t}} > \mathrm{C} = < \Phi, \mathrm{u} > \tag{2.5}$$

Se o conjunto $\{\phi_i\}$ for linearmente independente, então $R_{\phi\phi}$ é positiva definida [8] e dessa forma o sistema descrito tem solução única para *C*. Mas a solução requer grande esforço computacional porque deve ser feita a inversa de uma matriz de dimensão muito grande, mas que pode ser feita independente da imagem e offline.

Um novo indicador de eficiência de sistemas de detecção de padrões foi criado recentemente por Ben-Aire e Rao [8], chamada de *DSNR* – Discriminative Signal To Noise Ratio definida como,

$$\mathrm{DSNR}_{\log}(j) = 10\log\left[\frac{c_j^2}{\sum_{i=1,i\neq j}^{t} c_i^2}\right]$$

ou

$$\mathrm{DSNR}(j) = \frac{c_j^2}{\sum_{i=1,i\neq j}^{t} c_i^2} = \frac{c_j^2}{\left(\sum_{i=1}^{t} c_i^2\right) - c_j^2} \tag{2.6}$$

Como pode ser visto acima, a *DSNR* representa o quanto um sistema pode concentrar energia em um único coeficiente c_j.

Outro ponto importante que Ben-Aire e Rao pesquisaram foi em encontrar um operador linear ótimo que maximizasse a *DSNR*. Desta forma, considere θ o operador ótimo e [*U*] a matriz definida pela equação 2.7 formada pelas amostras do sinal de entrada.

$$\begin{bmatrix} u(1) & u(2) & \cdots & u(t) \\ u(t) & u(1) & \cdots & u(t-1) \\ \cdots & \cdots & \cdots & \cdots \\ u(2) & u(3) & \cdots & u(1) \end{bmatrix} = [U] \tag{2.7}$$

A partir da equação acima, o operador deve satisfazer a condição abaixo:

$$[U]\theta = C\,, \tag{2.8}$$

onde

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_t \end{bmatrix}. \tag{2.9}$$

Para se conseguir a maximização da *DSNR*, basta ver a equação 2.6 e maximizá-la, ou seja:

$$\frac{c_j^2}{\left(\sum_{i=1}^{t} c_i^2\right) - c_j^2} \rightarrow max \tag{2.10}$$

ou

$$\frac{\sum_{i=1}^{t} c_i^2}{c_j^2} \rightarrow min\,. \tag{2.11}$$

Ao comparar a equação 2.8 com a 2.11, percebe-se que o numerador desta é na verdade a norma quadrada do vetor $C = [U]\theta$. Desta forma, fazendo algumas substituições entre as equações acima e considerando que U_j como sendo o vetor obtido ao se transpor a linha j da matriz *[U]*, temos a seguinte minimização:

$$\frac{([U]\theta)^t([U]\theta)}{\left(U_j^t\theta\right)^2} = \frac{\theta^t[U]^t[U]\theta}{\left(U_j^t\theta\right)^2} \rightarrow min \tag{2.12}$$

Calculando o gradiente da equação 2.12 e igualando a zero, tem-se a equação abaixo,

$$\frac{2\left(U_j\theta^t[U]^t[U]\theta - [U]^t[U]\theta\theta^t U_j\right)}{\left(U_j^t\theta\right)^3} = 0 \tag{2.13}$$

que recai na seguinte igualdade,

$$U_j\theta^t[U]^t[U]\theta = [U]^t[U]\theta\theta^t U_j \tag{2.14}$$

Uma análise dimensional da equação acima mostra que os termos $\theta^t [U]^t [U]\theta$ e $\theta^t U_j$ são unidimensionais e que têm o objetivo de normalizar a amplitude do operador θ. Assim chega-se na equação 2.15 abaixo, que é única [10]:

$$\theta = k\left([U]^t [U]\right)^{-1} U_j = k\left(R_{uu}\right)^{-1} U_j, \tag{2.15}$$

onde k é a constante de normalização e $R_{uu} = [U]^t [U]$.

2.1.2 Filtros discriminativos em duas dimensões

A primeira etapa para a confecção do filtro discriminativo em duas dimensões é definir uma função-objetivo discriminativa ($DSNR_2$) [7], que não só leva em conta a energia em uma determinada amostra, mas também a energia em relação às outras amostras.

$$DSNR_2(i,j) = \frac{c_{i,j}^2}{\left(\sum_m \sum_n c_{m,n}^2\right) - c_{i,j}^2} \tag{2.16}$$

Os coeficientes $c_{m,n}$ são obtidos com a convolução bidimensional entre o sinal de entrada $u_{m,n}$ (janela com a imagem) e o operador linear Θ, com resposta ao impulso $\theta_{m,n}$ que se quer calcular. O coeficiente $c_{i,j}$ é aquele onde se quer concentrar a energia de saída do filtro.

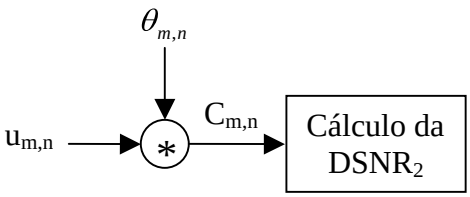


Figura 2.1. Cálculo da $DSNR_2$.

Para este cálculo acima, considere U uma janela M_1 x N_1 e Θ um filtro M_2 x N_2. Então a imagem resultante da convolução de U com Θ tem dimensões M_1+M_2-1 x N_1+N_2-1 e pode ser expressa da seguinte forma:

$$c_{m,n} = \sum_{m'} \sum_{n'} u_{m-m',n-n'} \theta_{m',n'} \tag{2.17}$$

Assim como na equação (2.11) percebe-se facilmente que a maximização da $DSNR_2$ é a minimização da equação abaixo:

$$f(\Theta) = \frac{\sum_{m} \sum_{n} c_{m,n}^2}{c_{i,j}^2}, \tag{2.18}$$

que da equação 2.17 chega-se a;

$$f(\Theta) = \frac{\sum_{m} \sum_{n} \left(\sum_{m'} \sum_{n'} u_{m-m',n-n'} \theta_{m-m',n-n'} \right)^2}{\left(\sum_{m'} \sum_{n'} u_{i-m',j-n'} \theta_{m',n'} \right)^2} \tag{2.19}$$

O $f(\Theta)$ é mínimo quando seu gradiente é igual a *0* e através do método do gradiente tem-se a seguinte equação para $\nabla f(\Theta)$,

$$\nabla f(\Theta) = 2 \left\{ c_{i,j} \sum_{m} \sum_{n} u_{m-r,n-s} c_{m,n} - u_{i-r,j-s} \sum_{m} \sum_{n} c_{m,n}{}^2 \right\} c_{i,j}{}^{-3} \tag{2.20}$$

O problema para esta formulação é que pode existir um outro gabarito que ao passar pelo operador dê uma $DSNR_2$ mais alta do que a do gabarito que originou o operador. Com isso, outras abordagens foram realizadas por Mendonça [7] visando a uma melhor estimativa para o operador linear Θ. Dentre elas tem-se a abordagem alternativa onde se cria uma nova função-objetivo que tem como principal função maximizar a $DSNR_2$ quando U variar. Outro enfoque também foi dado pela abordagem mista [7], onde é construída uma nova função-objetivo que combina tanto o método apresentado aqui nesta seção como a abordagem mista citada acima.

2.2 Restauração do impulso

A filtragem discriminativa apresentada [11-12] e a restauração do impulso são problemas com grande correlação. Como já foi dito, o objetivo do filtro discriminativo é: ao se convoluir uma imagem com um operador para determinado padrão, encontra-se um sinal de saída com a energia concentrada em uma de suas amostras. A restauração de impulso tem por objetivo obter uma imagem final de um impulso no local onde o padrão está localizado, e o restante da imagem teria o valor zero.

Fazendo uma associação entre as duas técnicas, pode-se ver que um filtro discriminativo ideal é aquele que restaura o impulso (alta concentração de energia) distorcido por um padrão e corrompido pelo resto da imagem que seria considerado um ruído. O ganho ao tratar a filtragem discriminativa como restauração de impulso é que, desta forma, podem ser encontradas soluções analíticas, diferentes das abordagens realizadas na seção 2.1 que levam a solução de otimização numérica.

O problema da filtragem discriminativa via restauração do impulso é representado inicialmente na imagem como:

$$g(m,n) = f(m-m_o, n-n_o) + b(m,n), \tag{2.21}$$

onde $g(m,n)$ é uma imagem composta por $f(m-m_o, n-n_o)$, que é o padrão centrado no ponto (m_o, n_o) equivalente à $f(m,n) * \delta(m-m_0, n-n_0)$ (impulso distorcido pelo padrão), e $b(m,n)$ que é o restante da imagem. Fazendo a associação com a restauração do impulso, é como seria um impulso δ centrado em (m_o, n_o) distorcido por um operador linear $f(m,n)$ e adicionado ao ruído $b(m,n)$.

Para facilitar os cálculos matemáticos, passa-se a notação acima para forma matricial, representando a imagem como um vetor coluna [11-12]. Desta forma a imagem $g(m,n)$ através da concatenação de suas linhas transpostas se torna $G(k)$, o padrão centrado em (m_o, n_o) se representa por $F(k-k_0)$, o impulso torna-se $\delta(k)$ e o restante da imagem $B(k)$. Assim a fórmula acima é representada da seguinte forma:

$$G(k) = F(k) * \delta(k-k_0) + B(k) \tag{2.22}$$

A convolução realizada na equação anterior é circular e a imagem tem dimensões $(2\mathrm{T}+1)\times(2\mathrm{T}+1)$, onde $-\mathrm{T} \le \mathrm{m},\mathrm{n} \le \mathrm{T}$. Desta forma as equações (2.21) e (2.22) podem ser representadas por

$$\mathrm{g} = \mathrm{F}\delta + \mathrm{b}, \tag{2.23}$$

onde

$$g = \begin{bmatrix} g(-T,-T) \\ g(-T,-T+1) \\ \vdots \\ g(T,T) \end{bmatrix}, \delta = \begin{bmatrix} \delta(-T,-T) \\ \delta(-T,-T+1) \\ \vdots \\ \delta(T,T) \end{bmatrix}, b = \begin{bmatrix} b(-T,-T) \\ b(-T,-T+1) \\ \vdots \\ b(T,T) \end{bmatrix}, \tag{2.24}$$

$$F = \begin{bmatrix} F(0) & F\left((2T+1)^2-1\right) & \dots & F(1) \\ F(1) & F(0) & \dots & F(2) \\ \dots & \dots & \dots & \dots \\ F\left((2T+1)^2-1\right) & F\left((2T+1)^2-2\right) & \dots & F(0) \end{bmatrix},$$

onde $g, \delta \; e \; b$ são vetores $(2T+1)^2 \times 1$ e F é uma matriz $(2\mathrm{T}+1)^2 \times (2\mathrm{T}+1)^2$.

Assim, o objetivo deste problema é estimar o vetor δ dados o vetor *g*, a matriz *F* e o vetor *b* que representa o ruído e que a priori é desconhecido.

Começando então uma análise do vetor impulso δ, considera-se inicialmente que ele pode ser restaurado com igual probabilidade em cada uma de suas amostras. Com isso, tem-se que a sua matriz autocorrelação corresponde a:

$$E\{\delta\delta^t\} = \frac{1}{N} I, \tag{2.25}$$

onde $N = (2T+1)$. Note que $E\{\delta_i \delta_j\}$ é igual a *1/N* quando δ_j está na posição *i (i=j)*, senão será 0 em qualquer outra posição. Por isso, é de fácil percepção que *I* é uma matriz identidade *NxN*.

Algumas outras considerações devem ser feitas também para que uma solução seja mais facilmente encontrada. Por exemplo, será considerado que o ruído *b* possui média

nula e sua matriz de covariância representado por C_b / N. Sendo assim, a melhor estimativa de δ (quer dizer $\hat{\delta}$) , que minimize o erro médio quadrático $E\left[\left\|\delta - \hat{\delta}\right\|^2\right]$ e que seja linear em relação a g ($\hat{\delta} = Ag$) é aquela em que o erro $\delta - \hat{\delta}$ é descorrelacionado de *g*. Assim

$$E\left\{\left(\delta - \hat{\delta}\right)g^t\right\} = 0, \tag{2.26}$$

ou

$$\begin{aligned} E\left\{\left(\delta - \hat{\delta}\right)g^t\right\} &= E\left\{\left(\delta - Ag\right)g^t\right\} = E\left\{\left(\delta - AF\delta - Ab\right)\left(F\delta + b\right)^t\right\} \\ &= E\left\{\delta\hat{\delta}F^t\right\} + E\left\{\delta b^t\right\} - E\left\{AF\delta\delta^t F^t\right\} \\ &- E\left\{AF\delta b^t\right\} - E\left\{Ab\delta^t F^t\right\} - E\left\{Abb^t\right\} \\ &= E\left\{\delta\hat{\delta}\right\}F^t + E\left\{\delta b^t\right\} - AFE\left\{\delta\delta^t\right\}F^t \\ &- AFE\left\{\delta b^t\right\} - AE\left\{b\delta^t\right\}F^t - AE\left\{bb^t\right\} = 0 \end{aligned} \tag{2.27}$$

Considerando δ e *b* descorrelacionados, faz-se

$$E\left\{\delta b^t\right\} = 0 \tag{2.28}$$

$$E\left\{bb^t\right\} = C_b / N \tag{2.29}$$

Para que as equações acima sejam válidas, o vetor *b* deve ter média nula e, para isso, a imagem em questão deve ter sua média retirada. Com isso, a equação 2.27 pode ser representada da seguinte forma

$$F^t - AFF^t - AC_b = 0 \tag{2.30}$$

assim,

$$A = F^t\left(FF^t + C_b\right)^{-1} \tag{2.31}$$

e

$$\hat{\delta} = F^t\left(FF^t + C_b\right)^{-1} g \tag{2.32}$$

Para utilizar o estimador linear na forma usada nas referências [11-12], são feitas as seguintes manipulações:

$$
\begin{aligned}
\hat{\delta} &= F^t\left(FF^t + C_b\right)^{-1} g \\
&= F^t\left(FF^t + C_b\right)^{-1} C_b\left(F^t\right)^{-1} F^t C_b^{-1} g \\
&= F^t\left(FF^t + C_b\right)^{-1}\left(F^t C_b^{-1}\right)^{-1} F^t C_b^{-1} g \\
&= \left(\left\{FF^t + C_b\right\}\left(F^t\right)^{-1}\right)^{-1}\left(F^t C_b^{-1}\right)^{-1} F^t C_b^{-1} g \\
&= \left(F + C_b\left(F^t\right)^{-1}\right)^{-1}\left(F^t C_b^{-1}\right)^{-1} F^t C_b^{-1} g \\
&= \left(F^t C_b^{-1}\left\{F + C_b\left(F^t\right)^{-1}\right\}\right)^{-1} F^t C_b^{-1} g \\
&= \left(F^t C_b^{-1} F + I\right)^{-1} F^t C_b^{-1} g
\end{aligned}
\tag{2.33}
$$

2.3 Filtragem discriminativa via restauração do impulso

O objetivo agora é confeccionar Filtros Discriminativos similares da seção anterior usando a restauração do impulso como base. Para isso, no entanto, exprime-se novamente a imagem como a soma do impulso em uma posição desconhecida, convoluído por um padrão *u(m,n)*, e o restante da imagem considerado como ruído:

$$g(m,n) = u(m,n) * \delta(m - m_o, n - n_o) + b(m,n), \tag{2.34}$$

ou

$$g = U * \delta + B. \tag{2.35}$$

De forma a facilitar os cálculos as equações acima são representadas na forma matricial, onde o ponto de partida é a equação abaixo, onde se busca a representação dos vetores e matrizes correspondentes.

$$g_{m,n} = \left\{\sum_{m'}\sum_{n'} u_{m-m',n-n'} * \delta_{m',n'}\right\} + b_{m,n}, \tag{2.36}$$

onde $-T \le m', n' \le T, -k \le m, n \le k, -q \le m - m', n - n' \le q$ e,

$$2k+1 = M_1 + M_2 - 1 \tag{2.37}$$

$$2T+1 = M_2 \text{ e } 2q+1 = M_1 \tag{2.38}$$

Abaixo se tem a representação vetorial para que o cálculo bidimensional seja possível. As imagens são representadas como uma concatenação de linhas transpostas, sem perda de generalidade.

$$g = \begin{bmatrix} g_{-k,-k} \\ g_{-k,-k+1} \\ \cdots \\ g_{-k,+k} \\ \cdots \\ g_{+k,+k} \end{bmatrix}, \tag{2.39}$$

$$\delta = \begin{bmatrix} \delta_{-T,-T} \\ \delta_{-T,-T+1} \\ \cdots \\ \delta_{-T,+T} \\ \cdots \\ \delta_{+T,+T} \end{bmatrix}, \tag{2.40}$$

onde *g* tem dimensão $(2k+1)^2 \times 1$ e δ tem $(2T+1)^2 \times 1$.

Combinando as equações acima se chega à matriz *F* em função das amostras $u_{m,n}$ do padrão, que é parecida com a matriz *F* apresentada na seção anterior.

$$F = \begin{bmatrix} u_{-k+T,-k+T} & u_{-k+T,-k+T-1} & \cdots & u_{-k+T,-k-T} & u_{-k+T-1,-k+T} & \cdots & u_{-k-T,-k-T} \\ u_{-k+T,-k+1+T} & u_{-k+T,-k+T} & \cdots & u_{-k+T,-k+1+T} & u_{-k+T-1,-k+1+T} & \cdots & u_{-k+T,-k+1-T} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ u_{+k+T,+k+T} & u_{+k+T,+k+T-1} & \cdots & u_{+k+T,+k-T} & u_{+k+T-1,+k+T} & \cdots & u_{+k-T,+k-T} \end{bmatrix} \tag{2.41}$$

Fazendo uma comparação entre as matrizes acima e a equação 2.23 percebe-se que não existe um casamento dimensional dos vetores, a menos que se faça algum truncamento ou aproximação. Isto porque *g* tem dimensão $(2k+1)^2 \times 1$, $\delta \rightarrow (2T+1)^2 \times 1$ e *F* tem (2k+1) (2T+1) x (2k+1) (2T+1).

Com esta análise verifica-se que uma solução com convolutores lineares não existe para a matriz *A*, lembrando que $\hat{\delta} = Ag$. Para se ter uma solução aproximada deve-se realizar alguns truncamentos nas matrizes *F* e *A*, mas estas aproximações podem gerar erros não desprezíveis.

A seguir, são mostradas as novas realizações para *F* e *A* que possuem dimensões de $(2T+1)^2 \times (2T+1)^2$. Note que *g* deve ser truncado para $(2T+1)^2 \times 1$.

$$F=\begin{bmatrix}
u_{0,0} & u_{0,-1} & \cdots & u_{0,-2T} & u_{-1,0} & u_{-1,-1} & \cdots & u_{-1,-2T} & u_{-2,0} & \cdots & u_{-2T,-2T} \\
u_{0,1} & u_{0,0} & \cdots & u_{0,-2T+1} & u_{-1,1} & u_{-1,0} & \cdots & u_{-1,-2T+1} & u_{-2,1} & \cdots & u_{-2T,-2T+1} \\
\cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \\
u_{0,2T} & u_{0,2T-1} & \cdots & u_{0,0} & u_{-1,2T} & u_{-1,2T-1} & \cdots & u_{-1,0} & u_{-2,2T} & \cdots & u_{-2T,0} \\
u_{1,0} & u_{1,-1} & \cdots & u_{1,-2T} & u_{0,0} & u_{0,-1} & \cdots & u_{0,-2T} & u_{-1,0} & \cdots & u_{-2T+1,-2T} \\
u_{1,1} & u_{1,0} & \cdots & u_{1,-2T+1} & u_{0,1} & u_{0,0} & \cdots & u_{0,-2T+1} & u_{-1,1} & \cdots & u_{-2T+1,-2T+1} \\
\cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \\
u_{1,2T} & u_{1,2T-1} & \cdots & u_{1,0} & u_{0,2T} & u_{0,2T-1} & \cdots & u_{0,0} & u_{-1,2T} & \cdots & u_{-2T+1,0} \\
u_{2,0} & u_{2,-1} & \cdots & u_{2,-2T} & u_{1,0} & u_{1,-1} & \cdots & u_{1,-2T} & u_{0,0} & \cdots & u_{-2T+2,-2T} \\
\cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \\
u_{2T,2T} & u_{2T,2T-1} & \cdots & u_{2T,0} & u_{2T-1,2T} & u_{2T-1,2T-1} & \cdots & u_{2T-1,0} & u_{2T-2,2T} & \cdots & u_{0,0}
\end{bmatrix} \tag{2.42}$$

$$A=\begin{bmatrix}
\theta_{0,0} & \theta_{0,-1} & \cdots & \theta_{0,-2T} & \theta_{-1,0} & \theta_{-1,-1} & \cdots & \theta_{-1,-2T} & \theta_{-2,0} & \cdots & \theta_{-2T,-2T} \\
\theta_{0,1} & \theta_{0,0} & \cdots & \theta_{0,-2T+1} & \theta_{-1,1} & \theta_{-1,0} & \cdots & \theta_{-1,-2T+1} & \theta_{-2,1} & \cdots & \theta_{-2T,-2T+1} \\
\cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \\
\theta_{0,2T} & \theta_{0,2T-1} & \cdots & \theta_{0,0} & \theta_{-1,2T} & \theta_{-1,2T-1} & \cdots & \theta_{-1,0} & \theta_{-2,2T} & \cdots & \theta_{-2T,0} \\
\theta_{1,0} & \theta_{1,-1} & \cdots & \theta_{1,-2T} & \theta_{0,0} & \theta_{0,-1} & \cdots & \theta_{0,-2T} & \theta_{-1,0} & \cdots & \theta_{-2T+1,-2T} \\
\theta_{1,1} & \theta_{1,0} & \cdots & \theta_{1,-2T+1} & \theta_{0,1} & \theta_{0,0} & \cdots & \theta_{0,-2T+1} & \theta_{-1,1} & \cdots & \theta_{-2T+1,-2T+1} \\
\cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \\
\theta_{1,2T} & \theta_{1,2T-1} & \cdots & \theta_{1,0} & \theta_{0,2T} & \theta_{0,2T-1} & \cdots & \theta_{0,0} & \theta_{-1,2T} & \cdots & \theta_{-2T+1,0} \\
\theta_{2,0} & \theta_{2,-1} & \cdots & \theta_{2,-2T} & \theta_{1,0} & \theta_{1,-1} & \cdots & \theta_{1,-2T} & \theta_{0,0} & \cdots & \theta_{-2T+2,-2T} \\
\cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \\
\theta_{2T,2T} & \theta_{2T,2T-1} & \cdots & \theta_{2T,0} & \theta_{2T-1,2T} & \theta_{2T-1,2T-1} & \cdots & \theta_{2T-1,0} & \theta_{2T-2,2T} & \cdots & \theta_{0,0}
\end{bmatrix} \tag{2.43}$$

Uma forma de evitar os truncamentos e obter respostas mais precisas seria usar a convolução circular, ao invés da linear, e também que se respeite a condição que $M_1 = M_2 = T$. Dessa forma, as nova matrizes a serem usadas durante a restauração do impulso são

$$F_{circ} = \begin{bmatrix} u_{0,0} & u_{0,-1} & \cdots & u_{0,+1} & u_{-1,0} & \cdots & u_{+1,+1} \\ u_{0,+1} & u_{0,0} & \cdots & u_{0,+2} & u_{-1,+1} & \cdots & u_{+1,+2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ u_{-1,-1} & u_{-1,-2} & \cdots & u_{-1,0} & u_{-2,-1} & \cdots & u_{0,0} \end{bmatrix} \tag{2.44}$$

$$A_{circ} = \begin{bmatrix} \theta_{0,0} & \theta_{0,-1} & \cdots & \theta_{0,+1} & \theta_{-1,0} & \cdots & \theta_{+1,+1} \\ \theta_{0,+1} & \theta_{0,0} & \cdots & \theta_{0,+2} & \theta_{-1,+1} & \cdots & \theta_{+1,+2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \theta_{-1,-1} & \theta_{-1,-2} & \cdots & \theta_{-1,0} & \theta_{-2,-1} & \cdots & \theta_{0,0} \end{bmatrix} \tag{2.45}$$

As matrizes F_{circ} e A_{circ} serão chamadas daqui por diante de *F* e *A*. Uma nova notação baseada por blocos é adotada, para facilitar os cálculos. A nova matriz, responsável por essa mudança na notação, é chamada de H_r, que é um operador que transforma uma linha *r* de uma janela genérica v (2*N*+1 x 2*N*+1) numa matriz circulante de dimensão 2*N*+1 x 2*N*+1, de acordo com a equação abaixo.

$$H_r(v) = \begin{bmatrix} v_{r,0} & v_{r,-1} & v_{r,-2} & \cdots & v_{r,+3} & v_{r,+2} & v_{r,+1} \\ v_{r,+1} & v_{r,0} & v_{r,-1} & \cdots & v_{r,+4} & v_{r,+3} & v_{r,+2} \\ \cdots & & \cdots & & \cdots & & \cdots \\ v_{r,+N} & v_{r,+N-1} & v_{r,+N-2} & \cdots & v_{r,-N+2} & v_{r,-N+1} & v_{r,-N} \\ v_{r,-N} & v_{r,+N} & v_{r,+N-1} & \cdots & v_{r,-N+3} & v_{r,-N+2} & v_{r,-N+1} \\ \cdots & & \cdots & & \cdots & & \cdots \\ v_{r,-1} & v_{r,-2} & v_{r,-3} & \cdots & v_{r,+2} & v_{r,+1} & v_{r,0} \end{bmatrix} \tag{2.46}$$

Dessa forma, usando a matriz H_r tem-se a nova representação para as matrizes F e A:

$$F = \begin{bmatrix} H_0(U) & H_{-1}(U) & \cdots & H_{-T}(U) & H_T(U) & \cdots & H_2(U) & H_1(U) \\ H_1(U) & H_0(U) & \cdots & H_{-T+1}(U) & H_{-T}(U) & \cdots & H_3(U) & H_2(U) \\ \cdots & & \cdots & & & \cdots & & \cdots \\ H_T(U) & H_{T-1}(U) & \cdots & H_0(U) & H_{-1}(U) & \cdots & H_{-T+1}(U) & H_{-T}(U) \\ H_{-T}(U) & H_T(U) & \cdots & H_1(U) & H_0(U) & \cdots & H_{-T+2}(U) & H_{-T+1}(U) \\ \cdots & & \cdots & & & \cdots & & \cdots \\ H_{-2}(U) & H_{-3}(U) & \cdots & H_{T-1}(U) & H_{T-2}(U) & \cdots & H_0(U) & H_{-1}(U) \\ H_{-1}(U) & H_{-2}(U) & \cdots & H_T(U) & H_{T-1}(U) & \cdots & H_1(U) & H_0(U) \end{bmatrix} \quad (2.47)$$

$$A = \begin{bmatrix} H_0(\Theta) & H_{-1}(\Theta) & \cdots & H_{-T}(\Theta) & H_T(\Theta) & \cdots & H_2(\Theta) & H_1(\Theta) \\ H_1(\Theta) & H_0(\Theta) & \cdots & H_{-T+1}(\Theta) & H_{-T}(\Theta) & \cdots & H_3(\Theta) & H_2(\Theta) \\ \cdots & & \cdots & & & \cdots & & \cdots \\ H_T(\Theta) & H_{T-1}(\Theta) & \cdots & H_0(\Theta) & H_{-1}(\Theta) & \cdots & H_{-T+1}(\Theta) & H_{-T}(\Theta) \\ H_{-T}(\Theta) & H_T(\Theta) & \cdots & H_1(\Theta) & H_0(\Theta) & \cdots & H_{-T+2}(\Theta) & H_{-T+1}(\Theta) \\ \cdots & & \cdots & & & \cdots & & \cdots \\ H_{-2}(\Theta) & H_{-3}(\Theta) & \cdots & H_{T-1}(\Theta) & H_{T-2}(\Theta) & \cdots & H_0(\Theta) & H_{-1}(\Theta) \\ H_{-1}(\Theta) & H_{-2}(\Theta) & \cdots & H_T(\Theta) & H_{T-1}(\Theta) & \cdots & H_1(\Theta) & H_0(\Theta) \end{bmatrix} \quad (2.48)$$

A partir das equações acima, percebe-se que as mesmas são circulantes por blocos. A questão agora é garantir que $F^t\left(FF^t + C_b\right)^{-1}$ tenha seus termos aparecendo nos formatos genéricos nas equações anteriores fazendo desta forma, que A seja um operador convolutor. Mas, por [13], vê-se que a inversa de uma matriz circulante por blocos, se ela existir, também é circulante por blocos.

É visto em [7], que ao fazer F circulante por blocos ao invés de só circulante como na abordagem original, se permite que o padrão mantenha sua forma com qualquer deslocamento (m_0, n_0) tanto na horizontal como na vertical.

2.4 Discriminação de múltiplos padrões

Dentre as aplicações do reconhecimento de padrões, uma delas é poder desenvolver um método, neste caso um filtro, que consiga separar alguns tipos de

padrões daqueles que não queremos reconhecer. Por exemplo, pode-se fazer um filtro que dê uma $DSNR_2$ alta para as vogais "A", "E", "O" e dê uma resposta baixa para todas as outras letras, basta para isso que se confeccione um filtro somente com as vogais acima citadas. A esse filtro chama-se de Filtro tipo "OU" [7].

2.4.1 Filtro tipo "OU"

O cálculo do filtro OU é baseado na restauração de impulso. Dessa forma, adota-se novamente a notação anterior, onde *g* é um vetor coluna formado pela concatenação das linhas transpostas da imagem, F_i é a matriz circulante referente ao padrão *i* (como definida na equação 2.24) e *b* é o vetor ruído aditivo correspondente ao resto da imagem. Assim, como anteriormente, o objetivo é estimar o vetor δ (impulso) que indica a posição do padrão que se está procurando.

Como na seção 2.2, supõe-se que o ruído *b* seja gaussiano, de média nula e matriz covariância *1/N* C_b e que o vetor aleatório δ tenha matriz autocorrelação 1/*N* vezes a matriz identidade **I**. A melhor estimativa para δ ou $\hat{\delta}$, que minimize o erro médio quadrático $E\left[\left\|\delta-\hat{\delta}\right\|^2\right]$ e que seja linear em relação a *g* é aquela onde δ - $\hat{\delta}$ seja descorrelacionado de *g* (princípio da ortogonalidade). Assim deve-se calcular *A* tal que

$$E\left\{\left(\delta-\hat{\delta}\right)g^t\right\}=E\left\{\left(\delta-Ag\right)g^t\right\}=0 \tag{2.49}$$

Sabendo que cada padrão *i* pode ter probabilidade $p_{i,}$ pode-se representar a equação anterior por,

$$E\left\{\left(\delta-Ag\right)g^t\right\}=\sum_i p_i E\left\{\left(\delta-AF_i\delta-Ab\right)\left(F_i\delta+b\right)^t\right\}$$

$$=\sum_i p_i\left\{E\left\{\delta\delta^t F_i^t\right\}+E\left\{\delta b^t\right\}-E\left\{AF\delta\delta^t F^t\right\}-E\left\{AF\delta b^t\right\}-E\left\{Ab\delta^t F^t\right\}-E\left\{Abb^t\right\}\right\} \tag{2.50}$$

$$=\sum_i p_i\left\{E\left\{\delta\delta^t\right\}F^t+E\left\{\delta b^t\right\}-AFE\left\{\delta\delta^t\right\}F^t-AFE\left\{\delta b^t\right\}-AE\left\{b\delta^t\right\}F^t-AE\left\{bb^t\right\}\right\}=0$$

Considerando δ e b descorrelacionados, pode-se aplicar as seguintes relações:

$$E\left\{\delta\hat{\delta}\right\} = \frac{1}{N} I, \tag{2.51}$$

$$E\left\{\delta b^t\right\} = 0, \tag{2.52}$$

$$E\left\{bb^t\right\} = \frac{1}{N} C_b, \tag{2.53}$$

Assim, a equação 2.50 pode ser reescrita da seguinte forma,

$$\sum_i p_i \left\{F_i^t - AF_i F_i^t - AC_b\right\} = 0 \tag{2.54}$$

Conseqüentemente,

$$A = \sum_i \left\{p_i F_i^t\right\} \left\{C_b + \sum_i p_i F_i F_i^t\right\}^{-1}. \tag{2.55}$$

Note que a equação acima depende de um somatório de cada padrão ponderado pelas suas probabilidades correspondentes (primeiro termo) e também da covariância do restante da imagem no segundo termo. Ou seja, dependendo da aplicação, seria de grande importância um pré-conhecimento do restante (“background”) da imagem de forma a fazer um filtro mais preciso.

2.4.2 Limiar de detecção

A escolha do limiar para a detecção de um certo padrão é de grande importância em trabalhos de reconhecimento de padrões utilizando a restauração do impulso, e este varia para cada aplicação. Como pode ser visto em [7], o cálculo deste limiar pode ser realizado pelo método de Neyman-Pearson, onde se supõe uma distribuição do ruído, se fixa uma probabilidade para falsas detecções e calcula-se então o limiar correspondente.

Mas esse método é mais eficaz quando se têm as texturas da imagem bem caracterizadas. Neste trabalho, como o objetivo é poder fazer a detecção dos gabaritos independente da textura da imagem, o valor de limiar é escolhido experimentalmente tendo como base os valores utilizados nos trabalhos realizados por Mendonça [7].

2.5 Exemplo com o filtro “OU”

Neste exemplo é mostrado um filtro OU (equação 2.55) que foi treinado pelas vogais “A”, “E” e “O”, que deverão neste caso apresentar uma $DSNR_2$ alta. Em comparação serão apresentadas outras letras para que possa perceber a resposta do filtro OU. O exemplo será dado para letras com fontes *Arial* e tamanho 8, contidas em janelas 9x9, A matriz de covariância C_b tem seu valor igual a 0, porque a cor de plano de fundo da imagem é preta, ou seja, com valor 0. Todas as letras, a menos da letra *I* foram alinhadas deixando uma coluna no lado direito, todas também estão alinhadas com a parte inferior da imagem. Abaixo, na figura 2.2, apresentam-se as letras utilizadas aqui neste primeiro teste:

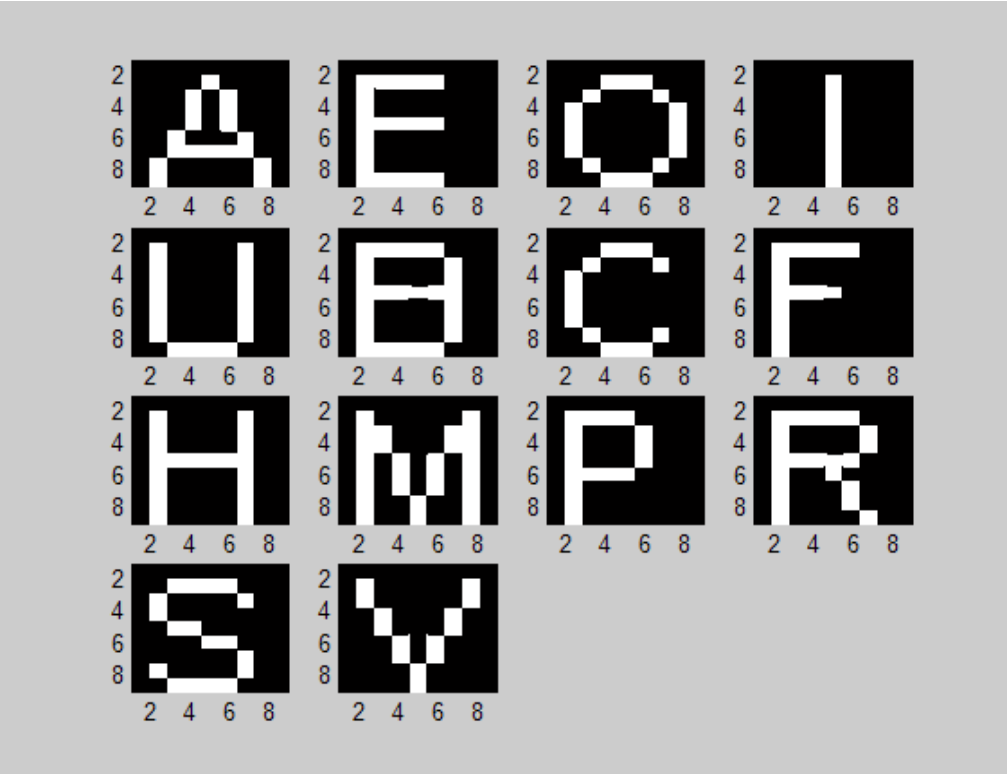


Figura 2.2. Letras a serem utilizadas no primeiro exemplo.

