**UNIVERSIDADE PRESBITERIANA MACKENZIE**

**MAURO ULISSES SCHNEIDER**

# EMPREGO DE COMITÊ DE MÁQUINAS PARA SEGMENTAÇÃO DA ÍRIS

São Paulo
2010

**MAURO ULISSES SCHNEIDER**

# EMPREGO DE COMITÊ DE MÁQUINAS PARA SEGMENTAÇÃO DA ÍRIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Presbiteriana Mackenzie, como parte das exigências para obtenção do Titulo de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. CLODOALDO APARECIDO DE MORAES LIMA

São Paulo
2010

S359e Schneider, Mauro Ulisses.
Emprego de comitê de máquinas para segmentação da íris / Mauro Ulisses Schneider – 2010.
85 f. : il. ; 30 cm

Dissertação (Mestrado em Engenharia Elétrica) – Universidade Presbiteriana Mackenzie, São Paulo, 2010.
Bibliografia: f. 56-58.

1. Sistemas biométricos. 2. Comitê de máquinas. 3. Segmentação de íris. 4. Agrupamento por máquinas de vetores suporte. 5. K-means. 6. Mapa auto organizável de Kohonen. I. Título.

CDD 005.8

**MAURO ULISSES SCHNEIDER**

# EMPREGO DE COMITÊ DE MÁQUINAS PARA SEGMENTAÇÃO DA ÍRIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Presbiteriana Mackenzie, como parte das exigências para obtenção do Titulo de Mestre em Engenharia Elétrica.

Aprovado em 23/08/2010

BANCA EXAMINADORA

_____

Prof. Dr. Clodoaldo Aparecido de Moraes Lima
Universidade Presbiteriana Mackenzie

_____

Prof. Dr. Nizam Omar
Universidade Presbiteriana Mackenzie

_____

Prof. Dr. André Luís Vasconcelos Coelho
Universidade de Fortaleza

# AGRADECIMENTOS

Primeiramente, agradeço a Deus que me fortaleceu e não me deixou desanimar com as dificuldades que apareceram durante esta pesquisa.

Agradeço a toda a minha família pelo incentivo dado, principalmente a minha esposa Maria, e meus filhos Erick e Letícia, que souberam compreender os momentos de isolamento na produção desta pesquisa.

Ao meu orientador, prof. Dr. Clodoaldo Aparecido de Moraes Lima, que é mais do que um orientador, um amigo, sempre disposto a ajudar, com diretrizes seguras, contribuindo com sua grande sabedoria nesta dissertação.

A todos os professores do programa de pós-graduação em Engenharia Elétrica do Mackenzie, em especial, a professora Dra Pollyana Notargiacomo Mustaro pela grande ajuda em metodologia científica.

Ao meu grande amigo Luiz Angelo D'Amore, pelo grande incentivo e na ajuda de visão computacional, que contribuiu largamente com o início desta dissertação.

Agradeço também aos meus amigos do curso de Engenharia Elétrica que direta ou indiretamente contribuíram com esta dissertação.

Ao departamento de Ciência da Computação da Universidade de Beira Interior em Portugal, pela disponibilização da base de dados de imagens da íris (UBIRIS).

Agradeço ao MackPesquisa pelo suporte ao desenvolvimento desta pesquisa.

# Resumo

A utilização de sistemas biométricos vem sendo amplamente incentivados pelo governo e entidades privadas a fim de substituir ou melhorar os sistemas de segurança tradicionais. Os sistemas biométricos são cada vez mais indispensáveis para proteger vidas e bens, sendo robustos, confiáveis, de difícil falsificação e rápida autenticação. Em aplicações de mundo real, os dispositivos de aquisição de imagem e o ambiente nem sempre são controlados, podendo em certas circunstâncias produzir imagens ruidosas ou com grandes variações na tonalidade, textura, geometria, dificultando a sua segmentação e por conseqüência a autenticação do indivíduo. Para lidar eficazmente com tais problemas, nesta dissertação é estudado o emprego de comitês de máquinas em conjunto com técnicas de processamento de imagens digitais para a segmentação da íris. Os componentes estudados na composição do comitê de máquinas são agrupamento por vetores-suporte, *k-means* e mapas auto-organizáveis. Para a avaliação do desempenho das ferramentas desenvolvidas neste trabalho, os resultados obtidos são comparados com trabalhos relacionados na literatura. Foi utilizada a base de dados pública UBIRIS disponível na internet.



**Palavras-chave:** *Sistemas Biométricos, Comitê de Máquinas, segmentação da íris, agrupamento por máquinas de vetores suporte, k-means, mapas auto organizável de Kohonen.*

# Abastract

The use of biometric systems has been widely stimulated by both the government and private entities to replace or improve traditional security systems. Biometric systems are becoming increasingly indispensable to protecting life and property, mainly due to its robustness, reliability, difficult to counterfeit and fast authentication. In real world applications, the devices for image acquisition and the environment are not always controlled and may under certain circumstances produce noisy images or with large variations in tonality, texture, geometry, hindering segmentation and consequently the authentication of the an individual. To deal effectively with such problems, this dissertation investigates the possibility of using committee machines combined with digital image processing techniques for iris segmentation. The components employed in the composition of the committee machines are support vector clustering, k-*means* and self organizing maps. In order to evaluate the performance of the tools developed in this dissertation, the experimental results obtained are compared with related works reported in the literature. Experiments on publicity available UBIRIS database indicate that committee machine can be successfully applied to the iris segmentation.



**Keywords:** *biometric systems, committee machine, iris segmentation, support vector clustering, k-means, self organizing maps.*.

# LISTA DE FIGURAS

LISTA DE TABELAS

# SUMÁRIO

# 1 INTRODUÇÃO

Na sociedade atual, a identificação precisa e rápida dos indivíduos é uma necessidade cada vez maior. Como os métodos tradicionais de identificação baseados em cartões e senhas podem ser facilmente fraudados, nos últimos anos têm sido propostos métodos mais efetivos e seguros de identificação de pessoas baseados em Biometria. A Biometria é a ciência que estabelece uma identidade para um indivíduo baseada em seus atributos pessoais, físicos, químicos ou comportamentais (JAIN; FLYNN; ROSS, 2008).

Métodos de identificação de pessoas sempre foram muito importantes para toda a sociedade. No mundo atual, as pessoas sempre precisam carregar algum documento para identificá-las, como se a individualidade somente existisse para pessoas que portam tais documentos (PROENÇA e ALEXANDRE, 2006). Partindo do fato de que não existem pessoas completamente idênticas, a necessidade da utilização de tais documentos se extingue quando se dispõe de métodos capazes de diferenciar cada indivíduo, sem confundi-los com seus semelhantes. Esse é o principal objetivo das pesquisas em biometria. Um sistema biométrico é essencialmente um sistema de reconhecimento de padrões que realiza a identificação de um indivíduo através da determinação da autenticidade de uma(s) característica(s) fisiológica(s) e/ou comportamental (is) (FAUNDEZ-ZANUY, ELIZONDO, *et al.*, 2007).

Um fator importante no projeto de um sistema real é determinar como um indivíduo é identificado. Dependendo do contexto, um sistema biométrico pode ser tanto um sistema de verificação (autenticação) ou um sistema de identificação (reconhecimento). Na verificação o sistema é projetado para responder a seguinte questão: Esta pessoa é quem ela diz ser? Já na identificação a questão é: Quem é esta pessoa? Ambos são largamente empregados para controle de acesso, vigilância, segurança computacional, etc. Tanto na verificação quanto na identificação há quatro fases distintas: captura, extração, comparação e resposta (Figura 1.1).

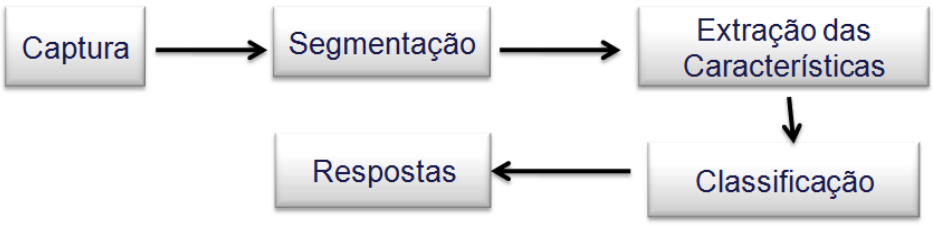



Figura 1.1 – Fases de um sistema biométrico.

A íris humana tem sido utilizada em sistemas de reconhecimento automático, bem como outros dados biométricos, para efeitos de verificação e identificação. Uma vez que suas características são únicas para cada indivíduo e estável com a idade, isto é, não varia com a idade. A íris tem um grande potencial de utilização na avaliação biométrica não invasiva (MIRA e MAYER, 2003).

As necessidades emergentes de um acesso seguro e rápido requerem técnicas não cooperativas. Como exemplo, podemos pensar ao acesso em um edifício onde os usuários não precisam olhar através de um pequeno buraco para ter sua íris reconhecida, mas em vez disso, um sistema de captura de imagem que recupere os dados necessários da íris ao se aproximar de uma porta. Isto é muito menos invasivo e permiti a difusão de sistemas de reconhecimento da íris para aplicações cotidianas (PROENÇA e ALEXANDRE, 2006).

Proença e Alexandre (2006) citam algumas vantagens em se utilizar um sistema de reconhecimento da íris não cooperado (PROENÇA e ALEXANDRE, 2006).

**Segurança**: Como a cooperação não é necessária, os usuários não precisam ter conhecimento sobre o reconhecimento através da íris. Obviamente, é muito mais difícil enganar o sistema, quando o sujeito não sabe quando e onde o sistema está fazendo a tarefa de reconhecimento.

**Comodidade para o usuário**: A cooperação dos usuários sobre o processo de captura de imagem dura alguns segundos e muitas vezes, é necessário repetir o processo. O fato de que os usuários não terão que fazer essa tarefa irá aumentar a sua comodidade.

**Numero total de pessoas reconhecidas**: Sistemas não cooperado terão um número maior de reconhecimento do que os sistemas cooperativos.

Em sistemas não cooperados, os dados biométricos são ruidosos, sua localização pode ser realizada inadequadamente e as imagens estão sujeitas a grandes variações. Na literatura não existe metodologia sistemática e eficaz capaz de propor modelos adequados para lidar no tratamento destes problemas. Portanto, os comitês de máquinas se apresentam como alternativas promissoras. Existem versões estáticas de comitês, na forma de ensembles de componentes, e versões dinâmicas, na forma de misturas de especialistas. Esta dissertação se dedica ao estudo dos componentes estáticos de um ensemble, tais componentes são formados por: agrupamento por vetores suporte, *k-means* e mapas auto-organizáveis.

## 1.1 OBJETIVO

O objetivo desta dissertação é investigar o emprego de comitê de máquinas em conjunto com técnicas de processamento de imagens digitais para a segmentação da íris em imagens ruidosas ou com grandes variações na tonalidade, textura e geometria. Para alcançar este objetivo, técnicas de geração e combinação de componentes são investigadas. Devido ao não conhecimento prévio da localização exata da íris na imagem e sua área, toda a abordagem está restrita ao caso de treinamento não supervisionado.

## 1.2 CONTRIBUIÇÕES DESTE TRABALHO

As principais contribuições deste trabalho podem ser descritas como:

- Estudo e implementação de técnicas tradicionais de processamento de imagem para segmentação da íris;
- Emprego de Redes Neurais Artificiais juntamente com a transformada de Hough visando diminuir o custo computacional na busca da circunferência da pupila;
- Estudo e implementação de comitês baseadas em Agrupamento por Vetores Suporte, k-*means* e mapas auto organizáveis;

## 1.3 ESTRUTURA DOS CAPÍTULOS

Para melhor entendimento da pesquisa realizada por esta dissertação, no capítulo 2, são descritas as técnicas tradicionais de processamento de imagem e segmentação. Já no capítulo 3, é apresentada uma introdução sobre sistemas biométricos e o estado da arte de trabalhos relacionados à segmentação da íris. No capítulo 4, é apresentada uma revisão sucinta de comitê de máquinas e os componentes empregados nesta dissertação. E enfim, o capítulo 5, apresenta os resultados obtidos com as técnicas desenvolvidas nesta dissertação.

# 2 PROCESSAMENTO DE IMAGEM DIGITAL

Uma imagem digital é representada por uma matriz $M \times N$, sendo que $N$ corresponde a largura e $M$ à altura. A função $f(M, N)$ denota um ponto na imagem e é chamado de pixel. Em imagens em tons de cinza (Figura 2.1a) um pixel é definido no intervalo (0, 255) (0 para preto e 255 para branco) e em imagens coloridas (Figura 2.1b) um pixel é definido por vetor composto por 3 atributos, indicando as cores vermelho, verde e azul (do inglês *red, green e blue* - RGB) sendo que o valor de cada componente está no intervalo (0, 255).

(a) Tons de Cinza (b) Imagem Colorida

Figura 2.1 – Exemplo de imagem colorida e em tons de cinza (PROENÇA e ALEXANDRE, 2006).

Alguns métodos de segmentação de imagem, como a binarização, demandam uma imagem representada por tons de cinza. Para transformar uma imagem colorida em tons de cinza, calcula-se a média das três cores do pixel – vide Equação (2.1).

$$f(M,N) = \frac{f(M,N,R) + f(M,N,G) + f(M,N,B)}{3} \tag{2.1}$$

## 2.1 FUNDAMENTO DE COR

Cor é uma propriedade importante na análise de imagens realizada pelos seres humanos com ou sem o auxílio do computador. A identificação de objetos e a interpretação de uma cena podem ser simplificadas com o uso de cor (PEDRINI e SCHWARTZ, 2008).

As características usadas para distinguir uma cor de outra são o brilho, o matiz e a saturação. O brilho ou luminância representa a noção de intensidade luminosa da radiação. O

matiz é uma propriedade associada ao comprimento de onda predominante na combinação de ondas de luz. A saturação expressa a pureza do matiz ou, de modo similar, o grau de mistura do matiz original com a luz branca. O matiz e a saturação, quando tomados juntos, são chamados de crominância e uma cor pode ser caracterizada pelo seu brilho e crominância.

Os modelos ou espaços de cores permitem a especificação de cores em um formato padronizado para atender a diferentes dispositivos gráficos ou aplicações que requeiram a manipulação de cores. Um modelo de cor é essencialmente uma representação tridimensional na qual cada cor é especificada por um ponto no sistema de coordenadas tridimensionais. O universo de cores que podem ser reproduzidas por um modelo é chamado de espaço.

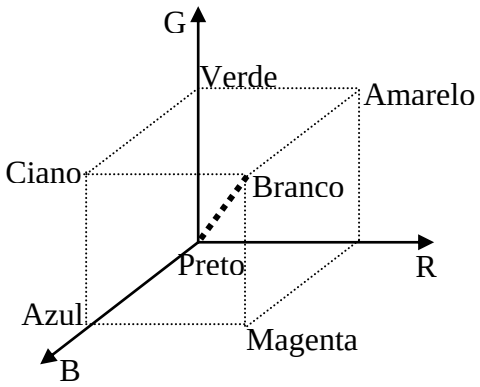


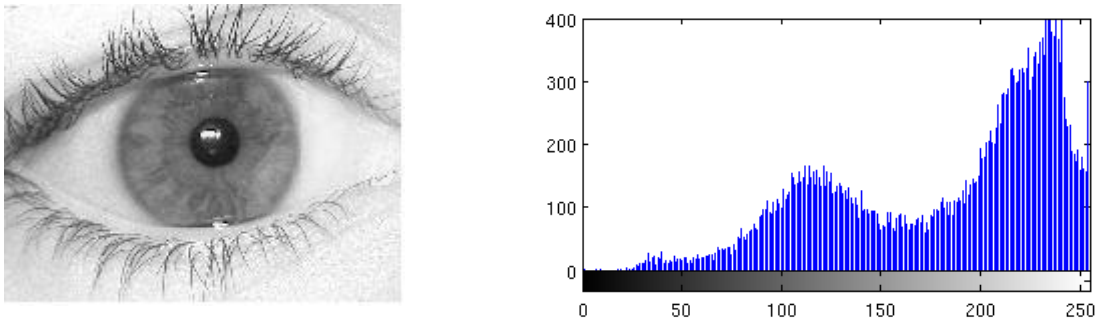

Figura 2.2 – Modelo RGB (PEDRINI e SCHWARTZ, 2008).

O modelo de cor comumente usado para a representação de imagens digitais é o RGB (SONKA, HLAVAC e BOYLE, 1993). Este modelo é baseado em um sistema de coordenadas cartesianas, em que o espaço de cores é um cubo (Figura 2.2). As cores primárias, vermelho (R – *red*), verde (G – *Green*) e azul (B – *blue*) estão em três vértices do cubo. As cores primárias complementares, ciano, magenta e amarelo, estão em outros três vértices. O vértice sobre à origem é o preto e o mais afastado da origem corresponde à cor branca. Neste modelo, a escala de cinza se estende através da diagonal do cubo, ou seja, a reta que une a origem (preto) ao vértice mais distante (branco).

## 2.2 HISTOGRAMA

O histograma de uma imagem corresponde à distribuição dos níveis de cinza, o qual pode ser representado através de um gráfico indicando o número de pixels na imagem para cada nível de cinza.

Através do histograma de uma imagem é possível obter várias medidas estatísticas de uma imagem, como mínimo, máximo, média, variância e desvio padrão dos tons de cinza dos pixels (PEDRINI e SCHWARTZ, 2008). A Figura 3.3a ilustra uma imagem em tons de cinza e a Figura 3.3b, o histograma correspondente.

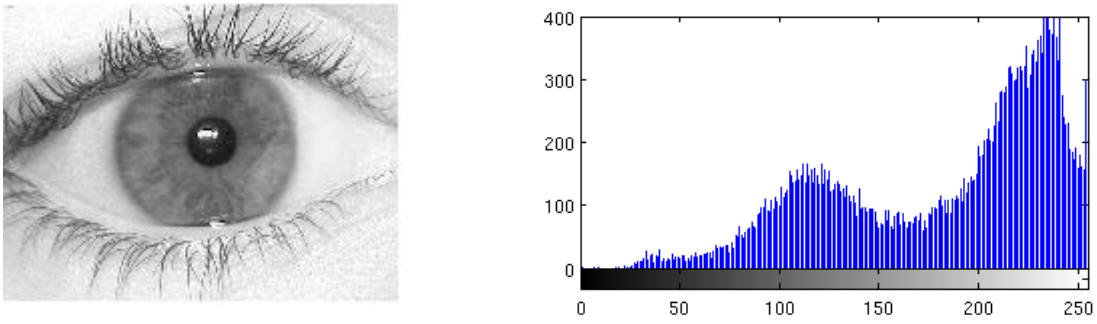



(a) Tons de cinza (b) Histograma

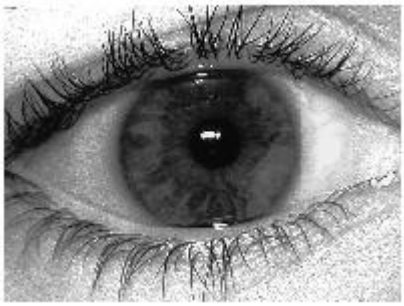
Figura 2-3 – Histograma da imagem em tons de cinza.

## 2.3 EQUALIZAÇÃO DO HISTOGRAMA

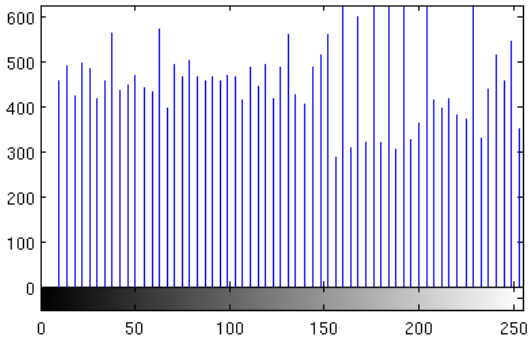
O método de equalização do histograma modifica a imagem em tons de cinza de forma a obter uma distribuição uniforme do histograma, ou seja, os níveis de cinza devem aparecer na imagem aproximadamente com a mesma frequência. Nas Figuras 3.4a e 3.4b é apresentada uma imagem em tons de cinza e o histograma correspondente. Já nas Figuras 3 4c e 3.4d é apresentada a mesma imagem em tons de cinza e o histograma correspondente após a equalização.




(a) Tons de cinza (b) Histograma



(c) Imagem Equalizada




(d) Histograma Equalizado

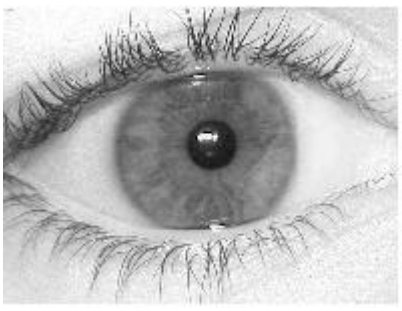
Figura 2-4 – Histograma equalizado.

A equalização do histograma possui a vantagem de ser completamente automática em relação às técnicas manuais para modificação de contraste. Entretanto, há situações nas quais a equalização de histograma pode degradar uma imagem. Um exemplo é quando a imagem a ser transformada possui um histograma com grande concentração de pixels em poucos níveis de cinza (SONKA, HLAVAC e BOYLE, 1993).

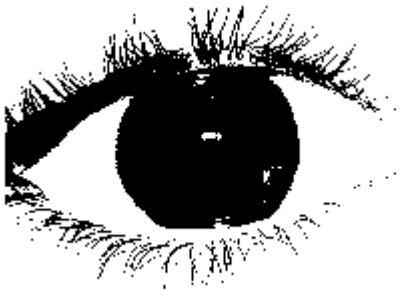
## 2.4 BINARIZAÇÃO

A binarização consiste em transformar uma imagem de níveis de cinza em uma imagem com duas cores (branco e preto) ou binária. Uma das maneiras de se obter uma imagem binarizada é através da análise de seu histograma, definindo uma cor como limiar, desta forma, os pontos da imagem que possuem uma cor menor que este limiar são modificados para 0, e as cores acima deste limiar, são modificadas para 255. A Equação (2.2) demonstra como aplicar a binarização:

$$f(M,N) = \begin{cases} 0, & se\ f(M,N) \leq LIMIAR \\ 255, & se\ f(M,N) > LIMIAR \end{cases} \quad (2.2)$$



(a) Tons de cinza



(b) Binarizada

Figura 2.5 – Imagem Binarizada.

