

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ  
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA  
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

TELMO FRIESEN

SISTEMA DE VISÃO COMPUTACIONAL PARA A CLASSIFICAÇÃO  
E MEDIDA DA POSIÇÃO ANGULAR DE OBJETOS METÁLICOS

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA

2014

**TELMO FRIESEN**

**SISTEMA DE VISÃO COMPUTACIONAL PARA A CLASSIFICAÇÃO  
E MEDIDA DA POSIÇÃO ANGULAR DE OBJETOS METÁLICOS**

Trabalho de Conclusão de Curso apresentado ao Departamento Acadêmico de Informática como requisito parcial para obtenção do grau de Engenheiro no Bacharelado em Engenharia de Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Gustavo Benvenutti Borba

**CURITIBA**

**2014**

## **AGRADECIMENTOS**

Agradeço primeiramente à Deus pela inspiração para realização deste trabalho. Agradeço a meus pais pelo amor e dedicação ao me dar a oportunidade de cursar Engenharia de Computação. E finalmente agradeço ao professor Gustavo Benvenutti Borba pela orientação do trabalho.

## RESUMO

FRIESEN, Telmo. Sistema de visão computacional para a classificação e medida da posição angular de objetos metálicos. 69 f. Trabalho de Conclusão de Curso – Bacharelado em Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

Técnicas de visão computacional podem ser ferramentas úteis para a automação de linhas de produção em tarefas como, por exemplo, a fabricação de componentes eletrônicos, a inspeção do acabamento em objetos metálicos, a produção de circuitos impressos, entre outros. Muitas vezes, existem etapas da produção em indústrias metalúrgicas, nas quais se necessita identificar objetos metálicos e sua respectiva orientação angular, seja para análise, separação ou mesmo somente para classificação desses objetos. O desenvolvimento deste projeto é motivado pela posterior aplicação em uma linha de produção real. A utilização do projeto deve reduzir os custos operacionais da linha, diminuindo a margem de erro e aumentando a velocidade da produção. O objetivo deste trabalho é desenvolver um sistema capaz de classificar objetos metálicos e medir seu respectivo ângulo de rotação com relação ao eixo x do plano cartesiano, sendo os objetos previamente conhecidos pelo sistema utilizando-se técnicas de aprendizagem de máquina. O desenvolvimento do projeto é dividido em três etapas. A primeira etapa consiste no estudo de técnicas de segmentação de imagem e implementação no software MATLAB. A segunda etapa consiste no estudo de técnicas de descrição de imagens e também implementação no MATLAB. Finalmente, na terceira etapa é implementado no MATLAB um sistema de redes neurais capazes de classificar objetos e medir seu ângulo. Para cada etapa do desenvolvimento do projeto adota-se a metodologia de desenvolvimento em espiral, onde a cada ciclo de desenvolvimento são agregadas novas funcionalidades ao sistema. Para o teste e a validação do sistema é desenvolvido um ambiente de testes, onde objetos especificamente selecionados são posicionados de forma automática, possibilitando a captura de imagens do objeto em diversas posições angulares. O sistema é dividido em três módulos: segmentação, descrição e classificação dos objetos. Após a aquisição da imagem do objeto a ser classificado o módulo de segmentação seleciona a área de interesse da imagem. No módulo de descrição são extraídas características da área de interesse da imagem, formando um descriptor que é fornecido ao terceiro módulo do sistema que classifica o objeto e mede a sua respectiva orientação angular. Portanto, o resultado do trabalho é um sistema capaz de classificar corretamente 100% dos objetos testados e medir o ângulo desses objetos com precisão de  $\pm 4.5^\circ$  no pior caso, utilizando para isso técnicas de processamento de imagens e aprendizado de máquina.