A tabela 2.1 contém o valor das $DSNR_2$ para cada uma das letras que estão no conjunto acima. As probabilidades de cada uma das letras que participaram do treinamento (“A”, “E”, “O”) são iguais a 1/3.

Tabela 2.1. Tabelas de $DSNR_2$ com probabilidades iguais.

Letra	A	E	O	I	U	B	C
$DSNR_2$	0,3674	0,4681	0,7465	0	0,0519	0,3937	0,5532
Letra	F	H	M	P	R	S	V
$DSNR_2$	0,2985	0,1070	0,1504	0,1603	0,2254	0,1312	0,0658

Neste exemplo a $DSNR_2$ das letras "*A"*, *"E"*, *"O"*, como esperado apresentam valores altos (acima de 0,3), mas isso não quer dizer que o filtro esteja funcionando perfeitamente: analisando a tabela acima percebe-se que outras letras também apresentaram $DSNR_2$ com valor alto. Além disso há casos de $DSNR_2$'s maiores do que as encontradas para letras que fizeram o treinamento do filtro "OU".

O principal motivo para a ocorrência destes problemas é que, como se trata de letras de baixa resolução (fonte tamanho 8), uma simples adição ou subtração de poucos pixels pode transformar uma letra em outra. Por exemplo, as letras "*B"* e "*C"*, por serem parecidas com as letras "*E"* e "*O"*, possuem DSNR alta e maior que a da letra "*A"*. A diferença entre a letra *"E"* e *"B"* é de 4 pixels e entre as letras *"O"* e *"C"* de 3 pixels.

Uma tentativa de corrigir o problema acima citado seria manipular os coeficientes p_i, presentes na equação 2.55 do filtro OU, das letras que confeccionaram o filtro, de forma a tentar igualar suas DSNR's e conseqüentemente deixar as DSNR's de "*B"* e "*C"* menores que a da letra "*A"*.

Na verdade estes coeficientes p_i (probabilidades) são usadas, nesta tese, como pesos de ajuste, que tem como objetivo gerar uma $DSNR_2$ maior para um determinado padrão de imagem. Isto significaria que, se cada padrão tiver uma probabilidade igual ao peso dado, resultaria num filtro "OU" com menor erro quadrático para a estimação do impulso.

Os valores destes pesos, na tabela 2.2, foram determinados experimentalmente de modo a melhorar a detecção do "A" . Os novos valores de $DSNR_2$ obtidos encontram-se na tabela 2.3.

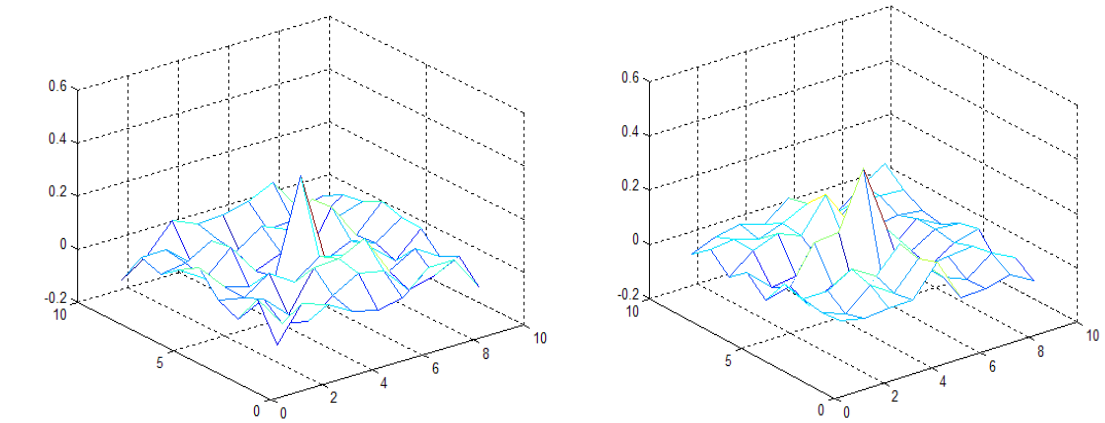
Tabela 2.2. Tabelas de pesos.

Letra	p_i
A	0,3623
E	0,3478
O	0,2899

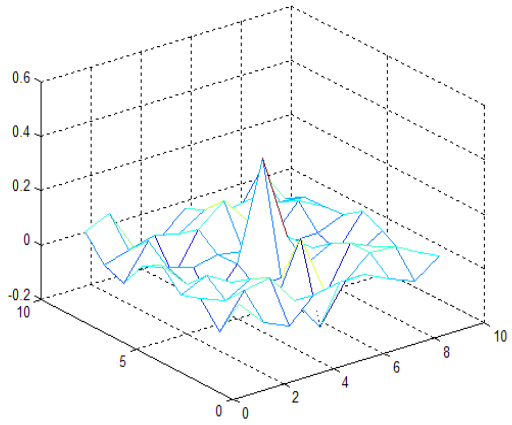
Tabela 2.3. Tabelas de $DSNR_2$ modificada pelos pesos.

Letra	A	E	O	I	U	B	C
$DSNR_2$	0,4923	0,4820	0,5365	0	0,0480	0,4023	0,4344
Letra	F	H	M	P	R	S	V
$DSNR_2$	0,3132	0,1124	0,1496	0,1681	0,2354	0,1274	0,0658

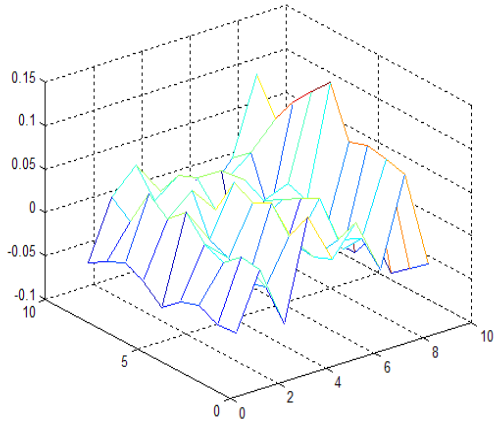
Para este teste, utilizando estas fontes, pode-se escolher o limiar de detecção de $DSNR_2$ no valor de 0,45. Desta forma, de acordo com a tabela 2.3, poder-se-ia discriminar sem erros as letras "*A", "E"* e *"O"*, que foram usadas para construir o filtro "OU", das outras letras que deram problemas antes da mudança dos pesos como as letras "*B"*, "*C"* e também a letra "F", que ainda possuem $DSNR_2$ bem altas. Abaixo segue o gráfico das DSNR's para cada uma das letras deste exemplo.



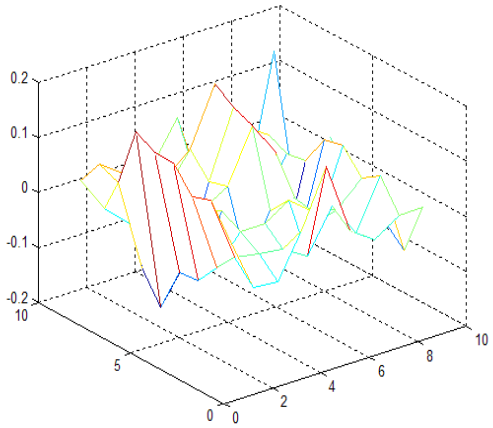
(a) (b)


0.6
0.4
0.2
0
-0.2
10
5
0
0
2
4
6
8
10

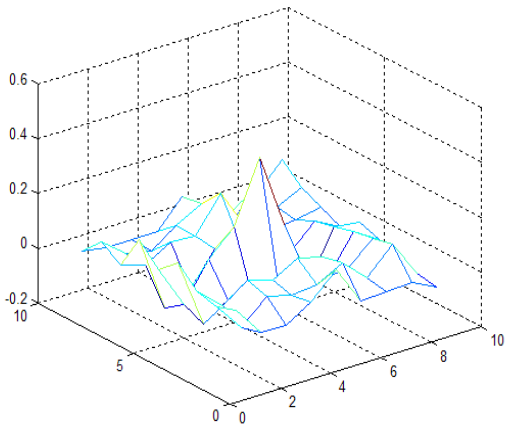
(c)


0.15
0.1
0.05
0
-0.05
-0.1
10
5
0
0
2
4
6
8
10

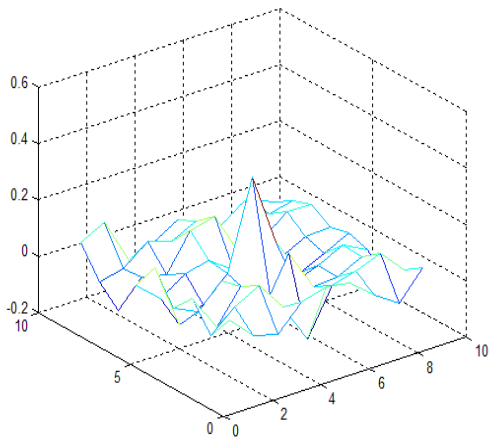
(d)


0.2
0.1
0
-0.1
-0.2
10
5
0
0
2
4
6
8
10

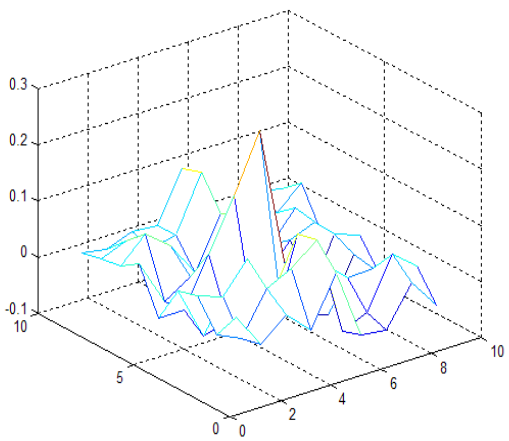
(e)


0.6
0.4
0.2
0
-0.2
10
5
0
0
2
4
6
8
10

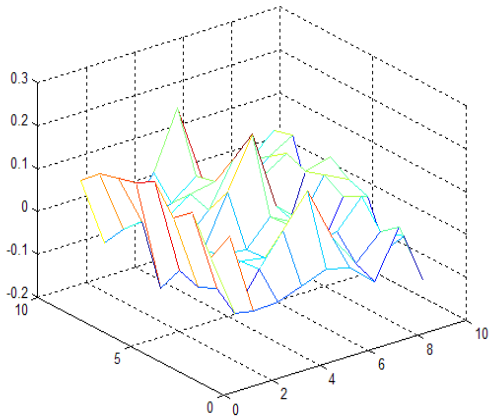
(f)


0.6
0.4
0.2
0
-0.2
10
5
0
0
2
4
6
8
10

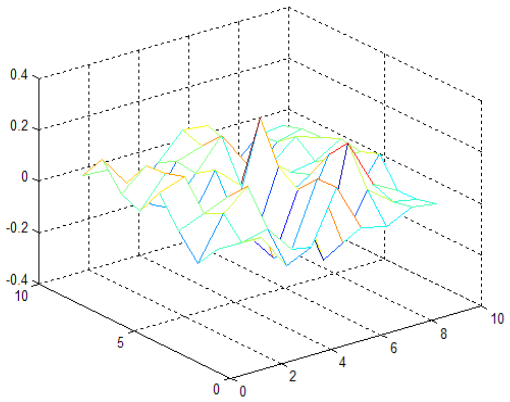
(g)


0.3
0.2
0.1
0
-0.1
10
5
0
0
2
4
6
8
10

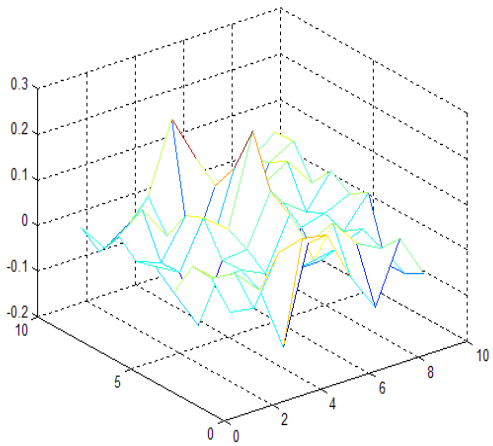
(h)



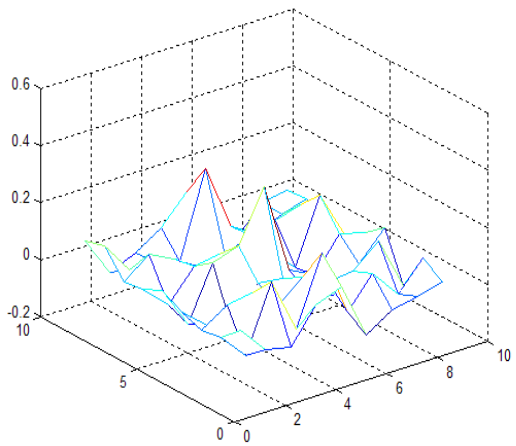
(i)



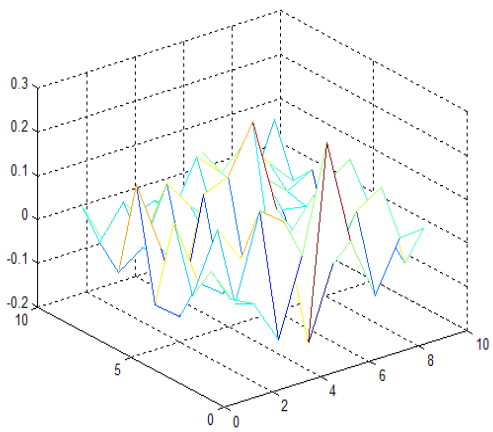
(j)



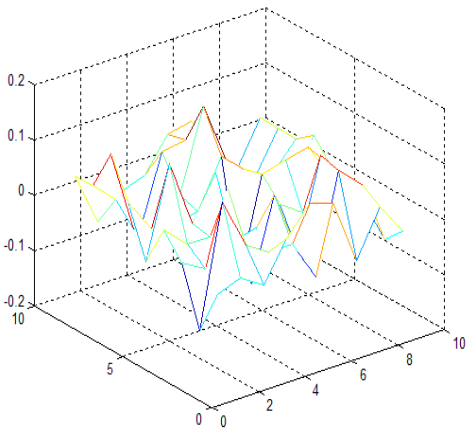
(k)



(l)



(m)



(n)

Figura 2.3. (a) Gráfico da DSNR da letra A (b) letra E (c) letra O (d) letra I (e) letra U (f) letra B (g) letra C (h) letra F (i) letra H (j) letra M (k) letra P (l) letra R (m) letra S (n) letra V

Capítulo 3

Discriminação de padrão robusta ao ruído

No capítulo anterior, foi apresentada uma aplicação da filtragem discriminativa via restauração de impulso, que é o filtro tipo “OU”. Como já foi dito, a sua função é discriminar um determinado grupo de padrões, que deu origem ao filtro, de qualquer outro que venha a ser testado. A partir disso, imaginou-se porque não criar um filtro “OU” que possa reconhecer, por exemplo, todos os tipos de letra “A”, e conseqüentemente outros filtros que detectassem cada uma das outras letras do alfabeto? Inicialmente, pensar-se-ia em pegar o máximo de fontes para uma determinada letra e fá-se-ia um filtro “OU” que as detectasse (veja tabela 3.3). Isto, teoricamente, daria certo já que se tentaria discriminar padrões que foram usados no treinamento do filtro “OU”, mas neste caso dependeríamos diretamente das fontes escolhidas para a confecção do filtro, o que poderia gerar erros ao se testarem outros tipos.

Imaginou-se então uma forma de representar as mais variadas formas de uma mesma letra e, a partir de algumas imagens mais genéricas (não necessariamente a própria fonte), confeccionar um filtro “OU” que pelo menos possa discriminar a maior parte delas.

Analisando novamente a equação 2.55, que é repetida abaixo, pode-se interpretar o primeiro termo do produto como definindo uma função probabilidade (PF) tal que, para cada padrão, tem-se uma probabilidade definida. O segundo termo é uma constante multiplicativa, mas que depende também de uma PF, mas agora de $F_i F_i^t$.

$$A = \sum_i \left\{ p_i F_i^t \right\} \left\{ C_b + \sum_i p_i F_i F_i^t \right\}^{-1} .$$

A equação acima sugere que se represente um padrão através de sua PF. Isto poderia ser feito criando imagens genéricas que gerassem esta PF. Como já foi explicado anteriormente, as probabilidades de cada padrão serão tratadas como pesos de

ajuste de forma a melhorar a $DSNR_2$ para um determinado gabarito. A seguir, uma figura com uma possível PF para a letra A (também chamada aqui de PF_A):

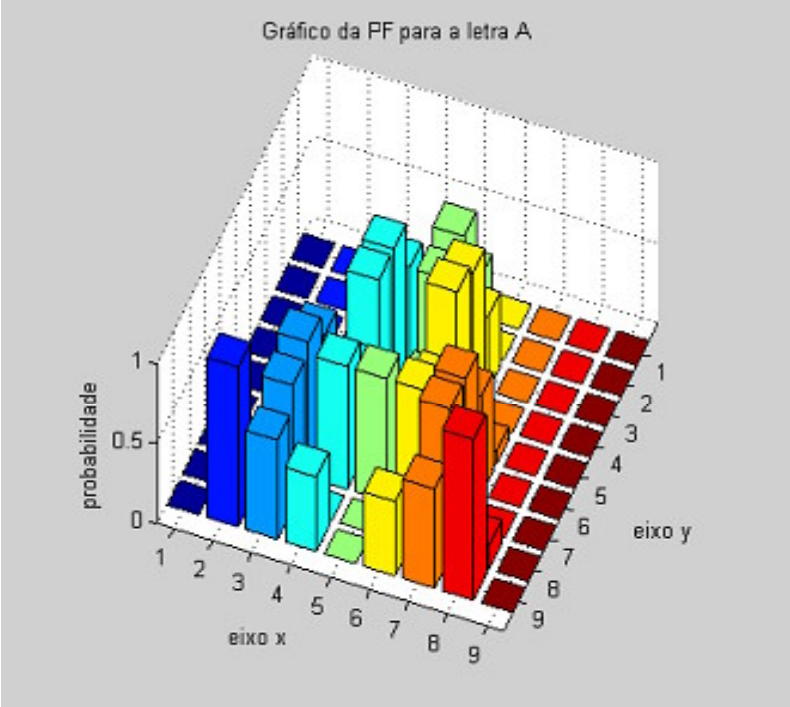


Figura 3.1. Gráfico com possível PF para a letra A - PF_A.

Considerando especificamente o caso da letra "A", tem-se que o segundo problema seria como determinar qual o conjunto de pseudo-A's (conjunto de padrões que se parecem com a letra "A"), presentes na PF_A, que seria usado para gerar o filtro "OU", ou seja, como a PF_A seria "decomposta". A partir disto, através de vários experimentos, surgiram alguns métodos de construção da PF_A (ou das imagens que a compõem) que evoluíram até surgir um algoritmo satisfatório.

Antes de começar a descrição dos métodos que serão propostos, é necessário explicar que, para todos os experimentos, a letra "A" de fonte tamanho 8 será usada como a fonte-teste, pois ela se mostra a letra que mais varia de forma de acordo com as fontes existentes. Todas elas estarão contidas em uma janela 9x9. A figura 3.2 e a tabela 3.1 mostram os conjuntos de fontes que foram utilizados como base nos métodos que são aqui apresentados. De forma a facilitar a visualização dos resultados de cada método que será proposto a seguir, no final deste capítulo é mostrada uma tabela resumo com os melhores resultados para cada método.

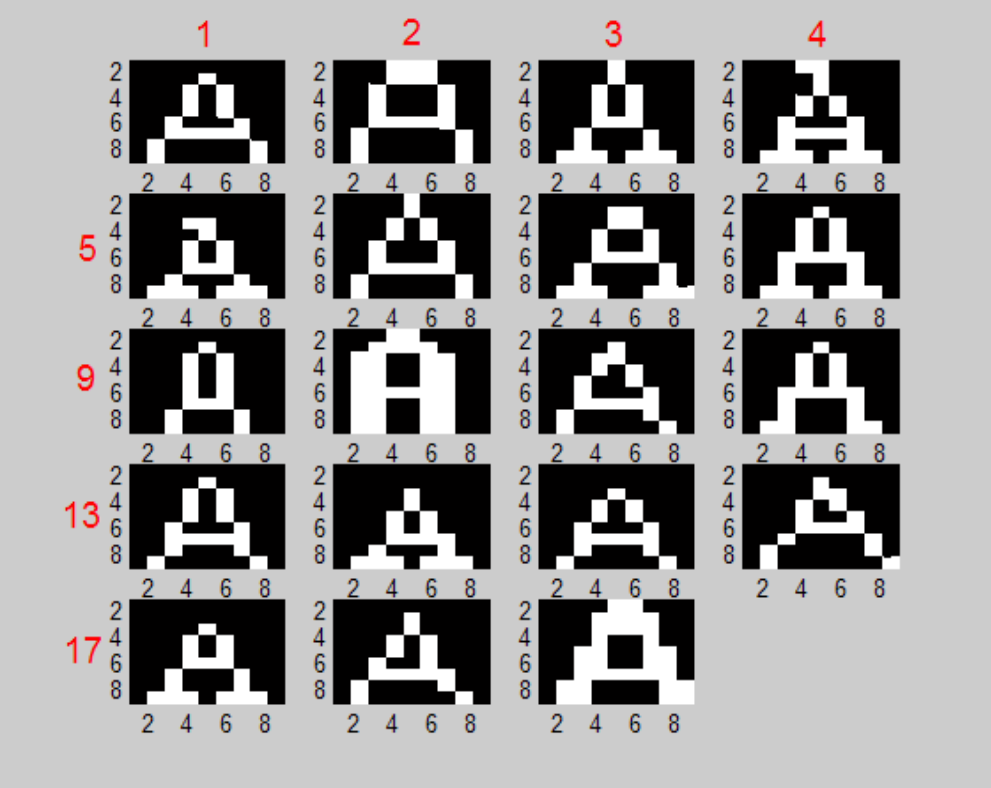


Figura 3.2. Conjunto de letras “A” usado nos testes.

Tabela 3.1. Nome das fontes utilizadas.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	11	Ms Outlook
2	Verdana	12	Bookman Old Style
3	Times New Roman	13	Lúcida Sans
4	Courier	14	Dauphin
5	Courier New	15	Lúcida Console
6	Ms Sans Serif	16	Kabel Bk Bt
7	Geórgia	17	Garamond
8	Book Antigua	18	Futura Lt BT
9	Arial Narrow	19	System
10	Fixed Sys		

Visto o conjunto de fontes que foram usadas nos testes desta tese, a primeira idéia para criar um filtro “OU” que detectasse qualquer tipo de fonte de uma determinada letra seria construir um filtro “OU” com o máximo de fontes conhecidas possível, no caso deste trabalho usando as 19 fontes mostradas na figura 3.2.

Outro dado importante para os testes é que o limiar de detecção é de $DSNR_2$ igual a 0,3. Este valor foi escolhido tendo como base trabalhos anteriores. Foi visto em [7]

que o valor do limiar foi escolhido experimentalmente, e na maior parte das vezes eram abaixo de 0,3. Por isso escolheu-se um valor que fosse acima da média dos já utilizados em [7], de forma a garantir uma discriminação correta dos padrões. É necessário dizer também que este valor é usado para os testes com todas as fontes e letras na fase de treinamento do filtro "OU", que serão apresentados neste capítulo e nos resultados para outras letras no Apêndice A.

Os resultados deste teste são calculados para dois casos diferentes de projeto do Filtro "OU". A diferença destes casos está relacionada com os pesos (p_i) dados, para cada uma das fontes na confecção do filtro "OU". Os pesos são ajustados de acordo com as fontes que não são detectadas, onde seus pesos são ajustados (eleva-se o valor dos pesos) pelo usuário até que a $DSNR_2$ para fonte for satisfatória ou estiver acima do limiar de discriminação pré-estabelecido. Abaixo são apresentados os resultados referentes a este filtro para dois casos:

1. O Caso I se refere à primeira coluna de resultados da tabela onde todas as fontes que compõem o filtro "OU" possuem o mesmo peso;
2. O Caso II, as fontes que compõem o filtro têm pesos diferentes.

Tabela 3.2. $DSNR_2$ obtidas pelo filtro com as fontes originais e primeiro método.

Fontes	Fontes originais I	Fontes Originais II
1	0,8112	0,7509
2	0,0439	0,0549
3	0,7444	0,6648
4	0,8974	0,7707
5	0,5176	0,4323
6	0,5421	0,5308
7	0,2830	0,3702
8	0,9204	0,8408
9	0,6408	0,5773
10	0,1183	0,1205
11	0,6888	0,6290
12	0,9128	0,8719
13	1,4086	1,1330
14	0,4605	0,3809
15	0,7916	0,6718
16	0,1745	0,3204
17	0,4659	0,4705
18	0,8121	0,7317
19	0,2979	0,3322

Desta forma, o primeiro caso com as fontes da figura 3.2 mostra que, fazendo um filtro “OU” (equação 2.55) com as fontes originais, algumas fontes não são detectadas mesmo participando da confecção do filtro. As fontes que não foram detectadas são 2,7,10,16 e 19, onde todas apresentaram $DSNR_2$ menor que 0,3.

Neste primeiro caso, todas as fontes foram treinadas com o mesmo peso e obtendo um desempenho de 14 acertos em 19 possíveis. No segundo caso (terceira coluna da tabela), houve uma mudança nos pesos de algumas fontes na confecção do filtro, de forma a encontrar um melhor resultado. Neste caso, pesos de algumas fontes que não haviam sido detectadas foram elevadas, que aqui no caso foram as fontes 16 e 19. Com isso, o filtro apresentou um desempenho melhor de 17 acertos em 19, onde somente as fontes 2 e 10 continuaram não sendo detectadas.

Abaixo, encontra-se a tabela com o valor dos pesos para os conjuntos das fontes.

Tabela 3.3. Pesos das fontes originais, para o caso II.

Fontes	Pesos
Fonte 16	2.5/21
Fonte 19	1.5/21
Outras fontes	1/21

Foi visto, através de testes, que aumentar o valor dos pesos das fontes 2 e 10 melhora o valor de suas $DSNR_2$, mas não o suficiente para que assuma o valor requerido. Visto também que o aumento de sua $DSNR_2$ implica na diminuição desta medida para as outras fontes. A seguir então serão apresentadas as explicações de cada método.

3.1 Método divisão em planos de freqüência

O primeiro teste realizado visando a encontrar uma PF_A foi somar as dezenove fontes-testes e, a partir delas, fazer uma distribuição da freqüência de seus pixels espacialmente. Desta forma, a partir de um conjunto limitado de fontes, gerou-se uma PF em duas dimensões que contém tanto os “A’s” da figura 3.2, como também outros tipos. A seguir, na figura 3.3, é apresentada a imagem-soma das letras “A”, representada em escala de cinza, onde a freqüência dos pixels varia da cor preta até a branca, ou seja, de 0 a 1.

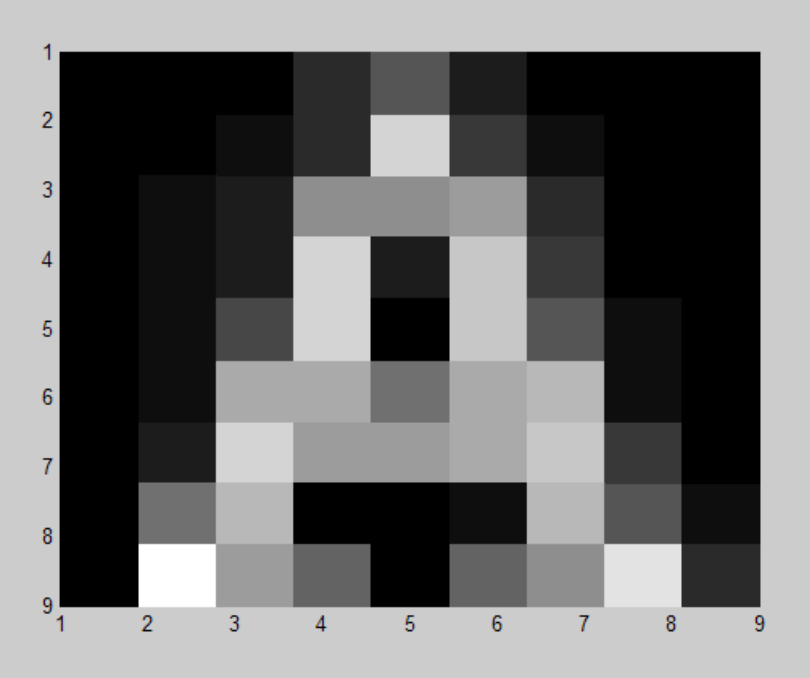


Figura 3.3. Soma das 19 letras “A”.

A soma das fontes-testes também pode ser representada em três dimensões onde no eixo z é representada a freqüência normalizada de cada pixel, veja figura 3.4 abaixo:

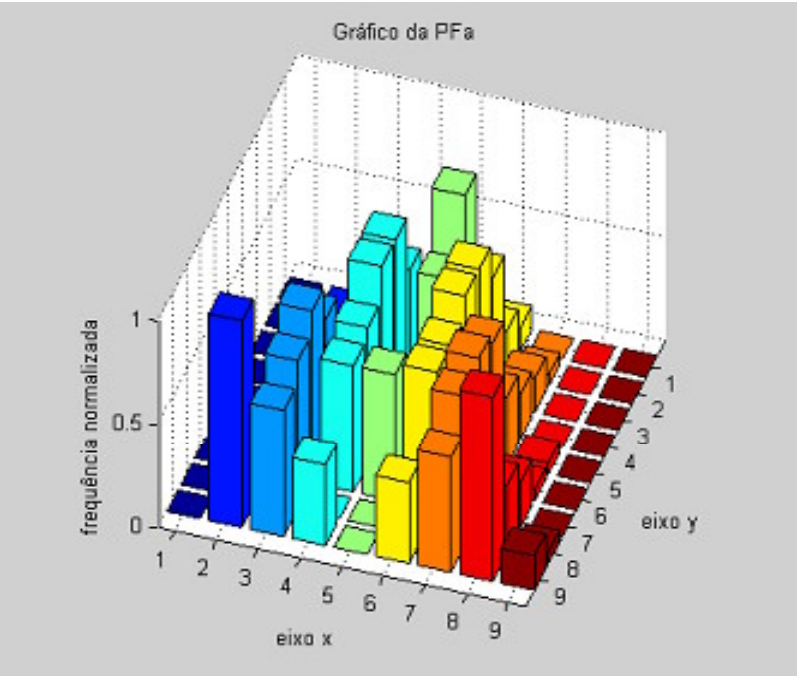


Figura 3.4. PF_A para o método de divisão em planos.

O segundo passo do método consiste em aplicar uma divisão em planos (ou fatias) na PF_A em cada nível de freqüência, ou seja, dividir a PF_A que está em três dimensões, em um conjunto de imagens bidimensionais, que representam cada nível de freqüência da PF, e que somadas a origine novamente. Veja a formulação abaixo:

Seja a PF_A, cuja amplitude varia de [0,A], e n o seu número de níveis de freqüência. Cada plano ou fatia i é tal que $x_i \in [a_i, a_{i+1})$, onde i varia de 1 até n e a_i corresponde a amplitude de cada nível de freqüência.

Desta forma, formar-se-iam alguns padrões de imagens chamados de pseudo-A's, que lembram as formas de uma letra "A". A figura 3.5 mostra então essas novas imagens geradas pela divisão, que são organizadas do plano mais baixo até os plano mais alto de freqüência da PF_A.

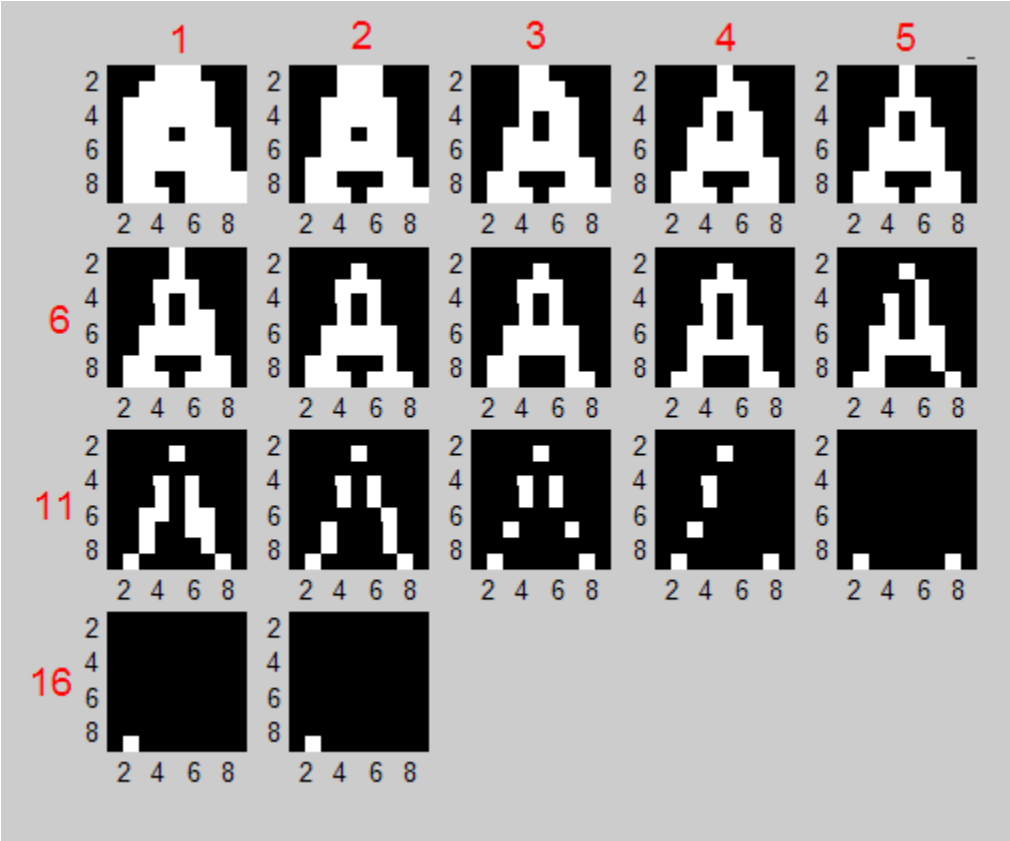


Figura 3.5. Imagens geradas pela divisão de planos de freqüência da PF_A.