A Figura 2.5 exemplifica o processo de binarização. Através da análise do histograma da Figura 2-4b se obteve um limiar de 160.

### 2.4.1 BINARIZAÇÃO DE OTSU

Um modelo automático de binarização que procura obter o melhor limiar para a separação dos pixels é proposto por Otsu (OTSU, 1979). O método considera que o histograma de uma imagem é composto de duas classes, que possuem características próprias, ou seja, sua média e desvio padrão. A variância ${\sigma^2}_T$ e a média $\mu_T$ globais da imagem são calculadas, e então, o método de Otsu procura maximizar a razão $n(T)$ da variância entre as classes ${\sigma^2}_B$ em relação à variância total, considerando todos os valores possíveis do limiar $T$ – Equação (2.3).

$$n(T) = \frac{{\sigma^2}_B}{{\sigma^2}_T} \tag{2.3}$$

A variância global é obtida através da Equação (2.4) para um intervalo de intensidade de cinza de (0 ,L-1).

$${\sigma^2}_T = \sum_{i=0}^{L-1} (i - \mu_T)^2 p_i \tag{2.4}$$

onde:

$$\mu_T = \sum_{i=0}^{L-1} i.p_i \tag{2.5}$$

à variância entre as classes pode ser obtida através da equação (2.6).

$${\sigma^2}_B = \omega_1 \omega_2 (\mu_1 \mu_2)^2 \tag{2.6}$$

onde:

$$\omega_1 = \sum_{i=0}^{T} p_i \tag{2.7}$$

$$\omega_2 = 1 - \omega_1 \tag{2.8}$$

$$\mu_1 = \frac{\mu_s}{\omega_2} \tag{2.9}$$

$$\mu_2 = \frac{\mu_T - \mu_s}{\omega_2} \tag{2.10}$$

$$\mu_s = \sum_{i=0}^{T} i.p_i \tag{2.11}$$

$$\sum_{i=0}^{L-1} p_i = 1, \qquad p_i = \frac{n_i}{n} \tag{2.12}$$

De acordo com a Equação (2.3), a razão $n(T)$ é calculada para todos os valores possíveis de *T*. Desta forma, o limiar ótimo pode ser determinado como:

$$T = \arg\max n(T) \tag{2.13}$$

Segundo Pedrini e Schwartz (2008), o método proposto por Otsu possui um bom desempenho em imagens com maior variância da intensidade, com a desvantagem de assumir que o histograma da imagem seja bimodal.

## 2.5 FILTRAGEM NO DOMÍNIO ESPACIAL

O domínio espacial refere-se ao próprio plano da imagem, ou seja, ao conjunto de pixels que compõe uma imagem. A transformação de um ponto da imagem *f(M, N)* depende do valor do próprio pixel e de outros pontos da vizinhança de *f(M, N)*. O processo de filtragem é realizado por meio de matrizes denominadas máscaras, as quais são aplicadas através do processo de convolução sobre a imagem.

Em processamento de sinal, um filtro linear é caracterizado pela sua resposta a um impulso. A resposta *g(x)* de um filtro linear *w(x)* para um sinal contínuo unidimensional *f(x)* é dada através da seguinte integral de convolução:

$$g(x) = \int_0^T f(t)\, w(x-t)dt \tag{2.14}$$

Imagens digitais são bidimensionais e discretas. Assim a convolução de uma imagem *f*(*M*, *N*) por um filtro *w* é dada por:

$$G(M,N) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} f(m-i, n-j)w(i,j) \tag{2.15}$$

Para obter a imagem *G(M, N)*, o filtro é aplicado em todos os pixels da imagem, obedecendo a restrições de borda.

## 2.6 FILTRAGEM GAUSSIANA

O operador de suavização gaussiana consiste em uma operação de convolução utilizada para remover detalhes e ruídos de uma imagem. O filtro é obtido através da função gaussiana bidimensional. A função gaussiana com média zero é definida como:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{2.16}$$

Como a imagem é uma coleção discreta de pixels, é necessário produzir uma aproximação discreta da função de distribuição. Na teoria, a distribuição gaussiana é assintótica e positiva em qualquer valor. Logo, seria necessária uma máscara de convolução infinitamente grande, mas na prática esta se torna zero para valores mais distante do que três ou quatro desvios-padrões em relação à média, o que permite então a truncagem da máscara a partir deste ponto. A Equação (2.17) ilustra uma máscara com uma escala $\sigma = 1.4$ e dimensão *5×5*.

$$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \tag{2.17}$$

Já a Equação (2.18) ilustra uma máscara com a mesma escala, mas com dimensão *3×3*.

$$\frac{1}{273}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.18}$$

Filtros gaussianos apresentam diversas características que os tornam particularmente úteis em processamento de imagens (PEDRINI e SCHWARTZ, 2008):

- As funções gaussianas em duas dimensões são simétricas, com isto, o grau de suavização realizado através do filtro será o mesmo em todas as direções.
- A suavização é realizada substituindo cada pixel por uma média ponderada dos pixels vizinhos, tal que o peso dado a um vizinho decresce monotonicamente com a sua distância ao pixel central.
- O grau de suavização está relacionado com o parâmetro de desvio padrão: quanto maior o valor, maior é o grau de suavização.
- A função gaussiana permite a decomposição das componentes *x* e *y*. Portanto, a convolução pode ser realizada processando a imagem com um filtro unidirecional, e

processar o resultado ortogonalmente com o outro filtro unidirecional, reduzindo desta forma o número de operações utilizadas na convolução gaussiana.

## 2.7 DETECÇÃO DE BORDAS

A detecção de borda é fundamental na análise de imagens. Uma borda é o limite entre duas regiões com propriedades relativamente distintas de nível de cinza (SONKA, HLAVAC e BOYLE, 1993). Bordas caracterizam os limites de objetos em uma figura, e são determinantes para a segmentação, registro e identificação de objetos em uma cena (LALIGANT e TRUCHETET, 2010).

A operação de identificação de mudanças locais significativas nos níveis de cinza da imagem pode ser descrita por meio do conceito de derivada. Como uma imagem depende de duas coordenadas espaciais, as bordas da imagem são expressas por derivadas parciais. A técnica para encontrar a força (intensidade) e a direção da borda na posição (*x*, *y*) de uma imagem *f*, é o gradiente, denotado por $\nabla f$, e é definido por:

$$\nabla f = \frac{\partial f(x,y)}{\partial x} i + \frac{\partial f(x,y)}{\partial y} j, \tag{2.19}$$

onde *i* e *j*, são vetores unitários das direções *x* e *y*, respectivamente. Uma variação rápida de *f(x, y)* ao longo da direção *x* e lenta ao longo da direção *y* indica a presença de uma borda praticamente vertical. Na forma vetorial, o gradiente da imagem pode ser expresso como:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{2.20}$$

A magnitude do gradiente equivale à maior taxa de variação de *f(x, y)* de acordo com a unidade de distância na direção de $\nabla f$, sendo determinada através da distância Euclidiana de acordo com a Equação (2.21). Para uma redução no esforço computacional, outra forma de se calcular a magnitude do gradiente é através da distância de Manhattan Equação (2.22).

$$mag(\nabla f) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \tag{2.21}$$

$$mag(\nabla f) = \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right| \tag{2.22}$$

A direção do vetor do gradiente de uma imagem representada por *f*(*x*, *y*) é obtida através do ângulo calculado na Equação (2.23). Em que $\theta(x,y)$ é o ângulo da direção do vetor $\nabla f$ na posição (*x*, *y*):

$$\theta(x,y) = arctan\left(\frac{\frac{\partial f}{\partial x}}{\frac{\partial f}{\partial y}}\right), \tag{2.23}$$

onde $\theta(x,y)$ é o ângulo medido em radianos do eixo *x* entre o ponto (*x*, *y*) - Figura 2.6. Uma mudança em intensidade pode ser detectada pela diferença entre os valores de pixels adjacentes. Bordas verticais podem ser detectadas pela diferença horizontal entre os pontos, enquanto bordas horizontais pela diferença vertical entre os pontos.

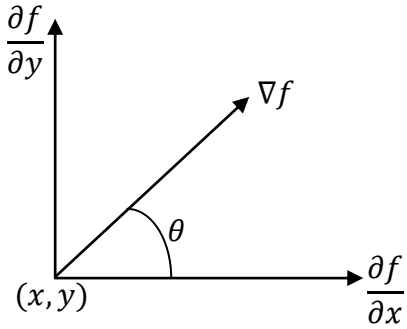



Figura 2.6 – Direção do gradiente.

Uma imagem digital é discreta por natureza e por isso as Equações (2.21) e (2.23), contendo derivadas, devem ser aproximadas através de diferenças. Uma forma simples de aproximação consiste em usar a diferença da Equação (2.24) na direção *x* e a Equação (2.25) na direção *y*.

$$\frac{\partial f}{\partial x} = f(x,y) - f(x+1,y) \tag{2.24}$$

$$\frac{\partial f}{\partial y} = f(x,y) - f(x,y+1) \tag{2.25}$$

Sobel (1990) propôs um operador que aproxima a magnitude do gradiente mediante a diferença de valores ponderados dos níveis de cinza da imagem (SOBEL, 1990). Ou seja:

$$\begin{aligned}\frac{\partial f}{\partial x} \approx\ & [f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1)] - \\ & [f(x-1,y-1) + 2f(x-1,y) + f(x-1,y+1)]\end{aligned} \tag{2.26}$$

$$\frac{\partial f}{\partial y} \approx [f(x-1,y+1) + 2f(x,y+1) + f(x+1,y+1)] - [f(x-1,y-1) + 2f(x,y-1) + f(x+1,y-1)] \quad (2.27)$$

Em que através das Equações (2.26) e (2.27) são obtidas as máscaras do operador Sobel – Equação (2.28), essas máscaras são convoluídas (Seção 2.5) com a imagem obtendo-se uma imagem com as bordas destacadas – vide Figura 2-7.

$$\frac{\partial f}{\partial x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad e \quad \frac{\partial f}{\partial y} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (2.28)$$

(a) Imagem original (b) Bordas detectadas com Sobel

Figura 2-7 – Detecção de bordas pelo operador Sobel.

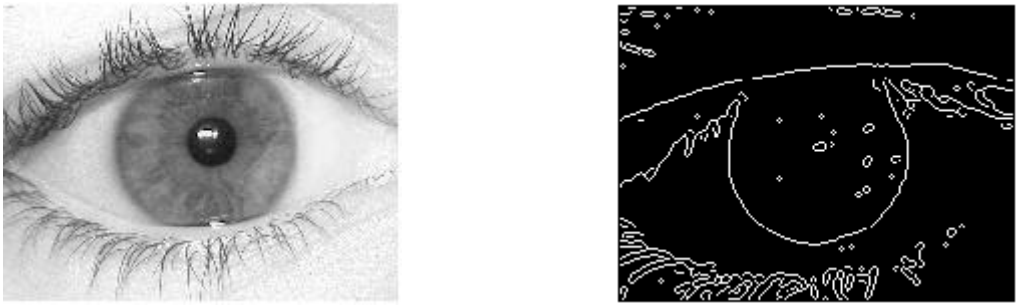
Outra abordagem para a detecção de bordas é proposta por Canny, que procura otimizar a localização de pontos da borda na presença de ruído (CANNY, 1987) (SONKA, HLAVAC e BOYLE, 1993).

O algoritmo é composto por cinco etapas:

1) Procede-se com a remoção de ruídos através da filtragem gaussiana (Seção 2.6).
2) Calcula-se a magnitude e a direção do gradiente.
3) As bordas são marcadas para os pontos cuja magnitude do gradiente seja localmente máxima na direção do gradiente.
4) Bordas são potencialmente determinadas através de dois limiares diferentes, $T_1$ e $T_2$, com $T_2 > T_1$. Essa etapa é conhecida como limiarização com histerese. Pontos da borda que possuem magnitude gradiente maior que $T_2$ são mantidos como pontos da borda. Qualquer outro ponto conectado a esses pontos da borda é

considerado como pertencente à borda se a magnitude de seu gradiente estiver acima de $T_1$.

O detector de bordas de Canny apresenta-se como o método com maior robustez em relação aos outros métodos (SONKA, HLAVAC e BOYLE, 1993). A Figura 2-8 exemplifica o método de Canny em uma imagem do olho humano.



(a) Imagem Original (b) Bordas detectadas

Figura 2-8 – Operador de Canny.

## 2.8 TRANSFORMADA DE HOUGH

Se uma imagem é composta de objetos cuja forma e tamanho são conhecidos, a segmentação pode ser vista como um processo de busca destes objetos dentro da imagem (ROSTEN, PORTER e DRUMMOND, 2010).

A transformada de Hough consiste em detectar um conjunto de pontos na imagem que pertençam a um contorno parametrizado específico (HOUGH, 1962). A transformada de Hough em conjunto com outras técnicas de segmentação é amplamente utilizada na detecção da íris e pupila (LI, LIU, *et al.*, 2009).

O método consiste em se definir um mapeamento entre o espaço da imagem *f*(*x*, *y*) com as bordas detectadas (Seção 2.7) e o espaço de parâmetros também conhecido como espaço de Hough. Para a detecção de circunferências, uma possível formulação em coordenadas cartesianas é apresentada na Equação (2.29). E na forma paramétrica é representada em coordenadas polares na Equação (2.30).

$$(x-a)^2 + (y-b)^2 = r^2 \tag{2.29}$$

$$\begin{aligned} x &= a + r\cos\theta \\ y &= b + r\,\mathrm{sen}\,\theta \end{aligned} \tag{2.30}$$

Nessas equações, *a* e *b* são as coordenadas do centro da circunferência, *r* é o raio e $\theta$ é a direção do gradiente. Para cada ponto na imagem *f*(*x*, *y*), a célula de acumulação (*a,b,r*) é incrementada se o ponto *(a,b)* estiver à uma distância de raio *r* do ponto (*x,y*). Desta forma, considera-se como uma circunferência na imagem, se o centro (*a*, *b*) e o raio *r*, são encontrados frequentemente no espaço de parâmetros.

## 2.9 EROSÃO E DILATAÇÃO COM MORFOLOGIA MATEMÁTICA

A linguagem da morfologia matemática é a teoria dos conjuntos, onde os conjuntos representam as formas em uma imagem. Esta oferece uma abordagem unificada e poderosa para vários problemas de processamento de imagem (GONZALEZ e WOODS, 2010).

A morfologia matemática consiste em uma metodologia para manipular imagens binárias ou em tons de cinza através de operadores morfológicos. Essa técnica de processamento e análise de imagem é utilizada em processos como, extração de componentes conexos, extração de bordas do objeto, afinamento de bordas (PEDRINI e SCHWARTZ, 2008).

Após o processo de binarização é possível aplicar a morfologia matemática para retirar ou reduzir irregularidades nos contornos, eliminar buracos no interior e remover ruídos, através das operações de erosão e dilatação (ZHANG e QIN, 2010).

Em imagens binárias, os conjuntos em questão são membros do espaço bidimensional de números inteiros $\mathbb{Z}^2$ em que cada elemento de um conjunto é um vetor bidimensional de coordenadas (*x,y*) de um pixel branco de uma imagem.

A erosão do conjunto *A* por *B* é definido na Equação (2.31). O conjunto resultante da erosão de *A* por *B* é o conjunto de todos os pontos *x*, tal que *B*, transladado por *x*, está contido em *A*. Considera-se *A* como uma imagem binária e *B* como um elemento estruturante.

$$A \ominus B = [x \,|(B)_x \subseteq A] \tag{2.31}$$

A erosão tem a vantagem de remover saliências e remover pequenas regiões isoladas que poderão ser ruído. Tem, no entanto o inconveniente de aumentar buracos no interior dos objetos reentrâncias nos seus contornos e de poder separar partes do mesmo objeto que se encontrem unidas por linhas finas.

A dilatação de *A* por *B* é definida na Equação (2.32). Essa equação baseia-se na reflexão de *B* em torno de sua origem, seguida da translação dessa reflexão por *x*. A dilatação

de $A$ por $B$ é, então, o conjunto de todos os deslocamentos de $x$ de forma que $\hat{B}$ e A se sobreponham pelo menos por um elemento não nulo.

$$A \oplus B = [x \,|(\hat{B})_x \cap A \neq \emptyset\,] \tag{2.32}$$

A dilatação tem a vantagem de eliminar reentrâncias nos contornos e buracos no interior dos objetos. Tem as desvantagens de realçar saliências, aumentar pequenas regiões que poderão ser ruído e ligar dois objetos distintos que estejam muito próximos.

Para reduzir estas desvantagens, usa-se normalmente qualquer combinação destas técnicas. Se o objetivo é erodir um objeto, é possível dilatá-lo primeiro para eliminar pequenas reentrâncias ou buracos e seguidamente erodi-lo duas vezes seguidas (SONKA, HLAVAC e BOYLE, 1993).

# 3 BIOMETRIA

Biometria é a ciência que estabelece uma identidade para um indivíduo através dos seus atributos pessoais, físicos, químicos ou comportamentais. Impressão digital, face, geometria da mão, íris e retina são exemplos de biometrias com atributos físicos; assinatura, caminhada e digitação são atributos comportamentais e DNA atributo químico.

A utilização de sistemas biométricos tem sido cada vez mais incentivada pelo governo e entidades privadas a fim de substituir ou melhorar os sistemas de segurança tradicionais (PROENÇA e ALEXANDRE, 2006).

Sistemas de autenticação biométrica são cada vez mais indispensáveis para proteger vidas e bens. Eles são robustos e confiáveis, de difícil falsificação, e podem autenticar os indivíduos rapidamente (FAUNDEZ-ZANUY, ELIZONDO, *et al.*, 2007).

Um sistema biométrico é essencialmente um sistema de reconhecimento de padrões que adquire dados biométricos de um indivíduo, extrai uma ou mais características definidas a partir dos dados coletados e faz a comparação com um conjunto de características armazenadas no banco de dados, executando uma ação com base nesta comparação.

Um sistema biométrico genérico pode ser visto como tendo quatro módulos principais (JAIN, FLYNN e ROSS, 2008): um módulo de aquisição; um de extração de características; comparação das características e um de banco de dados. A Figura 3.1 demonstra o funcionamento de um sistema biométrico para classificação de um indivíduo com a utilização da íris.

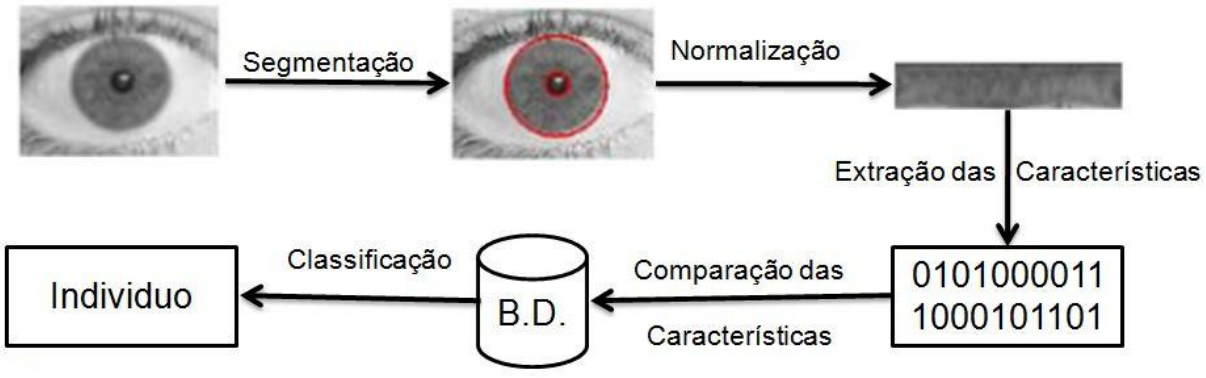



Figura 3.1 – Funcionamento de um Sistema Biométrico da íris.

- **Aquisição** - O processo de aquisição obtém os dados biométricos através de sensores biométricos e, com auxílio de um software, as características são extraídas e armazenadas em um banco de dados.

- **Extração de características** - No processo de extração é obtida uma representação digital do exemplar obtido, em que cada exemplar é associado a um único indivíduo.

- **Comparação** - As características extraídas são comparadas com os modelos armazenados para verificar qual o grau de similaridade entre as características extraídas do indivíduo com as armazenadas previamente no banco de dados.

- **Banco de dados** - O banco de dados funciona como repositório de informações biométricas e juntamente com as características extraídas, são armazenado os dados como PIN (número de identificação pessoal) e nome do indivíduo.

A Figura 3.22 apresenta a situação atual quanto à utilização dos sistemas biométricos existentes.

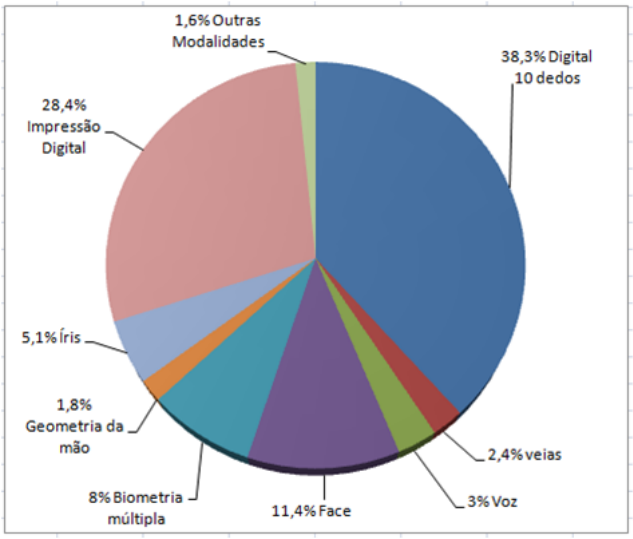



Figura 3.2 – Situação atual dos sistemas Biométricos (GROUP, 2009).