**Palavras-chave:** Processamento de Imagens, Análise de Componentes Principais, Descritores de Fourier, Classificação de Objetos, Reconhecimento de Padrões, Redes Neurais

## ABSTRACT

FRIESEN, Telmo. Computational vision system for classification and measure of the angular position of metallic objects. 69 f. Trabalho de Conclusão de Curso – Bacharelado em Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

Computer vision techniques can be useful tools for production line automation in tasks such as the manufacture of electronic components, inspection of finished metal objects, production of printed circuit boards, among others. Often, there are production stages in metallurgical industries in which the identification of metallic objects, and their respective angular orientation, is needed either for analysis, separation or even only to classify these objects. The objective of this work is to develop a system capable of classifying metallic objects and measure their respective angle of rotation with respect to the x axis of the Cartesian plane. The objects are previously known by the system using machine learning techniques. The project development is divided into three steps. The first step is the study of techniques for image segmentation and implementation in MATLAB software. The second step is the study of techniques for image description and also implementation in MATLAB. Finally, in the third step a set of neural networks capable of classifying objects and measuring their angle is implemented using MATLAB. For each step in the project development a spiral development methodology is adopted. In each cycle of the methodology new features are aggregated to the system. In order to test the system a test environment is developed, where specifically selected objects are placed automatically, allowing the capture of images of the object in different angular positions. The system is divided into three modules: segmentation, description, and classification of objects. After capturing an image of the object being classified, the segmentation module selects the region of interest from the image. In the description module features are extracted from the region of interest, forming a descriptor that is provided to the third module. The third module then classifies the object and measures its angular orientation. Therefore, the result of this work is a system able to classify objects and measure the angle of these objects using image processing and machine learning techniques.

**Keywords:** Image Processing, Principal Components Analysis, Fourier Descriptors, Object Classification, Pattern Recognition, Neural Networks

## LISTA DE FIGURAS

FIGURA 1 – VISÃO GERAL DO SISTEMA.	11
FIGURA 2 – IMAGENS COM E SEM RUÍDO E SEUS RESPECTIVOS HISTOGRA- MAS	15
FIGURA 3 – IMAGENS LIMIARIZADAS PELO MÉTODO DE <i>OTSU</i> E SEUS RES- PECTIVOS HISTOGRAMAS	16
FIGURA 4 – EXEMPLOS DE ELEMENTOS ESTRUTURANTES.	17
FIGURA 5 – EXEMPLO DE OPERAÇÃO MORFOLÓGICA: EROSÃO.	18
FIGURA 6 – EXEMPLO DE EROSÃO DO CONJUNTO A POR B.	19
FIGURA 7 – EXEMPLO DE DILATAÇÃO DO CONJUNTO A POR B.	19
FIGURA 8 – CONJUNTO A.	20
FIGURA 9 – CONJUNTO $A \circ B$ .	21
FIGURA 10 – CONJUNTO $A \bullet B$ .	21
FIGURA 11 – CONJUNTO A, DISCO MÁXIMO CENTRADO EM DIVERSOS PON- TOS Z E ESQUELETO REPRESENTADO PELAS LINHAS TRACEJA- DAS.	22
FIGURA 12 – EXEMPLO DE ESQUELETORIZAÇÃO E PODA.	24
FIGURA 13 – IMAGENS DE EXEMPLO E SUAS RESPECTIVAS MATRIZES DE CO- VARIÂNCIA.	26
FIGURA 14 – IMAGENS DE EXEMPLO E SEUS RESPECTIVOS AUTOVETORES.	27
FIGURA 15 – IMAGENS NORMALIZADAS.	28
FIGURA 16 – REPRESENTAÇÃO DA ÁREA PARA CÁLCULO DO MOMENTO DE INÉRCIA.	30
FIGURA 17 – IMAGEM ORIGINAL E IMAGENS RECONSTITUÍDAS UTILIZANDO <i>N</i> DESCRIPTORES DE <i>FOURIER</i> .	32
FIGURA 18 – MODELO DE <i>PERCEPTRON</i> .	34
FIGURA 19 – GRÁFICO DA FUNÇÃO DE ATIVAÇÃO SIGMOIDE.	35
FIGURA 20 – MODELO DE <i>MULTI-LAYER FEEDFORWARD NETWORK</i> .	36
FIGURA 21 – ESTRUTURAS DE REDES NEURAIS E SUA RESPECTIVA CAPACI- DADE DE DECISÃO.	37
FIGURA 22 – VISÃO GERAL DO SISTEMA DESENVOLVIDO.	42
FIGURA 23 – PRIMEIRA ETAPA DA SEGMENTAÇÃO.	44
FIGURA 24 – SEGUNDA ETAPA DA SEGMENTAÇÃO.	45
FIGURA 25 – PRIMEIRA ETAPA DA DESCRIÇÃO.	47
FIGURA 26 – ETAPA DE CLASSIFICAÇÃO.	49
FIGURA 27 – IMAGENS NORMALIZADAS PARA DIFERENTES ENTRADAS DO MESMO OBJETO.	50
FIGURA 28 – SEGUNDA ETAPA DA DESCRIÇÃO.	50
FIGURA 29 – REPRESENTAÇÃO DO AUTOVETOR (VERMELHO), DOS EIXOS DA IMAGEM (AMARELO) E DO EIXO DO OBJETO (AZUL).	51
FIGURA 30 – SEGUNDA ETAPA DA CLASSIFICAÇÃO.	52
FIGURA 31 – OBJETOS DE TESTE.	55
FIGURA 32 – MONTAGEM DO AMBIENTE DE TESTES.	56