Para este método de divisão em planos, foram analisados cinco casos, onde para cada um deles foi confeccionado um filtro "OU" diferente. A principal diferença entre os casos está no conjunto de imagens que irão compor o filtro (pseudo-A's). A escolha deste conjunto se baseou no fato de que algumas imagens-planos (as primeiras e as

últimas) contidas na figura acima não têm semelhança alguma com a letra “A”, com isso algumas combinações de imagens foram realizadas no intuito de se melhorar o desempenho do filtro “OU”. Abaixo, estão listados cinco casos de conjuntos de imagens-planos que serão usadas no treinamento do filtro:

- Caso I – Serão usadas todas as imagens que se originaram da divisão por planos;
- Caso II – As imagens 14,15,16,17 são excluídas da confecção do filtro;
- Caso III – As imagens 1,14,15,16,17 são excluídas da confecção do filtro;
- Caso IV – As imagens 1,2,14,15,16,17 são excluídas da confecção do filtro;
- Caso V – As imagens 1,2,13,14,15,16,17 são excluídas da confecção do filtro.

Abaixo é apresentado o resumo passo a passo deste método através do diagramas de blocos:

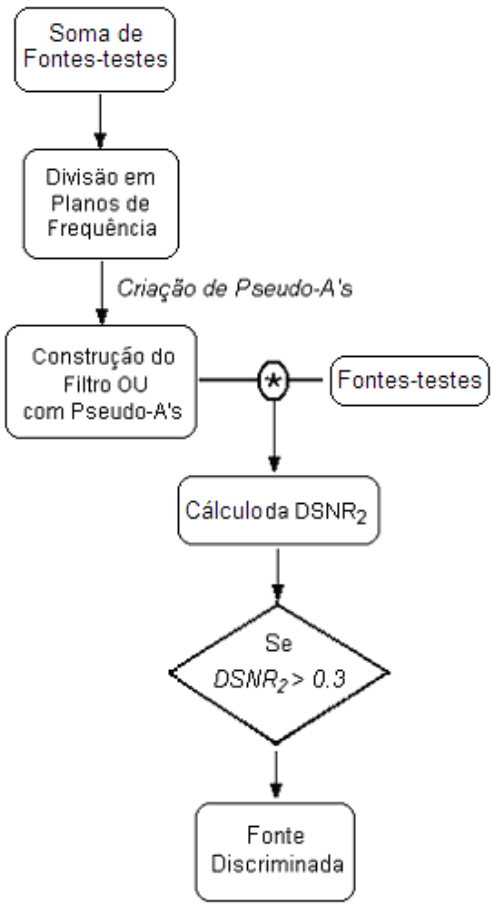


Figura 3.6. Diagrama de blocos descrevendo este primeiro método

Com relação à simulação do método, o valor de $DSNR_2$ limite que será utilizado, neste e nos outros testes desta tese para a discriminação de um gabarito, será de 0,3. Como já foi comentado, este valor foi escolhido após vários experimentos e também por apresentar um valor alto em relação aos demais valores apresentados através de simulações já realizadas em [7]. Abaixo encontra-se a tabela com a $DSNR_2$ para cada um dos casos e para cada fonte-teste (figura 3.2). Também é apresentado o valor médio de $DSNR_2$ para cada caso e também o valor do desvio padrão médio. Estes parâmetros visam ajudar na escolha do melhor caso, onde se deseja um compromisso entre uma média alta de $DSNR_2$ e pouca dispersão entre seus valores. Estes parâmetros são usados em todos os testes realizados nesta tese.

Tabela 3.4: Resultados com o valor da $DSNR_2$ para cada um dos casos.

Fonte	**Caso I**	**Caso II**	**Caso III**	**Caso IV**	**Caso V**
1	0,6358	0,6367	0,6744	0,6864	0,6902
2	0,0104	0,0043	0,0037	0,0004	0,0000
3	0,6472	0,5363	0,5321	0,5204	0,3866
4	0,8789	0,9226	0,9937	1,0594	1,0168
5	0,4520	0,4380	0,4438	0,4316	0,4509
6	0,4778	0,4626	0,4809	0,4700	0,4551
7	0,2505	0,1982	0,1889	0,1841	0,1548
8	0,8877	0,7830	0,8134	0,8389	0,7407
9	0,5019	0,6219	0,6143	0,6133	0,6579
10	0,1005	0,0964	0,0730	0,0694	0,0613
11	0,5567	0,5551	0,6175	0,5931	0,5902
12	0,7837	0,6159	0,6211	0,6367	0,5546
13	1,1471	1,0423	1,1630	1,2872	1,1998
14	0,4396	0,4239	0,4320	0,4069	0,4019
15	0,5914	0,5717	0,6357	0,6498	0,6467
16	0,1266	0,1140	0,1107	0,0917	0,0797
17	0,4480	0,3966	0,4091	0,4134	0,3177
18	0,6743	0,6401	0,6936	0,6762	0,6174
19	0,2755	0,2877	0,2532	0,2354	0,2380
Média da $DSNR_2$ (μ)	0,5203	0,4920	0,5134	0,5192	0,4874
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,2232	0,2112	0,2342	0,2494	0,2450
$\left(\overline{DP}\right)$**/(μ)**	0,4289	0,4292	0,4562	0,4805	0,5028
Acertos	14	14	14	14	14

Mesmo com estas tentativas para melhorar os resultados, todos os cinco casos apresentaram o mesmo problema, onde as fontes 2,7,10,16,19 (figura 3.2) apresentaram

$DSNR_2$ menor que 0,3, e que por isso não podem ser consideradas letra “A”, e desta forma, os cinco casos apresentaram 14 acertos em 19 possíveis.

Analisando os resultados das 19 fontes para os cinco casos, nota-se que o primeiro caso, além de apresentar os melhores resultados para as fontes com $DSNR_2$ baixa, ainda possui a maior média de $DSNR_2$ e a menor relação $\left(\overline{DP}\right)/(\mu)$. Por isso, ele foi escolhido como o resultado que representará este método, e portanto, estará presente na tabela resumo no fim deste capítulo. Para as outras fontes que já possuíam $DSNR_2$ maior que 0,3, houve apenas uma variação neste valor com algumas aumentando e outras diminuindo a $DSNR_2$.

Outra questão importante é que observando a figura 3.2, percebe-se que justamente as fontes 2, 10 e 16, que são os casos mais críticos, são as que mais se diferenciam morfologicamente do restante do grupo. Isso pode ser uma das causas porque apresentam uma $DSNR_2$ tão baixa.

A conclusão que pode se tirar deste primeiro método é que, apesar de ter uma boa margem de acerto (14 acertos em 19), este método não oferece liberdade ao usuário, como no exemplo em que se utilizaram as próprias fontes-testes para a confecção do filtro “OU” de poder mudar, por exemplo, os pesos das fontes na equação do filtro. Neste método da divisão por planos, o usuário fica preso às imagens geradas pela divisão e manipular suas probabilidades se torna uma tarefa bastante difícil, já que algumas das imagens-plano têm pouca semelhança com as fontes usadas como teste.

3.2 Método de dilatações progressivas usando um padrão fino como padrão base

A idéia de se representar uma letra por uma PF (função densidade probabilidade) e, a partir dela, extrair imagens que irão participar da confecção do filtro “OU” é a principal idéia deste trabalho mas, após alguns testes e inclusive o método apresentado na seção anterior, foi visto que encontrar um método prático e consistente que realizasse esta decomposição é muito difícil. Por isso, pensou-se num método que, ao invés de decompor a PF, criasse um conjunto de imagens-bases, que representasse esta função de probabilidade.

A primeira idéia que se teve foi a de se encontrar um padrão base que serviria como ponto de partida para a geração dos outros gabaritos que treinariam o filtro. Este padrão base seria o mais fino possível, de preferência com apenas um pixel de largura. A geração dos outros padrões a partir do padrão fino seria realizada através da sua soma com uma realização de ruído aleatório. A escolha de se somar o ruído aleatório é devido ao fato de se querer "crescer" o padrão fino gradualmente e de forma aleatória, formando assim vários padrões que formariam a PF.

A soma do padrão fino com o ruído aleatório é limitada por um padrão grosso, isto porque, se não houver limite de tamanho para o padrão resultante da soma, seria muito provável que outros gabaritos que nada se assemelham com as letras fossem criados.

A escolha do melhor conjunto de gabaritos para a confecção do filtro "OU" depende diretamente da qualidade do ruído aditivo e, por isso, vários testes foram realizados. Mais a frente é explicado como o ruído é criado e quais os seus parâmetros que podem ser manipulados. Nos casos dos testes, serão apresentados os "A" fino e o "A" grosso e conseqüentemente os resultados para a letra "A".

3.2.1 Criação do "A" fino e "A" grosso

Como já foi dito acima, todos os padrões que confeccionarão o filtro OU (pseudo-A's) partirão de um "A" fino chegando no máximo à forma de um "A" grosso. A partir disso, a primeira coisa a se fazer seria definir qual o "A" grosso seria utilizado. Ao contrário do que muitos poderiam pensar, de início é mais importante definir o "A" grosso para então se decidir o seu "A" fino correspondente.

Neste caso, aproveitou-se a divisão em planos realizada pelo método anterior e selecionou-se o primeiro plano, no qual estão reunidas todas as posições onde existem pixels (de acordo com as fontes contidas na figura 3.2, que são as fontes-bases para todas as simulações), conforme ilustrado na figura 3.7:

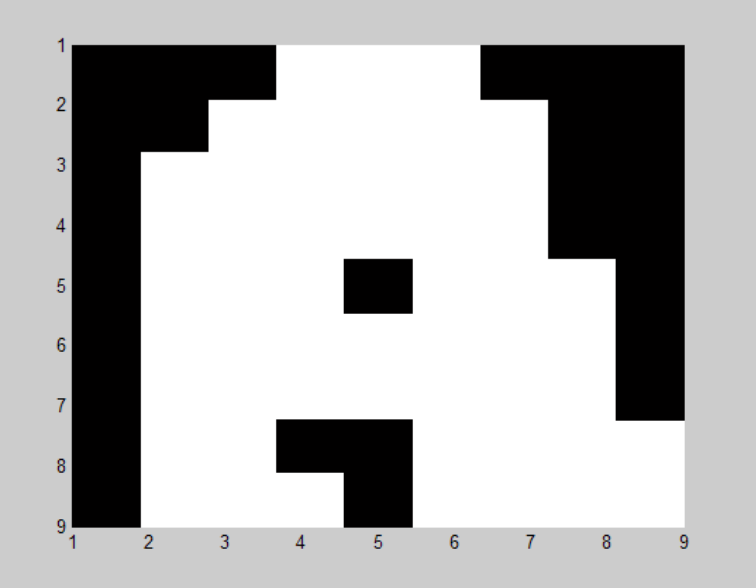
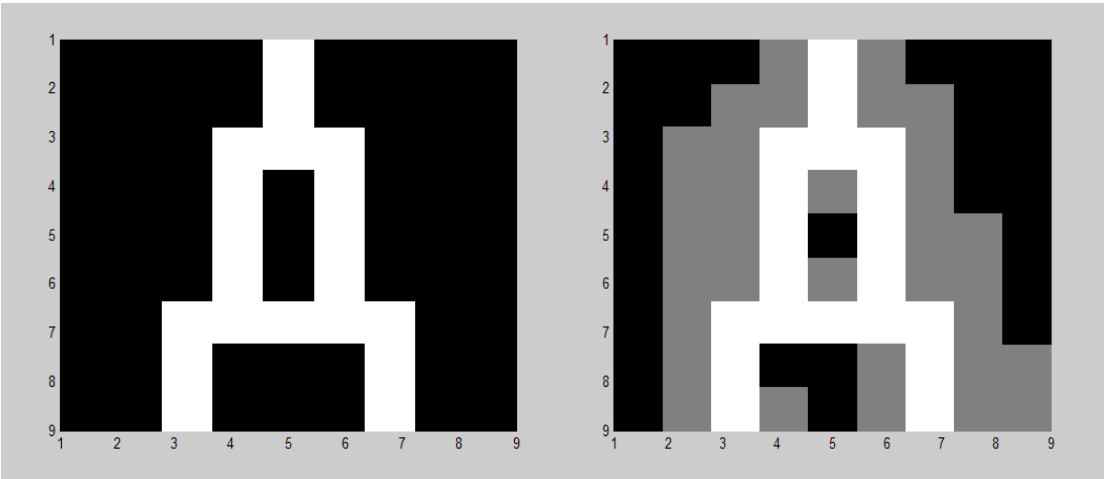


Figura 3.7. “A” grosso limite máximo para os pseudo-A’s.

Para criar o “A” fino, foram realizadas várias tentativas. Primeiro se pensou em utilizar o nono plano gerado pela divisão em planos no método anterior, mas este padrão não era o mais fino possível. Com isso, o estudo se direcionou em encontrar um esqueleto para o “A” grosso. Mas todas as tentativas geraram padrões muito pouco parecidos com as letras “A”, cheio de falhas e buracos. Então, através de uma breve análise visual da letra “A” grossa, formou-se o “A” fino abaixo, que está contido no “A” grosso e que possui largura igual a 1 pixel em sua maior parte.



(a) (b)

Figura 3.8. a) “A” fino que gerará os padrões para o filtro OU
b) “A” fino contido no “A” grosso.

3.2.2 Ruído aleatório aditivo

A criação do ruído aleatório que será adicionado ao "A" fino é essencial a este método. Como já foi dito, é a partir da soma dele com o "A" fino que será possível "crescer" o "A" de forma gradual e aleatória construindo a PF_A e posteriormente o filtro "OU". A realização do ruído branco aleatório é composta por várias fases, que serão explicadas abaixo:

- O primeiro passo é criar a imagem de um ruído aleatório uniformemente distribuído no intervalo de (0,1) [14-15].

- O segundo passo é projetar um filtro gaussiano bidimensional [16-18] que será usado pra filtrar o ruído aleatório. O objetivo de se filtrar o ruído é que, manipulando os parâmetros do filtro (tamanho do filtro e desvio padrão), o usuário tem um maior controle sobre o ruído, podendo deixá-lo mais correlacionado e, conseqüentemente, mais concentrado em uma região. Abaixo encontra-se a equação do filtro gaussiano:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \qquad (3.1)$$

  onde σ é o desvio padrão do filtro gaussiano. Outro parâmetro importante é o tamanho do filtro bidimensional. Os mais comuns utilizados são de tamanho 3x3, 5x5 e 7x7.

- Após a filtragem gaussiana, o próximo passo seria transformar o ruído, que ainda está em escala de cinza, para binário. Para isso, é aplicado um limiar ao ruído, conforme o procedimento abaixo:
  Seja p_{xy} um pixel na posição (x,y)
  Seja t o valor de limiar
  Se p_{xy}>t então p_{xy}=1
  Senão p_{xy}=0

Após a geração do ruído, algumas restrições ainda são feitas, porque nem todo o ruído será considerado válido para compor os pseudo-A's. A primeira restrição é que, como já foi dito, os pseudo-A's variaram do "A" fino ao grosso, e por isso só poderão existir pixel ou ruído na região contida entre o "A" fino o "A" grosso (veja a região cinza da figura 3.8 b).

A segunda restrição é que o "A" fino não pode "engordar" em 2 pixels de largura, no mesmo sentido, em um único passo de soma. Para fazer este controle de se somar somente ruídos de largura 1 por vez, viu-se a necessidade de se definir uma região que contenha os pixels que possam ser usados em uma iteração de soma. Para isso, buscou-se então da teoria de morfologia matemática uma ferramenta chamada dilatação por reconstrução [19-20].

A reconstrução de uma imagem binária pode ser explicada e exemplificada facilmente através da idéia de conjunto. Imagine um subconjunto S pertencente a uma imagem X. Imagine também um outro subconjunto Z que pertença a S. Desta forma, pode-se dizer que S é marcado por Z, logo Z é chamado de marcador e S de máscara. O objetivo é, através do marcador Z, recuperar o conjunto S. A partir deste conceito, faz-se a analogia com o problema tratado aqui na tese onde o "A" fino é o marcador e o "A" grosso é a máscara que se quer recuperar. A pergunta agora é: como fazer esta reconstrução?

A reconstrução é realizada através de uma operação morfológica chamada dilatação. As operações morfológicas são fortemente influenciadas pelo elemento estruturante, que é uma máscara bidimensional que indica o tamanho e a influência dos pixels vizinhos ao pixel que está sofrendo a filtragem. A dilatação de um conjunto X por um elemento estruturante B é definida da seguinte forma:

$$dil^{B}(X) = X \; dil \; B = \{x \in X : B_x \cap X \neq 0\}, \tag{3.2}$$

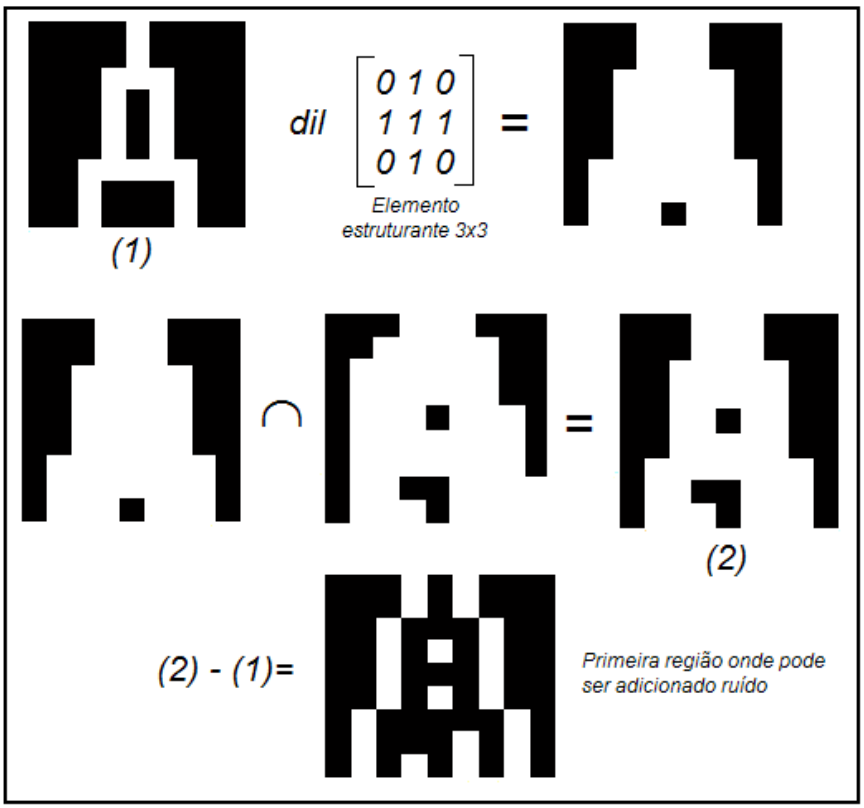
ou seja, se a interseção da imagem X com o elemento estruturante B for diferente de 0, então o pixel x centrado pelo elemento estruturante assume o valor 1, senão será 0. É necessário dizer que o elemento estruturante percorre todos os pixels da imagem e o efeito da filtragem é realizado em todo o pixel centrado por ele.

A dilatação por reconstrução é realizada a partir da dilatação do marcador, como explicado acima, condicionada à máscara. Dependendo da imagem e do elemento

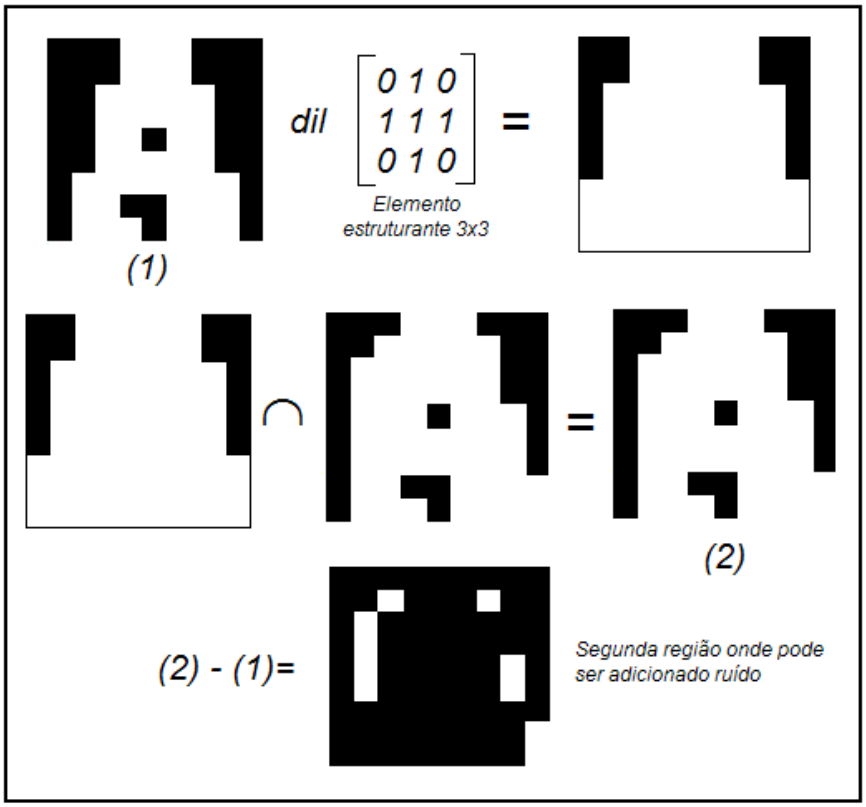
estruturante, pode haver dilatações condicionais sucessivas até que se chegue ao objetivo final, que é a máscara. Abaixo, está descrita a fórmula da dilatação por reconstrução:

$$dil_{cX}^{B}(Z) = (Z\ dil\ B) \cap X \tag{3.3}$$

No caso desta tese, a reconstrução do "A" fino até o "A" grosso levou duas iterações, aplicando duas vezes a equação 3.2. As regiões onde possam existir pixels, que serão adicionados como ruído, são definidas como a diferença entre a primeira dilatação e o "A" fino e a diferença entre a segunda dilatação com a primeira. Veja exemplo a seguir:



(a)



(b)

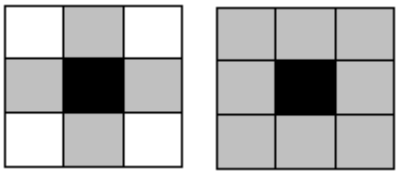
Figura 3.9. a) Dilatação por reconstrução e primeira região onde pode ser adicionado pixel
b) Segunda iteração da dil. por reconstrução e segunda região onde pode ser adicionado pixel.

Como pode ser visto, nenhuma das duas regiões onde possa ser adicionado ruído possui objetos com largura de 2 pixels. Com isso, o controle da adição de apenas um pixel por iteração pode ser feito corretamente.

A outra restrição é que só são aceitos ruídos (pixels) que sejam conectados à letra "A" fina, com conectividade 4. A definição de conectividade em uma imagem relaciona como os pixels vizinhos se relacionam com um determinado pixel e é de grande importância para a definição de conceitos como regiões e bordas.

A conectividade 4 (para imagens binárias) diz que um pixel é conexo a outro se ele for vizinho direto na posição acima ou abaixo, na esquerda ou a direita e se tiver o mesmo valor (no caso 1 ou 0). A conectividade 8 relaciona o pixel com os mesmos pixels da conectividade 4 mais os pixels que são vizinhos na direção diagonal (os pixels também devem ter o mesmo valor). Os pixel que são conectados pertencem a um mesmo objeto, e com isso, em determinadas operações morfológicas, principalmente as que se analisam objetos-imagens, a definição de conectividade pode mudar o resultado

da filtragem. Para maiores informações, ver [16-20]. Abaixo as ilustrações de conectividade:



a) b)

Figura 3.10. Relação de conectividade com o pixel central:
a) Pixels vizinhos na cor cinza, que tem conectividade 4.
b) Pixels vizinhos na cor cinza que tem conectividade 8.

Após a definição de conectividade, fica mais fácil compreender o controle de adição de ruído que é realizado após a sua soma com a letra “A”. Nesta imagem resultante, todo o objeto que não for conexo receberá um índice (*label*) e um contador é incrementado a cada objeto encontrado. Estabelecido isso, é realizada então uma contagem no momento da leitura da imagem, onde é verificado o seu valor que, se for maior que 1, mostra que existe mais de um objeto na imagem, ou seja, o ruído não estará conectado a letra “A”, portanto sem conectividade 4, e então o padrão será excluído do conjunto que comporá o filtro “OU”.

Outra questão importante para a criação do ruído, mas mais específica para os testes práticos realizados aqui nesta tese, é o tamanho da janela do ruído que será convoluído com o filtro gaussiano. Como o tamanho dos padrões testados é de 9x9, então o ruído bidimensional também deverá estar contido numa janela 9x9. Mas, para que esta janela esteja numa região válida da filtragem, a janela inicial do ruído deve ter tamanho (9+M+1)x(9+M+1), sabendo que o filtro tem tamanho (2M+1)x(2M+1).

No caso desta tese, após alguns testes modificando o tamanho do filtro, foi decidido que este será de tamanho 3x3 e, como o tamanho da janela usado pelos padrões aqui testados é de 9x9, o tamanho inicial do ruído, de acordo com a relação mostrada acima, deve ser pelo menos de 11x11. Após esta filtragem, o ruído é então redimensionado para 9x9 cortando simplesmente 2 linhas e 2 colunas. A seguir é mostrado um esquema que ilustra desde a criação do ruído até a soma ao “A” fino formando assim o pseudo-A que participará da confecção do filtro OU.

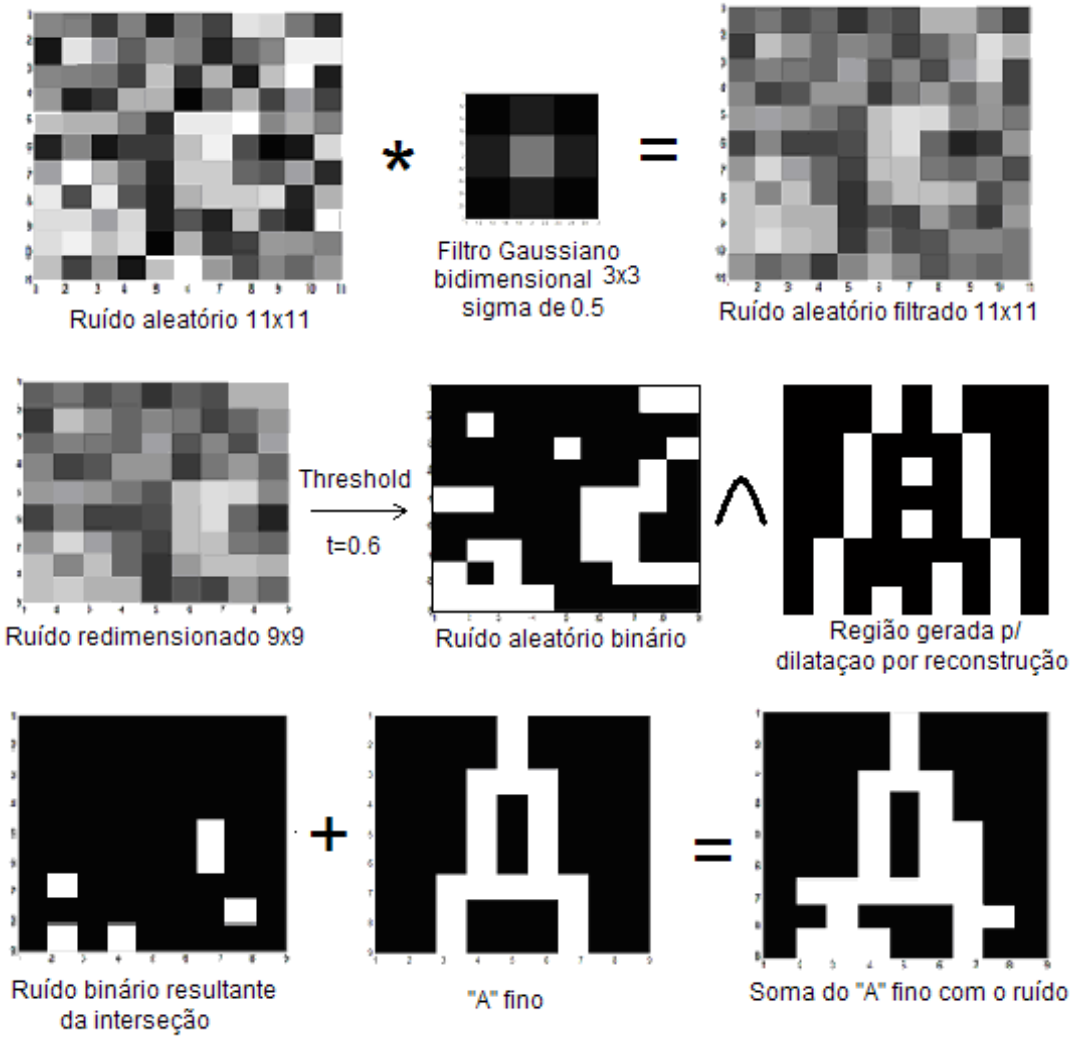


Figura 3.11. Esquema para a criação do Ruído e do Padrão que comporá o filtro “OU”

Abaixo encontra-se o resumo deste método através de diagrama de blocos. Note que os processos descritos nas figuras 3.09 e 3.11 estão presentes no diagrama abaixo.

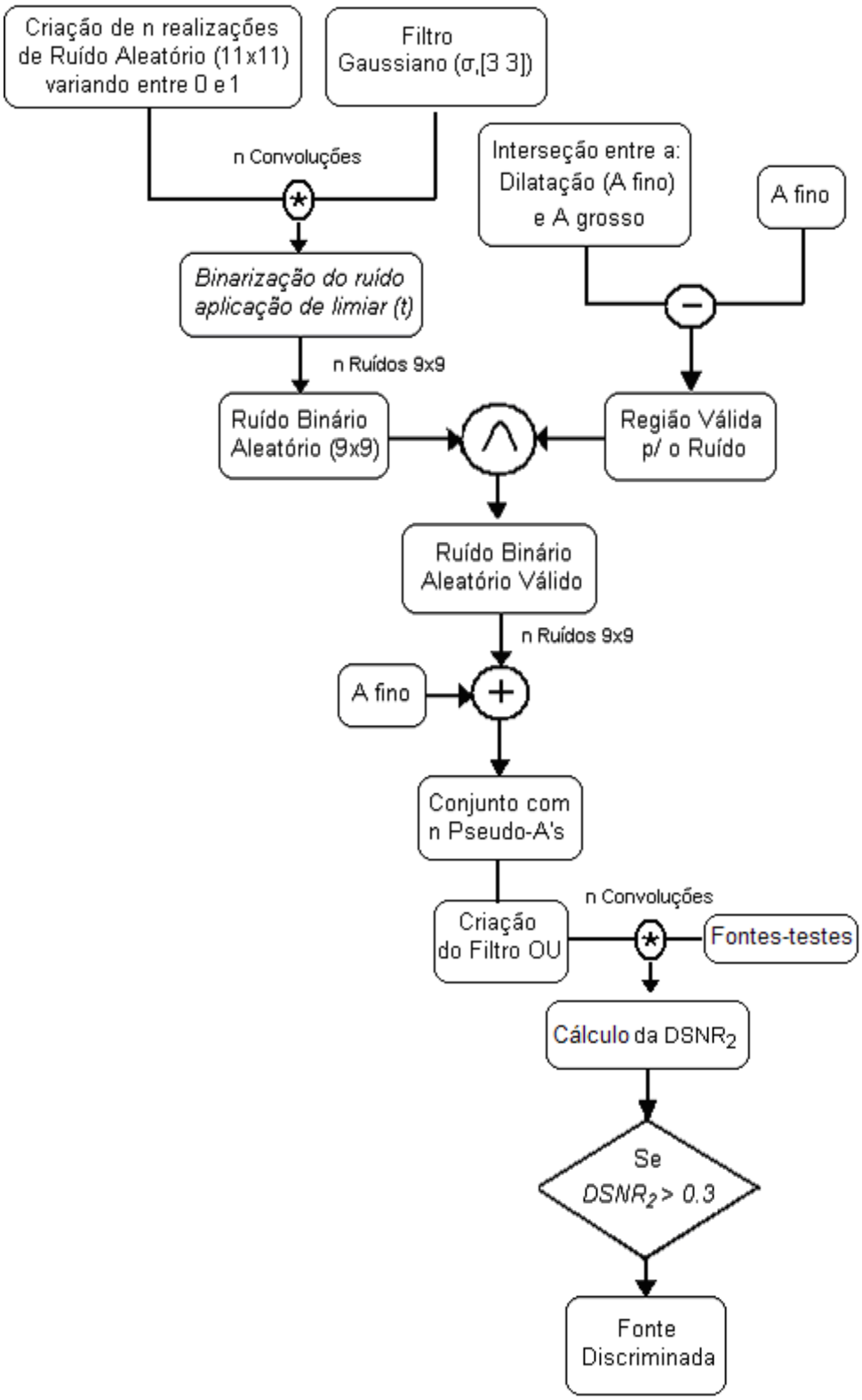


Figura 3.12. Diagrama de Blocos que representa todo o processo. Este é repetido até que se discrimine o máximo de fontes-testes.

3.2.3 Resultados das simulações

As simulações desta seção consistem em testar o filtro "OU" realizado pelos pseudo-A's gerados por este método, nas fontes-testes (figura 3.2). Para este teste, foram feitas várias simulações mudando o valor do desvio padrão (σ) do filtro gaussiano bidimensional e do valor de limiar (t) usado na transformação do ruído aleatório em binário. Para cada conjunto de valores de σ *e* t, o programa é executado várias vezes (até 2000 iterações) até que seja encontrado um conjunto que discrimine o máximo de fontes-testes possíveis.

A seguir, apresenta-se uma tabela que descreve os melhores resultados de $DSNR_2$ encontradas para as fontes-testes variando os valores de σ *e* t. Também são apresentados a média das DSNR's de cada caso e seu desvio padrão médio. Assim como no método anterior, o valor da $DSNR_2$ limite para mostrar que um padrão foi discriminado é de 0,3.

Tabela 3.5: Resultados do método de dilatação progressiva usando apenas um padrão fino variando o valor de σ e t.