A seguir serão apresentados brevemente os dois sistemas biométricos mais utilizados conforme o gráfico da Figura 3.2 e, posteriormente, com uma maior abrangência, o sistema biométrico com utilização da íris.

## 3.1 IMPRESSÃO DIGITAL

A impressão digital é a representação da epiderme de um dedo, existindo um padrão de intercalação das cristas e dos vales (JAIN, FLYNN e ROSS, 2008).

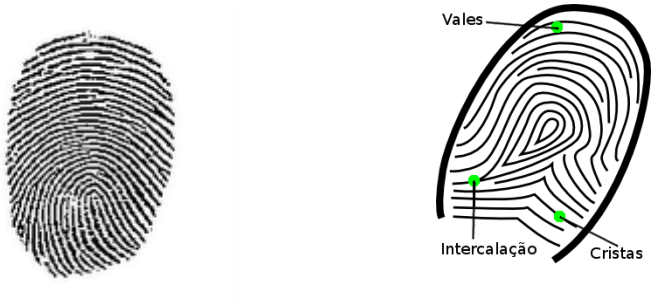



Figura 3.3 – Exemplo de impressão digital.

A biometria da digital consiste em encontrar os padrões nas intercalações das cristas e vales (Figura 3.3) (JAIN, HONG e BOLLE, 1997). Alguns fatores podem prejudicar a identificação ou autenticação do indivíduo no processo de biometria da digital. Trabalhos manuais podem ao longo do tempo modificar as intercalações das cristas do indivíduo (BLEHA, SLIVINSKY e HUSSEIN, 1990) formando um novo padrão na sua digital. O dispositivo de coleta da digital pode ficar sujo ou oleoso, acrescentando ruídos na imagem coletada (BLEHA, SLIVINSKY e HUSSEIN, 1990).

## 3.2 RECONHECIMENTO DA FACE

Segundo (PHILLIPS, MARTIN, *et al.*, 2000) os atributos referentes à dimensão, proporção e físicos da face de uma pessoa são únicos.

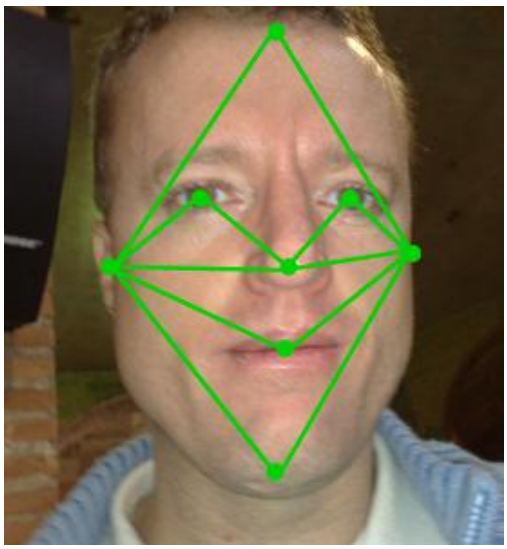


Figura 3.4 – Geometria da face.

A biometria da face consiste em um sistema de análise da geometria facial (Figura 3.4) neste sistema, as características são distâncias entre boca e orelha, nariz e orelha, olho e nariz, olho e orelha, queixo e orelha e testa e orelha. A robustez do sistema é determinada através da quantidade de características extraídas da face (PHILLIPS, MOON, *et al.*, 2000).

Geralmente, as imagens da face, comparadas com imagens da digital e íris, possuem uma quantidade maior de atributos, com o crescimento da base de dados, o sistema perde desempenho ao fazer a comparação do indivíduo com o restante das imagens para sua identificação (PHILLIPS, MARTIN, *et al.*, 2000).

## 3.3 ÍRIS

A íris é formada no início da vida em um processo chamado morfogênese. Depois de completamente formada, a textura é estável ao longo da vida. A íris tem um padrão único de olho para olho e de pessoa para pessoa Figura 3.5 (PROENÇA e ALEXANDRE, 2006).

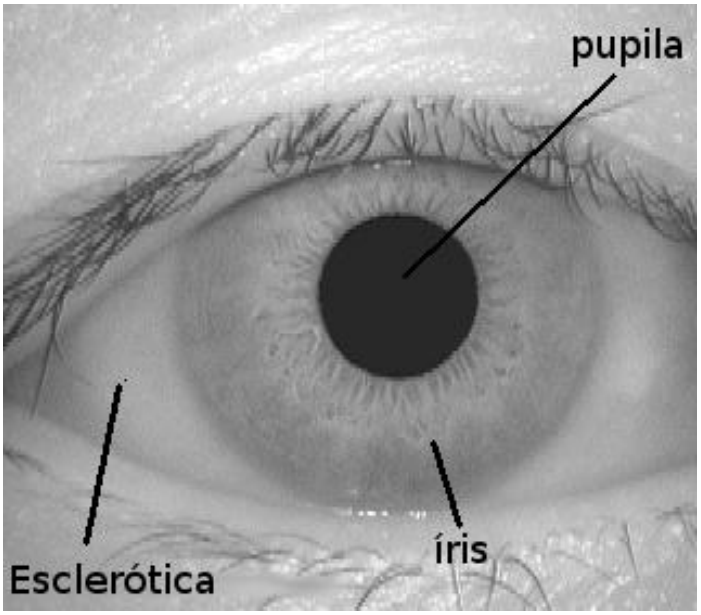



Figura 3.5 – Olho humano (DAUGMAN, 1993).

A biometria da íris consiste em uma varredura a fim de analisar padrões nas ranhuras que a formam. Óculos, lentes de contato e mesmo cirurgia ocular não modificam as características da íris.

### 3.3.1 MÉTODO DE DAUGMAN

A metodologia proposta por John Daugman (DAUGMAN, 1993) é certamente a mais citada na literatura de reconhecimento de íris. É licenciada para a Iridian Technologies, que detém 99,5% dos sistemas comerciais de reconhecimento de íris do mercado (PROENÇA e ALEXANDRE, 2006). Proposta em 1992, foi a primeira metodologia efetivamente implementada em um sistema biométrico.

Daugman assume que a pupila e a íris possuem uma forma circular e propõe a seguinte integral parametrizada para a sua segmentação (DAUGMAN, 1993).

$$max_{r,x0,y0} \left| \mathrm{G}\sigma(\mathrm{r}) * \frac{\partial}{\partial r} \oint_{r,x0,y0} \frac{f(x,y)}{2\pi r} ds \right| \tag{3.1}$$

Este operador busca no domínio (*x*, *y*) da imagem $f(x,y)$, contendo um olho, o conjunto de parâmetros de centro e raio $(r, x_0, y_0)$ que apresenta o valor máximo da derivada parcial da integral da imagem normalizada ao longo de um caminho circular *ds*. A função $G\sigma(r)$ *é* um filtro gaussiano para a redução de ruídos da imagem.

Para normalização da íris, com o objetivo de compensar distâncias entre o indivíduo e a câmera e contrações da pupila, é aplicada a Equação (3.2), transformando a imagem original *f*(*x*,*y*), com coordenadas cartesianas, em uma nova imagem *f*(*r*,*q*) em coordenadas polares. A normalização transforma o anel que contém a pupila em um retângulo com dimensões fixas.

$$x(r,\theta) = (1 - r)\, x_p\, (\theta) - r x_s(\theta) \tag{3.2}$$

Para a extração das características da íris, Daugman propõe um método chamado *Wavelets 2D de Gabor*, que mapeia a íris em vetores e adquire os dados relevantes da imagem, como frequência espacial, orientações e posições. Com estes dados, é possível mapear o código da íris (Figura 3.6).

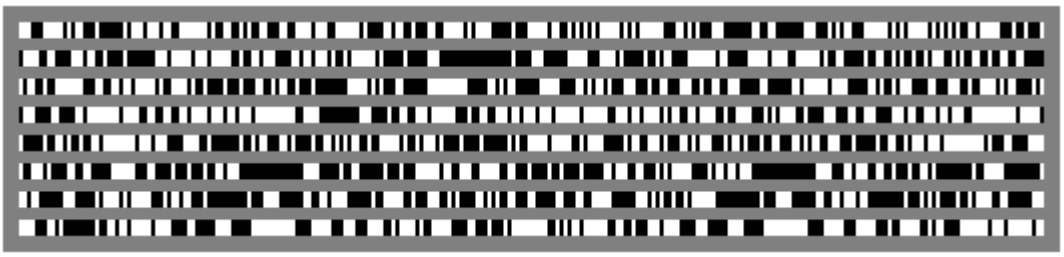


Figura 3.6 – IrisCode (DAUGMAN, 1993).

O autor define o filtro de Gabor para mapeamento da íris definido em um sistema de coordenadas polares duplo de $(r, \theta)$ como:

$$H(r,\theta) = exp^{-i\omega(\theta-\theta_0)} exp^{-(r-ro)^2/\alpha^2} exp^{-(-\theta-\theta_0)/\beta^2} \quad (3.3)$$

Os parâmetros $\alpha$ e $\beta$ variam em proporção inversa a $\omega$ para gerar uma similaridade no conjunto de filtros centralizados no intervalo de $\theta_0$ e $ro$. A classificação é obtida através do cálculo de similaridade entre duas íris (A e B) através da equação:

$$HD = \frac{1}{2048} \sum_{i=1}^{2048} A_i (XOR) B_i \quad (3.4)$$

Este cálculo de similaridade entre duas íris é conhecido como distância de Hamming e define a porcentagem de diferença entre os códigos de duas íris quaisquer. Normalmente, se aceita um padrão de 50% de bits semelhantes entre dois códigos de íris diferentes (*HD = 0,5*).

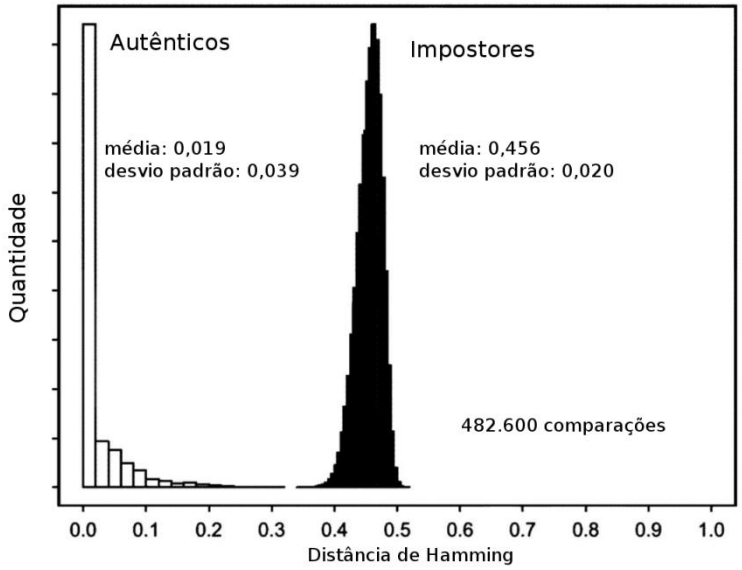



Figura 3.7 – Distribuição da distância de Hamming (DAUGMAN, 1993).

Na Figura 3.7 é possível visualizar que quanto maior o percentual da distância de Hamming, maior é a chance de aceitar duas íris diferentes como sendo iguais, e quanto menor o percentual, maior é a chance de se comparar errado a mesma íris.

### 3.3.2 MÉTODO DE WILDES E CAMUS

Os autores propuseram uma metodologia para busca da íris e da pupila baseada na equação de Daugman (CAMUS e WILDES, 2002) e que pode ser vista na Equação (3.5).

$$C = \sum_{\theta=1}^{n}((n-1)\big|\big|g(\theta,r)\big|\big| - \sum_{\emptyset=\theta+1}^{n}\big|\big|g(\theta,r) - g(\emptyset,r)\big|\big| - \frac{I(\theta,r)}{n} \qquad (3.5)$$

em que $||g(\theta,r)||$ é o gradiente sobre $\theta,r$ da imagem normalizada. Após esse procedimento segue-se a detecção de borda usando-se a transformada de Hough.

Este método é eficaz em imagens em que o contraste da região da íris com a da pupila são claramente separados. Em imagens ruidosas ou com baixo contraste da íris com a pupila a precisão do algoritmo decai de forma significativa (PROENÇA e ALEXANDRE, 2006).

### 3.3.3 MÉTODO DE PROENÇA E ALEXANDRE

Proença e Alexandre (2006) propuseram um algoritmo de segmentação baseado na textura da imagem. Os autores empregam métodos de agrupamento de dados em um conjunto formado através do cálculo do momento geométrico de segunda ordem para cada pixel da imagem, produzindo uma imagem intermediária, na qual se aplica o detector de bordas de Canny (CANNY, 1987). Em seguida, a íris e pupila são localizadas usando a transformada de Hough (PROENÇA e ALEXANDRE, 2006). O momento geométrico de segunda ordem para cada pixel é calculado como:

$$M_{pq} = \sum_{-W/2}^{W/2} \left( \sum_{-W/2}^{W/2} \left( f(m,n) x_m^p . y_n^q \right) \right) \tag{3.6}$$

em que $M_{pq}$ é o momento geométrico de segunda ordem *pq*, *f(m, n)* é a intensidade do pixel de uma imagem *f*, *x* e *y* são as coordenas de vizinhança e *W* a largura.

Os autores concluíram que este momento não dispunha de capacidade discriminante suficiente e propuseram então a aplicação da tangente hiperbólica como um mapeamento não linear:

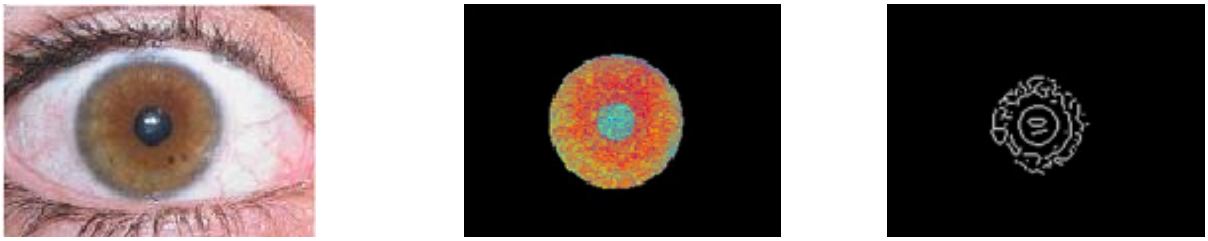
$$F_{pq}(i,j) = \frac{1}{L} \sum_{(a,b) \in W_{ij}} \left( \tanh \left( \sigma \left( M_{pq}(a,b) - \bar{M} \right) \right) \right) \tag{3.7}$$

em que $F_{pq}$ é a imagem resultante do momento $M_{pq}$ com média $\bar{M}$ e $W_{ij}$ é uma janela $L\ X\ L$ com média centralizada na posição $(i,j)$. $\sigma$ é o parâmetro que controla a forma da função logística e que foi determinado por tentativa e erro, sendo que 0,01 foi o melhor valor para a maioria dos casos. Os autores realizaram a clusterização da imagem resultante com três diferentes algoritmos: k-*means*, *fuzzy* k-*means* e mapas auto-organizáveis de Kohonen.

### 3.3.4 MÉTODO DE KHEIROLAHY

Todos os métodos para a segmentação da íris são baseados em modelos para localização da circunferência da pupila e da íris. Em imagens ruidosas ou de baixo contraste

da pupila com a íris, estes métodos apresentam uma deficiência na segmentação da pupila. Os autores propuseram então uma otimização do mapeamento de cores da imagem para a diminuição dos ruídos e melhorar o desempenho na segmentação da pupila com os métodos de localização de circunferência (KHEIROLAHY, EBRAHIMNEZHAD e SEDAAGHI, 2009). A Figura 3.8 ilustra o processo empregado pelos autores.



(a) Imagem Original (b) Imagem (c) Pupila segmentada

Figura 3.8 – Otimização do mapeamento de cores da íris (KHEIROLAHY, EBRAHIMNEZHAD e SEDAAGHI, 2009).

O método determina a circunferência da pupila aplicando primeiramente uma binarização na imagem com o objetivo de separar a íris do restante da imagem, e em seguida as bordas são destacadas com o operador de Canny (CANNY, 1987). Aplica-se morfologia matemática para reconstruir os pixels da borda eliminados na binarização e finalmente a circunferência da íris é localizada com a transformada de Hough (HOUGH, 1962).

Com a íris localizada na imagem, e para destacar a pupila, procede-se com uma transformação no modelo de cores empregando o método de *Levenberg-Marquardt* (LM) com o objetivo de eliminação de ruídos – Equação (3.8).

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \equiv M_{3x3}\, x \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.8}$$

em que $M$ é a matriz de cores mapeadas e parametrizada com $\theta = [a_{11}, a_{21}, a_{31}, a_{12}, a_{22}, a_{32}, a_{13}, a_{23}, a_{33}]$ o qual é calculado através do algoritmo de otimização LM e ($R'$,$G'$,$B'$) e ($R$,$G$,$B$) são os espaço de cor das imagens mapeada e original, respectivamente. Ou seja, empregando o algoritmo de *Levenberg-Marquardt* o objetivo é minimizar a seguinte função de erro:

$$e = \left( \frac{\vec{A}.\vec{B}}{|\vec{A}|.|\vec{B}|} \right) = \cos(\propto) \tag{3.9}$$

$$\vec{A} = \left[\frac{1}{nR'_d}\sum_{m=1}^{linhas}\sum_{l=1}^{colunas} R'(m,l)_d\right]\vec{i} + \left[\frac{1}{nG'_d}\sum_{m=1}^{linhas}\sum_{l=1}^{colunas} G'(m,l)_d\right]\vec{j} + \left[\frac{1}{nB'_d}\sum_{m=1}^{linhas}\sum_{l=1}^{colunas} B'(m,l)_d\right]\vec{k} \quad (3.10)$$

$$\vec{B} = \left[\frac{1}{nR'_f}\sum_{m=1}^{linhas}\sum_{l=1}^{colunas} R'(m,l)_f\right]\vec{i} + \left[\frac{1}{nG'_f}\sum_{m=1}^{linhas}\sum_{l=1}^{colunas} G'(m,l)_f\right]\vec{j} + \left[\frac{1}{nB'_f}\sum_{m=1}^{linhas}\sum_{l=1}^{colunas} B'(m,l)_f\right]\vec{k} \quad (3.11)$$

em que nas Equações (3.10) e (3.11), $nR', nG', nB'$ denotam o número total de pixels dentro e fora do contorno inicial para cada mapa de cor, e $\vec{A}, \vec{B}$ são os vetores de cor. A imagem mapeada (*R', G', B'*), claramente define a região da pupila com redução de ruídos e pode ser observada na Figura 3.88b. Por fim, a imagem mapeada (*R', G', B'*) é binarizada. A Figura 3.88c demonstra o resultado da binarização com a detecção de bordas de Canny, onde é possível observar a circunferência da pupila destacada.

# 4 APRENDIZADO DE MÁQUINA

Aprendizado de máquina é uma área da ciência da computação que se preocupa com o desenvolvimento de algoritmos que permitem com que os computadores desenvolvam comportamentos baseados em dados coletados de sensores ou banco de dados. (PRABHU e VENATESAN, 2007) (MITCHELL, 1997). Através de amostras dos dados o aprendizado divide-se em supervisionado, não supervisionado ou por reforço.

No aprendizado supervisionado o algoritmo recebe um conjunto de dados de treinamento para os quais os rótulos da classe associada são conhecidos. Cada exemplo é descrito por um vetor com os atributos de entrada e pela saída associada. No aprendizado supervisionado o objetivo é construir um classificador que possa determinar corretamente a classe de novos exemplos ainda não rotulados.

Para o aprendizado não supervisionado o algoritmo recebe amostras não identificadas, os métodos de agrupamento de dados aplicam um processo de decisão de modo a agrupar amostras que apresentam características semelhantes e separar aquelas com características distintas. Em imagens digitais com o processo de agrupamento de dados é possível formar conjuntos distintos de pixels da imagem. Para os algoritmos de agrupamento de dados aplicados a imagens com o olho, a características mais importantes é a sua capacidade de classificar, na mesma classe, todos os pixels pertencentes à íris e todos os demais em uma classe diferente (PROENÇA e ALEXANDRE, 2006).

A seguir é descrito uma rede neural artificial do tipo *perceptron* múltiplas camadas que emprega aprendizado supervisionado e em seguida os métodos de aprendizado não supervisionado empregados nesta dissertação.

## 4.1 REDES NEURAIS ARTIFICIAIS

Redes Neurais Artificiais são uma implementação de um programa de computador ou circuito eletrônico de forma paralela e distribuída. Elas são compostas por unidades de processamento simples (neurônios artificiais) com funções matemáticas que podem ser lineares ou não lineares, em que as conexões entre estes neurônios (sinapses) são compostas por pesos que representam o conhecimento desta rede, e são determinados através de um processo de aprendizagem.

### 4.1.1 NEURÔNIO BIOLÓGICO

A Figura 4.1 ilustra-se de uma forma simplificada os componentes de um neurônio biológico.

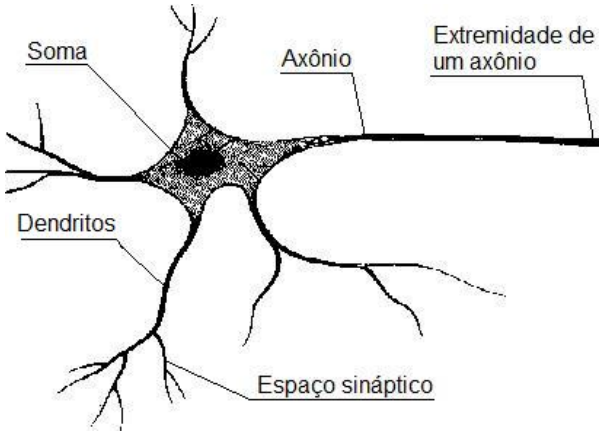



Figura 4.1 – Neurônio biológico (HAYKIN, 2008).

Os sinais procedentes dos neurônios pré-sinápticos são transmitidos para o corpo do neurônio pós-sináptico, que são comparados com os outros sinais recebidos pelo mesmo. Se o percentual de excitação do neurônio é suficientemente alto então ele envia um sinal que é captado nos neurônio seguinte.