σ/ t / **Fontes**	**0,4/0,5**	**0,52/0,55**	**0,52/0,6**	**0,52/0,65**	**0,6/0,5**	**0,6/0,7**	**0,7/0,6**
1	0,3464	0,3421	0,3606	0,3407	0,3497	0,1859	0,1907
2	0,0084	0,0085	0,0019	0,0007	0,0090	0,0026	0,0000
3	0,5212	0,5581	0,4547	0,5321	0,5882	0,3895	0,4373
4	0,4424	0,5003	0,6326	0,4960	0,5362	0,3492	0,4882
5	0,3757	0,4195	0,3837	0,3070	0,4153	0,3019	0,4820
6	0,2848	0,2606	0,3005	0,2368	0,2488	0,2422	0,2377
7	0,1025	0,1046	0,0850	0,1113	0,1632	0,0624	0,0638
8	0,3212	0,3771	0,3326	0,3505	0,3799	0,2306	0,2750
9	0,6450	0,6795	0,8472	0,7860	0,4654	0,8214	0,7307
10	0,0879	0,0872	0,0678	0,0974	0,1061	0,0501	0,0762
11	0,3474	0,3192	0,3327	0,3229	0,2470	0,1686	0,2198
12	0,3585	0,3913	0,3560	0,4121	0,3160	0,2048	0,3050
13	0,4429	0,4460	0,4983	0,4419	0,3439	0,1967	0,4195
14	0,3747	0,3959	0,3414	0,3097	0,4465	0,3040	0,4943
15	0,3310	0,3141	0,3945	0,3175	0,3124	0,1312	0,2680
16	0,0805	0,0968	0,0556	0,0729	0,0937	0,0397	0,0147
17	0,3061	0,3012	0,2527	0,2767	0,3951	0,2127	0,2187
18	0,3540	0,3567	0,3960	0,2838	0,3032	0,1625	0,2218
19	0,2346	0,1922	0,2341	0,1926	0,2365	0,2201	0,2208
Média da $DSNR_2$ (μ)	0,3140	0,3237	0,3330	0,3099	0,3135	0,2251	0,2823
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,1150	0,1294	0,1405	0,1274	0,1159	0,1119	0,1453
$\left(\overline{DP}\right)$**/(μ)**	0,3664	0,3997	0,4218	0,4111	0,3698	0,4974	0,5148
Acertos	13	13	13	12	12	5	8

A tabela acima mostra que este método de dilatação progressiva tem, como melhor resultado, 13 acertos em 19 padrões Ou seja, este método apresentou resultados um pouco piores do que o apresentado no método da divisão em planos de freqüência.

O número de padrões que compõem o filtro OU varia entre 15 e 25 padrões, mas foi visto que em média 20 padrões são suficientes para que o filtro apresente bons resultados. Um número maior de padrões mal-condiciona a matriz do operador e gera resultados ruins. Em todos os casos deste método, os padrões que participaram da confecção do filtro tiveram o mesmo peso.

A influência de σ (desvio padrão do filtro gaussiano) e t (valor de limiar) nos resultados só apresenta diferenças marcantes ao aplicar uma grande elevação em um dos valores, acima de 0,6. Nestes casos, pouco ruído é adicionado à letra "A" fina e por isso só as fontes-testes parecidas com o "A" fino apresentam valores altos de $DSNR_2$. Para outros casos com valores baixos (menores que 0,6), a resposta do filtro é muito parecida. Como é visto, na tabela 3.5, o filtro responde melhor quando o "A" fino recebe uma boa quantidade de ruído (figura 3.13) que pode ser conseguido com valores médios de limiar aliado com valores baixos de σ.

Figura 3.13. Pseudo-A's que responsáveis por 13 acertos p/ σ=0.4 e t=0.5 .

Como pode ser visto na figura acima, com valores de $\sigma = 0{,}4$ *e* $t = 0{,}5$, os pseudo-A's aparecem bastante ramificados e em algumas situações pouco lembram uma letra A. Este conjunto de pseudo-A's apresentou o melhor resultado por apresentar o maior número de acertos e também levando-se em conta um compromisso que visa um valor médio de $DSNR_2$ alto e baixa relação de $\left(\overline{DP}\right)$**/(μ).**

Portanto, a conclusão que se tem deste método é que, apesar da boa idéia de se criar uma PF a partir da soma do "A" fino com o ruído, de se ter um maior controle de como o ruído é gerado e, portanto, como o pseudo-A também é gerado, o método não funcionou satisfatoriamente. Os resultados apresentaram-se um pouco piores que o

primeiro método e mantendo alguns problemas que também existiam no método anterior, como a dificuldade de se manipular os pesos dos padrões que compõem o filtro "OU".

Com isso, algumas modificações são realizadas no próximo método de forma a melhorar os resultados aqui apresentados. De forma ilustrativa, é mostrada abaixo a PF_A formada a partir do conjunto de padrões da figura 3.13 e que, portanto, está presente na tabela resumo no fim deste capítulo.

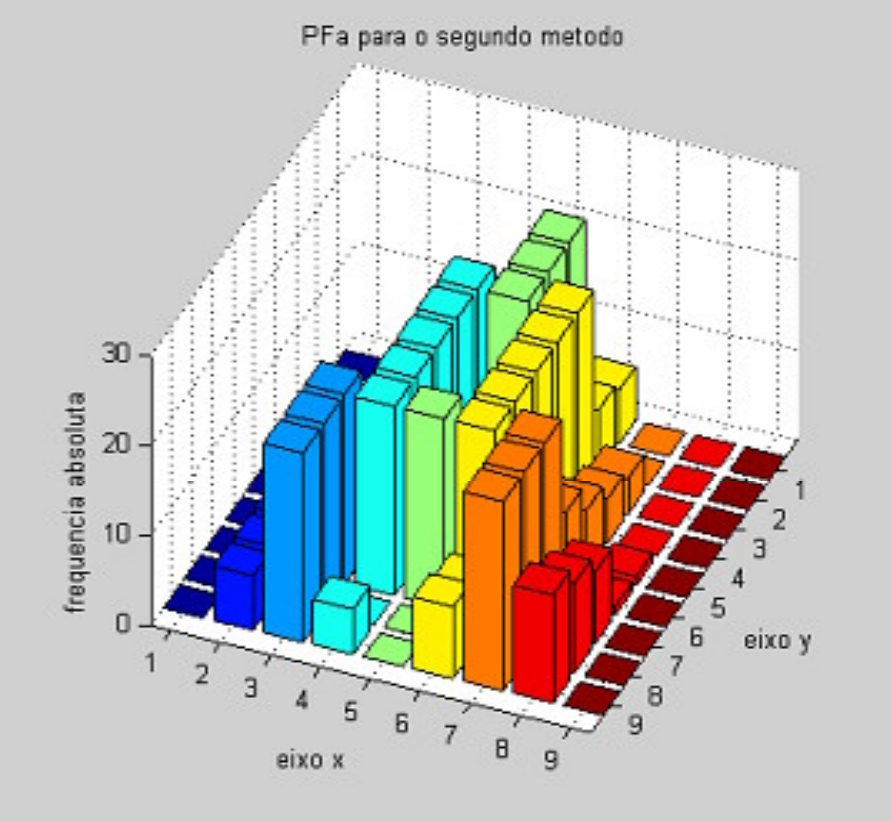


Figura 3.14. PF_A para o segundo método.

3.3 Métodos dilatações progressivas usando vários padrões finos como padrão base

Nesta seção, o método via dilatações progressivas usando apenas um padrão fino como ponto de partida para a criação dos pseudo-A's sofrerá algumas modificações. O objetivo desta seção é procurar um algoritmo em que o usuário possa ter maior controle sobre o conjunto de pseudo-A's criado a partir do "A" fino, podendo dar-lhes pesos diferentes (equação 2.55), caso seja necessário. Isto seria muito útil e importante, pois seria possível dar maior ênfase a uma fonte com $DSNR_2$ baixa e assim melhorar a performance do filtro "OU".

Desta forma, imaginou-se que, se cada fonte-teste tivesse seu próprio "A" fino e conseqüentemente seu próprio conjunto de pseudo-A's. Isto tornaria viável a idéia de se controlar a resposta do filtro para cada fonte-teste através das mudanças de pesos na confecção do filtro "OU". Isto seria similar a quando se fez um filtro OU a partir das próprias fontes-testes, só que agora o conjunto de imagens que participaria do treinamento do filtro "OU" seria bem maior e teria padrões corrompidos por ruído, o que daria maior robustez ao filtro "OU".

O importante de se deixar claro para este método aqui é que as outras características do método usando somente um "A" fino, como todo o processo de geração do ruído, a geração de pseudo-A's a partir do "A" fino somado ao ruído e limitado pelo "A" grosso serão mantidas.

As outras mudanças, além do "A" fino, que serão observadas são: novos padrões para os "A" grossos e a existência de ruído com sinal positivo e negativo, ou seja, o ruído poderá tanto adicionar quanto subtrair pixels da imagem. Além destas novas características, um novo dado de grande importância deve ser explicado, que é qual o grupo de pseudo-A é escolhido para compor o filtro "OU":

A primeira questão a ser explicada é que cada pseudo-A é proveniente da soma do "A" fino com uma realização nova de ruído, ou seja, para tantos pseudo-A's desejados, a rotina que cria o ruído aleatório será chamada.

A segunda questão é que, como se têm agora vários "A" finos, os processos de criação de pseudo-A's mostrados na seção anterior será repetido para cada novo padrão fino e, no final cada conjunto de pseudo-A's criados, serão agrupados num único conjunto para confeccionar o filtro "OU" final. Sabendo disso, o critério para se

escolher o melhor conjunto, que é usado em todos os métodos desta seção, é dado sob duas formas:

- Caso I – O conjunto de pseudo-A's formado, como já explicado acima, é convoluído com cada uma das 19 fontes-testes; este processo é repetido até que se encontre o conjunto que discriminou o maior número possível das fontes-testes. Cada vez que se cria o conjunto de padrões que treinará o filtro "OU", novas realizações de ruído são feitas de acordo com o número máximo de padrões estipulados pelo usuário. É necessário dizer que todo este algoritmo é repetido para cada um dos "A" finos, que no final se juntarão para construir um único filtro OU.

- Caso II – Este segundo caso é similar ao primeiro, só que o critério de escolha do melhor conjunto é que, ao invés de escolher o conjunto que discrimina o maior número de fontes possíveis, escolhe-se o conjunto que gera a maior $DSNR_2$ para a fonte correspondente ao "A" fino que gerou os pseudo-A's, ou seja, se o programa está sendo rodado para a fonte 3 (fig. 3.2), com os pseudo-A's gerados pelo seu "A" fino, procura-se o conjunto que dê a maior $DSNR_2$ para esta fonte; isto é também repetido para as 19 fontes-testes que, no final, também serão agrupados para o treinamento do filtro OU final.

Os Casos I e II apresentados acima são contemplados com todos os pseudo-A's com o mesmo valor de peso. Outros dois casos a serem mostrados nos resultados que virão em cada um dos 4 métodos desta seção são os Casos III e IV, que são os melhores conjuntos obtidos pelos Casos I e II, só que com alguns conjuntos com pesos diferentes, de modo a melhorar o desempenho do filtro final. Como já explicado, os pesos são elevados de acordo com a necessidade de se aumentar a $DSNR_2$ para determinada fonte.

Portanto, sabendo de todas essas novas informações, a seguir são apresentados os 4 métodos (ou sub-métodos) desta seção com seus respectivos resultados para os 4 casos explicados acima.

3.3.1 Fontes-testes como "A" fino

Neste primeiro sub-método, a mudança que é realizada em relação ao método da seção 3.2 é com relação aos "A" finos. Como já foi explicada anteriormente, a idéia central deste novo método é que cada fonte-teste tenha seu próprio "A" fino e, por isso, a idéia que se teve foi: ao se analisar a figura 3.2, percebeu-se que as fontes-testes já são muito finas (fonte 8) e por isso poderiam se utilizadas como "A" finos. Com isso, cada uma das fontes-testes formará um conjunto de pseudo-A's a partir delas mesmas e, juntas, farão parte da construção do filtro "OU".

Com relação ao "A" grosso, o mesmo será mantido igual ao método anterior, que é o primeiro padrão (fig. 3.6) gerado pelo método de divisão de planos em freqüência (seção 3.1). O controle de ruído, apresentado no subitem 3.2.2, agora não será mais feito com conectividade 4, mas sim 8, visto que as próprias fontes que são usadas como "A" fino têm em sua maioria conectividade 8.

A seguir. são mostrados os resumos deste método através de pseudo-código, tanto para o Caso I, quanto para o Caso II. Para os dois casos já serão considerados que os ruídos aleatórios binários já são válidos (ver seu processo de criação nas figuras 3.11 e 3.12) :

Algoritmo 3.1: Algoritmo para o método usando as próprias fontes-testes como "A" fino e utilizando o Caso I como método de busca para o conjunto de pseudo-A

```
// x é determinado pelo usuário
// n = no. de pseudo-A's definido p/ usuário
// no. de A finos e fontes-testes = 19
Para i = 1 até 19,
    Se FontesCertas < x,
        Para j = 1 até n,
            Pseudo_A (i,j) = Afino (i) + Ruído (j);
        Fim Para
        FiltroOU (i) = ConstroiFiltro(Pseudo_A(i,1:n));
        Para k = 1 até 19,
            RespConv(k) = Conv ( FiltroOU(i), FonteTeste(k));
            DSNR(k) = CalcDSNR (RespConv(k));
            Se  DSNR(k)> 0.3,
                FontesCertas = FontesCertas+1;
            Fim Se
        Fim Para
    Fim Se
  Senão RepetePrograma ()
Fim Para
// Teste final com todos os melhores conjuntos de Pseudo-A's
FonteCerta=0;
FiltroOUFinal = ConstroiFiltro(Pseudo_A(1:19));
Para k = 1 até 19,
    RespConv(k) = Conv ( FiltroOUFinal, FonteTeste(k));
    DSNR(k) = CalcDSNR (RespConv(k));
    Se DSNR(k) > 0.3
        FonteCerta= FonteCerta+1;
    Fim Se
Fim Para
FonteCerta= FonteCerta-1;
```

Algoritmo 3.2: Algoritmo para o método usando as próprias fontes-testes como "A" fino e utilizando o Caso II como método de busca para o conjunto de pseudo-A.

```
// Max = DSNR máxima definida pelo usuário
// n = no. de pseudo-A's definido p/ usuário
// no. de A finos e fontes-testes = 19
Para i = 1 até 19,  // i é referente ao A fino
      Se DSNR(i) < MAX,
         Para j = 1 até n,
               Pseudo_A (i,j) = Afino (i) + Ruído (j);
         Fim Para
         FiltroOU (i) = ConstroiFiltro(Pseudo_A(i,1:n));
         Para k = 1 até 19,
               RespConv(k) = Conv ( FiltroOU(i), FonteTeste(k));
               DSNR(k) = CalcDSNR (RespConv(k));
         Fim Para
      Fim Se
     Senão RepetePrograma ()
Fim Para
// Teste final com todos os melhores conjuntos de Pseudo-A's
FonteCerta=0;
FiltroOUFinal = ConstroiFiltro(Pseudo_A(1:19));
Para k = 1 até 19,
      RespConv(k) = Conv ( FiltroOUFinal, FonteTeste(k));
      DSNR(k) = CalcDSNR (RespConv(k));
      Se DSNR(k) > 0.3
         FonteCerta= FonteCerta+1;
      Fim Se
Fim Para
FonteCerta= FonteCerta-1;
```

Com relação às simulações deste método, também foram testados vários conjuntos de σ (desvio padrão do filtro gaussiano [19-20]) e t (valor de limiar) afim de se estudarem as suas influências nos resultados. Assim, como nos casos anteriores, o valor de discriminação da $DSNR_2$ é de 0,3. A seguir são apresentados na tabela 3.6, somente os melhores resultados para os dois casos com pesos com valores iguais, que são os Casos I e II explicados anteriormente, e para os dois casos com a mudança de pesos que são o Caso III e IV. Neste primeiro método (somente neste primeiro método), será mostrado também um outro caso como forma ilustrativa da influência dos pesos sobre o valor da $DSNR_2$, que será o Caso V (também um caso especial de mudança dos pesos dos resultados obtidos no Caso II). Também são apresentados os valores médios e os desvios padrões médios das $DSNR_2$, que servem para ajudar a escolher, junto com o número de acertos, o melhor caso deste método.

Tabela 3.6: $DSNR_2$ para as 19 fontes-testes utilizando o método de dilatações progressivas usando as próprias fontes-testes como padrão fino.

Fontes	**Caso I**	**Caso II**	**Caso III**	**Caso IV**	**Caso V**
1	0,6627	0,7581	0,5998	0,7170	0,4771
2	0,0401	0,0451	0,0478	0,0520	0,2508
3	0,8485	0,8277	0,7979	0,7801	0,6384
4	0,7323	0,7660	0,6333	0,6981	0,4848
5	0,3660	0,4381	0,3244	0,3930	0,2579
6	0,3666	0,4871	0,3396	0,4698	0,4030
7	0,3024	0,3238	0,3631	0,3771	0,3772
8	0,8654	0,9419	0,8314	0,9095	0,7658
9	0,5143	0,5597	0,4635	0,5343	0,3347
10	0,1722	0,1499	0,1732	0,1535	0,2570
11	0,4770	0,6016	0,4151	0,5620	0,4170
12	0,8774	0,9393	0,8578	0,9294	0,8393
13	1,0964	1,2627	0,9070	1,1029	0,6437
14	0,3362	0,3948	0,3054	0,3525	0,2507
15	0,6082	0,6807	0,5327	0,6193	0,4111
16	0,2030	0,2165	0,3048	0,3103	0,3352
17	0,4509	0,4804	0,4751	0,4935	0,4168
18	0,5834	0,6875	0,5410	0,6537	0,4642
19	0,3253	0,3087	0,3325	0,3065	0,2822
Média da $DSNR_2$ (μ)	0,5173	0,5721	0,4866	0,5481	0,4372
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,2248	0,2439	0,1903	0,2146	0,1319
$\left(\overline{DP}\right)$**/(μ)**	0,4347	0,4263	0,3911	0,3915	0,3016
Acertos	16	16	17	17	14

Os Casos I e II para este primeiro método, usando vários “A” finos, já se mostrou melhor que os dois métodos anteriores, que são os de divisão em planos de freqüência (seção 3.1) e de dilatações progressivas usando apenas um “A” fino (seção 3.2), com 16 acertos em 19.

No Caso I, foram necessários 208 padrões para compor o filtro “OU”, uma média aproximada de 11 padrões por fonte; o Caso II teve 173 padrões (pseudo-A’s), média de 9 padrões por fonte. Esta diferença se deve principalmente à diferença no par $\sigma \; e \; t$ (desvio padrão do filtro gaussiano e limiar de binarização do ruído respectivamente), que age direto na geração do ruído. Quando se quer um conjunto de padrões que

discrimine o máximo de fontes (Caso I), deve-se escolher valores medianos de σ e t, entre 0,5 e 0,6 para os dois parâmetros. Quando se busca um conjunto que gere uma $DSNR_2$ bem alta para uma determinada fonte, deve-se aumentar o valor do σ e t para valores entre 0,6 e 0,75.

Isto é explicado facilmente, pois quando se elevam os valores de σ e t, poucos pixels são adicionados ao "A" fino, que aqui no caso é a própria fonte. Por isso, os seus pseudo-A's são muito parecidos com eles mesmos, o que faz com que a $DSNR_2$ seja bem alta, ao contrário de valores medianos de σ e t onde, com uma inserção maior de ruído (pixel), pode-se aproximar a morfologia de uma fonte-teste à outra e assim o conjunto está mais apto a discriminar outras fontes. Para se ter uma melhor visão deste método, cada conjunto de pseudo-A's para uma determinada fonte no Caso I discrimina em média 12 fontes-testes enquanto no Caso II, 5 fontes-testes.

O Caso III, como foi dito, é uma variação do Caso I no que se refere à mudança de pesos de algumas fontes. Como pode ser visto no Caso I, as fontes 2, 10 e 16 são as que apresentam valores de $DSNR_2$ menores que 0,3, sendo que a fonte 16 ainda possui um valor médio de 0.2.

Com isso, a primeira tentativa consistiu em aumentar o valor de pesagem do conjunto de pseudo-A's gerados por esta fonte 16, de forma a melhorar o desempenho do filtro "OU". Contudo, é necessário dizer que a equação do filtro "OU" (equação 2.55) se trata de uma média ponderada da pesos com o padrão e, portanto, o aumento do peso de um padrão visando ao aumento de sua $DSNR_2$ torna-se um compromisso com o valor da $DSNR_2$ das outras fontes, podendo tanto aumentar como diminuir dependendo da relação da morfologia entre os padrões.

Por isso, a primeira tentativa de se aumentar somente o peso do conjunto referente à fonte 16 não é o suficiente, necessitando, especificamente neste caso, do aumento também para o conjunto da fonte 14. Com estas modificações, o filtro "OU" conseguiu discriminar 17 fontes, sobrando apenas as fontes 2 e 10, que a cada teste se mostram as fontes mais problemáticas. Este resultado é considerado bom, porque iguala a resposta do filtrou "OU" utilizando as próprias fontes, só que agora têm-se fontes corrompidas com ruído, o que teoricamente daria maior robustez ao método. A seguir, apresenta-se a tabela de pesos para este caso.

Tabela 3.7: Pesagem das fontes para o Caso III.

Fontes	Pesos
Fonte 14	1,3/21,2
Fonte 16	2,9/21,2
Outras fontes	1/21,2

Para o Caso IV (variação do caso II com relação aos pesos), foi preciso apenas dobrar o peso do conjunto referente à fonte 16, do Caso II, em relação as outras. Assim, como no Caso III, também se conseguiu discriminar 17 fontes neste teste. Neste caso, o conjunto referente à fonte 14 não precisou de um aumento no seu peso, pois, como pode ser visto nos resultados para o Caso II, ela já possuía uma $DSNR_2$ mais alta do que no Caso I e, por isso, mesmo aumentando o peso do conjunto da fonte 16, a $DSNR_2$ do conjunto 14 não abaixou de 0,3.

O Caso V, como foi dito, é um caso específico de mudança de peso do Caso II. Nele quis-se mostrar que, aumentando os pesos dos conjuntos das fontes 2 e 10, na tentativa de melhorar suas $DSNR_2$, resulta na diminuição das $DSNR_2$'s de outras fontes abaixo de 0,3, mostrando desta forma a relação de compromisso entre o peso e as $DSNR_2$'s. Para este caso especificamente, foi distribuído um conjunto de pesos para algumas fontes, de forma a se conseguir que todas obtivessem repostas acima de 0,25, mas para isso algumas fontes que antes eram detectadas ($DSNR_2$ >0,3) agora têm valores menores. Isto serve em uma aplicação onde se pode baixar o valor do limiar de detecção para uma determinada letra, o que serve até como uma solução para que o desempenho do filtro seja considerado de 100%.

O importante a dizer para este método é que todo o treinamento é feito offline. Depois que se têm os pseudo-A's e o filtro já foi realizado, a convolução com a imagem é bem rápida.

A seguir é apresentada de forma ilustrativa a PF_A do Caso IV, escolhido como o melhor resultado deste método por apresentar o maior número de acertos aliado com o alto valor médio de $DSNR_2$ e a baixa razão entre $\left(\overline{DP}\right)/(\mu)$ e que, por isso, estará na tabela resumo no fim deste capítulo:

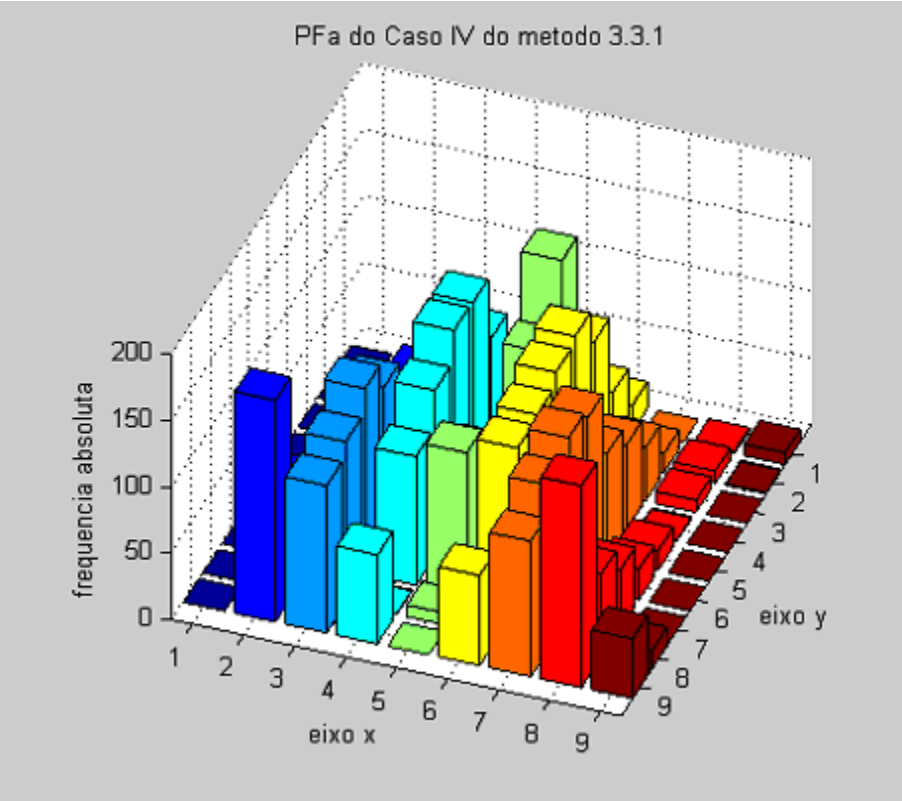


Figura 3.15. PF_A para o Caso IV do método 3.3.1.

3.3.2 Método com "A" finos "podados" e "A" grossos dilatados a partir do padrão fino

Neste segundo caso, algumas mudanças maiores serão apresentadas, mas mantendo ainda o conceito de ruído e de que todas as fontes que comporão o filtro "OU" partem de uma "A" fino até um "A" grosso. Aqui surge a idéia de se "podarem" as fontes originais (fig. 3.2), de forma a deixá-las mais finas ainda e usar este conjunto como os novos "A" finos. A intenção de se fazer isso é que possibilita que uma fonte mesmo com ruído seja menor que a fonte original. Isto de certa forma poderia funcionar até como um ruído de sinal negativo. Note na figura 3.16 a seguir que, em alguns casos, não se pode "podar" as fontes visto que elas já eram muito finas.

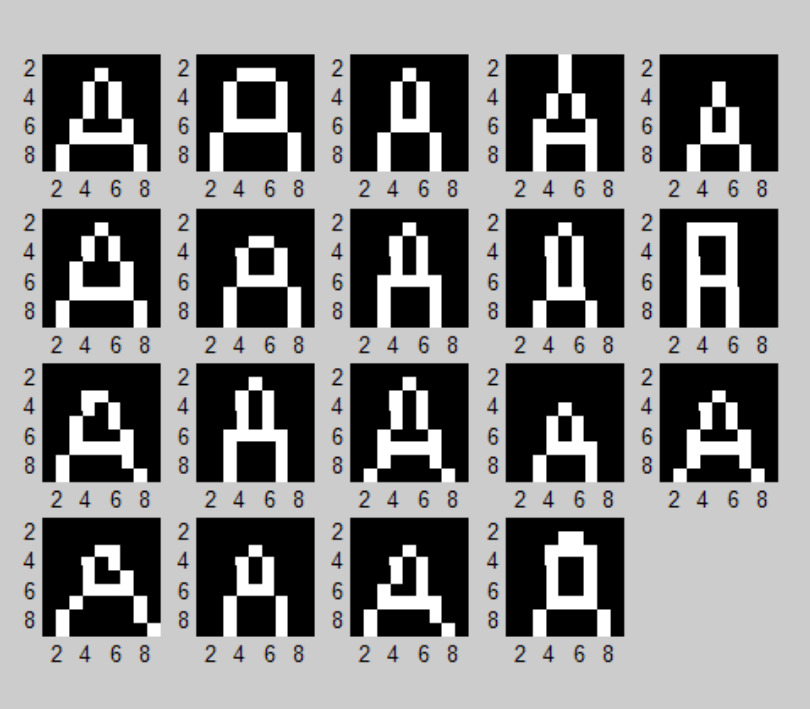


Figura 3.16. Novo conjuntos de “A” finos.

A segunda mudança é que cada fonte fina terá sua própria fonte grossa proveniente da operação morfológica da dilatação, já explicado na seção anterior. Como pode ser percebido na figura 3.17 abaixo, um dos principais efeitos da operação de dilatação é engrossar um determinado padrão seguindo a orientação do elemento estruturante. No caso deste método, a dilatação do padrão se deu em duas direções: vertical e horizontal, crescendo apenas um pixel em cada sentido.

Esta mudança tende a deixar os pseudo-A’s (“A” fino somado com ruído) mais parecidos com as fontes originais, já que a região onde pode ser adicionado pixel é mais limitada (figura 3.9 ilustra a região onde pode ser adicionado pixel). Esta mudança visa a melhorar os valor das $DSNR_2$’s, visto que este valor depende diretamente da semelhança das fontes que compõem o filtro com o padrão que se quer discriminar. A seguir é ilustrado o novo conjunto de “A” grossos.

Figura 3.17. Novos “A” grossos.

É necessário explicar também neste método que, similar ao item 3.3.1, um filtro “OU” é realizado para cada conjunto “A” fino/“A” grosso para a escolha do melhor conjunto e depois todos os conjuntos de pseudo-A’s de cada “A” fino formam um grupo que dará origem ao filtro “OU” final.

Os pseudo-códigos apresentados no item 3.3.1 podem ser utilizados também neste método só lembrando que os A’s finos e grossos devem ser trocados.

Assim como nos casos anteriores, as simulações deste método foram realizadas variando σ e t várias vezes, procurando sempre o melhor conjunto e, assim como no método do item 3.3.1 para o Caso I, os melhores resultados se deram para valores de σ e t entre 0,5 e 0,6 e, para o Caso II, entre 0,6 e 0,75, para ambos os parâmetros. O limiar de detecção neste método será também de 0,3.

A seguir serão apresentadas na tabela as $DSNR_2$ para as 19 fontes-testes nos quatro casos já conhecidos. Assim como nos métodos anteriores, o valor médio e o desvio padrão médio das $DSNR_2$ serão apresentados. Estes parâmetros servem para ajudar a escolher junto com o número de acertos o melhor caso deste método.

Tabela 3.8: $DSNR_2$ para as 19 fontes-testes utilizando o método de dilatações progressivas usando as fontes-testes "podadas" como padrão fino, e seus respectivos "A" grossos provenientes da dilatação.

Fontes	**Caso I**	**Caso II**	**Caso III**	**Caso IV**
1	0,9306	0,9247	0,8175	0,9369
2	0,0178	0,0294	0,0169	0,0234
3	0,4999	0,4048	0,4515	0,3616
4	0,6593	0,7199	0,6053	0,6776
5	0,3067	0,3179	0,3414	0,3343
6	0,4285	0,4410	0,4014	0,4550
7	0,2777	0,2838	0,3280	0,3155
8	0,6679	0,6325	0,5713	0,5386
9	0,6577	0,7352	0,6201	0,7049
10	0,1037	0,1070	0,0901	0,0919
11	0,6798	0,7415	0,5717	0,6473
12	0,7783	0,7889	0,6461	0,6654
13	1,2517	1,2331	0,9815	1,0603
14	0,2808	0,2928	0,3276	0,3200
15	0,7984	0,9072	0,7476	0,8447
16	0,2641	0,2753	0,3210	0,3101
17	0,3637	0,3535	0,3925	0,3398
18	0,7488	0,8678	0,6576	0,7639
19	0,2575	0,2404	0,3069	0,3074
Média da $DSNR_2$ (μ)	0,5249	0,5419	0,4840	0,5105
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,2577	0,2814	0,1961	0,2364
$\left(\overline{DP}\right)$**/(μ)**	0,4910	0,5193	0,4051	0,4631
Acertos	13	13	17	17

Como pode ser visto na tabela acima, os resultados dos Casos I e II se apresentaram piores do que seus correspondentes apresentados no método do item 3.3.1, onde as fontes discriminadas passaram de 16 para 13 nos dois casos. Felizmente as fontes que apresentaram $DSNR_2$ menores que 0,3 (neste caso), exceto as fontes 2 e 10, possuem ainda um valor alto que possa ser corrigido aumentando os pesos dos conjuntos de pseudo-A's formados por eles.

Com isso, os casos III e IV obtiveram uma melhora igualando o desempenho do método anterior de 17 acertos em 19. Para isso, no entanto, no Caso III precisou-se

elevar o peso do conjunto da fonte 14 e 16 em duas vezes e meia e do 19 em três vezes em relação as outras fonte. No caso IV, os conjuntos das fontes 14 e 16 tiveram seus pesos elevadas em duas vezes e da fonte 19 em três vezes. Abaixo é apresentada a PF_A do Caso IV, que foi escolhido como o melhor resultado deste método por apresentar o maior número de acertos com maior valor médio de $DSNR_2$ e que, por isso, também estará na tabela resumo no fim deste capítulo.

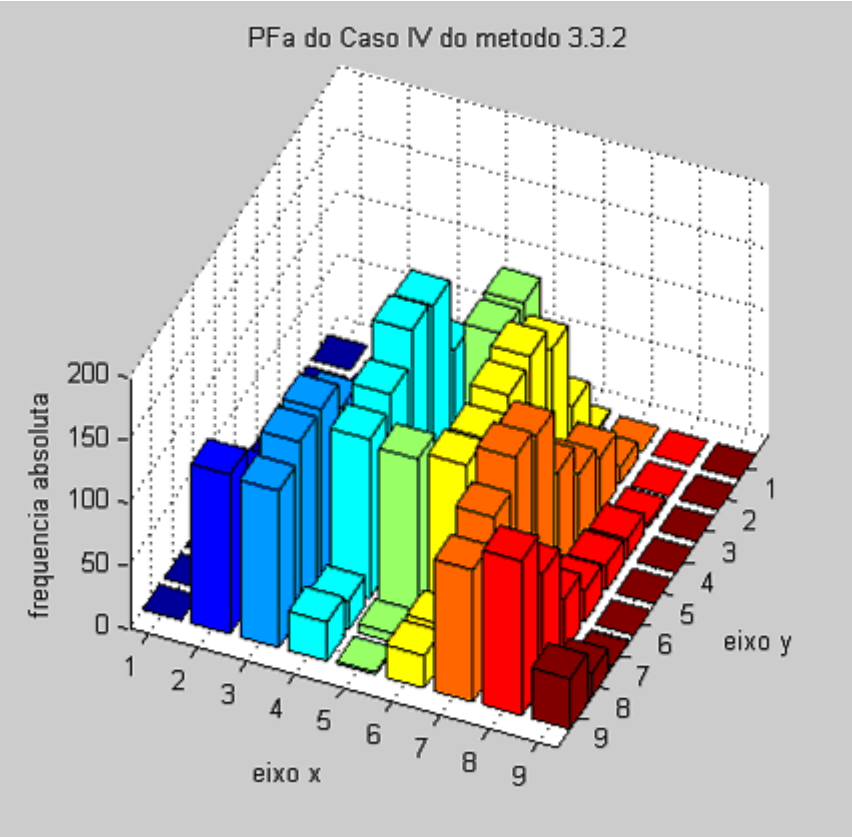


Figura 3.18. PF_A para o Caso IV do método 3.3.2.

3.3.3 Método usando ruído aditivo e subtrativo

O algoritmo desenvolvido neste item é praticamente igual ao caso apresentado no item 3.3.2. A diferença é que agora o ruído adicionado ao "A" fino pode ter sinal positivo ou negativo. Esta idéia de se ter um ruído negativo tem como objetivo preparar uma grande e diversificada base de padrões para a construção do filtro "OU", preparando-o para aplicações reais onde uma imagem possa ser corrompida com os mais diferentes tipos de ruído.

As simulações para este novo caso são um pouco mais complicadas, pois, além do controle dos parâmetros do ruído aditivo, agora controlam-se também os parâmetros do ruído subtrativo. Mas, para facilitar as simulações, fixou-se o valor do σ e t de forma que o ruído subtrativo retire poucos pixels do "A" fino, enquanto os parâmetros σ e t do ruído aditivo continuam sendo variados ao longo das simulações. Abaixo é mostrada a tabela de resultados para 4 casos já consagrados nas simulações anteriores.

Tabela 3.9: $DSNR_2$ para as 19 fontes-testes, utilizando o método de dilatações progressivas com as fontes-testes "podadas" como padrão fino, além de seus respectivos "A" grossos provenientes da dilatação, com o ruído agora podendo ter sinal positivo e negativo.

Fontes	**Caso I**	**Caso II**	**Caso III**	**Caso IV**
1	0,9937	0,9656	0,9451	0,9565
2	0,0179	0,0341	0,0170	0,0293
3	0,5226	0,4175	0,4680	0,3856
4	0,6410	0,6813	0,6359	0,6370
5	0,3053	0,3113	0,3451	0,3484
6	0,4470	0,4646	0,4391	0,4820
7	0,3060	0,2802	0,3561	0,3169
8	0,7089	0,6452	0,6092	0,5651
9	0,7210	0,6964	0,7042	0,6803
10	0,1028	0,1001	0,0874	0,0875
11	0,6428	0,7378	0,5669	0,6455
12	0,8143	0,7934	0,6854	0,6717
13	1,2211	1,2739	1,0765	1,0794
14	0,2660	0,2931	0,3106	0,3325
15	0,7118	0,8794	0,7064	0,8040
16	0,2741	0,2688	0,3012	0,3090
17	0,3374	0,3416	0,3413	0,3295
18	0,7010	0,8234	0,6343	0,7247
19	0,2494	0,2485	0,3040	0,3119
Média da $DSNR_2$ (μ)	0,5255	0,5398	0,5018	0,5104
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,2554	0,2777	0,2156	0,2285
$\left(\overline{DP}\right)$**/(μ)**	0,4860	0,5145	0,4296	0,4478
Acertos	14	13	17	17

Para este método, vê-se que nos Casos I e II, onde todos os padrões possuem o mesmo valor de peso, seus resultados pioraram em relação aos seus similares do item 3.3.1, regredindo de 16 para 14 acertos no Caso I e de 16 para 13 acertos no Caso II. Apesar disso, assim como no método 3.3.2, algumas fontes possuem um bom valor de $DSNR_2$ e, por isso, podem ser melhorada através da mudança do peso de alguns conjuntos de fontes.

Os casos III e IV mostram essa melhora onde se apresentam 17 acertos em 19, assim como também se mostrou na subseção 3.3.1 e 3.3.2. Para o caso III, os conjuntos referentes às fontes 14 e 16 receberam um peso duas vezes maior, e o conjunto 19 recebeu o peso três vezes maior que o restante do grupo (outras 16 fontes). No caso IV, os grupos 14 e 16 também receberam pesos duas vezes maior e o grupo 19 3.3 vezes maiores que o restante das fontes.

A não ser pela pequena melhora do Caso I deste método em relação ao apresentado no seu correspondente no subitem 3.3.2, este método não mostrou grandes avanços em resultados mesmo com a mudança no tipo de ruído. Por isso, novas mudanças devem ser realizadas.

A melhor forma de se perceber o desempenho do filtro OU final gerado em cada método é vendo os resultados para os Casos I e II de cada método apresentado nesta seção, pois todos os pseudo-A's estão com pesos iguais. Desta forma, pode-se ver qual o melhor conjunto de padrões gerados pelos métodos que discrimina mais letras sem o artifício de ajuste de peso. É importante salientar que, se cada fonte que não foi discriminada nestes casos tiver uma $DSNR_2$ mais ou menos alta, isso pode ser corrigido manipulando os pesos como mostrado nos Casos III e IV e aí pode se ver o máximo que aquele conjunto pode oferecer em resultados.

A seguir se encontra o gráfico da PF_A para o Caso IV, que foi escolhido como o melhor resultado deste método. Neste método, o Caso III também apresentou um resultado muito bom, pois os valores médios e a razão $\left(\overline{DP}\right)/(\mu)$ das $DSNR_2$ deram muito parecidos. Desta forma, optou-se pelo Caso IV, por apresentar a maior média, incluindo melhor resultado para a fonte 2.

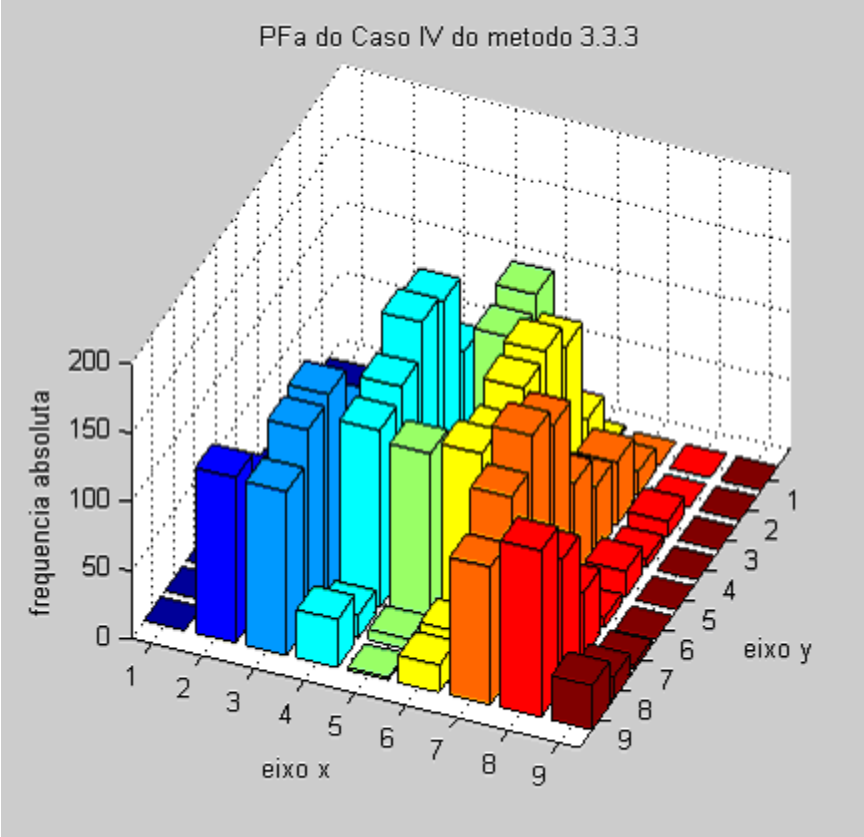


Figura 3.19. PF_A para o Caso IV do método 3.3.3.

Após estes três métodos é notório que a discriminação das fontes 2 e 10 do conjunto-teste (figura 3.2) é muito difícil, visto que suas morfologias são muito diferentes do restante do grupo. Assim como já foi comentado anteriormente, a tentativa de aumentar o peso destas fontes como forma de melhorar suas $DSNR_2$'s não surte o efeito desejado, pois seus valores de DSNR's são muito baixos.

3.3.4 Fontes-testes como "A" finos e seus respectivos "A" grossos provenientes da dilatação

Como podem ser percebidas, as apresentações dos métodos desta seção seguem uma linha de raciocínio, onde um é a evolução do outro. Por isso, este quarto algoritmo continua com o mesmo raciocínio do ruído ser aditivo e subtrativo apresentado do item 3.3.3. No entanto, os "A" finos agora são novamente as fontes-testes presentes na figura 3.2, visto que os melhores resultados obtidos para os Casos I e II foram os apresentados no item 3.3.1 onde se usavam as fontes-testes como "A" fino. Entretanto, a idéia de que cada "A" fino terá seu "A" grosso continua a mesma. Com isso, espera-se que os

resultados melhorem, gerando $DSNR_2$ maiores, pois agora cada "A" fino tem seu próprio "A" grosso.

Os pseudo-códigos apresentados no item 3.3.1 podem ser utilizados também neste método só lembrando que os A's finos e grossos devem ser trocados pelos novos conjuntos. Os testes que serão realizados aqui neste método obedecem aos mesmos casos já realizados nos três subitens anteriores. Logo, a tabela abaixo mostra os resultados para os quatro casos já conhecidos, junto com seus valores médios e desvios padrões médios, que ajudam na escolha do melhor conjunto do método.

Tabela 3.10: $DSNR_2$ para as 19 fontes-testes utilizando o método de dilatações progressivas com as fontes-testes como padrão fino, além de seus respectivos "A" grossos provenientes da dilatação, com o ruído podendo ter sinal positivo e negativo.

Fontes	**Caso I**	**Caso II**	**Caso III**	**Caso IV**
1	0,7240	0,8086	0,6799	0,7594
2	0,0224	0,0437	0,0302	0,0513
3	0,8176	0,7255	0,7287	0,6830
4	0,8419	0,8721	0,6938	0,7713
5	0,4239	0,4689	0,3378	0,4100
6	0,4221	0,5381	0,4306	0,5203
7	0,2636	0,3025	0,3389	0,3648
8	0,8373	0,8576	0,7613	0,8237
9	0,5886	0,6109	0,5339	0,5752
10	0,1443	0,1374	0,1371	0,1426
11	0,6032	0,6834	0,5327	0,6338
12	0,8405	0,8570	0,8016	0,8405
13	1,2202	1,3056	0,9910	1,0979
14	0,3774	0,4231	0,3008	0,3669
15	0,7354	0,7656	0,5973	0,6682
16	0,1868	0,2130	0,3137	0,3293
17	0,5075	0,4720	0,4880	0,4823
18	0,6719	0,7765	0,5829	0,7222
19	0,2695	0,3082	0,3236	0,3060
Média da $DSNR_2$ (μ)	0,5525	0,5879	0,5055	0,5552
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,2479	0,2509	0,1946	0,2130
$\left(\overline{DP}\right)$**/(μ)**	0,4487	0,4269	0,3849	0,3836
Acertos	15	16	17	17

Como previsto, os Casos I e II apresentaram resultados melhores que os seus respectivos apresentados nos itens 3.3.2 e 3.3.3 e ainda deu um resultado mais parecido com o 3.3.1, onde só o Caso I deu uma queda de 16 acertos para 15.

Nos Casos III e IV, todos os algoritmos propostos neste método de dilatação progressiva apresentaram o mesmo desempenho de 17 acertos. No caso III, o conjunto referente à fonte 16 recebeu o peso duas vezes maior e o conjunto 19 três vezes maior que o restante das fontes. No Caso IV, apenas o conjunto 16 recebeu peso duas vezes maior que o restante. Abaixo, apresenta-se a PF_A para o Caso IV, que foi escolhido como o melhor resultado por apresentar o maior número de acertos, com maior valor médio de $DSNR_2$ e menor razão $\left(\overline{DP}\right)/(\mu)$ deste método, que por isso também estará na tabela resumo no fim deste capítulo:

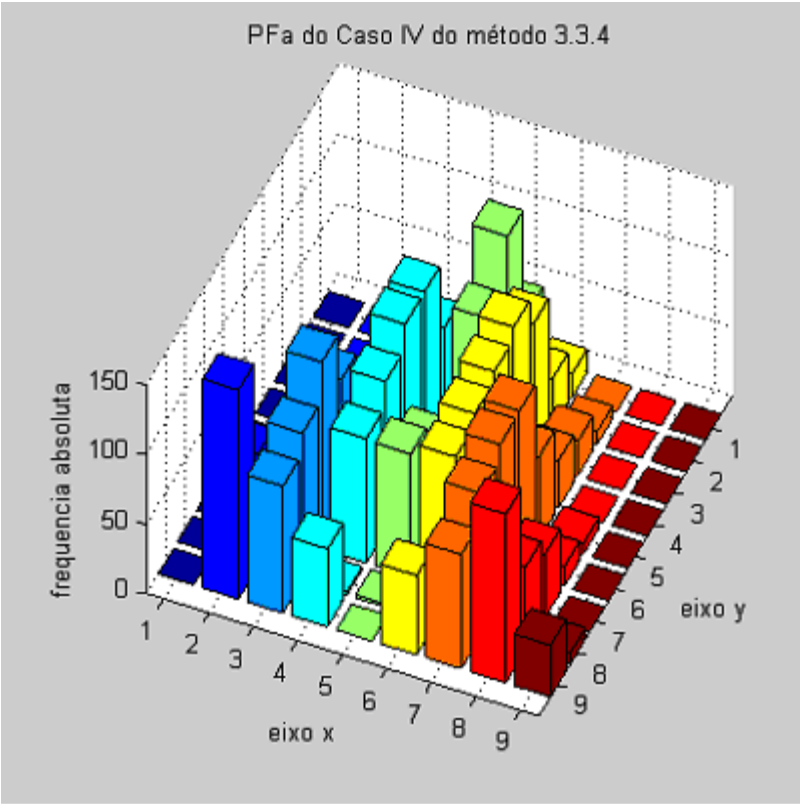


Figura 3.20. PF_A para o Caso IV do método 3.3.4.

3.3.5 Conclusões finais

Após os quatro métodos apresentados nesta seção, viu-se que, com a idéia de se formar um conjunto de pseudo-A's a partir de A's finos diferentes, obteve-se uma melhora em relação aos resultados do método utilizando apenas um padrão como "A" fino (seção 3.2). A utilização de "A" grossos que dependem de cada "A" fino buscou aumentar o valor da $DSNR_2$ das fontes-testes, o que pode ser observado na maior parte dos casos. Outra mudança importante foi a criação de ruído aditivo e subtrativo. Esta mudança é importante, pois visa aumentar o leque de variedades na morfologia que o pseudo-A pode apresentar se aproximando assim de uma aplicação mais real onde não se sabe como é o ruído.

Outros testes também foram importantes, como os dois modos de se buscar o melhor conjunto de fontes que comporão o filtro "OU". Este teste foi muito útil, pois permitiu perceber a influência do σ (desvio padrão do filtro gaussiano) e do *t* (constante de limiar usada na binarização do ruído) na criação do conjunto de pseudo-A's. Uma observação importante para este método é que, no geral, de 150 a 200 fontes (pseudo-A's) foram necessárias para compor o filtro "OU" final, enquanto o método de dilatações progressivas usando apenas um "A" fino, seção 3.2, tinha em média 20 fontes. Com isso, o treinamento do filtro "OU" se tornou mais lento, mas, em compensação, os resultados também melhoraram do método apresentado na seção 3.2 para os quatro apresentados no método desta seção. Mas como o treinamento só é realizado uma vez, este tempo não é muito importante, pois este não precisa ser realizado online igual a discriminação dos padrões.

Enfim, por último, como já foi citado nos itens anteriores, o maior problema detectado foi com relação às fontes 2 e 10. Como já explicado, ao analisar a figura 3.2, onde estão todas as fontes-testes, percebe-se que as fontes 2 e 10 têm suas formas as mais diferentes do grupo e, por isso, apresentam valores baixos de $DSNR_2$. Foi percebido que, mesmo as duas fontes tendo representantes na construção do filtro OU, nada garante que a resposta dele para estes padrões seja alta. Isto porque, como também já foi explicado, o filtro "OU" é composto por uma média ponderada dos gabaritos que o compõe e, como a maioria das fontes são iguais entre si e muito diferentes da 2 e 10, mesmo com o aumento de seus pesos, não é suficiente para que o filtro responda bem a esses padrões. A seguir será apresentado o quarto e último método que apresenta uma pequena mudança na tentativa de fazer um filtro que responda melhor as fontes 2 e 10.

3.4 Método usando um autoteste dos conjuntos de padrões-bases para o filtro “OU”

3.4.1 Descrição do método

O método que é apresentado nesta seção é uma evolução dos anteriores, onde os conceitos básicos continuam os mesmos. Para este caso, os “A” finos continuam como as fontes-testes e os “A” grossos são as dilatações dos “A” finos. Uma das mudanças que este método apresenta foi com relação ao ruído subtrativo. Nos métodos anteriores, qualquer pixel do “A” fino podia ser retirado. Neste método, haverá um controle de quais pixels podem ser retirados, assim como já havia um controle de quais pixels podiam ser adicionados.

O controle da retirada do pixel utiliza também uma operação morfológica que é chamada de erosão. Esta é dual da operação de dilatação já explicada no item 3.2.2. A erosão de um conjunto X por um elemento estruturante B é definida da seguinte forma [19-20]:

$$ero^{B}(X) = X \; ero \; B = \{x \in E : B_x \subset X\}, \tag{3.4}$$

onde *E* é um conjunto não vazio. A equação acima revela que, se o elemento estruturante B_x (elemento estruturante centrado no pixel x) estiver contido dentro do conjunto X, então o pixel x pertence a um conjunto não vazio que é a imagem resultante. Ao contrário da dilatação, onde basta que apenas um pixel do elemento estruturante esteja contido em X, aqui na erosão todos os pixels devem está contidos em X para que o pixel centrado por B_x receba o valor 1. O efeito da erosão em uma imagem é afiná-la de acordo com as relações entre pixels vizinhos empregado pelo elemento estruturante. Outras informações também podem ser vistas em [19-20].

Visto a teoria de erosão, pode-se explicar melhor agora como é feito o controle da retirada de pixel. A idéia que se teve foi de erodir o “A” fino na direção vertical e horizontal de forma a deixar somente os pixels que seriam como a origem da fonte. Após a erosão, seria realizada então a subtração do “A” fino pela sua erosão. Esta diferença conteria então os pixels que podem ser retirados pelo ruído. É claro que, após

a retirada do pixel, a imagem também sofre o controle pra ver se sua conectividade é 8. Veja o esquema abaixo que ilustra o processo:

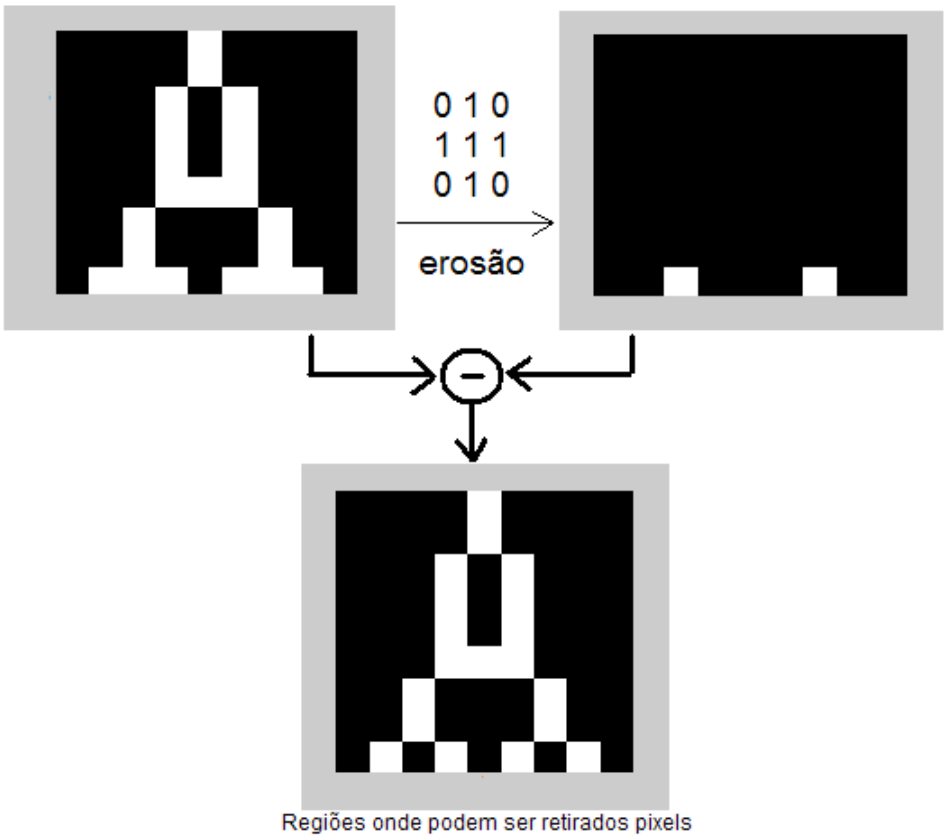


Figura 3.21. Processo para retirada de pixel.

Outra mudança que teve neste método e que é a mais importante foi no processo de como escolher o melhor conjunto de pseudo-A's que comporão o filtro OU. No último método apresentado, esta escolha poderia ser feita sob duas formas:

A primeira (Caso I) era rodar o programa para cada um dos "A" finos e procurar pelo conjunto que discriminasse o máximo de fontes possível sendo então o escolhido para aquela fonte. Na segunda forma (Caso II), o programa também era executado para cada fonte fina, só que agora se procuraria pelo conjunto que gerasse a maior $DSNR_2$ para aquela determinada fonte.

A principal motivação para mudar estes dois métodos de escolha foi que, apesar dos quatro algoritmos apresentados no método anterior apresentarem bons resultados, as fontes 2 e 10 se apresentaram com valor de $DSNR_2$ muito baixo em todos os casos.

Desta forma, imaginou-se inicialmente o seguinte processo para a escolha do conjunto:

1. Limitar o número máximo de fontes que comporão o conjunto. Nesta tese, experimentalmente, este número foi escolhido como 18;
2. Executar o programa 10 vezes para vários pares de σ e t. Só que as fontes que seriam testadas pelo filtro "OU" seriam as próprias fontes que compuseram o filtro. Desta forma, poder-se-ia ver qual o desempenho do filtro com as próprias fontes que o criaram;
3. Calcular a média de $DSNR_2$ de cada uma das 10 realizações feitas e posteriormente calcular a média total para um determinado par de σ e t.
4. A realização escolhida seria aquela que teve a maior média dentro do conjunto de σ e t com a maior média total.

Este processo acima busca por uma forma mais rápida de se achar um conjunto de boa qualidade. No item 2 da descrição do processo, comentou-se em executar o programa 10 vezes. Este número foi fixado, pois, como o processo inicialmente consiste em uma soma de um termo fixo ("A" fino) com um aleatório (ruído), a probabilidade de se ter um conjunto de boa qualidade após 10 iterações ou 1000 seria a mesma.

A partir disto, iniciaram-se então as simulações deste método para a letra A. Foram estabelecidos inicialmente 23 pares diferentes de σ e t, onde, para cada par, o programa rodou pelo menos 10 vezes para cada uma das 19 fontes-testes. Os melhores resultados se apresentaram para valores de σ e t altos, entre 0,65 e 0,75, onde poucos pixels são adicionados e subtraídos. Com valores mais baixos, os padrões ficam muito ruidosos e o filtro "OU" aplicado neles mesmos oferece valores mais baixos de $DSNR_2$. Para este método, de forma a facilitar os testes, os valores de σ e t serão iguais tanto para o ruído aditivo quanto para o subtrativo. Veja um exemplo a seguir da escolha de um conjunto:

Tabela 3.11. Exemplo da escolha do melhor conjunto.

Sigma	4,819	4,403	3,262	1,799	2,549	2,096	4,372	2,114	5,319	2,238	
0,75	4,421	5,735	3,587	2,453	2,365	3,465	2,701	2,705	5,245	5,085	
Thres	3,153	4,065	4,197	3,620	3,444	3,468	3,124	2,206	2,951	6,748	
0,70	8,074	2,552	3,682	2,824	4,596	4,206	4,633	3,869	7,085	6,468	
	3,314			3,539	3,544	2,428	5,943	1,897	5,232		
	5,376			2,613	5,627	3,539		6,803	6,948		
	3,845			3,101		1,378		5,083			
	8,555			2,878		1,739					
	5,250					5,626					
						4,258					Média Final
						5,451					
Média	5,201	4,189	3,682	2,853	3,688	3,423	4,156	3,525	5,463	5,135	4,132

No exemplo mostrado na tabela acima, percebe-se que o melhor conjunto, de acordo com o método proposto, seria a nona realização, com média de 5,46. No entanto, em alguns casos nem sempre a realização com maior média se mostra a melhor solução para uma determinada fonte.

A explicação está novamente na fórmula do filtro “OU”, que, como já foi exposto, baseia-se numa média ponderada de todos os padrões que comporão o filtro. Imagine por exemplo no caso de uma fonte que teve como vencedor um conjunto com apenas 3 ou 4 padrões. Qual seria o peso deste conjunto frente a por exemplo 150 outros padrões? Muito pequeno claro. Por isso, em alguns testes, a escolha condicionada somente à maior média não esteve totalmente correta apresentando resultados abaixo do esperado.

Com isso, após alguns testes, mudando os conjuntos, observou-se que o melhor conjunto dentre os já existentes seria aquele que tivesse uma média alta e que também possuísse muitos padrões. Na verdade, é um compromisso entre estes dois fatores. Observando novamente a tabela acima, percebeu-se que os conjuntos 1 e 6 apresentaram resultados melhores que o nono.

O conjunto 1, apesar de ter uma média de $DSNR_2$ menor que o conjunto 9, possui três padrões a mais. O conjunto 6 possui uma média bem menor que o 9, mas, em compensação, possui cinco padrões a mais, o que faz também possuir um resultado melhor. Com estes resultados, a nova regra seria então realizada através de um compromisso entre maior média e o maior número de padrões dentro de um conjunto. Por isso, a média total para um determinado par de σ e t não fornece mais um fator determinante para a escolha de um melhor conjunto, apesar de ainda poder direcionar a pesquisa.

Abaixo, encontra-se o pseudo-código para este método que faz o autoteste do conjunto de pseudo-A's criados com o filtro OU. No pseudo-código, o ruído utilizado já é considerado como o ruído válido com o controle de retirada de pixel (fig.3.22) e também seu controle de adição já mostrado nas figuras 3.12 e 3.13:

Algoritmo 3.3: Algoritmo para o método no qual se faz o autoteste com filtro OU dos pseudo-A's criados. Este programa é rodado 10 vezes para então se analisarem os conjuntos de $DSNR_2$ procurando pelo melhor conjunto.

```
// x é determinado pelo usuário
// n = no. de pseudo-A's definido p/ usuário
// no. de A finos e fontes-testes = 19
Para i = 1 até 19,
    Para j = 1 até n,
        Pseudo_A (i,j) = Afino (i) + Ruído (j);
    Fim Para
    FiltroOU (i) = ConstroiFiltro(Pseudo_A(i,1:n));
    Para k = 1 até 19,
        RespConv(k) = Conv ( FiltroOU(i), FonteTeste(k));
        DSNR(k) = CalcDSNR (RespConv(k));
    Fim Para
 Fim Para
// Teste final com todos os melhores conjuntos de Pseudo-A's
FonteCerta=0;
FiltroOUFinal = ConstroiFiltro(Pseudo_A(1:19));
Para k = 1 até 19,
    RespConv(k) = Conv ( FiltroOUFinal, FonteTeste(k));
    DSNR(k) = CalcDSNR (RespConv(k));
    Se DSNR(k) > 0.3
       FonteCerta = FonteCerta + 1;
    Fim Se
Fim Para
FonteCerta= FonteCerta-1;
```

3.4.2 Simulações do método

Para as simulações deste método, serão apresentados 4 casos que utilizam o mesmo conjunto de pseudo-A's na confecção do filtro OU. A diferença neste caso aqui está nos pesos estipulados de alguns conjuntos. Veja abaixo a apresentação dos 4 casos:

- Caso a - Teste onde todas as fontes possuem o mesmo peso;
- Caso b - Teste com ajuste nos pesos de alguns conjuntos, de forma a melhorar o desempenho do filtro paras as fontes 2 e 10;
- Caso c - Teste com ajuste de peso em que todas as fontes têm $DSNR_2$ maior que 0,28;
- Caso d - Teste em que se desconsideram as fontes 2 e 10 do conjunto que construirá o filtro "OU".

Os ajustes do pesos nos casos "b" e "c" são realizados pelo usuário a partir de uma breve análise da resposta do filtro "OU" para as 19 fontes-teste.

É observado quais as fontes que estão com a $DSNR_2$ abaixo do limiar de detecção, depois aumenta-se o valor do peso referente ao conjunto de pseudo-A's gerados por esta fonte. Isto é realizado até que se melhore as reposta do filtro paras as fontes 2 e 10, no caso b. Para o caso c, o ajuste é realizado em qualquer um dos conjuntos de fontes que apresentem $DSNR_2 < 0{,}28$.

A seguir, encontra-se a tabela com o resultado final pra este método testando o filtro "OU" com as 19 fontes-testes. Também serão apresentados os seu valores médios e desvios padrões médios, que servem para ajudar na escolha do melhor caso junto com o número de acertos:

Tabela 3.12: $DSNR_2$ para as 19 fontes-testes usando o quarto método com autoteste dos conjuntos de pseudo-A's.

Fonte	***Caso a***	***Caso b***	***Caso c***	***Caso d***
1	0,6250	0,4219	0,3977	0,6503
2	0,1196	0,2594	0,2966	X
3	0,5662	0,5039	0,4816	0,4323
4	0,6191	0,4734	0,4503	0,6687
5	0,4364	0,3003	0,2820	0,5896
6	0,4396	0,3741	0,3649	0,4268
7	0,4501	0,4194	0,4163	0,4735
8	0,7245	0,5874	0,5657	0,5510
9	0,6246	0,4058	0,3738	0,6244
10	0,1358	0,2718	0,2813	X
11	0,4354	0,3219	0,3084	0,4400
12	0,7484	0,6318	0,6091	0,5339
13	0,8711	0,5528	0,5113	0,8371
14	0,3850	0,3000	0,2846	0,5163
15	0,5557	0,3888	0,3664	0,6342
16	0,3204	0,3074	0,3083	0,4241
17	0,4135	0,3547	0,3420	0,4204
18	0,5790	0,4056	0,3859	0,6339
19	0,3231	0,3169	0,3128	0,4217
Média da $DSNR_2$ (μ)	0,4933	0,3999	0,3863	0,5458
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,1552	0,0846	0,0766	0,0968
$\left(\overline{DP}\right)$**/(μ)**	0,3145	0,2115	0,1984	0,1774
Acertos	17	17	15	17

Como pode ser visto na tabela de resultados, o filtro "OU" originado por este método, sem o recurso de se corrigir o peso de alguns conjuntos (*Caso a*), apresentou o melhor resultado até agora com 17 acertos em 19, diferentemente dos outros métodos que acertou no máximo 16 fontes. Note que a fonte 2 obteve uma melhora no seu resultado de $DSNR_2$.

Apesar da melhora de apenas uma fonte de acerto a mais e da leve elevação no resultado das fontes 2 (com a fonte 10 se mantendo), este resultado é muito importante, pois possibilitou no *Caso b* um ajuste nos pesos tal que o filtro ainda discrimine as 17 fontes e tal que a $DSNR_2$ das duas fontes problemáticas apresentem valores bem mais

expressivos, de 0,2594 para a fonte 2 e 0,2718 para a fonte 10, resultados esses que ainda não tinham sido alcançados com o filtro discriminando 17 fontes.

O *Caso c* mostra a possibilidade de que, em se ajustando os pesos dos conjuntos, consegue-se um valor maior que 0,28 para todas as 19 fontes. Este tipo de aplicação serviria para discriminar todas as fontes caso o limiar de detecção fosse diminuído para 0,28. Teste similar a esse foi realizado no item 3.3.1, caso V, onde o máximo que se tinha conseguido era colocar todas as fontes com $DSNR_2$ acima de 0,25.

O Caso d é um caso especial onde se rejeitam as fontes 2 e 10 da confecção do filtro "OU". Este teste visa a assegurar uma $DSNR_2$ mais alta para todas as outras fontes. Este caso pode ser muito útil em casos de aplicações mais reais, isto porque nestes casos os ruídos, por exemplo, não estão limitados a serem conectados à letra. Nestas aplicações eles podem estar presentes em qualquer lugar (no caso desta tese em qualquer lugar da janela 9x9). Por isso é importante que a resposta do filtro OU para a letra sem ruído apresente uma $DSNR_2$ alta, pois, ao se adicionar ruído, a tendência é que este valor diminua até que seja desconsiderado, como a letra que se está procurando. Os pesos neste caso foram distribuídas de tal forma que todas as fontes tivessem uma $DSNR_2$ acima de 0,4.

Os conjuntos que tiveram seus pesos modificados estão discriminados na tabela abaixo para cada um dos casos:

Tabela 3.13: Peso dos conjuntos nos 4 casos deste método 3.4

Conjuntos de fontes	*Caso a*	*Caso b*	*Caso c*	*Caso d*
Conjunto p/ fonte 2	1/19	1,81/22,21	2/22,5	0
Conjunto p/ fonte 10	1/19	2,9/22,21	3/22,5	0
Conjunto p/ fonte 14	1/19	1,5/22,21	1,5/22,5	3/22,5
Conjunto p/ fonte 16	1/19	1/22,21	1/22,5	3/22,5
Conjunto p/ fonte 19	1/19	1/22,21	1/22,5	2,5/22,5
Total p/outras fontes	14/19	14/22,21	14/22,5	14/17,5
Peso total	1	1	1	1

Abaixo é apresentada a figura que ilustra a PF_A, do Caso a, que é considerado o melhor resultado deste método, tomando as $DSNR_2$ das fontes 2 e 10:

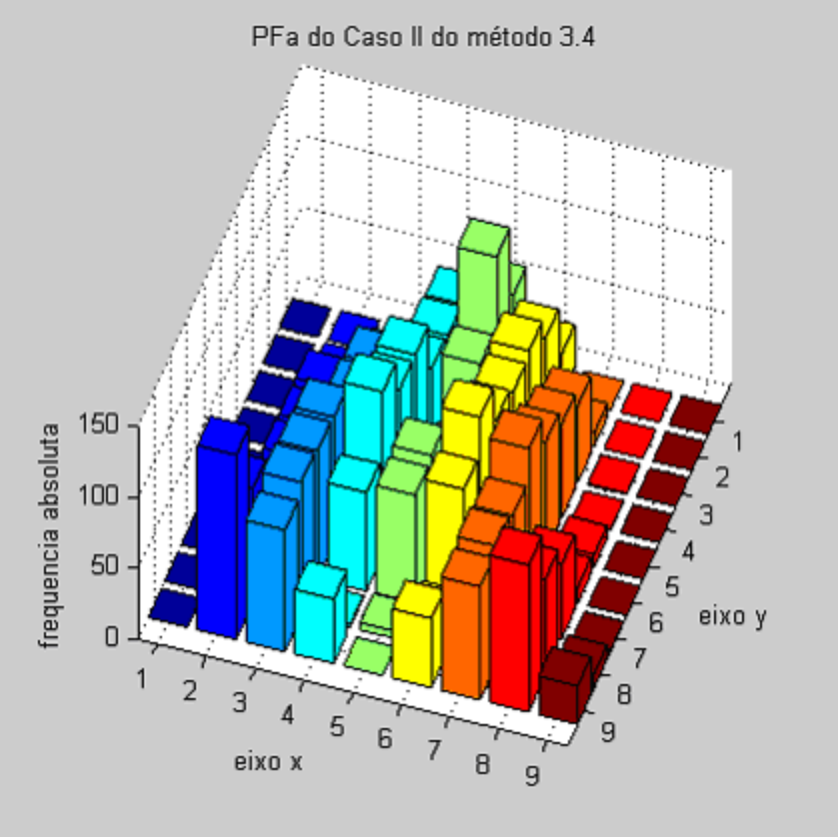


Figura 3.22. PF_A para o Caso II do método 3.4.