### 4.1.2 NEURÔNIO ARTIFICIAL

McCulloch e Pitts (1943) propuseram um modelo simplificado de um neurônio artificial, conforme ilustrado na Figura 4.2. As suas entradas ou dendritos (comparando com um neurônio biológico) recebem um conjunto de varáveis aqui indicadas como $x1$, $x2$,..., $xn$, e que representam a ativação dos neurônios anteriores (pré-sinápticos), e uma saída $y$ representando o comportamento de um axônio. Para representar o comportamento das sinapses, as entradas possuem pesos acoplados $w1$, $w2$,..., $wn$, com valores positivos ou negativos.

A ativação deste neurônio é determinada através de uma função de ativação que recebe como entrada a soma ponderada $xi$ $wi$. Se esta função receber um valor maior que um

determinado limiar, o neurônio é ativado produzindo como saída “1”, caso contrário, a saída é “0” indicando que ele não foi ativado.

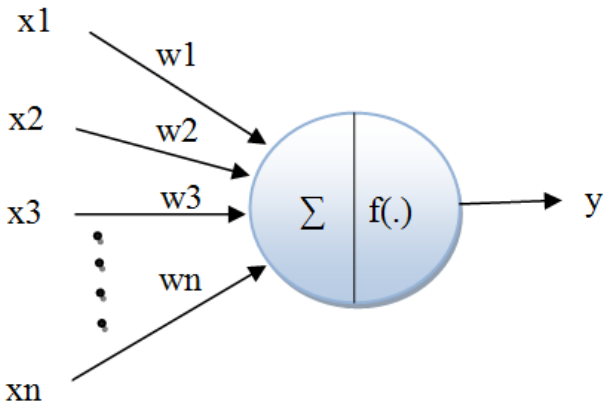



Figura 4.2 – Neurônio artificial.

### 4.1.3 FUNÇÃO DE ATIVAÇÃO

A função de ativação é responsável por determinar a saída *y* de um neurônio através dos valores de entrada *x* = (*x*1, *x*2, *x*3,..., *xn*) e dos valores dos pesos *w* = (*w*1, *w*2, *w*3,..., *wn*). Na versão de múltiplas camadas (MLP – *Multi Layer Perceptron*) como apresentado na seção (4.1.4) , sua função de ativação é a degrau, porém outros tipos de funções com propriedade não-linear são também utilizadas:

- Função de grau

$$v = \sum_{i=1} xi.wi \qquad f(v) = \begin{cases} 0, v < 0 \\ 1, v \geq 0 \end{cases} \tag{4.1}$$

- Função logística, onde “a” representa a inclinação da função, os resultados obtidos na saída desta função são valores no intervalo de {0,1}.

$$v = \sum_{i=1} xi.wi \qquad f(v) = \frac{1}{1 + e^{-av}} \tag{4.2}$$

- Função tangente hiperbólica, para representar valores no intervalo de {-1,+1}.

$$v = \sum_{i=1} xi.wi \qquad f(v) = \frac{1 - e^{-av}}{1 + e^{-av}} \tag{4.3}$$

### 4.1.4 PERCEPTRON

O Perceptron é uma rede neural de única camada introduzida por Rosenblatt (1957) com a proposta para reconhecimento de padrões. Rosenblatt também propôs um algoritmo para o ajuste dos pesos dos neurônios e provou sua convergência para padrões linearmente separáveis.

O Perceptron recebeu significativas críticas na publicação de Minsky & Papert (1969) devido à sua limitação de solucionar unicamente problemas lineares, causando impacto no interesse dos estudos de redes neurais artificiais, levando à abnegação da área durante a década de 70. Este desinteresse na comunidade científica mudou com a publicação de Hopfield (1982) e do método de retro-propagação para ajuste dos pesos em 1986.

A não linearidade é introduzida em redes neurais artificiais através de funções de ativação não linear e a adição de camadas intermediárias na estrutura da rede neural, denominada de perceptron de múltiplas camadas (do inglês *Multi Layer Perceptron* – MLP) – Figura 4.3.

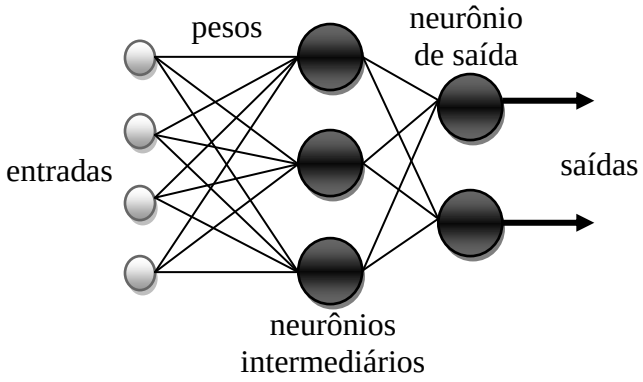



Figura 4.3 – Representação de uma rede com múltiplas camadas.

O treinamento das redes MLP é através do aprendizado supervisionado, com o método de retro-propagação (do inglês *back-propagation*). O algoritmo tem como objetivo encontrar uma função que permita classificar corretamente todas as amostras contidas no conjunto de treinamento. Primeiro calcula-se a saída da rede, obtendo *y*, em seguida são corrigidos os pesos sinápticos. O principal algoritmo para minimização do erro é o do gradiente descendente, porém, na literatura podem-se encontrar outros algoritmos com uma maior eficiência computacional, como o *QuickProp* proposto por (FAHLMAN, 1988), que considera a superfície do erro sendo localmente quadrática e procura saltar da posição atual na superfície para o ponto de mínimo da parábola.

## 4.2 K-*MEANS*

K-*Means* é um algoritmo de agrupamento de dados que consiste em particionar *n* observações em *k* grupos, em que cada observação pertence a um grupo mais próximo a ele. O objetivo do algoritmo é minimizar a variação dos vetores de atribuídos a cada cluster. O algoritmo é iterativo. Após cada vetor ser atribuído a um dos clusters, os centros dos clusters são recalculados e os vetores são reatribuídos usando os novos centros de cluster. A equação a seguir descreve a função a ser minimizada:

$$J = \sum_{j=1}^{k} \sum_{x_j \in S_i}^{i} (x_j - \mu_i)^2 \tag{4.4}$$

onde, $(x_j - \mu_i)^2$ é a medida de distância escolhida entre o ponto $x_j$ e o centro do cluster $\mu_i$. O algoritmo k-*means* é composto dos seguintes passos.

1. Colocam-se *k* pontos no espaço representado pelos objetos que estão sendo agrupados. Estes pontos representam o grupo inicial dos centróides.
2. Atribuir para cada grupo o objeto que tem o centróide mais próximo a ele.
3. Com todos os objetos distribuídos, recalcula a posição dos *k* centroides.
4. Repetir as etapas 2 e 3 enquanto os objetos estiverem se movendo. Desta forma os objetos são separados em grupos, podendo calcular a métrica de minimização.

O algoritmo k-*means* foi primeiramente proposto por MacQueen (MACQUEEN, 1967) e vem sendo utilizado em trabalhos de segmentação de imagens. Wang propõe a utilização do k-*means* para diminuir a quantidade de cores de imagens de satélites para análise (WANG e LEESER, 2007). Proença utilizou k-*means* na binarização de imagens com o olho, para separar as cores da pupila das demais na imagem (PROENÇA e ALEXANDRE, 2006). Em um trabalho mais recente, mostra-se que a binarização de Otsu é equivalente ao k-*means* (DONGJULIU e JIANYU, 2009).

## 4.3 MAPAS AUTO-ORGANIZÁVEIS

Os mapas auto-organizáveis (do inglês *Self-Organization Map* - SOM) é um tipo de Rede Neural Artificial que é treinada usando aprendizagem não supervisionada e foi desenvolvido por Teuvo Kohonen (KOHONEN, 1989).

Em um mapa auto-organizável, os neurônios estão colocados em nós de uma grade que é uni ou bidimensional (Figura 4.4). Os neurônios se tornam seletivamente sintonizados a vários padrões de entrada ou classes de padrões de entrada no decorrer de um processo de aprendizado (HAYKIN, 2008).

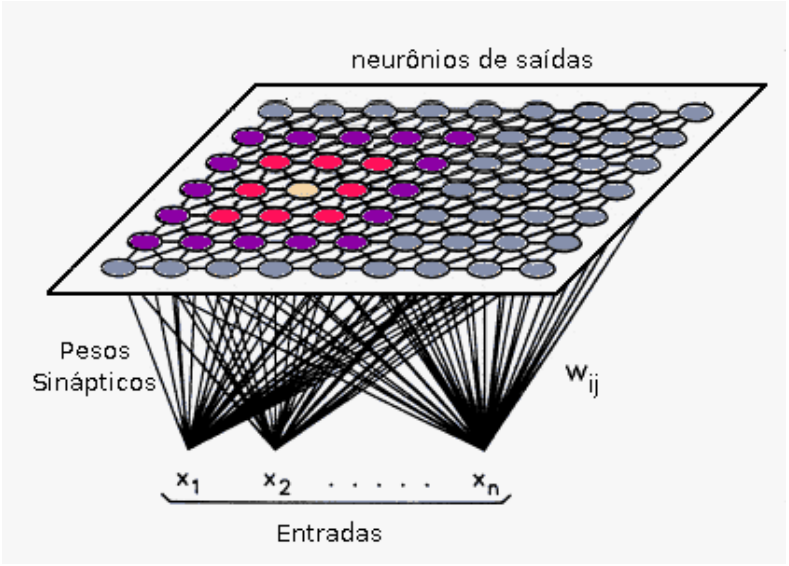



Figura 4.4 – Arquitetura bidimensional dos mapas auto-organizáveis (HAYKIN, 2008).

O principal objetivo do mapa auto-organizável é transformar um padrão de sinal incidente de dimensão arbitrária em um mapa discreto unidimensional (UETA, SUMI, *et al.*, 2006).

Os mapas auto-organizáveis são baseados em três fases:

- **Competição**: Para um padrão de entrada, os neurônios da grade calculam seus respectivos valores de uma função discriminante fornecendo a base para a competição entre os neurônios. O neurônio com o maior valor função discriminante é declarado vencedor. A forma mais usual de se obter este valor é através da distância euclidiana do vetor de entrada aos neurônios.

- **Cooperação**: O neurônio vencedor determina a localização espacial de uma vizinhança topológica de neurônios excitados, fornecendo assim a base para a cooperação entre os neurônios vizinhos. É necessário então encontrar uma forma de obter um valor que represente essa proximidade de vizinhança de um neurônio ativado. Desse modo, define-se a função de vizinhança $h_{ij}$, onde o índice *i*, refere-se ao neurônio vencedor e o índice *j* aos outros neurônios da grade. Uma escolha típica de $h_{ij}$ que satisfaz estas exigências é a função gaussiana:

$$h_{ij} = \exp\left(\frac{d_{ij}^2}{2\sigma^2}\right) \tag{4.5}$$

  onde, $d_{ij}^2$ é a distância euclidiana entre os neurônios considerados e o $\sigma$ é o parâmetro de largura da vizinhança topológica.

- **Adaptação**: Permite que os neurônios excitados aumentem seus valores individuais da função discriminante em relação ao padrão de entrada através de ajustes adequado aplicado a seus pesos sinápticos com base na seguinte expressão:

$$\Delta w_j = \eta h_{ij}(x - w_j) \tag{4.6}$$

  Onde, $\Delta w_j$ é o termo de atualização dos pesos do neurônio *j*, $\eta$ é a taxa de aprendizagem que pode variar com o tempo, $h_{ij}$ é a função de vizinhança e *x* são os padrões de entrada apresentados à rede.

O algoritmo para treinamento dos mapas auto-organizáveis pode ser visto a seguir:

1. Inicializar aleatoriamente os valores do vetor de pesos.
2. Retirar uma amostra *x* do espaço de entrada com certa probabilidade, em que o vetor *x* representa o padrão de ativação que é aplicado à grade.
3. Encontrar o neurônio vencedor *i(x)* no passo de tempo *n* usando o critério da mínima distância euclidiana: $i(x) = arg_j \min\|x(n) - w_j\|$
4. Ajustar os vetores de pesos sinápticos de todos os neurônios usando a Equação 4.3.
5. Continuar com o passo 2 até que não sejam observadas modificações significativas no mapa de características.

## 4.4 AGRUPAMENTO POR VETORES-SUPORTE

Agrupamento por vetores-suporte (do inglês *Support Vector Clustering* - SVC) é um método de aprendizado não supervisionado inspirado em máquinas de vetores-suporte (do inglês *Support Vector Machine* - SVM) (VAPNIK, 1999) e aplicado para resolver problemas de agrupamento de dados. O método foi proposto por Vapnik (1999), baseado em seu outro trabalho de aprendizado estatístico (VAPNIK, 1999) com o objetivo de obter um algoritmo de agrupamento com característica não paramétrica (BEN-HUR, 2008).

A idéia principal do SVC é mapear os dados através de uma função de kernel, como por exemplo, um kernel gaussiano para um espaço de características de alta dimensão e procurar uma esfera de raio mínimo que contenha a maioria dos dados mapeados neste espaço. Quando a esfera for mapeada de volta para espaço de dados, é possível obter a sua separação em vários clusters, cada um envolvendo um conjunto distinto de dados – vide Figura 4-5. O SVC possui algumas vantagens em relação aos outros algoritmos de agrupamento, como, por exemplo, capacidade de gerar clusters de formas arbitrárias e de lidar com *outliers*, empregando uma margem suave que permite com que a esfera no espaço de características não inclua todos os pontos (LEE e LEE, 2005) e (LEE e LEE, 2006).

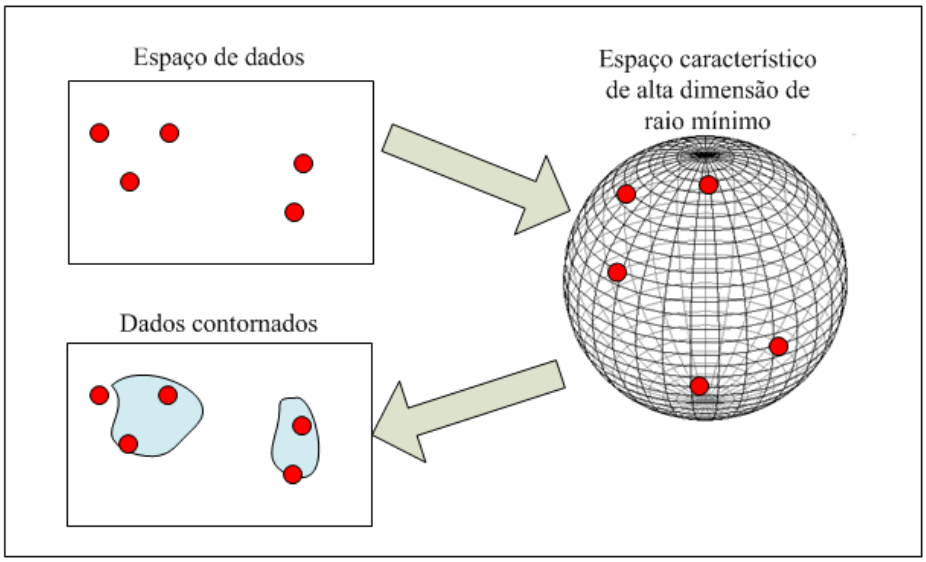



Figura 4-5 – Mapeamento para o espaço de características.

### 4.4.1 ALGORITMO SVC

Dado $\{x_i\} \subseteq x$ sendo um conjunto de dados de *N* pontos, com $x \in R^d$, a esfera de raio mínimo *R*, encapsulando todos os pontos no espaço de características pode ser descrito como:

$$\left\|\Phi\left(x_j\right) - a\right\|^2 \leq R^2 \quad \forall_j \tag{4.7}$$

em que ||.|| é a norma Euclidiana e *a* é o centro da esfera. Restrições são incorporadas adicionando a variável $\xi_j$.

$$\left\|\Phi\left(x_j\right) - a\right\|^2 \leq R^2 + \xi_j \tag{4.8}$$

Para resolver o problema de $\xi_j \geq 0$ é empregado o lagrangeano representado na Equação (4.9).

$$L = R^2 - \sum_j \left(R^2 + \xi_j - \left\|\Phi\left(x_j\right) - a\right\|^2\right)\beta_j - \sum \xi_j \mu_j + C \sum \xi_j \tag{4.9}$$

em que $\beta_j \geq 0$ *e* $\mu_j \geq 0$ são os multiplicadores de Lagrange, *C* é uma constante e $\sum \xi_j$ é um termo de penalidade. Definindo as derivadas de *L* em relação a *R*, *a* e $\xi_j$ respectivamente, tem-se:

$$\sum_j \beta_j \tag{4.10}$$

$$a = \sum_j \beta_j \Phi\left(x_j\right) \tag{4.11}$$

$$\beta_j = C - \mu_j \tag{4.12}$$

A distância de cada ponto de *x* no espaço de características ao centro da esfera é definido como:

$$R^2(x) = \left\|\Phi\left(x_j\right) - a\right\|^2 \tag{4.13}$$

Em virtude da equação quadrática e da definição de kernel, se obtém a distância como:

$$R^2(x) = K(x,x) - 2\sum_{\mathrm{j}} \beta_j K(x,x) + \sum_{ij} \beta_i \beta_j K(x_i, x_j) \tag{4.14}$$

em que $K\left(x_i, x_j\right)$ representa função kernel, geralmente é adotado a função de kernel gaussiana descrita por $K\left(x_i, x_j\right) = \exp\left(q\left\|x_i - x_j\right\|^2\right)$ e $R = \{R(x_i)\}$ o raio da esfera, sendo $x_i$ o vetor suporte. O contorno em volta dos pontos no espaço de dados é definido pelo conjunto:

$$\{x \mid R(x) = R\} \tag{4.15}$$

A Equação (4.15) descreve os pontos, chamados de vetores suporte, que estão na fronteira dos clusters. Já os pontos que possuem multiplicador de lagrange igual a C são chamados de vetores suporte limitados (do inglês *Bounded Support Vectors* - BSV), estes residem fora da esfera e os demais pontos estão dentro da esfera.

### 4.4.2 ATRIBUIÇÃO DOS GRUPOS

O algoritmo para a descrição dos clusters não diferencia pontos que estão entre os diferentes clusters. Para isto, é usada uma abordagem geométrica envolvendo *R(x)* baseada na seguinte observação: dado um par de dados que pertençam a diferentes clusters, qualquer caminho que conectá-los deve sair da esfera no espaço de características. Assim, o caminho contém um segmento de pontos *y* tal que *R(y) > R*. Isto leva à definição da matriz adjacente $A_{ij}$ entre o par de pontos $x_i$ e $x_j$ cujas imagens encontram-se na esfera do espaço de características:

$$A_{ij} = f(x) = \begin{cases} 1, & R\left(x_i + \lambda\left(x_j - x_i\right)\right) \leq R, \forall\, \lambda \in [0,1] \\ 0, & para\ os\ demais \end{cases} \tag{4.16}$$

Os clusters podem agora ser definidos como componentes conectados por um grafo induzido por *A*. Os vetores suporte limitados não são classificados através deste processo uma vez que suas imagens do espaço de características encontram-se fora da esfera. Pode-se decidir em mantê-los como não classificados ou atribuí-los a um cluster mais próximo.

## 4.5 COMITÊ DE MÁQUINAS

Comitê de máquinas agrega, de alguma forma, o conhecimento adquirido pelos componentes para chegar a uma solução global que é supostamente superior àquela obtida por qualquer um dos componentes isolados. Comitês de máquinas são aproximadores universais (HAYKIN, 1999) e podem se apresentar em versões estáticas, denominadas ensembles, ou em versões dinâmicas, denominadas misturas de especialistas. Esta dissertação tem como foco somente as estruturas estáticas.

Ensemble é o processo pelo qual vários modelos são estrategicamente gerados e combinados para resolver um problema particular, é utilizado para melhorar o desempenho de

um modelo ou reduzir a probabilidade de uma escolha indevida (KUNCHEVA, HADJITODOROV e TODOROVA, 2006).

Ensemble vem sendo utilizados em diversas áreas da inteligência computacional, como o método proposto por (BAKKER e HESKES, 2003) para combinar resultados de classificação com redes neurais artificiais. Na área de visão computacional, para segmentação de imagens, Lin emprega ensembles com k-*means* para obter uma face segmentada através da combinação das partições obtidas com o k-*means* (LIN e LIAO, 2007). Chang-ming utiliza ensemble para segmentar imagens de ultrassonografia, combinando o resultado do agrupamento de cores da imagem (MING, CHANG, *et al.*, 2008). Alguns estudos recentes mostraram também que ensembles para agrupamento de dados podem apresentar-se como um método robusto e estável (FERN e BRODLEY, 2003).

Um ensemble para partições consiste em analisar um conjunto de resultados de algoritmos de agrupamento e em seguida combiná-los usando uma função de consenso, para criar a partição final, que é considerada a abranger todas as informações contidas no conjunto. Os componentes de um ensemble podem ser obtidos utilizando diferentes técnicas, empregando diferentes algoritmos.

### 4.5.1 FUNÇÃO DE CONSENSO

O objetivo da clusterização é identificar um grupo de padrões em um conjunto de dados que são rotulados de acordo com as suas similaridades (HE, XU e DENG, 2005).

Seja $X = \{x_1, x_2, x_3, \dots, x_n\}$ representando um conjunto de dados, a partição destes *n* objetos em *k* clusters pode ser representado como um conjunto de *k* objetos $C_l = \{l = 1, \dots, k\}$ ou como um vetor de rótulos $\lambda \, \epsilon \, N^n$. A função $\phi$ atribui os rótulos para um conjunto de objeto. O conjunto de rótulos $\boldsymbol{\lambda}^{(\mathbf{1},\mathbf{2},\dots,\boldsymbol{r})}$ gerados são combinados em um único vetor de rótulo $\boldsymbol{\lambda}$ através da função de consenso Γ (figura 5.2) (STREHL e GHOSH, 2002) (HE, XU e DENG, 2005).