Os *Casos a* e *d* serão incluídos na tabela resumo deste capítulo como os melhores resultados deste método. O *Caso a* é considerado o melhor incluindo as fontes 2 e 10. Este teste é a base de comparação para os outros métodos apresentados. O *Caso d* (não incluem as fontes 2 e 10) também será incluído na tabela, pois, foi escolhido como o filtro OU que será utilizado nos testes finais do próximo capítulo. O *Caso b* não será considerado o melhor resultado apesar de ter 17 acertos, pois é um caso especial, onde o que se buscou foi elevar as $DSNR_2$ das fontes 2 e 10 em detrimento das outras fontes. A seguir, apresenta-se a tabela resumo com os melhores resultados do capítulo:

Tabela 3.14: Tabela Resumo com os melhores resultados de cada um dos testes apresentados neste capítulo.

Fonte	**Método 3.1 Caso I**	**Método 3.2 σ/t = 0.4/0.5**	**Método 3.3.1 Caso IV**	**Método 3.3.2 Caso IV**	**Método 3.3.3 Caso IV**	**Método 3.3.4 Caso IV**	**Método 3.4 *Caso a***	**Método 3.4 *Caso d***
1	0,6358	0,3464	0,7170	0,9369	0,9565	0,7594	0,6250	0,6503
2	0,0104	0,0084	0,0520	0,0234	0,0293	0,0513	0,1196	X
3	0,6472	0,5212	0,7801	0,3616	0,3856	0,6830	0,5662	0,4323
4	0,8789	0,4424	0,6981	0,6776	0,6370	0,7713	0,6191	0,6687
5	0,4520	0,3757	0,3930	0,3343	0,3484	0,4100	0,4364	0,5896
6	0,4778	0,2848	0,4698	0,4550	0,4820	0,5203	0,4396	0,4268
7	0,2505	0,1025	0,3771	0,3155	0,3169	0,3648	0,4501	0,4735
8	0,8877	0,3212	0,9095	0,5386	0,5651	0,8237	0,7245	0,5510
9	0,5019	0,645	0,5343	0,7049	0,6803	0,5752	0,6246	0,6244
10	0,1005	0,0879	0,1535	0,0919	0,0875	0,1426	0,1358	X
11	0,5567	0,3474	0,5620	0,6473	0,6455	0,6338	0,4354	0,4400
12	0,7837	0,3585	0,9294	0,6654	0,6717	0,8405	0,7484	0,5339
13	1,1471	0,4429	1,1029	1,0603	1,0794	1,0979	0,8711	0,8371
14	0,4396	0,3747	0,3525	0,3200	0,3325	0,3669	0,3850	0,5163
15	0,5914	0,331	0,6193	0,8447	0,8040	0,6682	0,5557	0,6342
16	0,1266	0,0805	0,3103	0,3101	0,3090	0,3293	0,3204	0,4241
17	0,4480	0,3061	0,4935	0,3398	0,3295	0,4823	0,4135	0,4204
18	0,6743	0,354	0,6537	0,7639	0,7247	0,7222	0,5790	0,6339
19	0,2755	0,2346	0,3065	0,3074	0,3119	0,3060	0,3231	0,4217
Média de DSNR (μ)	**0,5203**	**0,3140**	**0,5481**	**0,5105**	**0,5104**	**0,5552**	**0,4933**	**0,5458**
Desvio Padrão Médio $\left(\overline{DP}\right)$	0,2232	0,1150	0,2146	0,2364	0,2285	0,2130	0,1552	0,0968
$\left(\overline{DP}\right)$**/(μ)**	0,4289	0,3664	0,3915	0,4631	0,4478	0,3836	0,3145	0,1774
Acertos	14	13	17	17	17	17	17	17

3.5 Conclusão

A partir da tabela resumo, alguns resultados se destacam como os melhores do capítulo, que são os do método 3.3.1 e 3.3.4 (Caso IV) e o método 3.4 (Caso a). Nos três casos, foram obtidos 17 acertos em 19 possíveis. A média de $DSNR_2$ para os dois primeiros se mostrou mais alta do que para o último, que apresentou um desvio padrão menor. Ou seja, o método 3.4, que utiliza o autoteste do conjunto de pseudo-A's, apresentou as $DSNR_2$'s melhor distribuída. Com isso, as fontes 2 e 10 tiveram suas $DSNR_2$ mais próximas do valor médio de todo o conjunto.

A partir destes três resultados comentados acima e fazendo uma ponderação entre o número de acertos, média e desvio padrão médio da $DSNR_2$ e da razão entre $\left(\overline{DP}\right)/(\mu)$, escolheu-se o método da seção 3.4 (que realiza o autoteste dos pseudo-A's antes de se criar o filtro OU final) como o melhor do capítulo, por ser o mais justo entre todas as fontes-testes e por não ter sido necessária nenhuma manipulação dos pesos para que se achassem os 17 acertos.

Além disso, este método também apresentou o melhor resultado com 17 acertos e com as fontes 2 e 10 com $DSNR_2$ acima de 0,25, valor esse que não tinha sido conseguido em nenhum dos outros métodos. Desta forma, este método se mostrou o mais consistente e não foi à toa que foi apresentado por último.

No entanto, como foi explicado anteriormente, no *Caso d* deste método, as imagens que virão no próximo capítulo conterão ruídos mais complexos que os simulados até agora. Por este motivo, para as aplicações mais reais do próximo capítulo, o filtro gerado no *Caso d* será o escolhido para as próximas simulações com a letra A. Foi decidido que é mais necessário que o filtro responda com $DSNR_2$ alta para as fontes mais comuns do que tentar discriminar todas até as menos comuns como a fonte 2 e 10.

A partir desta decisão, as escolhas das fontes-testes para as outras letras do alfabeto serão escolhidas de tal forma que problemas como esses gerados pelas fontes 2 e 10 da figura 3.2 não aconteçam. As tabelas de resultados geradas pelo filtro "OU" e as figuras das fontes para as outras letras estão apresentados no Apêndice A desta tese.

Capítulo 4

Simulações com imagens reais

Uma forma de avaliar-se a robustez do filtro OU é propor experimentos que exijam um bom desempenho dele sobre várias formas. Por isso, o método da seção 3.4, que utiliza dilatações progressivas, escolhido o melhor método desenvolvido aqui, passará por uma bateria de experimentos que exijam a robustez tanto para imagens binárias como imagens em escala de cinza.

A forma de se medir a robustez seria testar o desempenho do filtro na presença de ruído. O teste mais simples é testar o filtro na presença de ruído binário; outro teste é com o ruído estruturado, onde o plano de fundo da imagem (background) seria o ruído. Isto por que, pela formulação (equação 2.21) de filtro discriminativo por restauração de impulso, tudo o que não for o padrão que se queira encontrar é considerado ruído, ou seja, o restante da imagem. Com isso, são propostos quatros testes neste capítulo:

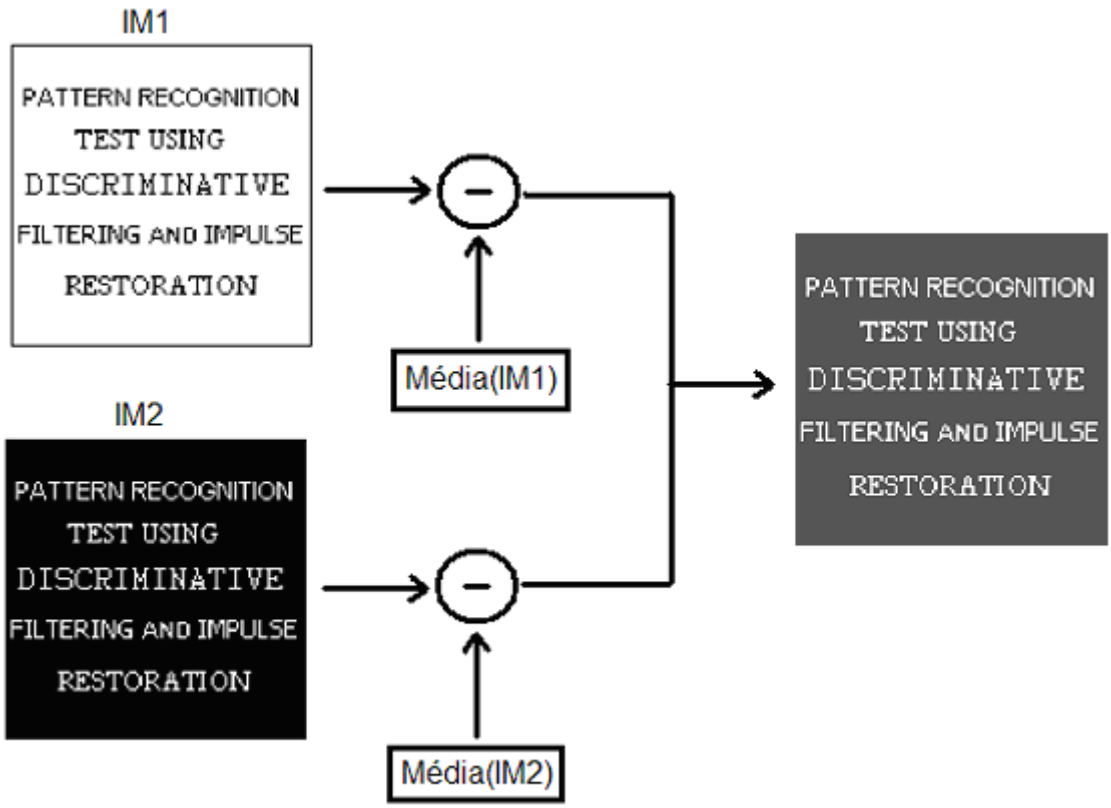
1. Imagem binária sem ruído – O teste mais simples, onde se testa o aproveitamento do filtro na ausência do ruído, que foi como o filtro foi treinado.
2. Imagem binária corrompida por ruído binário – Teste de robustez do filtro para imagens e ruídos binários.
3. Imagem da Lena com letras na cor cinza superpostas à imagem – Primeiro teste com ruído estruturado onde, a própria imagem em escala de cinza é o ruído.
4. Imagem da Lena com letras transparentes – Teste que exigirá mais do filtro OU; neste exemplo, o ruído estruturado, além de ser a própria imagem, está também presente na própria letra, que é transparente.

Os testes com imagens binárias descritos acima serão realizados em imagens de tamanho 128x128. Várias fontes de tamanho 8 serão utilizadas nestes experimentos. A imagem que será utilizada no terceiro e quarto experimentos será a da Lena com tamanho de 128x128 também.

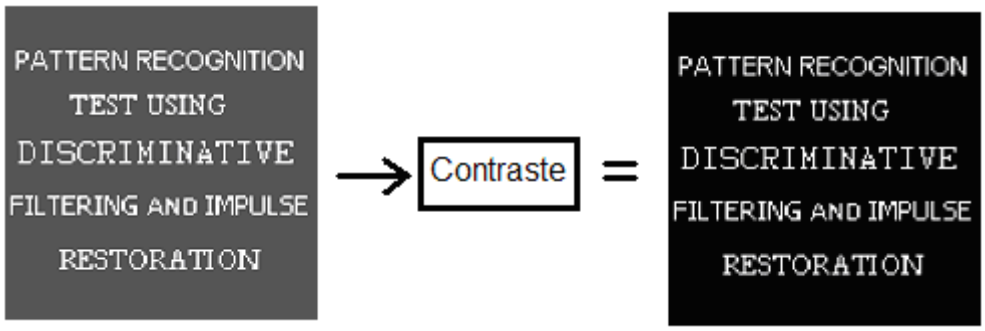
4.1 Imagens binárias sem ruídos

O primeiro teste a ser realizado será com imagem binária sem ruído. Neste teste tanto as imagens com fundo branco e letra preta, quanto as de fundo preto e letra branca apresentaram o mesmo resultado. Para isso, no entanto, como foi explicado na seção 2.2 sobre Restauração de Impulso, a média da imagem deve ser retirada (média nula) para que as equações sejam satisfeitas.

Além disso, para a imagem binária, mais um processo foi realizado, que foi o aumento de contraste da imagem. Como apresentado na figura a seguir, este aumento do contraste faz com que a cor cinza mais escura na imagem resultante se transforme em preta, e a cor cinza clara em branca. Com isso, força-se que a imagem resultante seja somente na cor preta e branca (fundo preto e letra branca, do mesmo modo como os filtros OU foram treinados).



(a)



(b)

Figura 4.1. a) Imagem binária menos sua média b) Aumento do contraste

Antes de explicar como ocorrerão as simulações, resta explicar uma característica importante sobre a filtragem discriminativa em uma imagem qualquer. Como já foi visto em [7], um padrão qualquer, como um ruído aleatório, pode oferecer uma $DSNR_2$ muito alta, determinando assim, uma falsa detecção. Mas, para contornar este problema, um teste é realizado logo após a convolução entre o filtro OU e o padrão.

Para a imagem binária, o teste consiste em realizar uma contagem de pixels dentro da janela correspondente ao padrão 9x9 que apresentou a $DSNR_2$ alta. Depois desta contagem, um limiar é estipulado de forma que, acima de um determinado valor, o padrão não seja considerado como a letra que se está procurando. Geralmente este valor de limiar está relacionado ao número mínimo de pixels que, ao ser retirado de uma letra, ainda mantenha sua morfologia similar com a letra original. Portanto, para cada letra, um valor diferente de limiar pode ser escolhido. Note que, para imagens em escala de cinza, o simples somatório de pixels não pode ser mais realizado, já que os pixels podem agora ter vários valores. Por isso, a energia da imagem é usada como medida de teste para este caso.

4.1.1 Simulações

Com relação às simulações que são apresentadas a seguir, será usada neste exemplo a imagem IM1, presente na figura acima, que possui tamanho 128x128, fundo branco e letras pretas. A frase que será testada é “PATTERN RECOGNITION TEST USING DISCRIMINATIVE FILTERING AND IMPULSE RESTORATION”.

O teste realizado nesta seção tem por objetivo discriminar as fontes “A, E, O e R” na imagem-teste. Para isso, são criados filtros OU para cada uma das letras mencionadas. Como a simulação é realizada para imagem sem ruído, o limiar pode ser fixado, de acordo com os valores mínimos obtidos em experimento, e contidos na tabela de resultados, presentes no Apêndice A desta tese. A seguir, encontram-se as imagens resultantes, marcadas pelas fontes discriminadas, para cada uma das letras. É mostrada também uma tabela com todas a letras encontradas, o nome de suas respectivas fontes, suas respectivas posições (x, y) e suas $DSNR_2$’s.

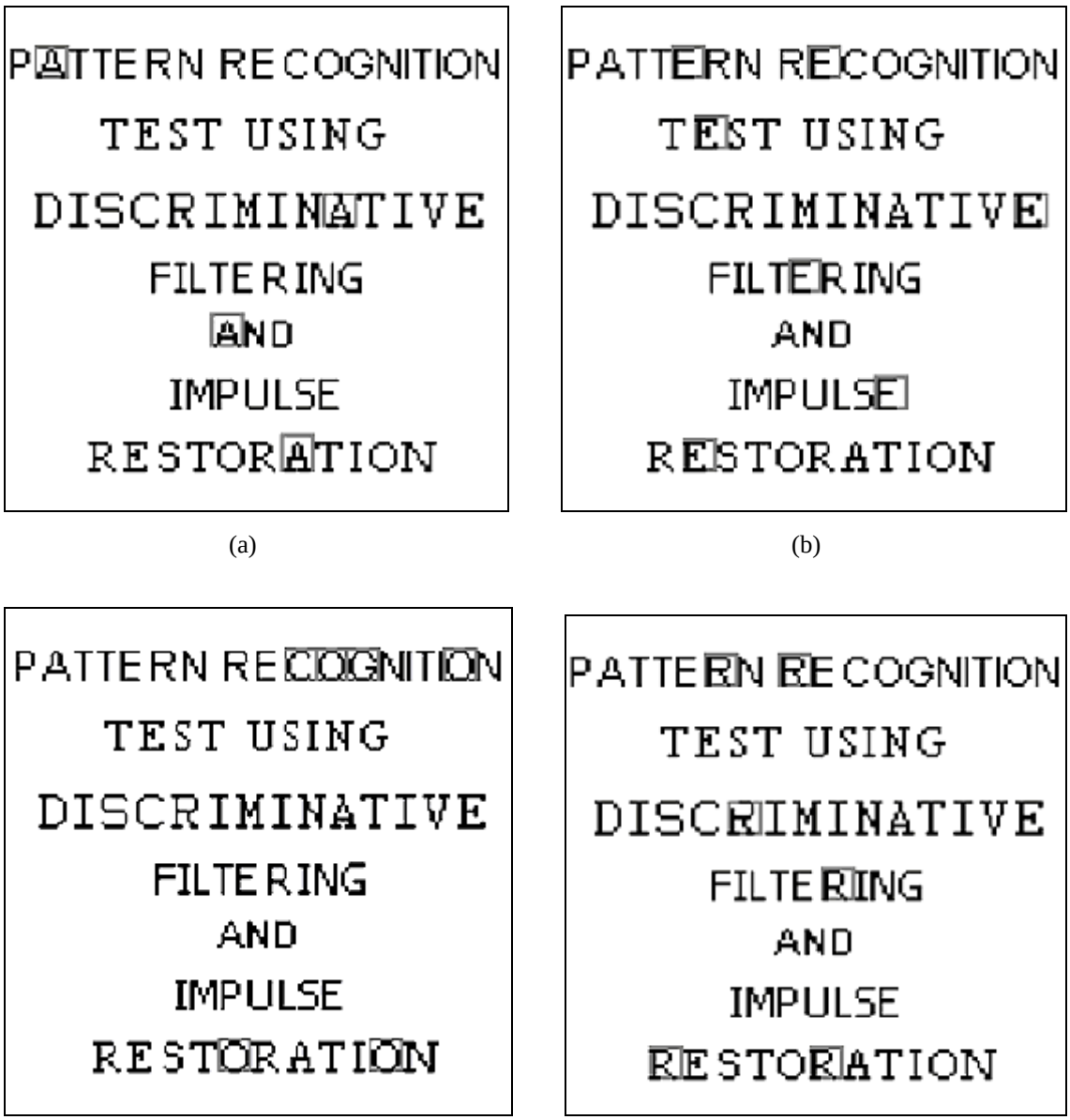


Figura 4.2. Imagens binárias referentes à:
a) Letra A b) Letra E c) Letra O d) Letra R.

Tabela 4.1: Resultados da filtragem discriminativa, utilizando a Imagem binária sem ruído, para as letras “A”, “E”, “O” e “R” com seus respectivos limiares de detecção, posição e $DSNR_2$.

Padrão	Nome da fonte	Limiar de detecção	Pos x	Pos y	$DSNR_2$
A1	Arial	0,42	17	7	0,6503
A2	Courier		55	81	0,6685
A3	Lucida Console		86	52	0,6342
A4	Times New Roman		118	70	0,5509
E1	Arial	0,52	17	27	0,5769
E2	Arial		17	62	0,5769
E3	Times New Roman		35	33	0,6530
E4	Courier		55	115	0,6027
E5	Tahoma		72	57	0,7956
E6	Tahoma		102	79	0,7956
E7	Times New Roman		118	30	0,6530
O1	Arial	0,56	17	70	0,7709
O2	Arial		17	78	0,6133
O3	Arial		17	86	0,5835
O4	Arial		17	111	0,6133
O6	Times New Roman		118	53	0,6133
O7	Times New Roman		118	92	0,6133
R1	Arial	0,43	17	35	0,5628
R2	Arial		17	54	0,5628
R3	Courier		55	41	0,6687
R4	Tahoma		72	65	0,4371
R5	Times New Roman		118	21	0,4764
R6	Times New Roman		118	62	0,4764

Neste primeiro experimento, os testes realizados com as letras “A”, “E” e “R” apresentaram 100% de acerto e não houve qualquer falsa detecção. O limiar alto de detecção foi importante para se evitarem falsas detecções, já que outros padrões podem ter $DSNR_2$ maior que 0.3.

Ao contrário das letras acima comentadas, a detecção da letra “O” não obteve o mesmo sucesso, mesmo usando um limiar de $DSNR_2$ alto, no valor de 0,52. Houve três falsas detecções: uma com a letra “C” (O1) e uma letra “G” (O3). O motivo das falsas detecções é a grande semelhança morfológica entre as três letras, o que é agravado ainda com os métodos aqui utilizados, pois, como se sabe, os filtros OU são realizados a

partir de padrões de letras adicionados ou subtraídos de pixels (ruído) e, por isso, um "O" com ruído tem grande chance de se parecer com "C" ou "G" e vice-versa.

Uma forma de tentar resolver o problema de falsa detecção seria criar um filtro OU_G e um OU_C e comparar as $DSNR_2$ geradas por eles com o filtro OU_O, onde a maior $DSNR_2$ identificaria a letra que está sendo testada (ver tabelas das respcecitvas letras no Apêndice A).

4.2 Imagens binárias com ruídos binários

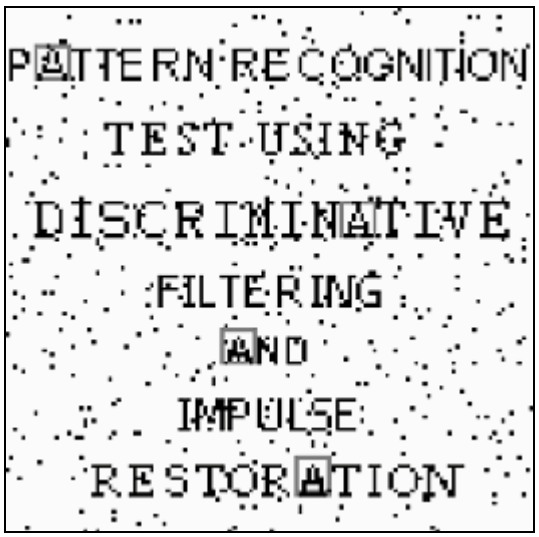
Os testes desta subseção são realizados nas mesmas imagens binárias testadas na seção anterior, só que agora corrompidas por ruído binário, realizado como descrito no capítulo anterior; abaixo é descrito novamente o processo:

- Gera-se um ruído aleatório com distribuição uniforme entre 0 e 1;
- Aplica-se o filtro gaussiano, no ruído, com a finalidade de deixá-lo mais correlacionado espacialmente (variando o seu desvio padrão σ);
- Aplica-se um limiar (t) ao ruído para transformá-lo de tons de cinza para binário onde: se o valor do pixel for acima de um limiar, recebe o valor 1, senão será 0.

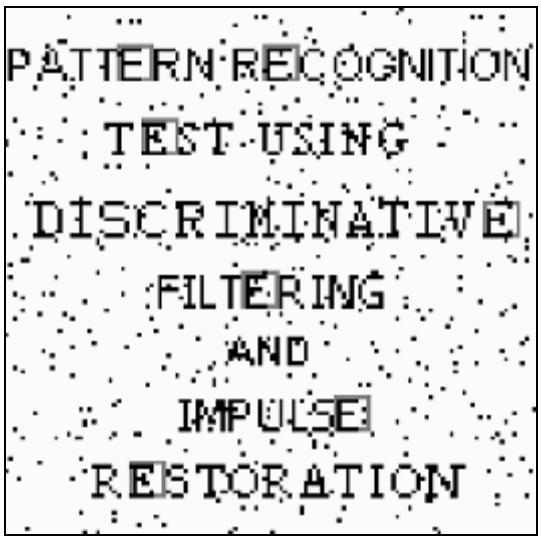
4.2.1 Simulações

O objetivo será de novo detectar as letras "A", "E", "O" e "R" corrompidas por ruído binário. Para isso, os dois parâmetros σ (desvio padrão do filtro) e *t* (limiar para transformação da imagem em binária), que influenciam na criação do ruído, foram escolhidos de forma que a imagem resultante não seja nem pouco e nem muito ruidosa.

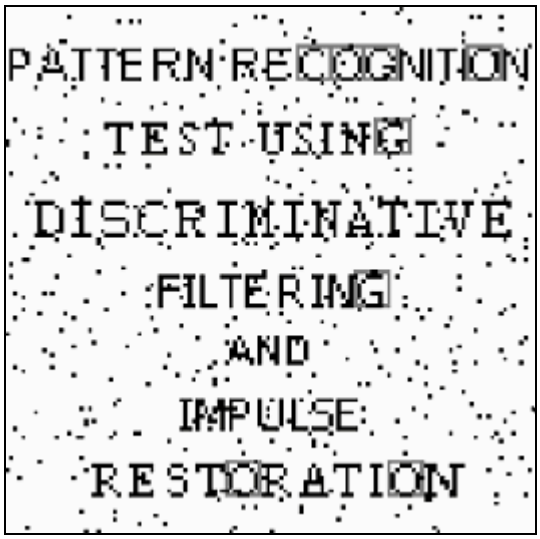
O limiar de detecção, para este caso, não pode seguir os valores da tabela do Apêndice, como no exemplo anterior, devido ao ruído presente na imagem e, por isso, foi estipulado de acordo com cada letra testada. A matriz de covariância "Cb" do ruído (ver equação 2.55) foi considerada como zero. Nas simulações desta seção, também é realizado o teste de contagem de pixels após a convolução, como já explicado na simulação anterior. Sendo assim, como foi mostrado na seção anterior, as imagens e tabelas serão apresentadas em seguida:



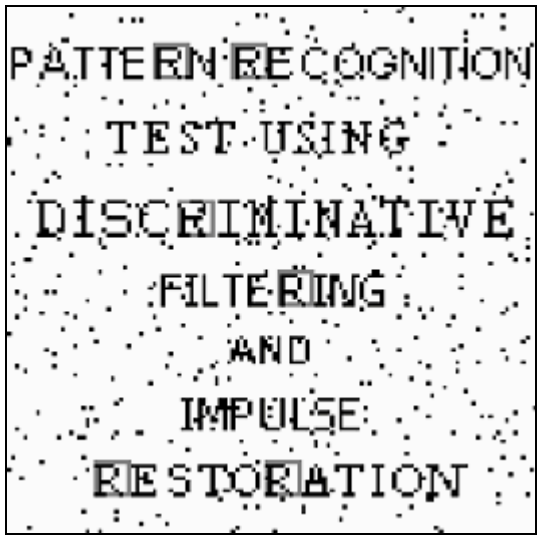
(a)



(b)



(c)



(d)

Figura 4.3. Imagens com ruídos filtrados, com σ = 0.2 e t = 0.03, referentes à:

a)Letra A b) Letra E c) Letra O d) Letra R.

Tabela 4.2: Resultados da filtragem discriminativa, tomando uma imagem binária com ruído, para as letras “A”, “E”, “O” e “R” com seus respectivos limiares de detecção, posição e $DSNR_2$, usando ruído binário filtrado com $\sigma = 0.2$ e $t = 0.03$

Padrão	Nome da fonte	Limiar de detecção	Pos x	Pos y	$DSNR_2$
A1	Arial	0,4	17	7	0,5576
A2	Courier		55	81	0,6686
A3	Lucida Console		86	52	0,5539
A4	Times New Roman		118	70	0,4098
E1	Arial	0,4	17	27	0,5529
E2	Arial		17	62	0,4196
E3	Times New Roman		35	33	0,6140
E4	Courier		55	115	0,6028
E5	Tahoma		72	57	0,5088
E6	Tahoma		102	79	0,6265
E7	Times New Roman		118	30	0,6140
O1	Arial	0,3	17	70	0,6732
O2	Arial		17	78	0,5847
O3	Arial		17	86	0,4390
O4	Arial		17	111	0,7710
O5	Times New Roman		35	89	0,3180
O6	Tahoma		72	84	0,3850
O7	Times New Roman		118	53	0,3039
O8	Times New Roman		118	92	0,4699
R1	Arial	0,35	17	35	0,4715
R2	Arial		17	54	0,4585
R3	Courier		55	41	0,5891
R4	Tahoma		72	65	0,4371
R5	Times New Roman		118	21	0,4764
R6	Times New Roman		118	62	0,3768

O teste com uma imagem corrompida pelo ruído apresentou resultados semelhantes ao teste da seção anterior. Como esperado, a $DSNR_2$ da maioria dos padrões apresentou uma queda devido ao ruído presente em cada um deles. Com isso, o limiar de detecção dos padrões também teve que ser diminuído, como mostrado na tabela acima.

Com relação aos padrões testados, as letras “A”, “E” e “R” apresentaram o mesmo desempenho, com 100% de acerto e nenhuma falsa detecção. Já para a letra “O”, houve

uma piora, pois existiram mais duas falsas detecções: a letra “G” nas palavras “USING” e “FILTERING”. Na verdade, estas letras “G” já possuíam $DSNR_2$ alta no primeiro teste, no valor de 0,48 e 0,5 respectivamente, só que não havia sido detectada porque o limiar era de 0,56. Mas, no presente teste, o limiar teve que ser diminuído para 0,3, porque o padrão O7 apresentou-se bastante corrompido por ruído e, por isso, as letras “G” entraram nos conjuntos dos padrões discriminados.

4.3 Imagem Lena sobrescrita com letras em tons de cinza

Os próximos testes que são realizados nesta seção têm como objetivo discriminar as letras “A”, “E”, “O” e “R” em tons de cinza sobre a imagem Lena de tamanho 128x128 em escala de cinza também. Este teste é um pouco mais complicado que os realizados com imagens binárias, pois, tanto as letras quanto o fundo da imagem estão em tonalidades de cinza. Em alguns lugares, inclusive, o tom de cinza da letra é praticamente igual ao do fundo da imagem, o que dificulta e até impossibilita a detecção do padrão.

Assim como nos exemplos anteriores, a imagem (com média zero), que será convoluída com o filtro OU, recebe um ajuste em seu contraste. O seu nível máximo e mínimo de luminância é ajustado para branco e preto (1 e 0), respectivamente. É importante dizer que a imagem resultante desta transformação continua em escala de cinza, só que com o seu nível DC elevado e com alguns ruídos filtrados, visto que, os pixels com menor intensidade de luminância são zerados, não influenciando mais na filtragem. Desta forma, consegue-se enfatizar melhor a letra do resto da imagem e, com isso, o resultado da filtragem se torna bem melhor, visto que, o filtro OU é treinado com letras brancas e fundo preto. Veja o exemplo abaixo:

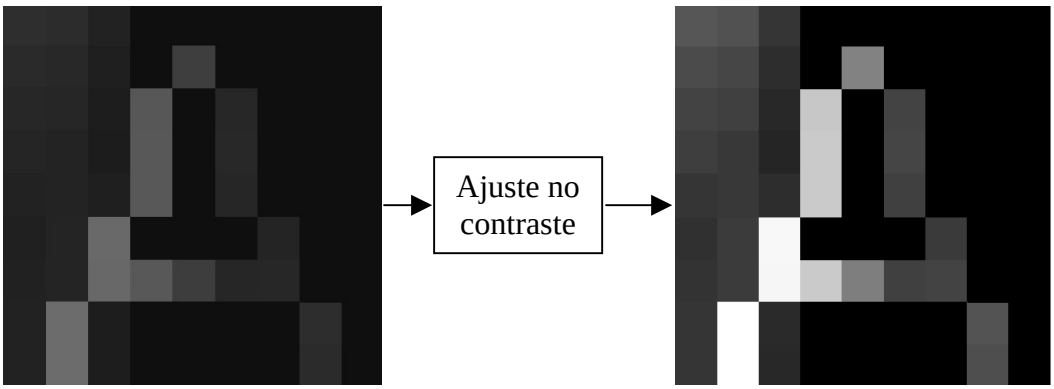


Figura 4.4. Imagem com média zero antes e após o ajuste de contraste.

Assim como nos exemplos para imagens binárias, neste e no próximo exemplo, também é realizado um teste após a convolução, para se evitar que padrões aleatórios sejam aceitos como padrões de letras. A diferença é que, para imagens em tons de cinza, o simples somatório de pixels não é mais válido. Por isso, realiza-se o cálculo da energia de cada um dos padrões que obteve $DSNR_2$ acima do limiar estabelecido e depois se estipula também um valor de corte para rejeitar os padrões indesejados. Veja abaixo como é calculada a energia de um padrão de imagem, no caso aqui, de tamanho 9x9:

$$E = \sum_{x=1}^{9} \sum_{y=1}^{9} \left[\left(pixel(x, y) \right)^2 \right], \tag{4.1}$$

onde pixel(x,y) é o valor da luminância do pixel.

No entanto, antes do cálculo da energia, procurou-se por um filtro que realçasse os padrões em relação ao fundo. Desta forma, o cálculo da energia seria melhor empregado na imagem, visto que só o padrão estaria realçado e o ruído estaria com intensidade tão baixa de luminância, que seria desprezível neste cálculo. Sendo assim, a energia calculada para o ruído, após a filtragem, seria muito pequena, fazendo com que este padrão fosse eliminado do conjunto de padrões discriminados.

Com esta idéia descrita acima, a filtragem a ser utilizada é um operador morfológico chamado Top-hat. Como já descrito, sua principal função é enfatizar as principais características (principais objetos) da imagem, em relação ao fundo da imagem. Ele faz isso equalizando o fundo da imagem com tons de cor escuros, enfatizando assim o padrão principal na imagem.

A equação do Top-hat depende de outro operador morfológico chamado abertura (*opening*) [19-20], que é a erosão de uma imagem seguida de sua dilatação pelo mesmo elemento estruturante, funções estas que já foram explicadas no capítulo anterior. Veja a formulação para o operador abertura abaixo:

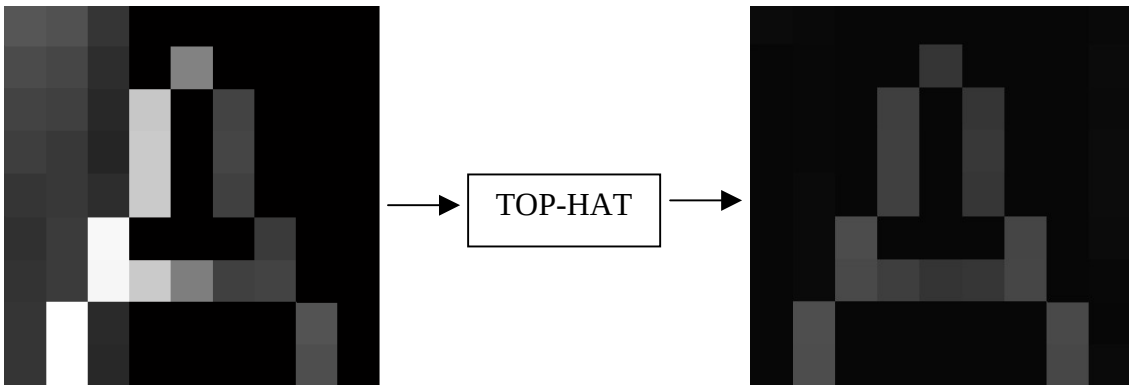
$$abe^B(X) = X \ \ abe \ B = dil^{\tilde{B}}\left(ero^B(X) \right), \tag{4.2}$$

onde B é o elemento estruturante, $ero^B(X)$ é a operação de erosão e $dil^B(X)$ a dilatação.

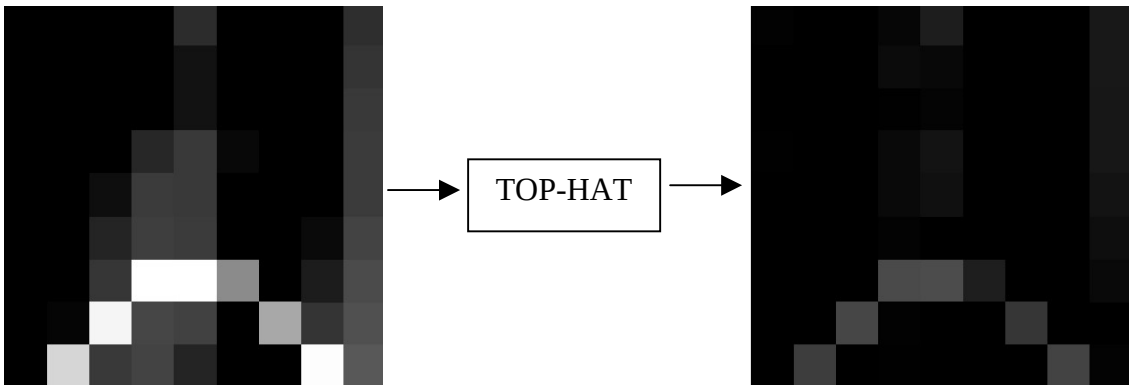
Com isso, pode-se definir a filtragem Top-hat, como a diferença da imagem original pela sua abertura [7-8], veja abaixo:

$$TH^{B}(X) = X - abe^{B}(X) \tag{4.3}$$

Então, posteriormente a essa filtragem, é feito o cálculo da energia para cada padrão 9x9 que obteve $DSNR_2$ maior que o limiar estabelecido. Veja o exemplo a seguir:



(a)



(b)

Figura 4.5. Padrões 9x9 antes e depois do Top-hat.
a) Leta “A” bem definida b) Parte superior de uma letra “O” com ruído.

Com o exemplo acima, percebe-se que as imagens resultantes do Top-Hat continuam em escala de cinza, só que com o nível DC mais baixo. Por isso, a energia de cada padrão é baixa, geralmente abaixo de 2. Com isso, se o limiar de energia do padrão, para este exemplo acima, fosse 1, o padrão na figura 4.5.b já seria eliminado do resultado final.

4.3.1 Simulações

A imagem a ser testada é a imagem-teste usada no primeiro exemplo (sem ruído), com fundo preto e letras em tom de cinza, que possui luminância de 212 (cinza claro). Esta imagem é superposta à imagem Lena formando, desta forma, a imagem a seguir, que foi propositalmente ampliada para ser melhor visualizada:

Figura 4.6. Lena escrita por letras com luminância de 212.

A seguir, apresenta-se o primeiro exemplo com as letras em tom de cinza no valor de 212 de luminância. A matriz de covariância C_b usada neste teste é igual a 0.

Primeiro serão apresentadas as imagens para as letras “A”, “E”, “O” e “R” marcadas pelas fontes discriminadas. Depois, será apresentada uma tabela que contém o nome da fonte para o padrão discriminado, o valor de sua $DSNR_2$, o limiar de detecção e a posição (x,y) do padrão. As fontes que não foram discriminadas terão um “X” no valor de sua $DSNR_2$.

(a) (b)

(c) (d)

Figura 4.7. Imagens da Lena 128x128 escrita por letras na cor cinza:

a) Imagem referente à letra A b) Imagem referente à letra E

c) Imagem referente à letra O d) Imagem referente à letra R

Tabela 4.3: Resultados da filtragem discriminativa, utilizando a imagem da Lena (128x128) em escala de cinza, para as letras “A”, “E”, “O” e “R” com seus respectivos limiares de detecção, posição e $DSNR_2$.

Padrão	Nome da fonte	Limiar de $DSNR_2$	Pos x	Pos y	$DSNR_2$
A1	Arial	0,4	17	7	0,5512
A2	Courier		55	81	X
A3	Lucida Console		86	52	0,6284
A4	Times New Roman		118	70	0,6654
E1	Arial	0,4	17	27	0,5769
E2	Arial		17	62	X
E3	Times New Roman		35	33	0,6530
E4	Courier		55	115	0,6027
E5	Tahoma		72	57	0,4064
E6	Tahoma		102	79	X
E7	Times New Roman		118	30	0,6382
O1	Arial	0,4	17	78	0,6116
O2	Arial		17	111	X
O3	Times New Roman		118	53	0,6188
O4	Times New Roman		118	92	0,5380
R1	Arial	0,4	17	35	0,5665
R2	Arial		17	54	0,4733
R3	Courier		55	41	0,6689
R4	Tahoma		72	65	0,4825
R5	Times New Roman		118	21	0,4764
R6	Times New Roman		118	62	0,4764

Como esperado, os resultados desta simulação foram inferiores ao apresentados para imagem binária, mas ainda assim foram animadores, já que o teste proposto, por ser com imagens e letras em tons de cinza, apresentou uma complexidade maior que os anteriores. É importante explicar nesta simulação que, como se trata de uma imagem mais complexa, a aplicação do limiar com o valor do tom de cinza das letras não faz sumir o fundo da imagem, já que este também esta em tons de cinza. Isso só funciona bem quando as letras estão na cor preta ou branca.

Para a letra “A”, apenas uma letra não foi detectada, que é a letra pertencente à palavra “DISCRIMINATIVE”. Isto ocorreu pois a letra “A” está numa região com alta freqüência de cor e, em determinado local, mais especificamente na perna esquerda da letra, a cor da letra e do restante da imagem é a mesma. Veja figura 4.6.

Por isso, esta letra “A” fica bastante disforme, não sendo, desta forma, detectada pelo método. Todas as outras letras “A” apresentaram $DSNR_2$ alta não apresentando problema algum.

Assim como o problema apresentado pela letra “A”, algumas letras “E” também não foram detectadas, que são as contidas em “RECOGNITION” e “IMPULSE”. Estas também tiveram parte de suas letras em um fundo de imagem com luminância muito próxima, dificultando assim sua discriminação.

Para letra “O”, houve algumas mudanças. as letras “C” e “G” da palavra “RECOGINITION” não foram detectadas, desta vez, como letra “O” graças ao ajuste de contraste e ao limiar escolhido no cálculo de energia. Um ponto negativo foi que o segundo “O” desta palavra não foi detectado, por apresentar os mesmos problemas com o fundo da imagem e sua cor, com já havia ocorrido para as letras “A” e “E”. As outras letras “O” foram discriminadas corretamente, com $DSNR_2 > 0,4$. As outras falsas detecções apresentadas nos teste anteriores não ocorreram, visto que já possuíam $DSNR_2$ mais baixas. Os testes com a letra “R” foram perfeitos para este caso, não apresentando falsa detecção e com todas a letras “R” discriminadas.

Com este teste, percebe-se que o ajuste de contraste e o cálculo da energia para eliminar padrões incorretos foram muito importantes para uma correta discriminação de padrões.

4.4 Imagem Lena com letras transparentes sobrepostas

Até agora, a luminância das letras era constante. Neste teste, foram confeccionadas letras transparentes e que foram superpostas sobre a Lena. O objetivo disto é testar a robustez do filtro OU em letras que variam de cor, de acordo com o fundo da imagem. Nos testes anteriores, considerava-se o ruído somente o fundo da imagem, visto que a letra só tinha uma cor. Neste teste, agora o ruído está inserido na própria letra.

Para isso, no entanto, foi realizada uma combinação linear entre a Lena e a imagem-teste, apresentada na seção 4.1 (figura 4.1.a), só que dividida por um fator de escala para deixar as letras na cor cinza. O controle usando a filtragem morfológica top-hat também foi realizada nesta simulação.

4.4.1 Simulações

Nesta simulação, a imagem-teste (fundo preto e letra branca - figura 4.1.a) foi dividida por 3.5. Sendo assim, as letras brancas, com valor de luminância de 255, assumem a cor cinza com 73 (255/3.5) de luminância. Desta forma, ao ser somada com a Lena, as letras terão sua cor mesclada podendo variar o valor de luminância de 73 até 255. Veja a imagem que é usada nas simulações a seguir:

Figura 4.8: Imagem resultante da soma entre a Lena e imagem usada como teste na seção 4.1 com fundo preto e letra branca dividida por um fator de escala.

A discriminação das letras “A”, “E”, “O” e “R” é novamente testada em imagens 128x128 e o mesmo filtro OU usado nos outros testes também serão usados neste experimento. A covariância da imagem (C_b) será também 0 como no teste anterior. Este valor não é o ideal. Em trabalhos futuros, esta matriz deverá ser estimada através de modelos já connhecidos, o que poderá gerar melhores resultados. Veja a imagens com as letras discriminadas a seguir e também a tabela com os resultados finais, onde as fontes que não foram discriminadas terão um “X” no campo referente a $DSNR_2$:

(a) (b)

(c) (d)

Figura 4.9. Imagens da Lena 128x128 escrita por letras na cor cinzas mesclada:

a) Imagem referente à letra A b) Imagem referente à letra E

c) Imagem referente à letra O d) Imagem referente à letra R

Tabela 4.4: Resultados da filtragem discriminativa, utilizando a Imagem da Lena com letras cinzas mescladas, com luminância entre 73 e 255, para as letras “A”, “E”, “O” e “R” com seus respectivos limiares de detecção, posição e $DSNR_2$.

Padrão	Nome da fonte	Limiar de detecção	Pos x	Pos y	$DSNR_2$
A1	Arial	0,3	10	11	0,3541
A2	Courier		55	81	X
A3	Lucida Console		86	52	0,3627
A4	Times New Roman		118	70	0,5354
E1	Arial	0,3	10	31	0,5390
E2	Arial		23	60	0,5286
E3	Times New Roman		37	33	0,6192
E4	Courier		55	115	0,6470
E5	Tahoma		72	57	X
E6	Tahoma		102	79	X
E7	Times New Roman		118	30	0,2041
O1	Arial	0,3	23	68	0,7872
O2	Arial		23	76	0,2500
O3	Arial		23	84	0,3968
O4	Arial		23	109	X
O6	Times New Roman		118	53	0,3029
O7	Times New Roman		118	92	X
R1	Arial	0,3	10	39	0,5501
R2	Arial		23	52	0,3657
R3	Courier		54	41	0,4194
R4	Courier		55	115	0,3565
R5	Tahoma		72	65	0,3029
R6	Times New Roman		118	21	0,4674
R7	Times New Roman		118	62	0,4817

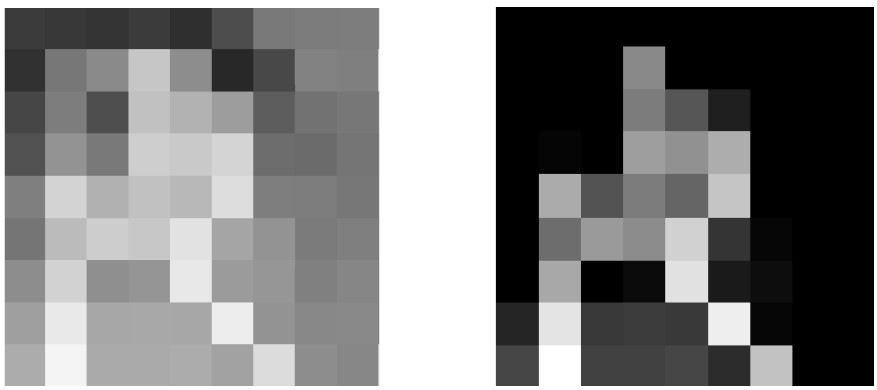
Como esperado, este experimento foi o que mais demandou de esforço do filtro Ou. A discriminação de fontes “transparentes” é muito difícil, visto que, tanto a letra como o fundo da imagem variam de cor. Mesmo assim, resultados animadores foram alcançados, visto que para letra “A”, por exemplo, o primeiro padrão, na palavra “PATTERN”, que corresponde ao padrão ilustrado na figura 4.4, foi discriminado com bom valor de $DSNR_2$, mesmo variando de cor. Entretanto, nesta simulação, a letra “A” da palavra “DISCRIMINATIVE” também não foi detectada, assim como nos testes da Lena com a letra cinza.

Para a letra “E”, mais uma letra não foi discriminada em relação ao teste anterior, que foi a letra da palavra “RESTORATION” que, neste caso, possui $DSNR_2$ 0,2041. Esta apresentou valor baixo, pois suas cores eram muito escuras e se confundiam com o fundo da imagem. Além disso, possui altas freqüências devido ao cabelo da Lena.

A letra “O” também teve um resultado inferior ao dos testes anteriores, onde somente um “O” teve $DSNR_2$ maior que 0,3. Neste caso, as falsas detecções das letras “C” e “G” continuaram. Como pode ser visto na tabela, a letra “O” localizada entre as letras “C” e “G” obteve um valor mediano de $DSNR_2$, de 0,25.

Para esta letra “O” e para a letra “E” com $DSNR_2$ de 0,20 (citada acima), poder-se-ia pensar em baixar o limiar de detecção, para que estas duas letras fossem incluídas no conjunto de padrões discriminados. Mas, isso geraria outras falsas detecções para ambos os casos. Sendo assim, esta proposta foi descartada.

Neste experimento, a única melhora se deu para a letra “R”, onde todas a letras foram reconhecidas. Porém, a falsa detecção apresentada no experimento anterior (letra “E”) se repetiu neste experimento. Para este teste, o filtro OU funcionou muito bem, principalmente para a letra “R” que está presente sobre o olho da Lena, que está bem disforme, veja a imagem desta letra antes e após a correção de contraste:



(a) (b)

Figura 4.10. Letra “R” sobre o olho da Lena. a) Imagem original. b) Imagem original após a subtração da média e o ajuste de contraste, que foi usada na convolução com o filtro OU.

4.5 Considerações sobre métodos práticos de uso da filtragem discriminativa

A partir dos resultados apresentados nesta seção, percebe-se o potencial que a filtragem discriminativa oferece para o reconhecimento de padrões. Foram apresentados experimentos com ruído aleatório binário e em tons de cinza.

No primeiro caso com ruído binário, o filtro não apresentou qualquer erro, exceto no caso da letra “O”, como já foi explicado. Para o caso utilizando imagens em tom de cinza, o filtro foi testado fora de seu contexto, já que não havia sido treinado para tais padrões de imagem. Algumas não detecções e falsas detecções se apresentaram, como esperado, já que, em alguns casos o próprio fundo da imagem se confunde com a letra. Na última seção, com os experimentos com letras transparentes, o processo se torna mais complicado ainda, pois tanto a letra quanto o fundo mudam de cor.

Uma forma de se corrigir estes problemas seria fazer um pré-processamento na imagem, de forma a realçar melhor os padrões de letras em relação ao fundo da imagem. Para isso, no entanto, buscou-se novamente a idéia do filtro morfológico Top-hat, como um modo simples de se garantir este realce.

O processo consiste em aplicar o Top-hat nos padrões, com média nula e sem o ajuste de contraste já explicado anteriormente, que serão convoluidos com o filtro OU. A imagem resultante, que se espera, é composta por um fundo de imagem mais equalizado (mais uniforme) e com o padrão de letra mais visível.

O segundo passo seria aplicar o limiar à imagem, transformando-a em binária, onde, se o valor do pixel for maior que o limiar, receberá o valor 1, senão será 0. O valor de limiar é definido pela média do padrão após o Top-hat. O importante deste pré-processamento, é que ele é automático, sem intervenção do usuário. Veja a seguir o esquema com a descrição do processo de filtragem:

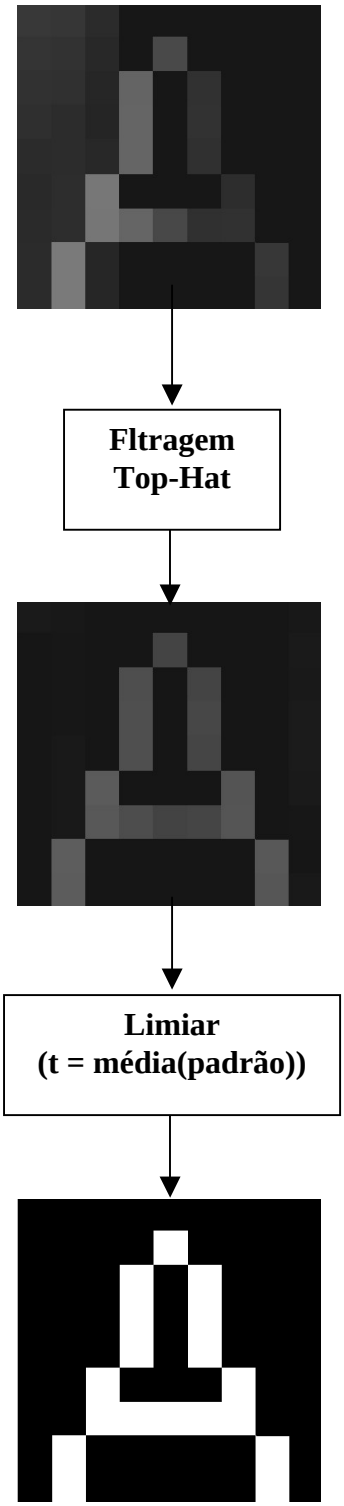


Figura 4.11. Esquema para o realce e binarização do padrão.

A seguir, apresentam-se as imagens marcadas pelos padrões discriminados, pelos seus respectivos filtros OU. Em seguida, encontra-se a tabela com os resultados para cada uma das imagens:

(a) (b)

(c) (d)

Figura 4.12. Imagens da Lena 128x128 escrita por letras na cor cinzas mesclada:

a) Imagem referente à letra A b) Imagem referente à letra E

c) Imagem referente à letra O d) Imagem referente à letra R

Tabela 4.5: Resultados da filtragem discriminativa após o pré-processamento, utilizando a Imagem da Lena com letras cinzas mescladas, com luminância entre 73 e 255, para as letras “A”, “E”, “O” e “R” com seus respectivos limiares de detecção, posição e $DSNR_2$.

Padrão	Nome da fonte	Limiar de detecção	Pos x	Pos y	$DSNR_2$
A1	Arial	0,3	10	11	0,6503
A2	Courier		55	81	X
A3	Lucida Console		86	52	0,4599
A4	Times New Roman		117	70	0,5222
E1	Arial	0,3	10	31	0,5769
E2	Arial		23	60	0,5769
E3	Times New Roman		37	33	0,3639
E4	Courier		55	115	0,3481
E5	Tahoma		72	57	0,3056
E6	Tahoma		102	60	0,3419
E7	Tahoma		102	79	0,2184
E8	Times New Roman		118	30	X
O1	Arial	0,38	23	76	0,4712
O2	Arial		23	84	0,5835
O3	Arial		23	109	0,6528
O4	Times New Roman		37	61	0,3884
O5	Times New Roman		37	89	0,4871
O6	Times New Roman		118	53	0,5293
O7	Times New Roman		118	92	0,3815
R1	Arial	0,35	10	39	0,5628
R2	Arial		23	52	0,5628
R3	Courier		54	41	0,4166
R5	Tahoma		72	65	0,3560
R6	Times New Roman		118	21	0,2399
R7	Times New Roman		118	62	0,2583

De acordo com a tabela acima, os resultados melhores que o teste sem o pré-procesamento. A letra “A” continuou com o mesmo problema, de não discriminar a letra “A” da palavra “DISCRIMINATIVE” e também apresentou um erro de ter reconhecido a letra “A” da palavra “RESTORATION” deslocada em um pixel na vertical.

A letra “E” apresentou uma falsa detecção e duas letras não foram encontradas, anteriormente três letras não haviam sido discriminadas. O filtro OU_O desta vez conseguiu encontrar todas as suas letras, mas apresentou três falsas detecções, uma para a letra “U” e outras duas para a letra “G”. Neste caso, a letra “C”, que sempre vinha sendo discriminada, foi eliminada no teste em que se calcula a energia do padrão.

A letra “R” apresentou um resultado inferior ao encontrado sem o pré-processamento. Desta vez duas letras não foram detectadas, mas ambas apresentaram $DSNR_2$ maior que 0,2. Em compensação nenhuma falsa detecção foi apresentada neste experimento.

A tabela 4.5 mostra alguns campos marcados na cor cinza. As marcações referentes ao índice da letra discriminada indicam uma falsa detecção e as existentes sobre os valores das $DSNR_2$’s indicam padrões que não foram detectados, mas que obtiveram uma $DSNR_2$ mediana.

Capítulo 5
Conclusões

Como já foi dito na introdução desta tese, o reconhecimento de padrões de imagem é uma ciência bastante útil em vários ramos da indústria e, por isso, várias técnicas que realizam este reconhecimento são pesquisadas e apresentadas em vários trabalhos pelo mundo. Com isso, surgiu o interesse em estudar este ramo da ciência e buscou-se por um método que fosse novo e eficiente para este fim. Foi então que se iniciou o estudo pela filtragem discriminativa.

A filtragem discriminativa por restauração de impulso teve sua origem nos trabalhos de Mendonça [7]. Neste trabalho, foi visto que esta filtragem, além de ser simples, tem um grande potencial de discriminação de padrões. Foi também apresentado um projeto de filtro (filtro OU) capaz de discriminar vários padrões de forma bastante simples, bastando apenas escolher um conjunto de padrões que o construiriam e que lhe dariam desta forma o potencial de reconhecer estes padrões ao longo de uma imagem.

Foi a partir deste trabalho que se decidiu criar um filtro OU (filtro que discrimina múltiplos padrões) que fosse robusto ao ruído em uma imagem. Desta forma, foram apresentados, no Capítulo 3, vários métodos que objetivaram dar robustez ao Filtro OU. Como já foi dito, um conjunto de padrões é responsável pela confecção do filtro OU é a escolha deste conjunto o ponto central desta tese. Isto porque, através da escolha certa deste conjunto de padrões de imagens, pode-se conseguir um filtro robusto.

A criação deste conjunto de imagens (pseudo-A’s, para a letra “A”) partiu de uma idéia central, onde cada letra possui uma PF (função densidade probabilidade) que descrevesse a maioria de padrões que uma letra poderia ter. A tarefa de cada um dos métodos criados foi encontrar uma forma de representar esta PF em um conjunto de imagens que, ao confeccionar o filtro OU, lhe desse a robustez ao ruído.

A apresentação dos métodos foi realizada a partir de um encadeamento lógico dos estudos desta tese, onde cada um se mostrou como uma evolução em relação ao método anterior. Por isso, o método da seção 3.4, que utiliza dilatações progressivas com o autoteste dos pseudo-A’s, foi o utilizado nas experiências finais apresentadas no Capítulo 4. Com relação a estes métodos que geram os pseudo-A’s e criam o filtro OU,

é importante ressaltar que o projeto dos filtros pode ser feito offline, deixando somente a filtragem em si para ser realizada online.

Algumas características importantes podem ser obtidas dos experimentos utilizando os métodos aqui apresentados. A principal delas é que o reconhecimento de padrões com letras de baixa resolução (fonte 8) é uma tarefa difícil, pois, como se tratam de imagens com poucos pixels, uma letra tem grandes chances de se parecer com a outra. Isso pode gerar alguns erros de detecção, como apresentado nos Capítulos 2 e 4.

Neste trabalho, este problema ficou mais evidente, pois as imagens que fazem parte do treinamento do filtro OU são imagens de um padrão de letra conhecido, adicionado a um ruído aleatório. Isso faz com que alguns resultados sejam muito ruins.

Veja o exemplo da letra “O”, com a letra “C” e “G”, olhando no Apêndice A desta tese. Pode-se ver as imagens de algumas fontes “O” e “C” usadas nos filtros, onde se percebem que elas são muito parecidas. Daí, conclui-se que, ao se adicionar ruído (com sinal positivo ou negativo), as chances de um “C” virar um “O” ou vice-versa são muito grandes e aí o filtro OU não se torna mais tão eficiente. Mas, enquanto não houver semelhança tal entre os padrões, os métodos se apresentam muito bem.

Como forma de avaliar o método da seção 3.4, um conjunto de experimentos foi proposto, onde a complexidade e a exigência de um bom filtro OU foi requisitada a cada experimento.

Os resultados com imagens binárias com ou sem ruído se apresentaram de forma satisfatória, onde só a letra “O” apresentou problemas. Para as imagens com ruído binário, é importante dizer que as imagens (pseudo-A’s) geradas pelos métodos do Capítulo 3 contemplam apenas padrões com ruído (pixels) conectados aos padrões de letras (com conectividade 4 ou 8). Nas imagens testadas, o ruído pode estar em qualquer lugar dentro da janela 9x9. Sendo assim, o filtro OU criado pode mostrar sua robustez a este tipo de ruído, com resultados animadores, visto que, como já dito, só a letra “O” teve problemas.

Para os testes utilizando letras cinza ou mescladas, pode-se ver o desempenho do filtro OU para um caso em que se exigiu muito da sua robustez. Mesmo com algumas falsas detecções e outras não-detecções, o filtro ainda mostrou um grande potencial, acertando vários padrões e alguns até bastante complicados, como os apresentados na figura 4.5.a e 4.10.

No geral, as não-detecções se apresentaram para padrões onde a letra se confundia, em cor, com o fundo da imagem, ou seja, por apresentar um baixo contraste

com o fundo. As falsas detecções devem-se, principalmente, à semelhança entre as letras (baixa resolução) que, ao serem adicionados com o ruído, geram uma confusão ao filtro.

Foi apresentado também que o filtro OU, em aplicações práticas e específicas, pode ser usado após um pré-processamento, com a finalidade de melhorar seus resultados de detecção.

É importante salientar também que todo o treinamento do filtro "OU" é feito offline. Como já foi dito a filtragem em si que é feita online. A convolução do filtro "OU" com as imagens, de tamanho 128x128, duraram em média quatro minutos em um programa feito em matlab, o que mostra o potencial do método para ser aplicado em experimentos reais, onde pode-se até paralelizar o programa.

Com isso, após a apresentação dos resultados, conclui-se que o método aqui apresentado tem um grande potencial como uma ferramenta para a discriminação de letras na presença de um ruído qualquer. É claro que existem ainda algumas limitações com relação a ruídos mais complexos, mas que podem ser solucionados com algumas modificações. Por isso alguns trabalhos futuros podem ser indicados como apresentado abaixo:

- Criar métodos ou a modificar o apresentado nesta tese, para a geração de pseudo-A's que inclua no seu treinamento o uso de ruídos não conectados às imagens, que possam aparecer em qualquer parte da janela.
- Criar um método que inclua ruídos correlacionados nos cantos das janelas, visando a filtragem correta de padrões finos como a letra I.
- Modelar melhor a covariância da imagem (matriz C_b); pode-se tentar, por exemplo, usar o modelo de Gaus-Markov, com coeficientes de correlação de 0,95. Esta mudança visa também a melhorar o desempenho do filtro OU.
- Gerar filtros OU para fontes maiores, o que teoricamente melhora os resultados dos testes, já que os ruídos aleatórios, retiram ou adicionam poucos pixels e assim uma letra tem menos chances de ser parecida com outra.
- Criar uma solução (um filtro) para a detecção de padrões rotacionados.

Bibliografia

[1] Theodoridis, S., Koutroumbas, K., Pattern Recognition, 2ª ed, Elsevier Academic Press, San Diego, 2003.

[2] Sá, J.P.M., Pattern Recognition: Concepts Methods and Applications, Springer-Verlag, 2001.

[3] Ronsenfeld, A., "Image Pattern Recognition", Proceedings of IEEE, vol.69 no.5, 1981.

[4] Miyazawa, K. ito, K., Aoki,T., Kobayashi, K., Nakajima, H.,"An Efficient Iris Recognition Algorithm Using Phase-Based Image Matching", IEEE International Conference on Image Processing, vo.2, pag.49-52, Genova, 2005.

[5] Zhang, L., Samaras, D.,"Face Recognition from a Single Training Image under Arbitrary Unknown Lighting Using Spherical Harmonics", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.28, no.3, 2006.

[6] Zhang, L., Lu, Y., Tan, C.L., "Italic Font Recognition Using Stroke Pattern Analysis", Proceedings of 17th International Conference on Pattern Recognition, 2004.

[7] Mendonça, A.P. "Algoritmos para a Discriminação de Padrões em Imagem", tese de Doutorado, COPPE/UFRJ, Rio de Janeiro-Brasil, 2003.

[8] Arie, J.B., Rao, R. "A Novel Approach for Template Matching by Nonorthogonal Image Expansion", IEEE Transations on Circuits and Systems for Video Technology, v.3, n.1, 1993.

[9] Nandy, D., Arie, J.B. "EXM Eigen Templates for Detecting and Classifying Arbitrary Junctions". In: Proceedings of the IEEE International Conference on Imaging Processing, Chicago-EUA, 1998.

[10] Rao, K.R., Arie, J.B. "Multiple Template Matching Using the Expansion Filter", IEEE Transactions on Circuits and Systems for Video Technology, v.4, n.5, Out. 1994.

[11] Abu-Naser, A. "Impulse Restoration-Based Template-Matching", Ph.D. dissertation, University of Illinois at Chicago, Chicago-EUA, 2000.

[12] Abu-Naser, A., Galatsanos, N.P., Wernick, M.N. "Impulse Restoration-Based Template-Matching Using the Expectation-Maximization Algorithm" In:

Proceedings of the IEEE International Conference on Image Processing, Kobe-Japão, 1999.

[13] Golub, G.H., Loan, C.F.V., Matrix Computations, 2ª ed. The Johns Hopkins University Press, Inglaterra, 1989.

[14] Papoulis, A., Probability Random Variables and Stochastic Processes, Mc Graw-Hill, New York, 3ª Edition, 1991.

[15] Helstrom, C.W., Probability and Stochastic Processes for Engineers, 2ª ed. Lodon, U.K., Macmillan, 1991.

[16] Gonzalez, R.C., Woods, R.E., Digital Image Processing, 2ª ed. Prentice Hall, Nova Jersey, 2002.

[17] Russ, J.C., The Image Handbook, 3ª ed., CRC Press, Florida, 1999.

[18] Jain, A.K., Fundamentals of Digital Image Processing, Prentice Hall, New Jersey, 1989.

[19] Facon, J., Morfologia Matemática: Teoria e Exemplos, Editora Universitária Champagnat da PUC-Paraná, Curitiba,1996.

[20] Soille, P. Morphological Image Analysis., 2ª ed. Springer-Verlag, Heidelberg, 2003.

Apêndice A

Imagens-Bases e Resultados Auxiliares

Este anexo tem como objetivo apresentar os conjuntos de fontes para cada letra que originou um filtro “OU”, uma tabela com os nomes de cada fonte e os resultados obtidos com o filtro “OU” gerado, pelo método da seção 3.4, por uma letra, e testado em outras letras do alfabeto. Estes dados serão mostrados novamente para a letra A, que já foi apresentada ao longo do Capítulo 3.

A.1 Imagens e nomes das fontes

As letras usadas nos testes desta tese não possuem um número fixo de fontes, pois dependendo da letra várias fontes têm a mesma forma. Todos os padrões são de tamanho 8 contidas numa janela 9x9, todos foram alinhados com a parte inferior da janela. Todas as fontes que serão mostradas são enumeradas crescentemente da esquerda para a direita de cima para baixo.

A.1.1 Letra A

Figura A.1. Fontes “A” usadas nos testes.

Tabela A.1. Nome das fontes “A” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	10	Bookman Old Style
2	Times New Roman	11	Lúcida Sans
3	Courier	12	Dauphin
4	Courier New	13	Lúcida Console
5	Ms Sans Serif	14	Kabel Bk Bt
6	Geórgia	15	Garamond
7	Book Antigua	16	Futura Lt BT
8	Arial Narrow	17	System
9	Ms Outlook		

A.1.2 Letra B

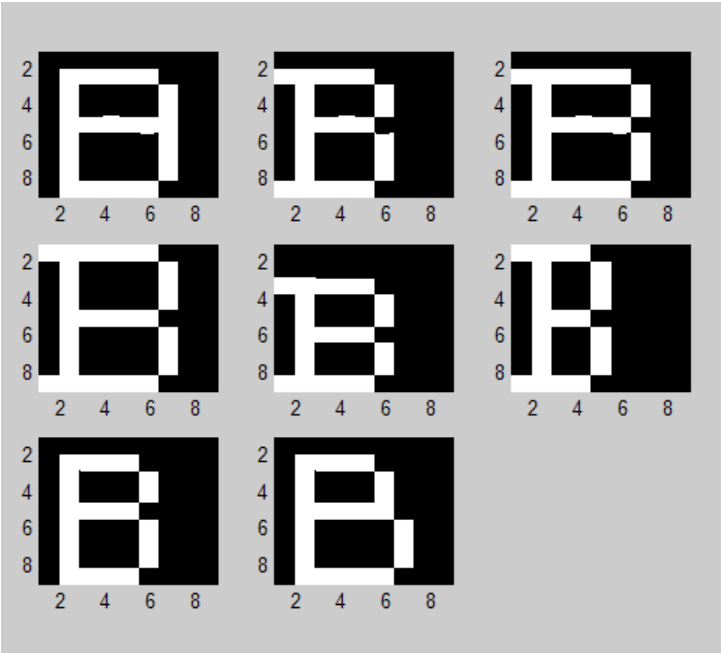


Figura A.2. Fontes “B” usadas nos testes.

Tabela A.2. Nome das fontes

“B” utilizadas nos testes finais.

Nº. da Fonte	Nome
1	Arial
2	Book Antigua
3	Bookman Old Style
4	Courier
5	Courier New
6	Ms Serif
7	Lúcida Console
8	Verdana

A.1.3 Letra C

Figura A.3. Fontes “C” usadas nos testes.

Tabela A.3. Nome das fontes “C” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	7	Tahoma
2	Book Antigua	8	Times New Roman
3	Bookman Old Style	9	Century gothic
4	Serifa Bt	10	BankGothic Md Bt
5	Courier	11	Courier New
6	Lúcida Console	12	Georgia

A.1.4 Letra D

Figura A.4. Fontes “D” usadas nos testes.

Tabela A.4. Nome das fontes

"D" utilizadas nos testes finais.

Nº. da Fonte	Nome
1	Arial
2	Book Antigua
3	Courier
4	Geórgia
5	Ms Serif
6	Times New Roman

A.1.5 Letra E

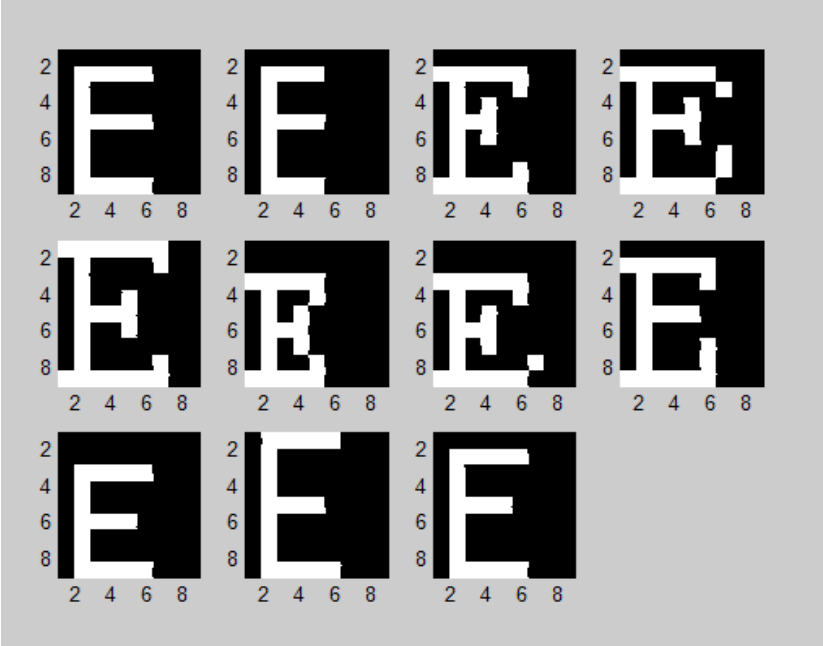


Figura A.5. Fontes "E" usadas nos testes finais.