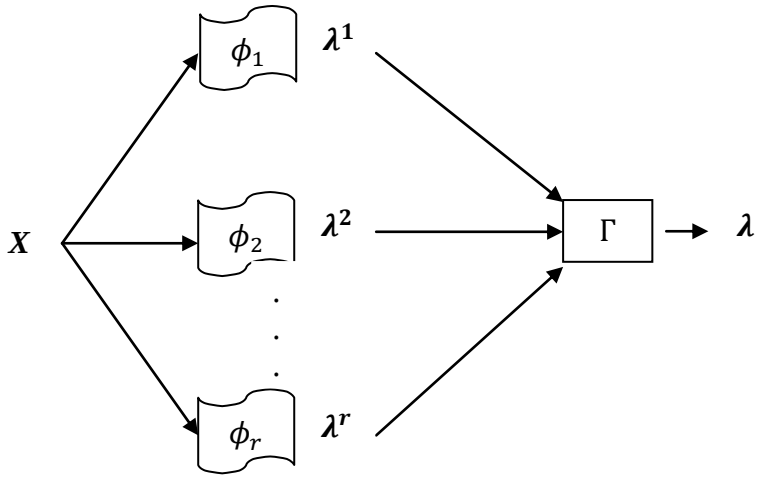



Figura 4-6 – Função de consenso.

Existem diversas técnicas para produzir um único vetor de rótulo $\boldsymbol{\lambda}$, entre elas inclui *simulated annealing* e algoritmo genético que podem resultar em uma solução razoável, porém com um alto custo computacional (STREHL e GHOSH, 2002). Estratégias com desempenho superior é proposta por Strehl e Ghosh (2002) a fim de se obter uma combinação de diferentes clusters e são descritas abaixo.

### 4.5.2 ALGORITMO BASEADO EM CLUSTERS DE PARTIÇÕES SIMILARES

O algoritmo baseado em clusters de partições similares (CSPA - *Cluster-based Similarity Partitioning Algorithm*) cria uma matriz de similaridade para cada componente, onde o elemento (*i, j*) será *1* se o objeto *i* e *j* pertencem ao mesmo cluster e *0* caso contrário. Em seguida, calcula a matriz de similaridade global. Baseado nesta matriz, constrói-se o grafo correspondente. Finalmente, utiliza-se um algoritmo de particionamento de grafo, como por exemplo o METIS (*Multilevel graph partitioning algorithm*), para gerar o agrupamento final. O algoritmo pode ser sumarizado da seguinte forma:

(i) Para cada componente *i* construa a matriz de similaridade;

(ii) Em seguida construa a matriz de similaridade global. Os elementos (*i ,j*) desta matriz representa a fração dos componentes, nos quais os objetos *i* e *j* são atribuídos ao mesmo cluster;

(iii) Baseado nesta matriz de similaridade constrói-se um grafo *G = (V, E)*. *V* contém *N* vértices cada um representando um objeto. Os pesos das arestas que liga os objetos *i* e *j* é dado pela matriz de similaridade global.

(iv) Particiona-se o grafo usando, por exemplo, o algoritmo METIS para produzir os agrupamentos finais.

### 4.5.3 ALGORITMO DE META-CLUSTERIZAÇÃO

O algoritmo, de meta-clusterização (MCLA - *Meta-Clustering Algorithm*) é baseado na clusterização de grupo de clusters (meta clusters). Os grupos formados em diferentes partições do conjunto de dados podem conter o mesmo conjunto de objetos ou um grande número de objetos compartilhados e que podem ser considerados similares entre si. Com base nestas informações o algoritmo constrói um grafo que modela o relacionamento entre os diferentes grupos pertencentes às diferentes partições. O algoritmo pode ser formalizado como:

(i) Constrói-se um grafo G = (V, E) em que V é composto por $K_t$ vértices, representando cada objeto do cluster.

(ii) O peso de similaridade das arestas entre dois clusters $C_i^m$ e $C_j^n$ é calculado usando a medida de Jaccard – equação (4.17).

$$\frac{\left|C_i^m \cap C_j^n\right|}{\left|C_i^m \cup C_j^n\right|} \tag{4.17}$$

(iii) Emprega-se o algoritmo de METIS para particionar o grafo. Cada cluster resultante tem o seu valor associado para cada objeto, relacionando a semelhança entre eles. O cluster final é obtido associando para cada objeto do meta-cluster com o maior valor associado a ele.

### 4.5.4 ALGORITMO DE PARTIÇÃO HIPERGRAFO

O algoritmo de partição de hipergrafo (HGPA - *HyperGraph-Partitioning Algorithm*) é uma abordagem para ensemble de cluster que re-particiona os dados usando a informação do cluster como indicações da força do vinculo entre os objetos. O problema de ensemble de cluster é formulado como particionamento de um hipergrafo pelo corte do número mínimo de

hiper-arestas. Considera-se que todas as hiper-arestas possuem um mesmo peso e que todos os vértices são ponderados igualmente. Nota-se que isto inclui *n*-forma de relacionamento das informações, enquanto que o CSPA apenas considerava pares de relacionamentos. O algoritmo HGPA busca por uma hiper-arestas que particiona um hipergrafo em *k* componentes desconectados de aproximadamente do mesmo tamanho. Para particionar o hipergrafo Strehl e Ghosh (2002) utilizaram o algoritmo HMETIS (*Hypergraph Multilevel graph partitioning algorithm*) que é uma extensão do METIS.

# 5 SIMULAÇÕES E RESULTADOS

Nos experimentos realizados, foi utilizada a base de dados UBIRIS.V1 (PROENÇA e ALEXANDRE, 2006), que é composta de 1877 imagens (200x150 pixels - 24 cores por bit) do olho de 241 pessoas. Uma importante característica destas imagens é que são imagens com ruído, obtidas em ambientes não controlados, permitindo assim, uma melhor avaliação dos métodos de segmentação da íris quanto à robustez. As imagens desta base de dados são divididas em duas sessões, a saber:

i. Na primeira sessão, as imagens possuem menos ruído, principalmente devido a reflexos, luminosidade e contraste.
ii. Já na segunda sessão, as imagens são capturadas com luminosidade natural, proporcionando o aparecimento de imagens heterogêneas com relação à reflexão, contraste, luminosidade e foco.

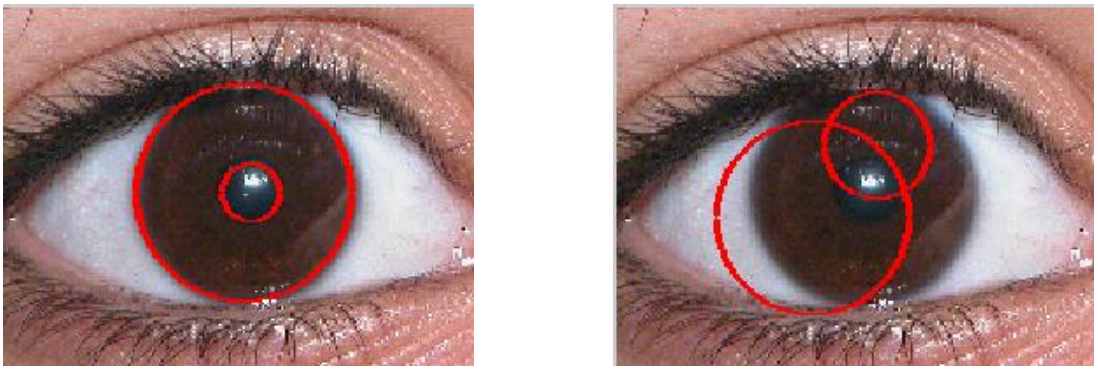
Todas as imagens de ambas as sessões são classificadas em relação aos parâmetros foco, reflexo e visibilidade da íris em três escalas de valores (bom, médio e ruim). Na Tabela 5.1, são apresentados os grupos das imagens de acordo com suas características.

Tabela 5.1 – Caracterização da base de dados de íris (PROENÇA e ALEXANDRE, 2006)

| Parâmetro | Bom | Médio | Ruim |
|---|---|---|---|
| Foco | 73,83% | 17,53% | 8,64% |
| Reflexo | 58,87% | 36,78% | 4,34% |
| Visibilidade Íris | 36,73% | 47,83% | 15,44% |

## 5.1 DESCRIÇÃO DOS EXPERIMENTOS

A segmentação da íris consiste em buscar numa imagem, com um olho, a circunferência da íris e da pupila. Em seguida extrair apenas a imagem da íris necessária para o processo de classificação e posterior identificação de um indivíduo (DAUGMAN, 1993). Os resultados são analisados visualmente nas imagens segmentadas. Considera-se acerto quando a circunferência em vermelho reproduzir a circunferência da pupila e a da íris (Figura 5-1a) e erro quando a circunferência em vermelho não corresponder à circunferência da íris ou da pupila (Figura 5-1b).



(a) Imagem corretamente segmentada (b) Imagem incorretamente segmentada

Figura 5-1 – Análise da imagem segmentada.

Foram realizados três experimentos distintos, conforme descrito abaixo:

1) O experimento # 1 consiste em buscar as circunferências da íris e da pupila empregando somente técnicas de processamento de imagem digital.
2) No experimento # 2 foram adotados os algoritmos de agrupamento de dados descrito no capítulo anterior (k-*means*, mapas auto-organizáveis e SVC). Este experimento tem como objetivo extrair o olho da imagem em diferentes grupos de cores. Após a extração, emprega-se uma estratégia de combinação (ensemble) dos resultados obtidos, utilizando alguns dos algoritmos descritos no capitulo anterior.
3) No experimento # 3 foi investigado o emprego de aprendizado supervisionado utilizando Redes Neurais Artificiais, de forma a reduzir a área a ser pesquisada pela transformada de Hough para buscar a circunferência da íris. Com isto, almeja-se uma redução no esforço computacional ao se aplicar a transformada de Hough.

## 5.2 EXPERIMENTO # 1

A Figura 5.2 ilustra a seqüência de passos e o resultado produzido, numa determinada imagem, empregada para localizar a circunferência da íris com a transformada de Hough (HOUGH, 1962). Após a aquisição da imagem, esta é transformada em tons de cinza e em seguida o histograma da imagem é equalizado a fim de se obter um maior contraste entre a íris e a esclerótica. Posteriormente, a imagem é binarizada com o método de Otsu (OTSU, 1979) que procura obter o melhor limiar para separar a íris da esclerótica. Em seguida, é realizada a detecção de borda com o operador de Canny (CANNY, 1987). A imagem é dilatada empregando técnicas de morfologia matemática com o objetivo de melhorar o desempenho da transformada de Hough.

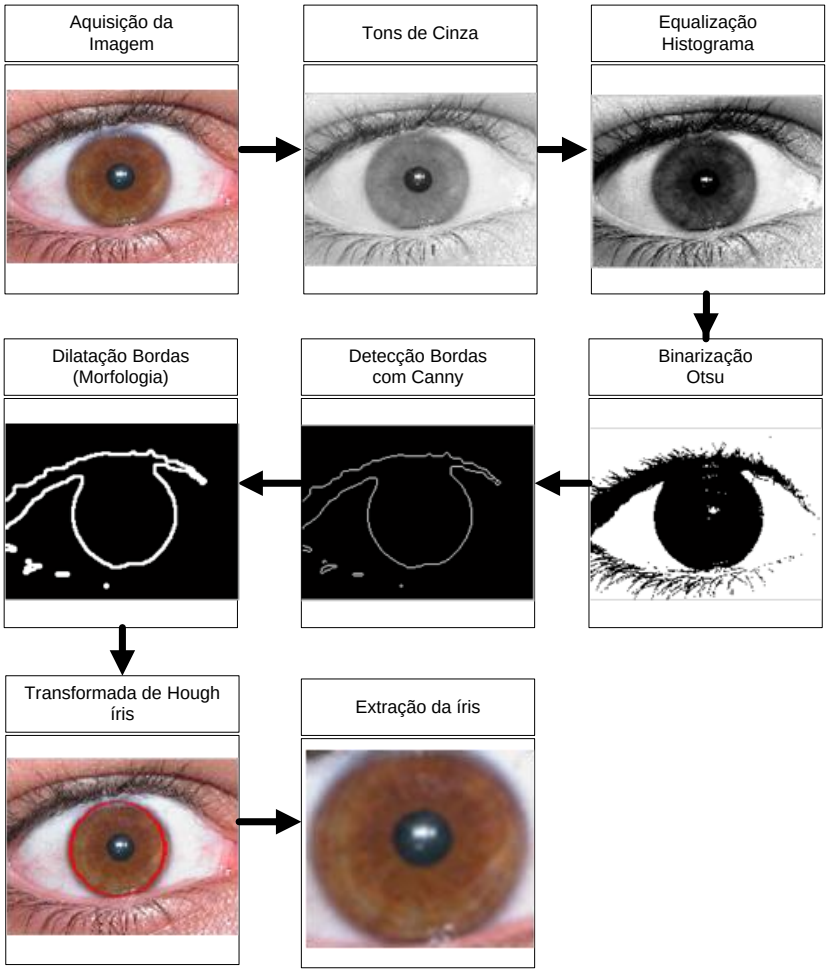



Figura 5.2 – Seqüência para segmentação da íris com processamento digital de imagem.

Aplicando a seqüência de passos descritos, foi possível obter 97% das duas sessões das íris segmentadas. Somente nas imagens em que a pálpebra ocultava parte ou totalmente a íris (Figura 5.3) não foi possível localizar a circunferência da íris.

Figura 5.3 – Imagens com a íris oculta.

Em imagens com baixo contraste da íris em relação à esclerótica ou em que parte da íris é ocultada (Figura 5.4) o método se mostrou robusto conseguindo obter uma imagem segmentada e conseqüentemente o contorno da íris.

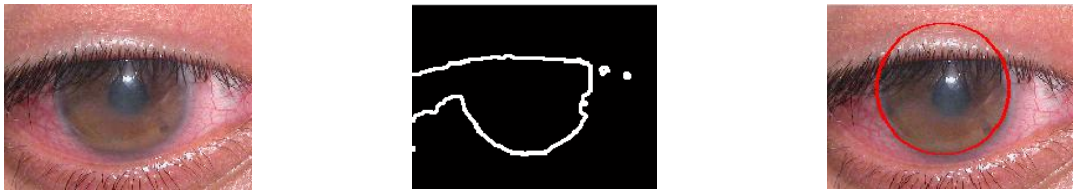


Figura 5.4 – Imagem com a íris segmentada.

Após a localização da circunferência da íris, esta é então extraída e gera-se uma nova imagem. Esta imagem é usada para realizar a busca pela pupila, cujo objetivo é separar a pupila e a íris em grupos distintos. A Figura 5.5 apresenta a seqüência de passos empregada para localizar a circunferência da pupila.

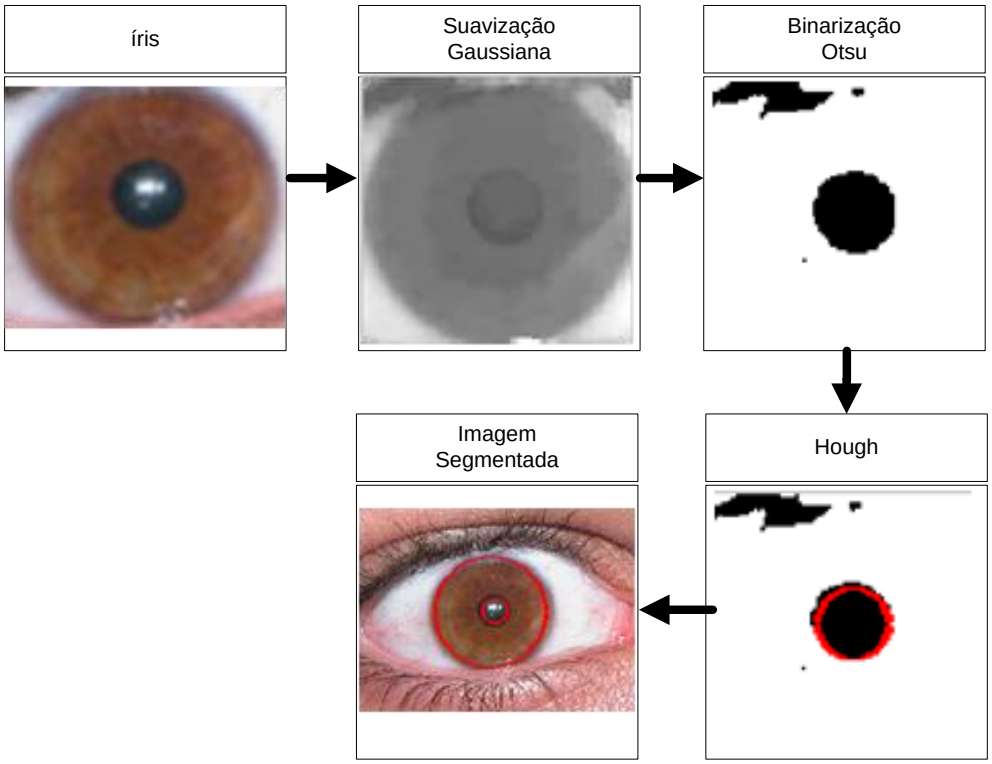


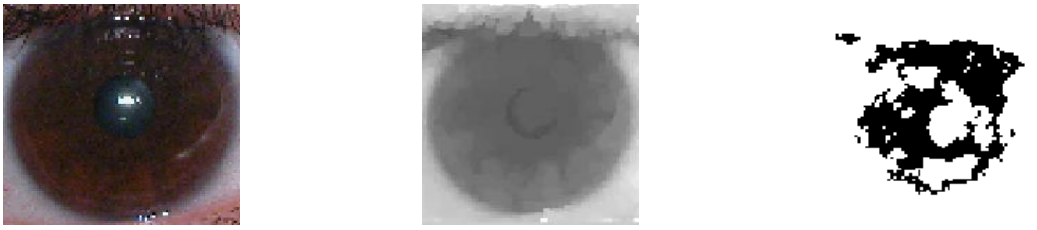

Figura 5.5 – Seqüência para segmentação da pupila com processamento de imagem digital.

Iniciando-se com a imagem composta apenas pela íris e pupila, esta é transformada em tons de cinza e aplica-se uma suavização gaussiana a fim de eliminar ruídos da imagem e obter uma imagem suavizada. Em seguida, a imagem é binarizada empregando o método de Otsu (OTSU, 1979) com o objetivo de separar a pupila do restante da imagem e, então, localizar a sua circunferência através da transformada de Hough (HOUGH, 1962). Por fim, se obtém a imagem final segmentada. A Tabela 5.2 apresenta o resultado final obtido para o

experimento #1, em que foi possível segmentar 90,83% das imagens da primeira sessão e 87,44% da segunda sessão.

Tabela 5.2 – Resultado com Processamento de Imagem Digital

| | Sessão 1 | Sessão 2 |
|---|---|---|
| Descrição | Segmentado | Segmentado |
| Hough | 90,83% | 87,44% |



(a) Original (b) Tons de Cinza (c) Binarizada

Figura 5-6 – Imagem não segmentada com Processamento de Imagem Digital.

Para imagens com baixo contraste entre a pupila e a íris (Figura 5-6a), a binarização resultou em imagens ruidosas (Figura 5-6c), não sendo possível aplicar a transformada de Hough para encontrar alguma circunferência na imagem.

Tabela 5.3 – Tabela comparativa com outros métodos da literatura (experimento #1)

| | Sessão 1 | Sessão 2 |
|---|---|---|
| Método | Segmentado | Segmentado |
| Daugman | 95,79% | 91,10% |
| Camus e Wildes | 96,78% | 89,29% |
| Kheirolahy | 98,00% | 96,38% |
| Proposta | 90,83% | 87,44% |

A Tabela 5.3 apresenta o resultado obtido por outros métodos reportados na literatura (PROENÇA e ALEXANDRE, 2006). Comparando-os com o proposto nesta dissertação, é possível observar que a proposta neste experimento é inferior ao demais em relação à taxa de acerto na segmentação das imagens. Visando melhorar o desempenho na segmentação, nos próximos experimentos foram empregadas técnicas de eliminação de ruído na imagem.

## 5.3 EXPERIMENTO # 2

O objetivo deste experimento é obter um conjunto distinto de dados extraídos de uma imagem para que os pixels que se encontram na pupila sejam rotulados diferentemente dos demais da imagem. A Figura 5.7 ilustra os passos para segmentar a íris empregando agrupamento de dados.

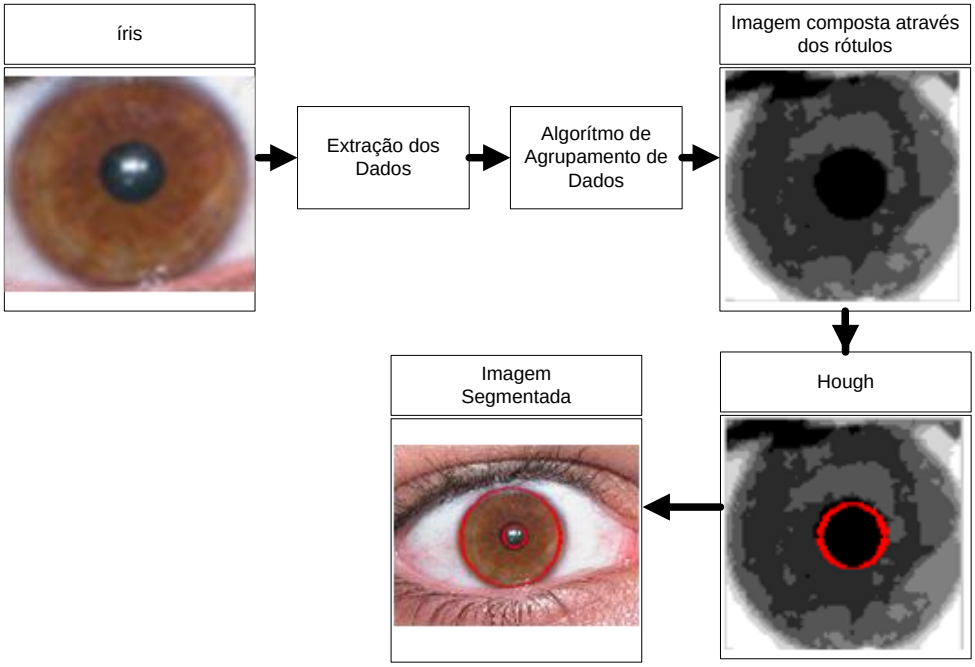



Figura 5.7 – Seqüência para segmentação com agrupamento de dados.