Tabela A.5. Nome das fontes "E" utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	7	Garamond
2	Futura Lt BT	8	Geórgia
3	Times New Roman	9	Lúcida Console
4	Bookman Old Style	10	Ms Sans Serif
5	Courier	11	Tahoma
6	Courier New		

A.1.6 Letra F

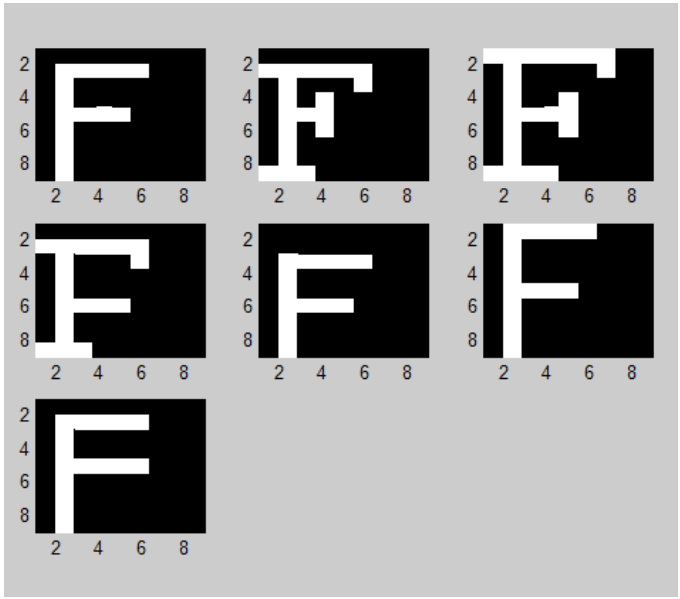


Figura A.6. Fontes “F” usadas nos testes finais.

Tabela A.6. Nome das fontes “F” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	5	Lúcida Sans
2	Times New Roman	6	Tahoma
3	Century Gothic	7	Verdana
4	Geórgia		

A.1.7 Letra G

Figura A.7. Fontes “G” usadas nos testes finais.

Tabela A.7. Nome das fontes “G” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da Fonte	Nome
1	Arial	5	Lúcida Sans
2	Book Antigua	6	Tahoma
3	Courier	7	Times New Roman
4	Geórgia	8	Verdana

A.1.8 Letra H

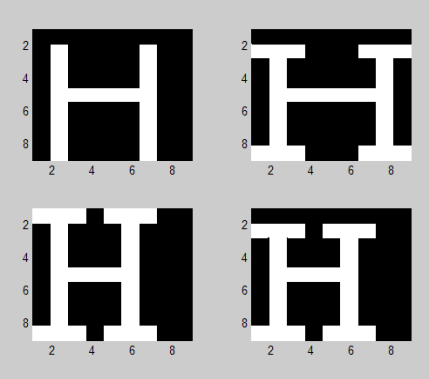


Figura A.8. Fontes “H” usadas nos testes finais.

Tabela A.8. Nome das fontes “H” utilizadas nos testes finais.

Nº. da Fonte	Nome
1	Arial
2	Book Antigua
3	Courier
4	Georgia

A.1.9 Letra J

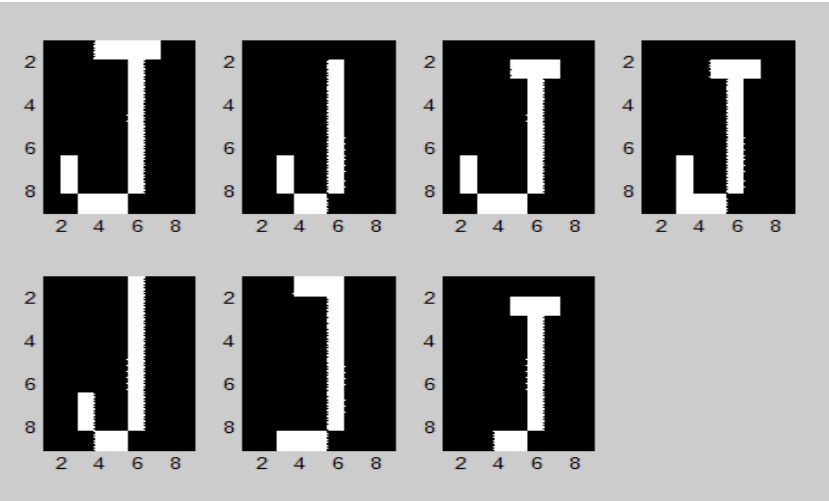


Figura A.9. Fontes “J” usadas nos testes finais.

Tabela A.9. Nome das fontes “J” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	5	Lúcida Sans
2	Times New Roman	6	Tahoma
3	Century Gothic	7	Verdana
4	Geórgia		

A.1.10 Letra K

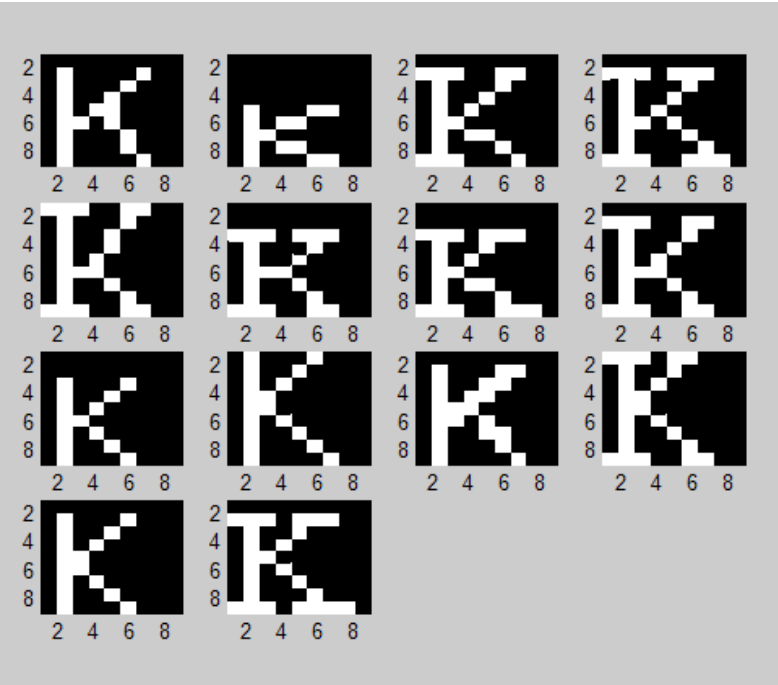


Figura A.10. Fontes “K” usadas nos testes finais.

Tabela A.10. Nome das fontes “K” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	8	Georgia
2	BankGothic Md BT	9	Lúcida Console
3	Book Antigua	10	Gautami
4	Bookman Old Style	11	Futura Md Bt
5	Courier	12	MS Serif
6	Courier New	13	Century Gothic
7	Garamond	14	Times New Roman

A.1.11 Letra L

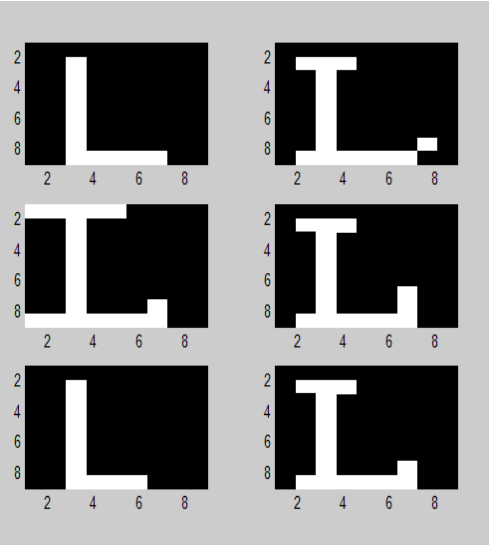


Figura A.11. Fontes “L” usadas nos testes finais.

Tabela A.11. Nome das fontes “L” utilizadas nos testes finais.

Nº. da Fonte	Nome
1	Arial
2	Book Old Style
3	Courier
4	Georgia
5	Tahoma
6	Times New Roman

A.1.12 Letra O

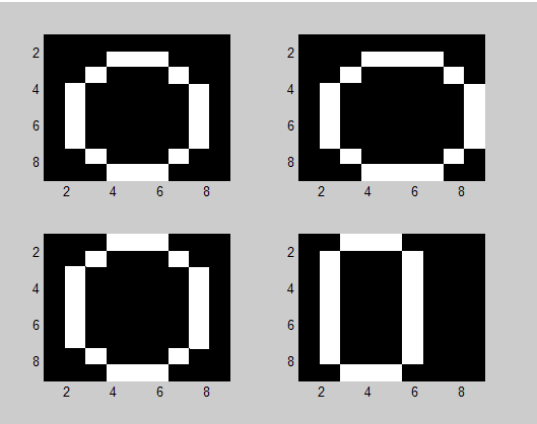


Figura A.12. Fontes “O” usadas nos testes finais.

Tabela A.12. Nome das fontes “O” utilizadas nos testes finais.

Nº. da Fonte	Nome
1	Arial/Times
2	Century Gothic
3	Courier
4	MS Serif

A.1.13 Letra R

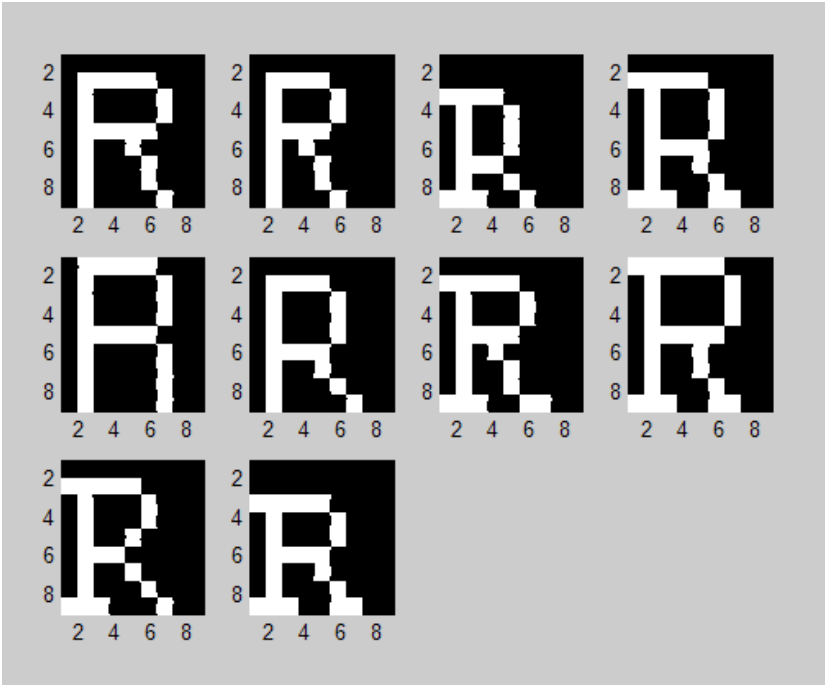


Figura A.13. Fontes “R” usadas nos testes finais.

Tabela A.13. Nome das fontes “R” utilizadas nos testes finais.

Nº. da Fonte	Nome	Nº. da fonte	Nome
1	Arial	6	Tahoma
2	Century Gothic	7	Times New Roman
3	Courier New	8	Courier
4	Geórgia	9	Book Antigua
5	MS Sans Serif	10	Serifa BT

A.1.14 Letra U

Figura A.14. Fontes “U” usadas nos testes finais.

Tabela A.14. Nome das fontes “H” utilizadas nos testes finais.

Nº. da Fonte	Nome
1	Arial
2	Book Antigua
3	Courier
4	Futura
5	Courier New
6	Lúcida Console

A.2 Resultados com outras letras

Abaixo serão mostradas tabelas com os resultados para todas as letras apresentadas acima, em relação à outras letras do alfabeto, apresentando suas $DSNR_2$’s. Todos os filtros OU foram construídos pelo método da seção 3.4.

Tabela A.15: Resultados do Filtro OU_A testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,6503	0,0075	0,0016	0,0111	0,0026	0,0017	0,0003	0,0138	0,0043	0,0049	0,0002	0,0018	0,0014	0,0016
2	0,4323	0,0149	0,0001	0,0001	0,0055	0,0400	0,0009	0,0139	0,0313	0,0031	0,0106	0,0047	0,0250	0,0003
3	0,6687	0,0013	0,0018	0,0056	0,0151	0,0098	0,0003	0,0008	0,0209	0,0152	0,0001	0,0040	0,0014	0
4	0,5896	0,0002	0	0,0033	0,0037	0,0052	0,0007	0,0092	0,0477	0,0349	0,0114	0,0021	0,0138	0,0050
5	0,4268	0,0079	0,0050	0,0223	0,0077	0,0268	0,0005		0,0264	0,0012	0,0002		0,0003	0,0082
6	0,4735	0,0012	0,0096	0,0037	0,0117	0,0017	0,0003		0,0085	0,0110	0,0149		0,0210	0,0166
7	0,5510	0,0298	0,0022		0,0755	0,0021	0,0002		0,0142	0,0377			0,0251	
8	0,6244	0,0284	0,0022		0,0023	0,0400	0,0005			0,0107			0,0016	
9	0,4400		0,0011		0,0266					0,0192			0,0129	
10	0,5339		0,0279		0,0010					0,0128			0,0130	
11	0,8371		0,0059		0,0022					0,0355				
12	0,5163		0,0014							0,0046				
13	0,6342									0,0152				
14	0,4241									0,0204				
15	0,4204													
16	0,6339													
17	0,4217													

Tabela A.16: Resultados do Filtro OU_B testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0108	0,6149	0,0778	0,3707	0,5695	0,2739	0,0796	0,2222	0,0814	0,0339	0,0002	0,0622	0,2410	0,2639
2	0,0019	0,8205	0,0173	0,3040	0,7824	0,2218	0,0267	0,1124	0,0107	0,0003	0,0011	0,0147	0,1699	0,1930
3	0,0067	0,9335	0,0268	0,2561	0,3545	0,2040	0,0356	0,1856	0,1330	0,0268	0,0131	0,0430	0,0628	0,1635
4	0	0,5330	0,0338	0,4758	0,5499	0,1398	0,0240	0,3308	0,0255	0,0706	0,0070	0,2037	0,1382	0,2627
5	0,0012	0,3800	0,0242	0,3373	0,2163	0,0321	0,1271		0,0077	0,0206	0,0059		0,1434	0,2525
6	0,0011	0,3485	0,0061	0,2237	0,1510	0,1096	0,0603		0,0517	0,0644	0,0040		0,0756	0,0376
7	0,0076	0,6342	0,0253		0,1990	0,3240	0,0563		0,0537	0,0029			0,2043	
8	0,0001	0,8351	0,0550		0,6201		0,0311			0,0307			0,1415	
9	0,0251		0,0223		0,1302		0,0796			0,0222			0,1468	
10	0,0019		0,0999		0,2432		0,0267			0,0139			0,0838	
11	0,0013		0,2205		0,4884		0,0356			0,0185				
12	0,0006		0,0164							0,0251				
13	0									0,0104				
14	0,0028									0,0704				
15	0,0018													
16	0,0066													
17	0,0005													

Tabela A.17: Resultados do Filtro OU_C testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0266	0,1240	1,8359	0,1150	0,1600	0,0641	0,8309	0,0268	0,0246	0,0310	0,0843	0,7756	0,0436	0,1705
2	0,0043	0,1124	1,5590	0,1686	0,1455	0,0433	0,9131	0,0148	0,0343	0,0295	0,0602	0,8284	0,0457	0,1717
3	0,0118	0,1159	1,7645	0,0727	0,1376	0,0301	0,3100	0,0381	0,0570	0,0264	0,0464	0,2389	0,0448	0,0767
4	0,0039	0,0568	1,5780	0,1150	0,1470	0,0534	0,9217	0,0720	0,0347	0,0723	0,0413	0,0851	0,0581	0,1250
5	0,0216	0,1339	0,4713	0,0517	0,1061	0,0818	1,0430		0,0283	0,0276	0,0604		0,0091	0,1695
6	0,0233	0,0424	0,6208	0,1702	0,1124	0,0141	0,9430		0,0112	0,0964	0,0606		0,0461	0,0359
7	0,0135	0,1089	2,1849		0,1920	0,0487	0,9437		0,0344	0,0388			0,0550	
8	0,0061	0,1079	1,7974		0,1510		1,0059			0,0684			0,0136	
9	0,0083		2,0474		0,2292					0,0440			0,0567	
10	0,0788		0,4679		0,0921					0,0370			0,0800	
11	0,0332		0,3999		0,1888					0,0363				
12	0,0007		1,5824							0,0291				
13	0,0038									0,0189				
14	0,0140									0,0314				
15	0,0098													
16	0,0029													
17	0,0091													

Tabela A.18: Resultados do Filtro OU_D testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0028	0,4541	0,1485	0,8210	0,3860	0,1806	0,1332	0,0438	0,0758	0,0366	0,0027	0,1612	0,1419	0,2879
2	0,0082	0,5064	0,0313	0,9252	0,4403	0,3792	0,0201	0,0500	0,0136	0	0,0047	0,0128	0,1009	0,2738
3	0,0015	0,5504	0,0658	0,7246	0,8582	0,1284	0,0121	0,1512	0,1291	0,1515	0,0218	0,0333	0,0411	0,1902
4	0,0034	0,1893	0,0840	1,1073	0,5955	0,3772	0,0156	0,2474	0,0406	0,1682	0,0159	0,2915	0,1759	0,3219
5	0,0134	0,1010	0,0172		0,1323	0,0278	0,1387		0,0039	0,1051	0,0111		0,0166	0,1662
6	0,0029	0,2742	0,0163		0,1095	0,0421	0,0308		0,0488	0,0836	0,0111		0,0806	0,0567
7	0,0103	0,3695	0,0439		0,1264	0,1469	0,0471		0,0461	0,0004			0,1703	
8	0,0015	0,3234	0,0927		0,7080		0,0163			0,0730			0,0903	
9	0,0025		0,0228		0,1254					0,1574			0,2276	
10	0,0701		0,0820		0,1533					0,0244			0,0204	
11	0,0096		0,1607		0,4554					0,0606				
12	0,0005		0,0219							0,0328				
13	0,0130									0,0188				
14	0,0054									0,1117				
15	0,0596													
16	0,0028													
17	0,0024													

Tabela A.19: Resultados do Filtro OU_E testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0013	0,3180	0,1768	0,1387	0,5769	0,2810	0,1057	0,0786	0,0730	0,0956	0,0147	0,1359	0,0451	0,3419
2	0,0006	0,3196	0,0891	0,3756	0,5418	0,3459	0,0758	0,0923	0,0049	0,0029	0,0018	0,0723	0,1955	0,2924
3	0,0024	0,4391	0,0813	0,0983	0,6530	0,3433	0,0797	0,2314	0,0660	0,1961	0,0139	0,1326	0,2570	0,1084
4	0	0,3264	0,1055	0,2000	0,6895	0,2528	0,0728	0,2842	0,0007	0,3495	0,0047	0,1083	0,0848	0,1383
5	0,0003	0,1280	0,1140	0,0663	0,6027	0,1157	0,1454		0,0018	0,1489	0,0339		0,0514	0,0971
6	0,0068	0,1996	0,0290	0,2758	0,5257	0,1724	0,1011		0,0082	0,1158	0,0063		0,0081	0,1382
7	0,0009	0,2431	0,1317		0,7789	0,1799	0,0887		0,0025	0,2321			0,2764	
8	0,0052	0,3688	0,1795		0,7888	0,3459	0,0771			0,2409			0,0846	
9	0,0066		0,1167		0,5554					0,4099			0,0819	
10	0,0161		0,2054		0,5238					0,0076			0,0588	
11	0,0146		0,2279		0,7956					0,0603				
12	0		0,0987							0,0761				
13	0,0378									0,1384				
14	0,0003									0,1467				
15	0,0079													
16	0,0046													
17	0,0017													

Tabela A.20: Resultados do Filtro OU_F testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0128	0,1954	0,0014	0,1629	0,3898	0,7085	0,0009	0,1820	0,0161	0,2157	0,0316	0,0028	0,2947	0,0402
2	0,0492	0,3604	0,0014	0,1709	0,4457	0,9508	0,0066	0,2725	0,0234	0,0134	0,0052	0,0050	0,3250	0,0682
3	0,0018	0,3074	0,0002	0,0704	0,5163	0,3790	0,0169	0,1538	0,0173	0,2964	0,0022	0,0171	0,1355	0,0305
4	0,0017	0,1242	0,0002	0,2621	0,4181	1,0172	0,0046	0,3687	0,0067	0,5262	0,0087	0,0512	0,2467	0,0957
5	0,0109	0,0734	0,0183	0,1028	0,2448	0,3341	0,0019		0,0183	0,0982	0,0291		0,1199	0,0198
6	0,0160	0,1746	0,0082	0,0925	0,0862	0,3077	0,0050		0,0002	0,1136	0,0105		0,1481	0,0008
7	0,0134	0,2342	0,0048		0,1455	0,6332	0,0009		0,0012	0,0671			0,4740	
8	0,0021	0,2470	0,0020		0,4821		0,0049			0,1208			0,1508	
9	0,0016		0,0046		0,2269					0,3069			0,3313	
10	0,0868		0,0252		0,1506					0,0444			0,0581	
11	0,0012		0,0430		0,4288					0,2091				
12	0,0048		0,0016							0,0120				
13	0,0000									0,1651				
14	0,0061									0,0488				
15	0,0053													
16	0,0478													
17	0,0003													

Tabela A.21: Resultados do Filtro OU_G testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0118	0,0662	1,2501	0,0585	0,0681	0,0272	1,6522	0,0099	0,0197	0,0040	0,0311	0,9328	0,0161	0,1046
2	0,0292	0,0298	1,0877	0,1494	0,0399	0,0150	1,7274	0,0046	0,0490	0,0127	0,0152	0,7857	0,0151	0,1118
3	0,0107	0,0694	1,0538	0,0331	0,0564	0,0011	0,6142	0,0097	0,0511	0,0085	0,0089	0,3007	0,0125	0,0321
4	0,0118	0,0221	1,1735	0,0596	0,0566	0,0377	1,8232	0,0349	0,0581	0,0247	0,0107	0,0383	0,0363	0,0744
5	0,0146	0,0301	0,3700	0,0188	0,0196	0,0356	1,2825		0,0480	0,0048	0,0210		0,0040	0,0634
6	0,0150	0,0032	0,1876	0,1542	0,0281	0	1,8689		0,0105	0,0265	0,0138		0,0315	0,0249
7	0,0254	0,0277	1,7803		0,0415	0,0229	2,2008		0,0481	0,0059			0,0155	
8	0,0172	0,0444	1,3645		0,0512		2,6910			0,0187			0,0023	
9	0,0158		1,4876		0,0876					0,0171			0,0184	
10	0,0975		0,0462		0,0132					0,0068			0,0250	
11	0,0244		0,0817		0,0747					0,0205				
12	0,0021		1,2716							0,0041				
13	0,0003									0,0017				
14	0,0174									0,0040				
15	0,0005													
16	0,0001													
17	0,0051													

Tabela A.22: Resultados do Filtro OU_H testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0013	0,1010	0,0141	0,0120	0,2452	0,3323	0,0051	0,7518	0,0178	0,2541	0,0003	0,0080	0,3407	0,0061
2	0,0000	0,3926	0,0013	0,0294	0,2930	0,3057	0,0031	0,9237	0	0,0041	0,0004	0,0004	0,1663	0,0075
3	0,0063	0,2067	0,0001	0,0341	0,2577	0,2806	0,0136	0,7934	0,0255	0,2357	0,0075	0,0141	0,0269	0,0277
4	0,0002	0,2070	0,0000	0,0454	0,1756	0,0587	0,0012	1,3387	0,0009	0,1240	0,0007	0,0711	0,0815	0,0933
5	0,0004	0,0204	0,0116	0,0528	0,3201	0,0035	0,0159		0,0002	0,0602	0,0105		0,3025	0,0199
6	0,0001	0,1172	0,0021	0,0050	0,0261	0,3212	0		0,0098	0,0683	0,0012		0,0420	0,0191
7	0,0019	0,2405	0,0007		0,0167	0,3912	0,0001		0,0039	0,0858			0,3452	
8	0	0,1477	0,0071		0,3531		0,0067			0,0459			0,4516	
9	0,0139		0,0040		0					0,2427			0,1528	
10	0,0269		0,0049		0,1700					0,0400			0,0346	
11	0,0005		0,0036		0,2130					0,0339			0,3407	
12	0,0026		0,0031							0,0356				
13	0,0046									0,1816				
14	0,0025									0,0988				
15	0,0017													
16	0,0064													
17	0,0074													

Tabela A.23: Resultados do Filtro OU_J testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0106	0,0053	0,0132	0,1483	0,0322	0,0021	0,0115	0,0018	0,7193	0,0250	0,0564	0,0066	0,0097	0,0264
2	0,0162	0,2943	0,0355	0,0123	0,0883	0,0016	0,0204	0,0034	1,5534	0,0378	0,0163	0,0229	0,0326	0,0260
3	0,0074	0,0068	0,0135	0,0793	0,0681	0,0041	0,0096	0,0679	1,5529	0,0247	0,0060	0,0087	0,0319	0,4098
4	0,0018	0,0028	0,0101	0,1451	0,0018	0,0024	0,0250	0,1422	2,0298	0,0000	0,0093	0,4514	0,0036	0,7832
5	0,0028	0,2482	0,0077	0,1016	0,0026	0,0023	0,0070		1,4215	0,0095	0,0637		0,0142	0,4946
6	0,0008	0,0005	0,0203	0,0125	0,0264	0,0263	0,0260		0,9947	0	0,0166		0,0239	0,0128
7	0,0242	0,3130	0,0417		0,0254	0	0,0221		1,9227	0,0054			0,0144	
8	0,0104	0,0334	0,0264		0,1071		0,0336			0,0016			0,0054	
9	0,0454		0,0320		0,0208					0,0008			0,0198	
10	0,0021		0,0172		0,0016					0,0024			0	
11	0,0023		0,3233		0,0193					0,0433				
12	0,0113		0,0317							0,0018				
13	0,0112									0,0161				
14	0,0019									0,0098				
15	0,0023													
16	0,0033													
17	0,0044													

Tabela A.24: Resultados do Filtro OU_K testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0101	0,0066	0,0003	0,0326	0,0321	0,0876	0,0137	0,0693	0,0046	0,4263	0,0041	0,0004	0,0910	0,0178
2	0,0019	0,0535	0,0101	0,0576	0,0312	0,2314	0,0008	0,1275	0,0597	0,0314	0,0021	0,0041	0,1388	0,0249
3	0,0005	0,0273	0,0009	0,0533	0,1952	0,0751	0,0010	0,0936	0,0000	1,5929	0,0002	0,0054	0,1127	0,0061
4	0,0080	0,0211	0,0002	0,0673	0,0674	0,2359	0,0006	0,1917	0,0204	0,5916	0,0013	0,0210	0,3510	0,0396
5	0,0374	0,1078	0,0062	0,0327	0,0802	0,1801	0,0017		0,0537	0,7642	0,0193		0,0283	0,0053
6	0,0073	0,1692	0,0138	0,0276	0,2069	0,0476	0,0002		0,0088	0,5514	0,0022		0,2107	0,0186
7	0,0012	0,0235	0,0113		0,1406	0,0639	0,0006		0,0093	0,1294			0,3166	
8	0	0,0082	0,0004		0,1509		0,0009			0,7374			0,2167	
9	0,0120		0,0068		0,0986					1,5935			0,4692	
10	0,0187		0,0460		0,0216					0,1482			0,4030	
11	0,0066		0,0434		0,0452					0,3761				
12	0,0004		0,0053							0,3102				
13	0,0036									0,7439				
14	0,0322									0,5087				
15	0,0033													
16	0,0063													
17	0,0063													

Tabela A.25: Resultados do Filtro OU_L testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0246	0,0091	0,0977	0,0090	0,0173	0,0053	0,0659	0,0006	0,0003	0,0035	1,6163	0,0815	0,0097	0,0133
2	0,0026	0,0034	0,0760	0,0097	0,0018	0,0309	0,0718	0,0085	0,0559	0,0001	1,4035	0,1021	0,0002	0,0079
3	0,0166	0,0032	0,0874	0,0013	0,0006	0,0001	0,0622	0,0003	0,0027	0,0117	1,0286	0,1109	0,0022	0,0070
4	0,0202	0,0196	0,0940	0,0178	0,0080	0,0291	0,0616	0,0002	0,0150	0,0002	1,2071	0,0148	0,0146	0,0061
5	0,0303	0,0001	0,0909	0,0174	0,0416	0,0053	0,0625		0,0494	0,0063	1,3366		0,0044	0,0024
6	0,0080	0,0062	0,1082	0,0214	0,0014	0,0163	0,0945		0,0081	0,0035	1,5718		0,0167	0,0159
7	0,0011	0,0004	0,1303		0,0180	0,0083	0,0945		0,0042	0,0286			0,0045	
8	0,0118	0,0072	0,1124		0,0013		0,0852			0,0087			0,0011	
9	0,0069		0,1269		0,0203					0,0001			0,0381	
10	0,0233		0,0361		0,0130					0,0152			0,0037	
11	0,0132		0,0006		0,0220					0,0010				
12	0,0015		0,1049							0,0063				
13	0,0030									0,0015				
14	0,0331									0,0038				
15	0,0019													
16	0,0097													
17	0,0414													

Tabela A.26: Resultados do Filtro OU_O testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0454	0,1205	0,7708	0,1426	0,2213	0,0423	0,5835	0,0146	0,1616	0,0562	0,1060	0,6133	0,0257	0,2345
2	0,0235	0,1639	0,8725	0,2653	0,1639	0,0540	0,6702	0,0338	0,1888	0,0004	0,0723	0,5717	0,0135	0,1859
3	0,0129	0,1242	0,6316	0,1862	0,2529	0,0608	0,2242	0,1527	0,2605	0,0762	0,0981	0,5928	0,0032	0,2643
4	0,0310	0,1153	0,6906	0,1359	0,1278	0,0539	0,5433	0,2075	0,2125	0,1065	0,0631	0,5611	0,0528	0,3884
5	0,0589	0,1578	0,2726	0,0951	0,1798	0,0176	0,4904		0,1277	0,0408	0,1049		0,0094	0,2161
6	0,0164	0,0355	0,1832	0,2859	0,0815	0,0157	0,5041		0,1803	0,0409	0,0935		0,0319	0,3455
7	0,0103	0,1766	0,9319		0,1777	0,0425	0,5355		0,2281	0,0310			0,0219	
8	0,0116	0,1426	0,8211		0,2412	0,0540	0,6837			0,1098			0,0364	
9	0,0605		0,9535		0,1905					0,0157			0,0372	
10	0,0007		0,1917		0,1495					0,0022			0,0395	
11	0,0081		0,2443		0,2297					0,0450				
12	0,0249		0,7282							0,0248				
13	0,0110									0,0114				
14	0,0116									0,1238				
15	0,0175													
16	0,0290													
17	0,0149													

Tabela A.27: Resultados do Filtro OU_R testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0380	0,1107	0,0006	0,0161	0,2071	0,2457	0,0075	0,4999	0,0132	0,2851	0,0011	0,0045	0,5628	0,0147
2	0,0366	0,1126	0,0015	0,0105	0,1737	0,2284	0,0005	0,2445	0,0768	0,1216	0	0,0022	0,4319	0,0063
3	0,0079	0,2012	0,0042	0,0014	0,1114	0,1710	0,0003	0,1514	0,0009	0,2070	0,0036	0,0010	0,4355	0,0010
4	0,0049	0,1874	0,0011	0,0294	0,1696	0,2189	0,0010	0,3174	0,0381	0,1749	0,0005	0,0043	0,5808	0,0147
5	0,0074	0,1150	0,0047	0,0487	0,1568	0,1563	0		0,0670	0,0994	0,0245		0,4322	0,0240
6	0,0051	0,1196	0,0001	0,0040	0,0980	0,2010	0,0067		0,0273	0,4275	0,0012		0,4371	0,0253
7	0,0413	0,0845	0,0060		0,0849	0,3509	0,0013		0,0318	0,2640			0,6905	
8	0,0304	0,1976	0,0001		0,1346	0,2284	0,0004			0,2510			0,6689	
9	0,0158		0,0052		0,0739					0,1189			0,4764	
10	0,0462		0,0286		0,0815					0,1302			0,5467	
11	0,0081		0,0157		0,1300					0,1297				
12	0,0043		0,0041							0,2102				
13	0,0001									0,0569				
14	0									0,1264				
15	0,0121													
16	0,0012													
17	0,0000													

Tabela A.28: Resultados do Filtro OU_U testados com outras letras. Cada coluna é nomeada pela letra que está sendo testada

Fontes	A	B	C	D	E	F	G	H	J	K	L	O	R	U
1	0,0101	0,1927	0,0626	0,3072	0,2198	0,0095	0,0513	0,0063	0,4090	0,0078	0,0695	0,0597	0,0086	0,7927
2	0,0140	0,2153	0,0201	0,2071	0,2239	0,0378	0,0190	0,0121	0,1078	0,0060	0,0161	0,0340	0,0041	0,7729
3	0,0049	0,1648	0,0464	0,2210	0,2148	0,0689	0,0440	0,0957	0,5211	0,0396	0,0003	0,0634	0,0172	0,8704
4	0,0047	0,1258	0,0397	0,2713	0,1516	0,0450	0,0240	0,1405	0,1958	0,0098	0,0130	0,7008	0,0349	1,3026
5	0,0047	0,1807	0,0500	0,1216	0,0956	0,0079	0,0601		0,0877	0,0111	0,1275		0,0001	1,1519
6	0,0125	0,0658	0,0291	0,2164	0,1169	0,0002	0,0351		0,2581	0,0300	0,0119		0,0085	0,2670
7	0,0051	0,2481	0,0419		0,1420	0,0122	0,0446		0,1669	0,0068			0,0130	
8	0,0071	0,1898	0,0674		0,2423		0,0368			0,0091			0,0176	
9	0		0,0408		0,2095					0,0316			0,0309	
10	0,0134		0,1540		0,1430					0,0113			0,0193	
11	0,0135		0,5422		0,2046					0,0056				
12	0,0401		0,0477							0,0003				
13	0,0678									0,0060				
14	0,0025									0,0043				
15	0,0663													
16	0,0054													
17	0,0027													