Após a aquisição da imagem composta apenas pela íris, são extraídos os dados da imagem que serão empregados como entrada para os algoritmos de clusterização. O conjunto de dados é composto por vetores que são formados pelas coordenadas do pixel e a cor no formato RGB. Assim, de uma imagem *f(M, N)* se obtém um conjunto de dados com $M \times N$ linhas, e cada linha é representada por um vetor composto de $\{X, Y, R, G, B\}$, em que:

- *X* corresponde à coluna do pixel.
- *Y* corresponde à linha do pixel
- *R* corresponde à cor na faixa do vermelho ($0 \leq R \leq 255$)
- *G* corresponde à cor na faixa do verde ($0 \leq G \leq 255$)
- *B* corresponde à cor na faixa do azul ($0 \leq B \leq 255$)

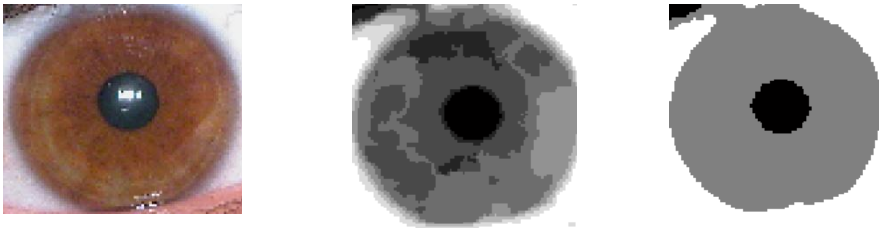
Em seguida, sobre o conjunto obtido são aplicados os algoritmo de aprendizado de máquinas descritos na seção 4 (k-*means*, agrupamento por vetores suporte e mapas auto-organizáveis). Em que cada algoritmo empregado irá atribuir um rótulo para cada pixel, formando *k* grupos. Como os algoritmos possuem inicialização aleatória, com exceção dos agrupamento por vetores suporte, a atribuição dos rótulos não é a mesma em todas as execuções. Desta forma, adotou-se um procedimento que após a atribuição dos rótulos, os rótulos que estão mais distantes da origem em termos de distância euclidiana são atribuídos o valor *k* e aqueles de menor distância é atribuído o rótulo 1. Em seguida, aplica-se uma função consenso para obter o resultado do cluster de cada pixel. Nesta dissertação será empregado o algoritmo HGPA descrito no Seção 4.5.1.

Com a resposta do agrupamento, é formada uma nova imagem em tons de cinza composta pela quantidade de cores igual ao número de grupos obtidos. Por fim é aplicada a transformada de Hough para localizar a circunferência da pupila que se destacou na imagem através da rotulação.

Tabela 5.4 – Resultado com k-*means*

| | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|
| Clusters | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| k = 02 | 1042 | 172 | 83,49% | 547 | 116 | 78,79% |
| k = 03 | 1093 | 121 | 88,93% | 571 | 92 | 83,89% |
| k = 04 | 1093 | 121 | 88,93% | 567 | 96 | 83,07% |
| k = 05 | 1108 | 106 | 90,43% | 584 | 79 | 86,47% |
| k = 06 | 1119 | 95 | 91,51% | 592 | 71 | 88,01% |
| k = 07 | 1119 | 95 | 91,51% | 598 | 65 | 89,13% |
| k = 08 | 1142 | 72 | 94,06% | 609 | 54 | 91,85% |
| k = 09 | 1106 | 108 | 90,24% | 587 | 76 | 87,05% |
| k = 10 | 1102 | 112 | 89,84% | 587 | 76 | 87,05% |
| k = 11 | 1085 | 129 | 88,11% | 568 | 95 | 83,27% |
| k = 12 | 1076 | 138 | 87,17% | 561 | 102 | 81,82% |

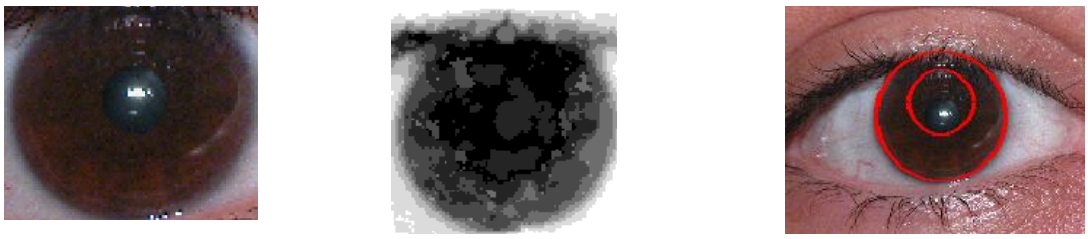
Utilizando o algoritmo k-*means* foi necessário indicar a priori quantidade de grupos que se deseja obter. O número de grupos foi variado de 2 até 12. Valores acima de 12 produziram os resultados insatisfatórios, portanto, não apresentados nesta dissertação. A Tabela 5.4 apresenta o resultado do experimento empregando o algoritmo k-*means*. Quando o número de grupos foi atribuído igual a 8 se obteve o melhor resultado. A Figura 5.8 ilustra imagens geradas quando o número de grupos foi igual a 8 (Figura 5.8b) e 3 (Figura 5.8c).



(a) Original (b) 8 grupos (c) 3 grupos

Figura 5.8 – Exemplo de agrupamento com k-*means*.

Em imagens com baixo contraste da pupila em relação à íris, como a da Figura 5.9a, o k-*means* não conseguiu rotular os pixels da pupila diferentemente dos outros pixels, não sendo possível a transformada de Hough localizar a circunferência da pupila na imagem resultante (Figura 5.9b), resultando em um erro de segmentação (Figura 5.9c).



(a) original (b) 8 grupos (c) falha na segmentação

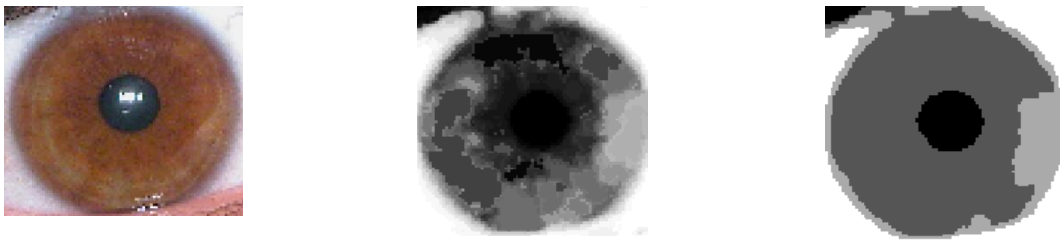
Figura 5.9 – Imagem ruidosa obtida com o k-*means* parametrizado com oito grupos ($k$=8).

Com os mapas auto-organizáveis, diferentemente do k-*means*, o número de grupos é determinado automaticamente. Entretanto, é necessário especificar alguns parâmetros, tais como: taxa de aprendizado, número de épocas para treinamento e o número de neurônios. Para a determinação dos valores destes parâmetros, realizaram-se algumas simulações com um número reduzido de imagens com características distintas. Valores entre 0,001 a 0,01 para a taxa aprendizado e valores entre 10 e 100 para o número de épocas, não influenciam na qualidade do resultado da segmentação. Sendo necessário especificar apenas o número de neurônios. Este foi variado no intervalo de dois a doze neurônios, sendo que valores maiores que estes produziram resultados insatisfatórios. A Tabela 5.5 apresenta o resultado com mapas auto-organizáveis sendo que o melhor resultado foi obtido com seis neurônios.

Tabela 5.5 – Resultado com mapas auto-organizáveis

| | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|
| Número Neurônios | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 02 x 02 | 1156 | 58 | 94,98% | 609 | 54 | 91,13% |
| 04 x 04 | 1169 | 45 | 96,15% | 601 | 62 | 89,68% |
| 06 x 06 | 1175 | 39 | 96,68% | 618 | 45 | 92,72% |
| 08 x 08 | 1071 | 143 | 86,65% | 558 | 105 | 81,18% |
| 10 x 10 | 1013 | 201 | 80,16% | 551 | 112 | 79,67% |
| 12 x 12 | 997 | 217 | 78,23% | 536 | 127 | 76,31% |

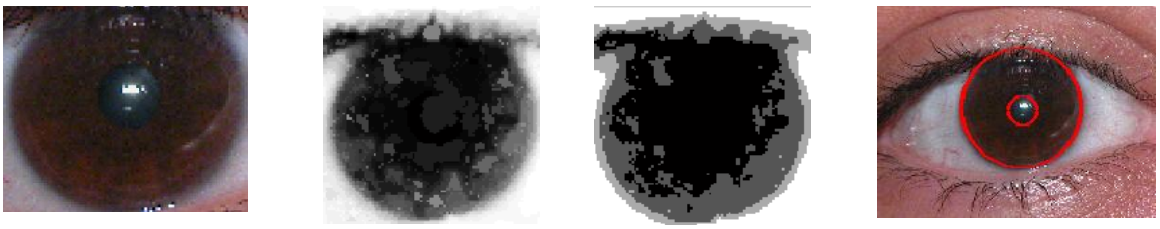
A Figura 5-10 ilustra um agrupamento formado quando são empregados mapas auto-organizáveis. A Figura 5-10b ilustra o resultado para grade de seis neurônios e a Figura 5-10c o resultado com grade de dois neurônios.



(a) Original (b) 6 neurônios (c) 2 neurônios

Figura 5-10 – Exemplo de agrupamento com mapas auto-organizáveis.

Na Figura 5-11, é apresentada uma imagem com baixo contraste entre a íris e a pupila (Figura 5-11a). Quando o mapa auto-organizável foi parametrizado com seis neurônios. O resultado foi uma imagem possível de ser segmentada (Figura 5-11b). No entanto, quando são empregados 2 neurônios, o resultado foi uma imagem ruidosa (Figura 5-11c), não sendo possível segmentá-la. A Figura 5-11d ilustra o resultado obtido após a aplicação da transformada de Hough sobre a imagem da Figura 5-11b.



(a) original (b) 6 neurônios (c) 2 neurônios (d) imagem segmentada

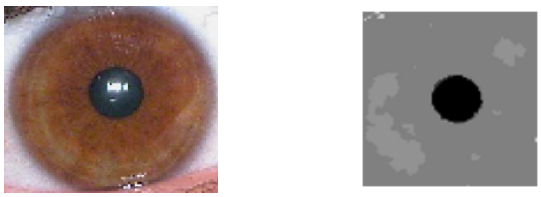
Figura 5-11 – Exemplo de baixo contraste entre a pupila e a íris de uma imagem segmentada com SOM.

A seguir, foi investigado o emprego de agrupamento por vetores suporte. Nesta dissertação, foi utilizado o toolbox em Matlab desenvolvido por (LEE e LEE, 2005). Este utiliza a função de kernel gaussiana. A rotulação do método é computacionalmente custosa, tendo complexidade exponencial em relação ao tamanho da entrada, o que dificulta a determinação da variância da função gaussiana. Para uma imagem, em um computador com uma configuração de processador core2duo e com 4G de memória, o tempo gasto é de aproximadamente oito minutos. Para tornar viável a utilização do método, quatro imagens distintas do conjunto foram escolhidas. Sobre estas foram realizadas vários execuções variando o valor do parâmetro do kernel. O valor do parâmetro igual a 0,06 produziu melhor o resultado.

A Tabela 5.6 apresenta o resultado com agrupamento por vetores suporte. Em imagens com baixo contraste da íris em relação à pupila, como a ilustrada na Figura 5-11a, o método resulta em um único rótulo gerando uma imagem totalmente preta. Entretanto, para imagens com alto contraste entre a íris e a pupila, o método resultou em uma separação com eficácia, comparando-se com os métodos anteriores. A Figura 5-12 ilustra um dos resultados obtido via SVC.

Tabela 5.6 – Resultado com agrupamento baseado em vetores-suporte

| | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|
| Descrição | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| SVC | 1095 | 119 | 89,13% | 591 | 72 | 87,82% |



(a) original (b) resultado

Figura 5-12 – Exemplo de segmentação via agrupamento baseado em vetores-suporte.

### 5.3.1 ENSEMBLE

O objetivo deste experimento é obter uma nova imagem através da combinação dos resultados obtidos anteriormente. A Figura 5.13 apresenta a seqüência de passos, em que os

melhores resultados dos quatro métodos anteriores são estrategicamente combinados (KUNCHEVA, HADJITODOROV e TODOROVA, 2006), gerando-se um novo conjunto com apenas dois rótulos, resultando uma nova imagem de apenas duas cores.

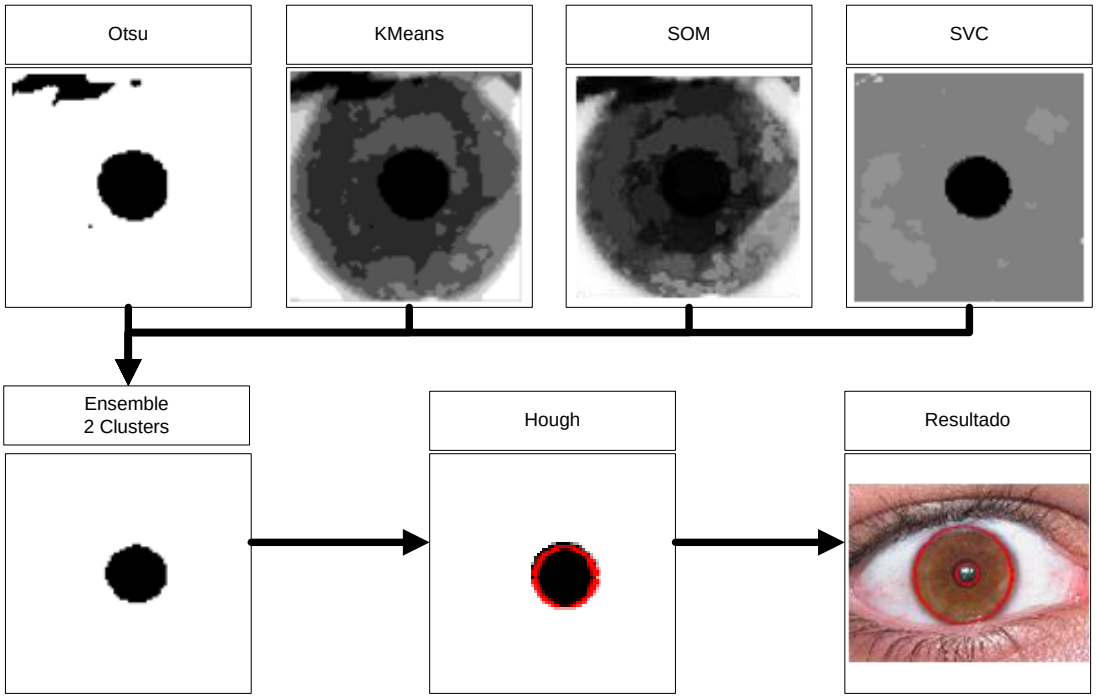



Figura 5.13 – Seqüência para segmentação da íris através da combinação de resultados (Ensemble).

A estratégia de combinação adotada foi a HGPA, sendo que as outras estratégias de combinação possuem complexidade computacional quadrática e não são viáveis para a aplicação em imagens (STREHL e GHOSH, 2002). A Tabela 5.7 apresenta o resultado obtido com ensembles. Pode-se observar uma pequena melhora, quando se compara com o melhor resultado do mapa auto-organizável.

Tabela 5.7 – Resultado com ensemble

| | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|
| Descrição | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| Ensemble | 1178 | 36 | 96,94% | 621 | 42 | 93,24% |

A Tabela 5.8 apresenta um comparativo dos resultados obtidos com os métodos propostos neste experimento com os principais métodos de segmentação da literatura com o mesmo conjunto de dados.

Tabela 5.8 – Comparativo com outros métodos da literatura (experimento #2)

| | Sessão 1 | Sessão 2 |
|---|---|---|
| Método | Segmentado | Segmentado |
| K-*means* (Proença) | 92,33% | 89,14% |
| SOM (Proença) | 97,69% | 96,68% |
| Fuzzy K-*means* (Proença) | 98,02% | 97,88% |
| Daugman | 95,79% | 91,10% |
| Camus e Wildes | 96,78% | 89,29% |
| Kheirolahy | 98,00% | 96,38% |
| K-*means* (Proposto) | 93,70% | 91,13% |
| SOM (Proposto) | 96,68% | 92,72% |
| SVC (Proposto) | 89,13% | 87,82% |
| Ensemble (Proposto) | 96,94% | 93,24% |

Os trabalhos descritos neste comparativo não mencionaram qual o tratamento foi adotado para as imagens distorcidas, como as apresentadas na Figura 5-14. Como nestes trabalhos apenas é apresentado o percentual de segmentação, não é possível constatar a real superioridade em seus resultados com os deste trabalho. Neste trabalho, no cálculo da taxa de acerto foram levadas em conta todas as imagens, inclusive as imagens da Figura 5-14.

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

Figura 5-14 – Imagens distorcidas do conjunto de dados da UBIRIS

No experimento com ensembles foi possível melhorar a segmentação de três imagens. Sendo que na imagem da Figura 5-14d composta com uma área maior do rosto do indivíduo, este método segmentou eficientemente.

## 5.4 EXPERIMENTO # 3

Neste experimento é proposta a aplicação de Redes Neurais Artificiais do tipo múltiplas camadas para a busca da pupila na imagem e posterior aplicação da transformada de Hough para localização da circunferência da íris.

Para o treinamento da rede foram extraídos manualmente 30 pontos da vertical e 30 pontos da horizontal (formando uma cruz) de cada imagem. A Figura 5-15 exemplifica a região de onde foram extraídos os pontos para a construção do conjunto de treinamento da rede. Estes pontos foram rotulados como pertencente à pupila.

Figura 5-15 – Exemplo da região de onde são extraídos os pontos para treinamento.

Foram também coletados manualmente 60 pontos que não se encontram na pupila formando o conjunto de treinamento, sendo estes rotulados como não pertencentes à pupila.

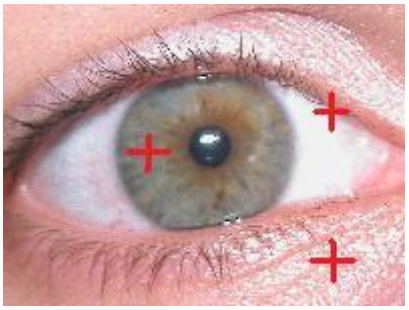


Figura 5-16 – Pontos aleatórios para treinamento

A rede neural utilizada é do tipo múltiplas camadas e é definida com 60 neurônios na entrada, uma camada intermediária e um neurônio na saída. A função de ativação dos neurônios é a tangente hiperbólica e a saída foi codificada no intervalo (*-1,+1*). Para os pontos considerados válidos, ou seja, os que pertencentes à pupila, a saída é estabelecida como 1 e para os outros pontos, com -1. A rede é treinada com o método de retro-propagação (do inglês *backpropagation*), utilizando o algoritmo QuickProp, proposto por (FAHLMAN, 1988), com

25 iterações. Para a avaliação do desempenho de treinamento da rede neural, foi considerado o erro quadrático médio (do inglês *Mean Square Error* - MSE).

Os experimentos foram divididos por etapas: na primeira foram processadas as 15 primeiras imagens do conjunto da primeira sessão, em seguida, as 30 primeiras e depois com as 45 primeiras, variando o número de neurônios na camada intermediária de 1 a 15; na segunda etapa, as imagens foram selecionadas do conjunto da primeira sessão, iniciando com 15 imagens, 30 e por fim 45, variando os neurônios da camada intermediária entre (1,15).

Tabela 5.9 – Síntese dos resultados com Rede Neural.

| | | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|---|
| Imagens | Número Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 15 S | 2 | 0,01345 | 1188 | 11 | 99,07% | 644 | 4 | 99,38% |
| 30 S | 6 | 0,01046 | 1173 | 11 | 99,06% | 629 | 4 | 99,36% |
| 45 S | 5 | 0,00852 | 1157 | 12 | 98,96% | 615 | 3 | 99,51% |
| 15 A | 5 | 0,00455 | 1188 | 11 | 99,07% | 644 | 4 | 99,38% |
| 30 A | 6 | 0,00017 | 1175 | 9 | 99,23% | 628 | 5 | 99,20% |
| 45 A | 8 | 0,00160 | 1159 | 10 | 99,14% | 613 | 5 | 99,18% |

A síntese dos resultados obtidos neste experimento pode ser analisada através da Tabela 5.9 e o resultado completo das simulações encontra-se no APÊNDICE A - . Através dos resultados, é possível observar que este método possui um desempenho superior aos outros deste trabalho, por ser supervisionado e treinado com as imagens do conjunto de dados da UBIRIS. No entanto, não é possível obter o mesmo desempenho com outros conjuntos. Em situações onde o ambiente e o dispositivo de aquisição da imagem são controlados, com um dispositivo de baixo custo, como uma câmera, este método se mostra com um desempenho superior aos demais, e com menor custo de implantação.

# 6 CONCLUSÕES E TRABALHOS FUTUROS

As ferramentas computacionais utilizadas durante o desenvolvimento da pesquisa e apresentadas ao longo desta dissertação mostraram-se viáveis na segmentação de imagens digitais, conforme pode ser observado no Capítulo 5. Apesar dos resultados não serem bastante convincentes, devido à pequena variação de desempenho, acredita-se que o caminho para o desenvolvimento de sistemas biométricos cada vez mais eficazes e robustos à variação da imagem esteja no emprego de SVC.

O algoritmo k-*means*, apesar de não ter apresentado bons resultados na segmentação, computacionalmente é bastante rápido em relação aos outros métodos abordados. Uma das desvantagens deste algoritmo na sua aplicação para segmentação de imagens é definir o melhor valor para o número de grupos.

Já os mapas auto-organizáveis de Kohonen apresentaram os melhores resultados entre os métodos não supervisionados estudados. Entretanto, a determinação dos pesos dos neurônios, em imagens com alta resolução, é um processo custoso computacionalmente, dificultando seu emprego para segmentação de imagens.

Agrupamento por vetores suporte tem a vantagem de não necessitar determinar a priori o número de grupos. Entretanto, o processo de rotulação exige grande capacidade de processamento computacional, o que o torna inviável sua aplicação em sistemas biométricos.

Os resultados obtidos com Redes Neurais Artificiais mostram-se bastante satisfatórios. Sendo que a principal desvantagem reside na necessidade de re-treinamento da rede em caso de mudança do ambiente ou do dispositivo de aquisição.

Pelo exposto acima, pode-se observar que os métodos abordados nesta dissertação possuem alguns problemas que podem inviabilizar a aplicação de tais modelos em sistemas biométricos. Portanto, é essencial desenvolver ferramentas com alto poder de generalização e que não sofra de tais problemas. O emprego de comitê de máquinas produziu uma pequena melhora na segmentação da imagem, porém, diminuindo a necessidade de interferência do usuário no projeto de aprendizado máquina.

## 6.1 PERSPECTIVAS FUTURAS

Os estudos realizados durante esta pesquisa levantaram várias questões importantes e que devem ser investigadas em outros trabalhos. São elas:

i. O emprego de agrupamento por vetores suporte apresentou em algumas imagens uma separação total da íris com o restante da imagem. Quando o objetivo era separar a pupila do restante da imagem, SVC também apresentou resultados que houve uma total separação da pupila do restante da imagem. Baseado neste fato acredita que o emprego de comitê de máquinas conjuntamente com técnicas de processamento de imagem digital seja possível a segmentação da íris sem a necessidade de outro método de localização de bordas como a transformada de Hough.

ii. Esta dissertação restringiu-se ao estudo de comitê de máquinas com estruturas estáticas. Não há na literatura estudos envolvendo estruturas dinâmicas para segmentação de imagem. Acredita-se que estas poderão apresentar resultados melhores.

iii. Estender as diversas propostas de comitês de máquinas propostas a problemas de treinamento não supervisionado para segmentação.

# BIBLIOGRAFIA BÁSICA


BAKKER, B.; HESKES, T. Clustering ensembles of neural network models. **Neural Networks**, v. 16, n. 2, p. 261-269, 2003.

BEN-HUR, A. Support vector clustering. **Scholarpedia**, v. 3, n. 6, p. 5187, 2008.

BLEHA, S.; SLIVINSKY, C.; HUSSEIN, B. Computer-Access Security Systems Using Keystroke Dynamics. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. PAMI-12, n. 12, p. 1217-1222, 1990.

CAMUS, T. A.; WILDES, R. P. **Reliable and fast eye finding in close-up images**. [S.l.]: [s.n.]. 2002. p. I: 389--394.

CANNY, J. **A Computational Approach to Edge Detection**. [S.l.]: [s.n.]. 1987. p. 184-203.

DAUGMAN, J. High Confidence Visual Recognition of Persons by a Test of Statistical Independence. **IEEE Trans. Pattern Anal. Mach. Intell**, v. 15, n. 11, p. 1148-1161, 1993.

DONGJULIU; JIANYU. **Otsu method and K-means**. [S.l.]: [s.n.]. 2009.

FAHLMAN, S. E. An Empirical Study of Learning Speed in Back-Propagation Networks. **Carnegie-Mellon University**, 1988. CMU-CS-88-162.

FAUNDEZ-ZANUY, M. et al. Authentication of Individuals using Hand Geometry Biometrics:

FERN, X.; BRODLEY, C. Random projection for high dimensional data clustering: A cluster ensemble approach. **Proceedings of the Twentieth International Conference on Machine Learning**, 2003. 186-193.

GONZALEZ, R.; WOODS, R. **Processamento Digital de Imagens**. 3 edição. ed. [S.l.]: Pearson Prentice Hall, 2010.

GROUP, B. International Biometric Group. **International Biometric Group**, 2009. Disponivel em: <http://www.biometricgroup.com/>. Acesso em: 01 jun. 2009.

HAYKIN, S. **Neural networks:** A comprehensive foundation. 3. ed. [S.l.]: Prentice Hall, v. ISBN: 978-0131471399, 2008.

HE, Z.; XU, X.; DENG, S. A cluster ensemble method for clustering categorical data. **Information Fusion 6**, 2005. 143–151.

HOUGH. Methods and means to recognize complex patterns. In: _____ **The Hough Transform reference.** [S.l.]: [s.n.], 1962.

HOUGH, P. **Method and Means for Recognizing Complex Patterns**. [S.l.]: [s.n.]. 1962.

JAIN, A. K.; FLYNN, P. J.; ROSS, A. A. **Handbook of Biometrics**. [S.l.]: Springer-Verlag, 2008.

JAIN, A. K.; HONG, L.; BOLLE, R. M. On-Line Fingerprint Verification. **IEEE Trans. Pattern Anal. Mach. Intell**, v. 19, n. 4, p. 302-314, 1997.

KHEIROLAHY, R.; EBRAHIMNEZHAD, H.; SEDAAGHI, M. Robust Pupil Boundary Detection by Optimized Color Mapping for Iris Recognition. **CSI Computer Conference**, 2009.

KOHONEN, T. **Self-Organization and Associative Memory**. Springer-Verlag., 1989.

KUNCHEVA, L. I.; HADJITODOROV, S. T.; TODOROVA, L. P. **Experimental Comparison of Cluster Ensemble Methods**. 2006.

LALIGANT, O.; TRUCHETET, F. A Nonlinear Derivative Scheme Applied to Edge Detection. **PATTERN ANALYSIS AND MACHINE INTELLIGENCE**, FEBRUARY 2010. VOL. 32, NO. 2.

LEE, J.; LEE, D. An Improved Cluster Labeling Method for Support Vector Clustering. **IEEE Transactions on Pattern Aanalysis and Machine Intelligence**, 2005. 27-(3), 461-464.

LEE, J.; LEE, D. An Improved Cluster Labeling Method for Support Vector Clustering. **IEEE Trans. Pattern Anal. Mach. Intell**, v. 27, n. 3, p. 461-464, 2005.

LEE, J.; LEE, D. Dynamic Characterization of Cluster Structures for Robust and Inductive Support Vector Clustering. **IEEE Trans. Pattern Anal. Mach. Intell**, v. 28, n. 11, p. 1869-1874, 2006.

LI, P. et al. Robust and accurate iris segmentation in very noisy iris images. **Image and Vision Computing**, v. III, p. 50-62, 2009.

LIN, W.-H.; LIAO, J.-C. **A Fast Face Segmentation Based on Color Spatial Features and K-Means Clustering Ensembles**. 2007.

MACQUEEN, J. **Some methods for classification and analysis of multivariate observations**. 1967.

MING, Z. C. et al. Segmentation of Ultrasound Image Based on Cluster Ensemble. **Digital Object Identifier**, v. 1, p. 418-421, 2008.

MIRA, J.; MAYER, J. **Image Feature Extraction for application of Biometric Identification of Iris - A Morphological Approach**. Computer Graphics and Image Processing. Brazil: XVI SIBGRAPI 2003. 2003. p. 391 - 398.

MITCHELL, T. **Machine Learning**. McGraw-Hill, v. ISBN: 0070428077, 1997.

OTSU, N. A threshold selection method from gray-level histograms. **IEEE J SSC**, v. 9, n. 1, p. 62-66, 1979.

PEDRINI, H.; SCHWARTZ, W. R. **Análise de Imagens Digitais**.: THOMSON, 2008.

PHILLIPS, P. J. et al. An Introduction to Evaluating Biometric Systems. **IEEE Computer**, v. 21, n. 2, p. 56-63, 2000.

PHILLIPS, P. J. et al. The Feret Evaluation Methodology for Face-Recognition Algorithms. **IEEE Trans. Pattern Anal. Mach. Intell**, v. 22, n. 10, p. 1090-1104, 2000.

PRABHU, S.; VENATESAN, N. **Data Mining and Warehousing**. NEW AGE INTERNATIONAL, v. ISBN : 978-81-224-2432-4, 2007.

PROENÇA, H.; ALEXANDRE, L. A. Iris segmentation methodology for non-cooperative recognition. **IEE Proceedings-Vision Image and Signal Processing**, v. 153, n. 2, p. 199-205, 2006.

ROSTEN, E.; PORTER, R.; DRUMMOND, T. Faster and Better: A Machine Learning Approach to Corner Detection. **PATTERN ANALYSIS AND MACHINE INTELLIGENCE**, JANUARY 2010. VOL. 32, NO. 1.

SOBEL, I. An 3x3 Isotropic Gradient Operator for Image Processing. **Machine Vision for Three-Dimensional Scenes**, 1990. pp. 376-379.

SONKA, M.; HLAVAC, V.; BOYLE, R. **Image Processing, Analysis, and Machine Vision**. [S.l.]: Thomson-Engineering, 1993.

STREHL, A.; GHOSH, J. Cluster Ensembles --- A Knowledge Reuse Framework for Combining Multiple Partitions. **Journal of Machine Learning Research**, v. 3, p. 583-617, 2002.

UETA, T. et al. **A Study on Contour Line and Internal Area Extraction Method by using the Self-Organization Map**. 2006.

VAPNIK, V. An overview of statistical learning theory. **IEEE Trans. Neural Networks**, v. 10, n. 5, p. 988-999, 1999.

WANG, X.; LEESER, M. **K-means Clustering for Multispectral Images Using Floating-Point Divide**. [S.l.]: IEEE Computer Society. 2007. p. 151-162.

ZHANG, J.; QIN, Z. Edge Detection Using Fast Bidimensional Empirical Mode Decomposition and Mathematical Morphology. **Digital Object Identifier**, Concord, Abril 2010. 139 - 142.

ZHOU, S. R.; LIANG, X. M.; ZHU, C. **Support Vector Clustering of Facial Expression Features**. 2008.

# APÊNDICE A - RESULTADO DAS SIMULAÇÕES COM RNA

Tabela A.1 – Resultado com as 15 primeiras imagens

| | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|
| Numero Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 2 | 0,01345 | 1188 | 11 | 99,07% | 644 | 4 | 99,38% |
| 5 | 0,00579 | 1187 | 12 | 98,99% | 644 | 4 | 99,38% |
| 12 | 0,00060 | 1190 | 9 | 99,24% | 642 | 6 | 99,07% |
| 2 | 0,00748 | 1188 | 11 | 99,07% | 643 | 5 | 99,22% |
| 7 | 0,00187 | 1188 | 11 | 99,07% | 643 | 5 | 99,22% |
| 4 | 0,00031 | 1186 | 13 | 98,90% | 644 | 4 | 99,38% |
| 3 | 0,00113 | 1187 | 12 | 98,99% | 643 | 5 | 99,22% |
| 8 | 0,00123 | 1188 | 11 | 99,07% | 642 | 6 | 99,07% |
| 6 | 0,00016 | 1189 | 10 | 99,16% | 641 | 7 | 98,91% |
| 1 | 0,07459 | 1190 | 9 | 99,24% | 640 | 8 | 98,75% |

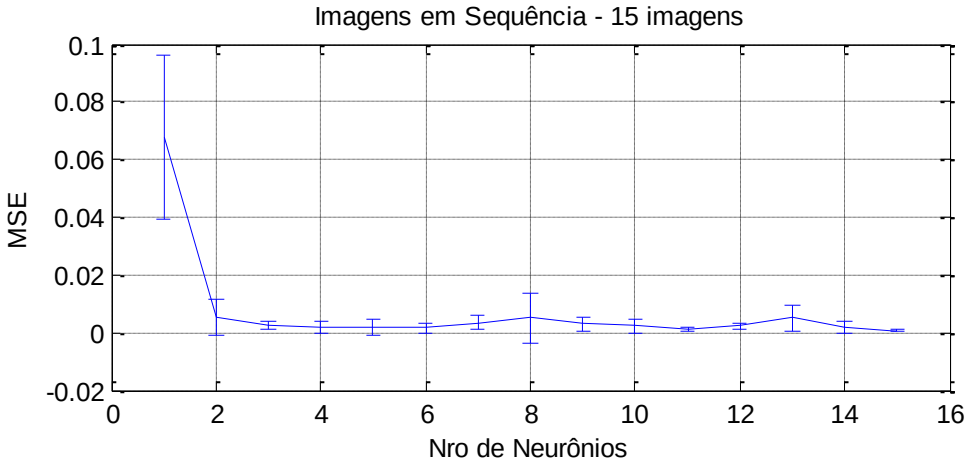



Figura A.1 – 15 imagens em seqüência

Tabela A.2 – Resultado com as 30 primeiras imagens

| | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|
| Numero Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 6 | 0,01046 | 1173 | 11 | 99,06% | 629 | 4 | 99,36% |
| 4 | 0,00300 | 1175 | 9 | 99,23% | 627 | 6 | 99,04% |
| 7 | 0,00247 | 1173 | 11 | 99,06% | 628 | 5 | 99,20% |
| 7 | 0,00391 | 1174 | 10 | 99,15% | 627 | 6 | 99,04% |
| 5 | 0,00020 | 1173 | 11 | 99,06% | 626 | 7 | 98,88% |
| 7 | 0,00114 | 1173 | 11 | 99,06% | 626 | 7 | 98,88% |
| 6 | 0,00042 | 1169 | 15 | 98,72% | 628 | 5 | 99,20% |
| 8 | 0,00033 | 1174 | 10 | 99,15% | 625 | 8 | 98,72% |
| 2 | 0,00025 | 1173 | 11 | 99,06% | 625 | 8 | 98,72% |
| 1 | 0,03642 | 1175 | 9 | 99,23% | 623 | 10 | 98,39% |

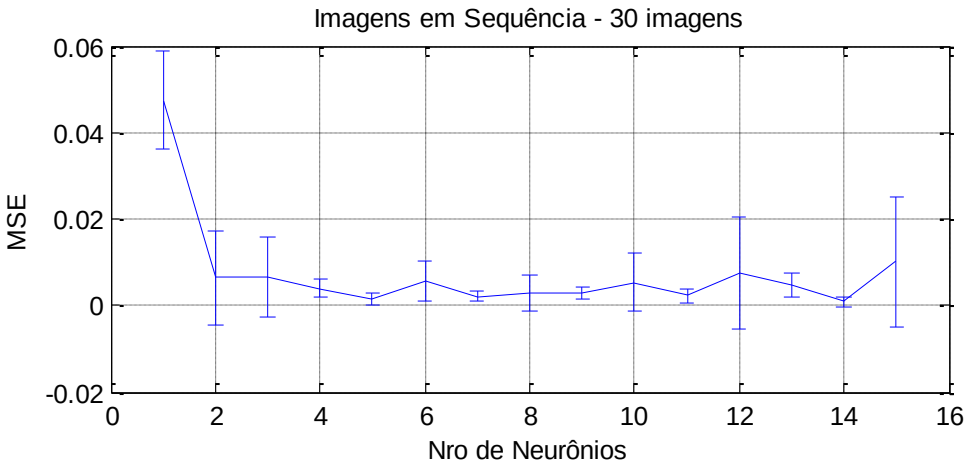



Figura A.2 – 30 imagens em seqüência

Tabela A.3 – Resultado com as 45 primeiras imagens

| | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|
| Numero Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 5 | 0,00852 | 1157 | 12 | 98,96% | 615 | 3 | 99,51% |
| 13 | 0,00164 | 1158 | 11 | 99,05% | 614 | 4 | 99,35% |
| 4 | 0,00705 | 1159 | 10 | 99,14% | 613 | 5 | 99,18% |
| 11 | 0,00221 | 1160 | 9 | 99,22% | 612 | 6 | 99,02% |
| 4 | 0,00385 | 1158 | 11 | 99,05% | 613 | 5 | 99,18% |
| 3 | 0,00483 | 1159 | 10 | 99,14% | 611 | 7 | 98,85% |
| 7 | 0,00121 | 1157 | 12 | 98,96% | 612 | 6 | 99,02% |
| 9 | 0,00119 | 1158 | 11 | 99,05% | 611 | 7 | 98,85% |
| 3 | 0,00048 | 1159 | 10 | 99,14% | 610 | 8 | 98,69% |
| 15 | 0,00516 | 1159 | 10 | 99,14% | 608 | 10 | 98,36% |

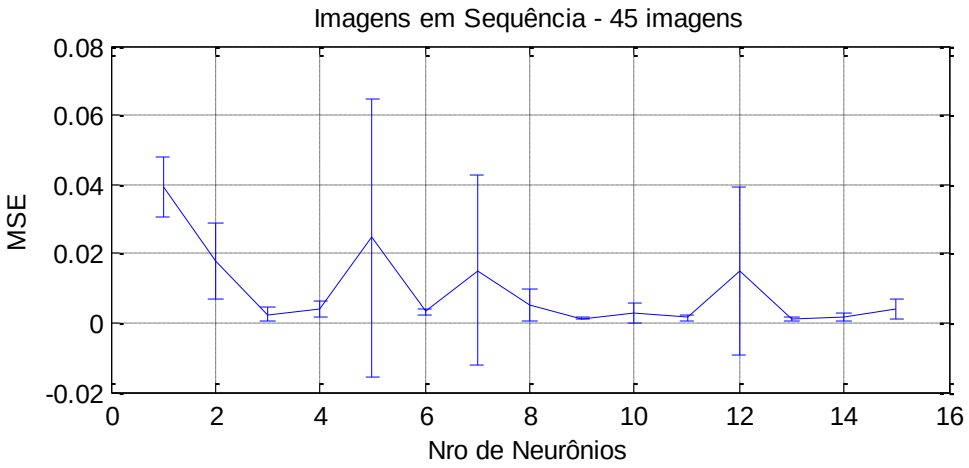



Figura A.3 – 45 imagens em seqüência

Tabela A.4 – Resultado com 15 imagens aleatórias

| | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|
| Numero Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 5 | 0,00455 | 1188 | 11 | 99,07% | 644 | 4 | 99,38% |
| 5 | 0,00684 | 1188 | 11 | 99,07% | 644 | 4 | 99,38% |
| 6 | 0,00426 | 1190 | 9 | 99,24% | 642 | 6 | 99,07% |
| 7 | 0,00017 | 1190 | 9 | 99,24% | 642 | 6 | 99,07% |
| 10 | 0,00068 | 1188 | 11 | 99,07% | 643 | 5 | 99,22% |
| 2 | 0,00048 | 1188 | 11 | 99,07% | 642 | 6 | 99,07% |
| 4 | 0,00130 | 1188 | 11 | 99,07% | 642 | 6 | 99,07% |
| 11 | 0,00359 | 1186 | 13 | 98,90% | 643 | 5 | 99,22% |
| 4 | 0,00116 | 1184 | 15 | 98,73% | 644 | 4 | 99,38% |
| 5 | 0,00025 | 1189 | 10 | 99,16% | 641 | 7 | 98,91% |

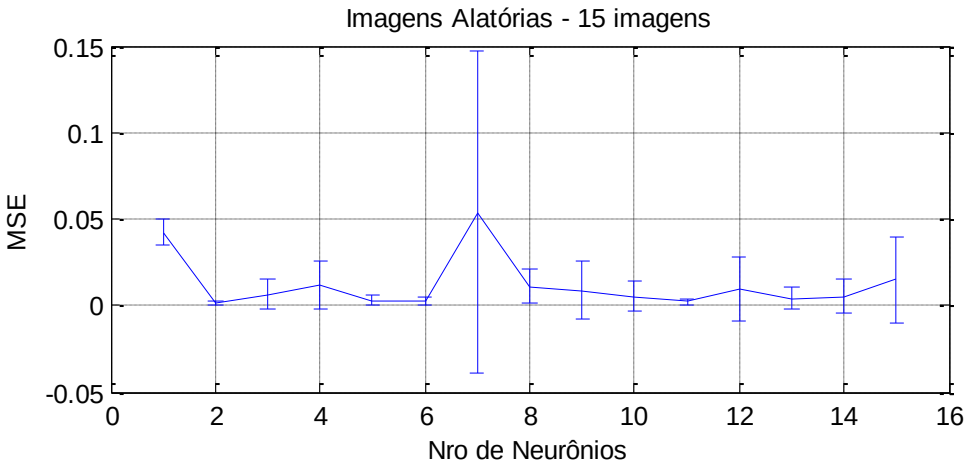



Figura A.4 – 15 imagens aleatórias

Tabela A.5 – Resultado com 30 imagens aleatórias

| | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|
| Numero Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 7 | 0,01561 | 1174 | 10 | 99,15% | 629 | 4 | 99,36% |
| 6 | 0,00017 | 1175 | 9 | 99,23% | 628 | 5 | 99,20% |
| 6 | 0,00305 | 1174 | 10 | 99,15% | 628 | 5 | 99,20% |
| 13 | 0,00763 | 1174 | 10 | 99,15% | 628 | 5 | 99,20% |
| 4 | 0,00019 | 1174 | 10 | 99,15% | 627 | 6 | 99,04% |
| 12 | 0,00014 | 1174 | 10 | 99,15% | 627 | 6 | 99,04% |
| 13 | 0,00030 | 1174 | 10 | 99,15% | 627 | 6 | 99,04% |
| 15 | 0,00809 | 1174 | 10 | 99,15% | 627 | 6 | 99,04% |
| 2 | 0,00386 | 1175 | 9 | 99,23% | 626 | 7 | 98,88% |
| 5 | 0,00055 | 1175 | 9 | 99,23% | 626 | 7 | 98,88% |

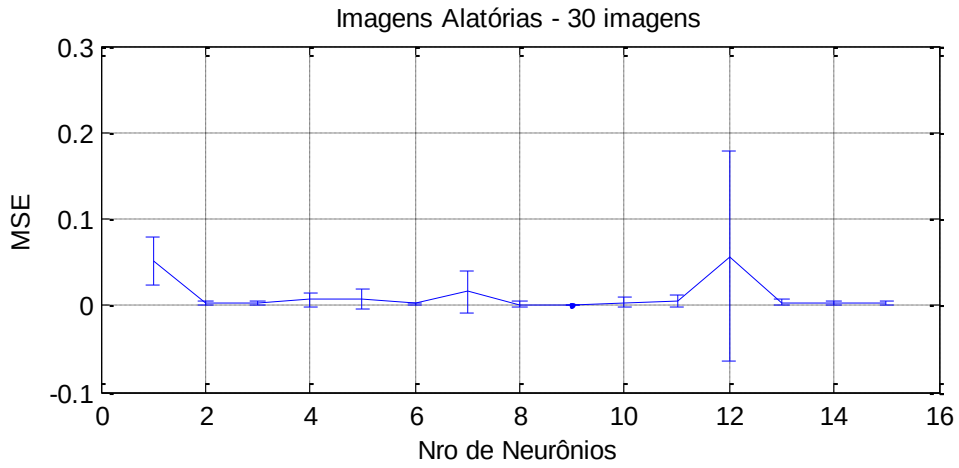



Figura A.5 – 30 imagens aleatórias

Tabela A.6 – Resultado com 45 imagens aleatórias

| | | Sessão 1 | | | Sessão 2 | | |
|---|---|---|---|---|---|---|---|
| Numero Neurônios | MSE | Acertos | Erros | Segmentado (%) | Acertos | Erros | Segmentado (%) |
| 8 | 0,00160 | 1159 | 10 | 99,14% | 613 | 5 | 99,18% |
| 13 | 0,00049 | 1159 | 10 | 99,14% | 613 | 5 | 99,18% |
| 2 | 0,00371 | 1158 | 11 | 99,05% | 613 | 5 | 99,18% |
| 5 | 0,00334 | 1156 | 13 | 98,88% | 613 | 5 | 99,18% |
| 2 | 0,00089 | 1155 | 14 | 98,79% | 613 | 5 | 99,18% |
| 1 | 0,03867 | 1159 | 10 | 99,14% | 610 | 8 | 98,69% |
| 2 | 0,02229 | 1159 | 10 | 99,14% | 610 | 8 | 98,69% |
| 7 | 0,00211 | 1157 | 12 | 98,96% | 611 | 7 | 98,85% |
| 15 | 0,00883 | 1151 | 18 | 98,44% | 614 | 4 | 99,35% |
| 5 | 0,00021 | 1153 | 16 | 98,61% | 612 | 6 | 99,02% |

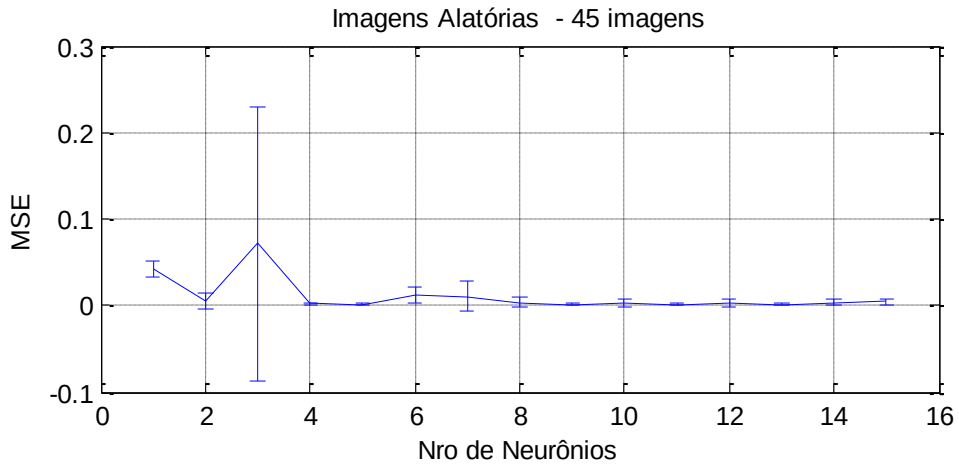



Figura A.6 – 45 imagens aleatórias

# APÊNDICE B - IMPLEMENTAÇÃO DOS MÉTODOS PROPOSTOS

A seguir apresenta-se as implementações em Matlab 7.9.0 (R2009b) dos métodos propostos desta pesquisa.

Programa inicial do Toolbox para executar todos os métodos propostos

```
cd clust/svc/stprtool/
stprpath
cd ../../..
close all;clear all;clc;
work_dir = pwd;
wp{1} = strcat(work_dir,'/imagem/');
wp{2} = strcat(work_dir,'/util/');
wp{3} = strcat(work_dir,'/clust/');
wp{4} = strcat(work_dir,'/ensemble/');
wp{5} = strcat(work_dir,'/iris/');
for p=1:length(wp)
   addpath( wp{p} );
end
type_segm = 100;
while (type_segm > 0)
    clc;
    disp(sprintf('[1] - Imagem'))
    disp(sprintf('[2] - Hough'))
    disp(sprintf('[3] - Método de Daugman'))
    disp(sprintf('[4] - IrisCode'))
    disp(sprintf('[5] - Entradas para o Cluster'))
    disp(sprintf('[6] - Kmeans'))
    disp(sprintf('[7] - SOM'))
    disp(sprintf('[8] - Ensemble'))
    disp(sprintf('[9] - Simulações'))
    disp(sprintf('[0] - Sair'))
    type_segm=input('Escolha uma opcao ');
    if type_segm==1 % Imagem
       cd imagem
       menu_imagem
       cd ..
    elseif type_segm==2 % Hough

    elseif type_segm==3 % MÃ©todo de Daugman

    elseif type_segm==4 % IrisCode

    elseif type_segm==5 % Entradas para o Cluster
        fator=input('VizinhanÃ§a: ');
        X=entradas(A,fator);
    elseif type_segm==6 % KMeans
        try
            valor=input('Clusters : ');
            [l,c] = kmeans(X, valor);
            l = rotula(l,c);
            Ik = lbl2img(l,nlin-(fator*2)+1,ncol-(fator*2)+1);
            figure(3);imshow(Ik);
        catch
            msgbox('Erro no kMeans. Tente mudar o numero de clusters');
        end
```

```
    elseif type_segm==7 % SOM
       cd clust/som
       valor=input('NÃ°mero Neuronios : ');
       try
          [tmp l c] = som(X,valor,20);
          l = rotula(l,c);
          Ik = lbl2img(l,nlin-(fator*2)+1,ncol-(fator*2)+1);
          figure(3);imshow(Ik);
       catch
            msgbox('Erro no SOM');
       end
       cd ../..

    elseif type_segm==8 % Ensemble
        fator = 7;
        c{1}.imagem = I;
        [nlin,ncol,ncor]=size(I);
        c{1}.fator = fator;
        c{1}.imsize = size(I);
        c{1}.nlin=nlin-(fator*2)+1;
        c{1}.ncol=ncol-(fator*2)+1;
        c{1}.entradas =entradas(A,fator);
        c = ckmeans(c,6);
        c = csom(c,2);
        c = csvc(c,0.005,0.4);
        c{1}.ensemble = ensemble(c);
        ploti(c);
        drawnow;
    elseif type_segm==9 % SimulaÃ§Ãµes
    end
end
```

Transformada de Hough

```
function [yr,xr,ar,rhraio] = findHough(Img,rmin, rmax, thresh, region)
    imSize = size(Img);
    if nargin < 5 region = [1,1,imSize(2),imSize(1)]; end
    yr=0;xr=0;ar=0;rhraio=0;
    if (thresh > 4)
      for hraio=rmax:-1:rmin
           [Y,X,A]=houghcircle(Img,hraio,thresh,region);
           if ( size(Y) ==  1)
              yr = Y; xr = X; ar = A;
              rhraio = hraio;
              break;
           end
       end
    else
       achou=0;
       for thresh=200:-10:5
          for hraio=rmax:-1:rmin
              [Y,X,A]=houghcircle(Img,hraio,thresh,region);
              if ( size(Y) ==  1)
                 yr = Y; xr = X; ar = A;
                 rhraio = hraio;
                 achou = 1;
                 break;
```

```
            end
        end
        if (achou == 1)
           break
        end
      end
    end
end

function [y0detect,x0detect,Accumulator] =
houghcircle(Imbinary,r,thresh,region)
    imSize = size(Imbinary);
    if nargin < 3 thresh = 4; end
    if nargin < 4 region = [1,1,imSize(2),imSize(1)]; end
    Accumulator = zeros(imSize);
    xmin = region(1); xmax = xmin + region(3) - 1;
    ymin = region(2); ymax = ymin + region(4) - 1;
    [yIndex xIndex] = find(Imbinary);
    for cnt = 1:length(xIndex)
        xCenter = xIndex(cnt); yCenter = yIndex(cnt);
        xmx=xCenter-r; xpx=xCenter+r;
        ypx=yCenter+r; ymx=yCenter-r;
        if xpx<xmin | xmx>xmax | ypx<ymin | ymx>ymax continue; end
        if ypx<=imSize(1) & ymx>=1 & xpx<=imSize(2) & xmx>=1
            x = 1;
            y = r;
            d = 5/4 - r;
            Accumulator(ypx,xCenter) = Accumulator(ypx,xCenter) + 1;
            Accumulator(ymx,xCenter) = Accumulator(ymx,xCenter) + 1;
            Accumulator(yCenter,xpx) = Accumulator(yCenter,xpx) + 1;
            Accumulator(yCenter,xmx) = Accumulator(yCenter,xmx) + 1;
            while y > x
                xpx = xCenter+x; xmx = xCenter-x;
                ypy = yCenter+y; ymy = yCenter-y;
                ypx = yCenter+x; ymx = yCenter-x;
                xpy = xCenter+y; xmy = xCenter-y;
                Accumulator(ypy,xpx) = Accumulator(ypy,xpx) + 1;
                Accumulator(ymy,xpx) = Accumulator(ymy,xpx) + 1;
                Accumulator(ypy,xmx) = Accumulator(ypy,xmx) + 1;
                Accumulator(ymy,xmx) = Accumulator(ymy,xmx) + 1;
                Accumulator(ypx,xpy) = Accumulator(ypx,xpy) + 1;
                Accumulator(ymx,xpy) = Accumulator(ymx,xpy) + 1;
                Accumulator(ypx,xmy) = Accumulator(ypx,xmy) + 1;
                Accumulator(ymx,xmy) = Accumulator(ymx,xmy) + 1;
                if d < 0
                    d = d + x * 2 + 3;
                    x = x + 1;
                else
                    d = d + (x - y) * 2 + 5;
                    x = x + 1;
                    y = y - 1;
                end
            end
            if x == y
                xpx = xCenter+x; xmx = xCenter-x;
                ypy = yCenter+y; ymy = yCenter-y;
                Accumulator(ypy,xpx) = Accumulator(ypy,xpx) + 1;
                Accumulator(ymy,xpx) = Accumulator(ymy,xpx) + 1;
                Accumulator(ypy,xmx) = Accumulator(ypy,xmx) + 1;
                Accumulator(ymy,xmx) = Accumulator(ymy,xmx) + 1;
```

```
            end
        else
            ypxin = ypx>=ymin & ypx<=ymax;
            ymxin = ymx>=ymin & ymx<=ymax;
            xpxin = xpx>=xmin & xpx<=xmax;
            xmxin = xmx>=xmin & xmx<=xmax;
            if ypxin Accumulator(ypx,xCenter) = Accumulator(ypx,xCenter) +
1; end
            if ymxin Accumulator(ymx,xCenter) = Accumulator(ymx,xCenter) +
1; end
            if xpxin Accumulator(yCenter,xpx) = Accumulator(yCenter,xpx) +
1; end
            if xmxin Accumulator(yCenter,xmx) = Accumulator(yCenter,xmx) +
1; end
            x = 1;
            y = r;
            d = 5/4 - r;
            while y > x
                xpx = xCenter+x; xpxin = xpx>=xmin & xpx<=xmax;
                xmx = xCenter-x; xmxin = xmx>=xmin & xmx<=xmax;
                ypy = yCenter+y; ypyin = ypy>=ymin & ypy<=ymax;
                ymy = yCenter-y; ymyin = ymy>=ymin & ymy<=ymax;
                ypx = yCenter+x; ypxin = ypx>=ymin & ypx<=ymax;
                ymx = yCenter-x; ymxin = ymx>=ymin & ymx<=ymax;
                xpy = xCenter+y; xpyin = xpy>=xmin & xpy<=xmax;
                xmy = xCenter-y; xmyin = xmy>=xmin & xmy<=xmax;
                if ypyin & xpxin Accumulator(ypy,xpx) =
Accumulator(ypy,xpx) + 1; end
                if ymyin & xpxin Accumulator(ymy,xpx) =
Accumulator(ymy,xpx) + 1; end
                if ypyin & xmxin Accumulator(ypy,xmx) =
Accumulator(ypy,xmx) + 1; end
                if ymyin & xmxin Accumulator(ymy,xmx) =
Accumulator(ymy,xmx) + 1; end
                if ypxin & xpyin Accumulator(ypx,xpy) =
Accumulator(ypx,xpy) + 1; end
                if ymxin & xpyin Accumulator(ymx,xpy) =
Accumulator(ymx,xpy) + 1; end
                if ypxin & xmyin Accumulator(ypx,xmy) =
Accumulator(ypx,xmy) + 1; end
                if ymxin & xmyin Accumulator(ymx,xmy) =
Accumulator(ymx,xmy) + 1; end
                if d < 0
                    d = d + x * 2 + 3;
                    x = x + 1;
                else
                    d = d + (x - y) * 2 + 5;
                    x = x + 1;
                    y = y - 1;
                end
            end
            if x == y
                xpx = xCenter+x; xpxin = xpx>=xmin & xpx<=xmax;
                xmx = xCenter-x; xmxin = xmx>=xmin & xmx<=xmax;
                ypy = yCenter+y; ypyin = ypy>=ymin & ypy<=ymax;
                ymy = yCenter-y; ymyin = ymy>=ymin & ymy<=ymax;
                if ypyin & xpxin Accumulator(ypy,xpx) =
Accumulator(ypy,xpx) + 1; end
                if ymyin & xpxin Accumulator(ymy,xpx) =
Accumulator(ymy,xpx) + 1; end
                if ypyin & xmxin Accumulator(ypy,xmx) =
```

```
Accumulator(ypy,xmx) + 1; end
                if ymyin & xmxin Accumulator(ymy,xmx) =
Accumulator(ymy,xmx) + 1; end
            end
        end
    end
    y0detect = []; x0detect = [];
    AccumulatorbinaryMax = imregionalmax(Accumulator);
    [Potential_y0 Potential_x0] = find(AccumulatorbinaryMax == 1);
    Accumulatortemp = Accumulator - thresh;
    for cnt = 1:length(Potential_y0)
        if Accumulatortemp(Potential_y0(cnt),Potential_x0(cnt)) >= 0
            y0detect = [y0detect;Potential_y0(cnt)];
            x0detect = [x0detect;Potential_x0(cnt)];
        end
    end
```

Mapas Auto Organizável de Kohonen - SOM

```
function [clusters rotulos ctrs] = som(P, neuron, epocas)
    clusters = [];
    rotulos = [];
    [s1 s2] = size(P);
    [W1 p1]=somtreina(P,neuron,neuron,10,[.01,8]);                %
Treinamento Ordenacao
    W=somtreina(P,neuron,neuron,epocas,[p1(1) .001 p1(2) 0],W1); %
Treinamento Convergencia
    [clusters rotulos ctrs] = dist(P,W);
end


function [W_out, p_out]=som_2d(P, nsofmx, nsofmy, epochs, phas, W)
  r=nsofmx;  c=nsofmy; cr=r+1; cc=c+1;
  [pats dimz]=size(P);
  if nargin < 6, W=randn(nsofmx,nsofmy,dimz)*.1; end
  n = 1:epochs;
  if length(phas) == 2,
    so = phas(2);
    t1 = epochs/(log(so+1));
    sigmas = so*exp(-n/t1);
    lrs    = phas(1)*exp(-n/epochs);
  else
    sigmas = (phas(3)-phas(4))*fliplr(n)/epochs+phas(4);
    lrs    = (phas(1)-phas(2))*fliplr(n)/epochs+phas(2);
  end
  mask=zeros(2*r+1,2*c+1);
  x=[-c:c];   y=[r:-1:-r];
  for i=1:length(x), for j=1:length(y),
        mask(j,i)=sqrt(x(i)^2+y(j)^2);
     end
  end
  sigma=sigmas(1);
  m = (1/(2*pi*sigma^2)*exp((-mask.^2/(2*sigma^2))));
  m = m/max(max(m));
  lbr =cr-floor(r/2);
  ubr =cr+floor(r/2);
```

```
  lbc =cc-floor(r/2);
  ubc =cc+floor(r/2);
  sigma=sigmas(length(sigmas));
  m = (1/(2*pi*sigma^2)*exp((-mask.^2/(2*sigma^2))));
  m = m/max(max(m));
  for epoch=1:epochs,
     [neword]=shuffle(P);
     sigma=sigmas(epoch);
     for pat=1:pats,
        in=neword(pat,:)';
        Dup=in(:,ones(r,1),ones(c,1));
        Dup=permute(Dup,[2,3,1]);
        Dif= W - Dup;
        Sse=sum(Dif.^2,3);
        [val1 win_rows]=min(Sse); [val2 wc]=min(val1); wr=win_rows(wc);
        WN(pat,1:2)=[wr wc];
        M1 = mask(cr-wr+1:cr+(r-wr),cc-wc+1:cc+(c-wc));
        M1 = (1/(2*pi*sigma^2)*exp((-M1.^2/(2*sigma^2))));
        M1 = M1/max(max(M1));
        M1=M1(:,:,ones(1,dimz));
        W = W + lrs(epoch)*(Dup-W).*M1;
     end
     p_out = [lrs(length(lrs)) sigma];
     W_out = W;
  end
end


function [ clusters labels ctrs] = dist( P, W )
   clusters = [];
   labels = [];
   ctrs = [];
   [nlin ncol] = size(P);
   [w1 w2 w3]  = size(W);
   deant = ones(nlin,1)*10000;
   pesos = reshape(W,(w1*w2),w3);
   for j=1:(w1*w2)
       de = zeros(nlin,1);
       %calcula a distancia euclidiana para todos os pontos
       for i=1:nlin
          de(i) = sqrt(sum(power( P(i,:) - pesos(j,:),2)));
       end

       %atribui um cluster para um determinado ponto
       for i=1:nlin
         if de(i) < deant(i)
            deant(i) = de(i);
            clusters(i) = j;
         end
       end
   end
   labels = ones(nlin,1);
   l=1;
   ctrs = [];
   for i=1:length(pesos)
      lbls = find(clusters==i);
      if (length(lbls) > 0 )
         labels(lbls) = l;
         l = l+1;
         ctrs = [ctrs; pesos(i,:)];
      end
```

```
   end
end
```

Programa para simulação de cada método de Clusterização

```
function labels = ckmeans(a,k)
    labels=[];
    for yyy=1:20
        disp(sprintf('KMeans: Tentativa:%i ',yyy));
        try
           [l,ctrs] = kmeans(a, k);
           labels = rotula(l,ctrs,1);
           break;
        catch
        end
    end
end


function labels = csom(a, neuron)
    cd clust/som
    [c l ctrs] = som(a,neuron,10);
    cd ../..
    labels = rotula(l,ctrs,1);
end


function labels = cssv(a,gaus,bsvs)
    cd clust/svc
    disp(length(a))
    try
        data.X = a';
        %options=struct('method','SEP-CG','ker','rbf','arg',0.005,'C',0.4);
        options=struct('method','SEP-CG','ker','rbf','arg',gaus,'C',bsvs);
        [model]=svc(data,options);
        labels = model.cluster_labels';
        disp('SVC SEP-CG');
    catch
        labels = [1];
        disp('ERRO no SVC');
    end
    cd ../..
end
```

Programa para simulação com Ensemble

```
function [l,cls] = ensemble(c,k)
    if ~exist('k')
       k = 0;
    end

    workdir=pwd;
    cls=[];
    for i=2:length(c)
       a = c{i}.rotulos;
```

```
        if (k>0)
           a(find(a>1))=2;
        end
        cls=[cls; a'];
    end
    cd ensemble/ClusterEnsembleV10
    l=clusterensemble(cls)';
    cd (workdir);
end
```

Programas de normalização

```
function [E] = entradas(I,fator)
    [nlin,ncol,ncor]=size(I);
    a = I;
    E = [];
    for i=fator:nlin-fator
       for j=fator:ncol-fator
          d = [ I(i,j,1) I(i,j,2) I(i,j,3) ]; % centro
          E = [E;d];
       end
    end
    E = double(E)/255;
end

function [ r ] = rotula( label, c, n )
   if ~exist('n')
      n = 1;
   end
   [nl nc] = size(c);
   r = zeros(length(label),1);
   a = zeros(nl,1);
   for j=1:nl
      a(j) = sqrt(sum(power(c(j,:),2)));
   end
   [tmp,is] = sort(a);
   for j=1:nl
      idxl = find(label==is(j));
      r(idxl) = j;
   end
end

function [ Img ] = lbl2img( label, h, w )
   a= (reshape(label',[w h])')-1;
   Img = mat2gray(a-1);
end
```

Segmentação da íris e pupila

```
function [c,d] = getIris(I,xr,yr,hraio)
    A = I(:,:,1);
    yi = yr - hraio;
    yf = yr + hraio;
    xi = xr - hraio;
    xf = xr + hraio;
    c.xc = xi;
    c.yc = yi;
```

```
    try
       A = A(yi:yf,xi:xf);
       c.cortada = I(yi:yf,xi:xf+3,:);
    catch
    end
    A=imcomplement(imfill(imcomplement(A),'holes'));     c.I1 = A;
    se = strel('square',3);A = imdilate(A, se);
    A=autolevel(A);                                      c.I2 = A;
    A=brilho(A,30);                                      c.I3 = A;
    A=autolevel(A);                                      c.I4 = A;
    fator = 2;
    [nlin,ncol]=size(A);
    c.fator = fator;
    c.nlin=nlin-(fator*2)+1;
    c.ncol=ncol-(fator*2)+1;
    c.entradas      = entradas(A,fator);
    d = distinct(c.entradas);
    c.A = A;
end



function [xp,yp,rp,b] = findPupila(b)
    se = strel('square',3);b = imdilate(b, se);
    b = edge(b,'canny');
    b = realca(b);
    [yp,xp,raios,rp]=findHough(b,4,30,0);
end
```

# Livros Grátis

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

Milhares de Livros para Download:

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