FIGURA 33 – MONTAGEM DO AMBIENTE DE TESTES. ....	57
FIGURA 34 – MATRIZ DE CONFUSÃO PARA 8 OBJETOS (FORMANDO 16 CLASSES). ....	58
FIGURA 35 – ERRO MÉDIO DA ESTIMATIVA DO ÂNGULO PARA 8 OBJETOS. ..	59
FIGURA 36 – DESVIO PADRÃO DO ERRO DA ESTIMATIVA DO ÂNGULO PARA 8 OBJETOS. ....	59
FIGURA 37 – ERRO MÁXIMO DA ESTIMATIVA DO ÂNGULO PARA 8 OBJETOS. ..	60
FIGURA 38 – MATRIZ DE CONFUSÃO PARA 16 OBJETOS (FORMANDO 31 CLASSES). ....	61
FIGURA 39 – ERRO MÉDIO DA ESTIMATIVA DO ÂNGULO PARA 16 OBJETOS. ..	62
FIGURA 40 – DESVIO PADRÃO DO ERRO DA ESTIMATIVA DO ÂNGULO PARA 16 OBJETOS. ....	62
FIGURA 41 – ERRO MÁXIMO DA ESTIMATIVA DO ÂNGULO PARA 16 OBJETOS. ..	63
FIGURA 42 – MATRIZ DE CONFUSÃO PARA 8 OBJETOS - REDE COM 1 SAÍDA. ..	64

## **LISTA DE TABELAS**

TABELA 1 – DESCRIPTOR PARA PRIMEIRA ETAPA .....	47
TABELA 2 – ETAPAS DO DESENVOLVIMENTO EM ESPIRAL .....	66

## **LISTA DE SIGLAS**

RGB      *Reg, Green and Blue*

PCA      *Principal components analysis*

## SUMÁRIO

<b>1 INTRODUÇÃO .....</b>	<b>10</b>
1.1 MOTIVAÇÃO E JUSTIFICATIVA .....	10
1.2 OBJETIVOS .....	10
1.3 METODOLOGIA .....	11
1.4 ESTRUTURA DO DOCUMENTO .....	12
<b>2 FUNDAMENTAÇÃO TEÓRICA .....</b>	<b>13</b>
2.1 SEGMENTAÇÃO .....	13
2.1.1 Subtração de fundo .....	13
2.1.2 Limiarização .....	14
Limiarização global pelo método de <i>Otsu</i> .....	15
2.1.3 Operações morfológicas básicas .....	16
Erosão .....	18
Dilatação .....	19
Abertura e Fechamento .....	20
2.1.4 Esqueletonização .....	21
Poda ( <i>Pruning</i> ) .....	23
2.1.5 Normalização .....	24
Matriz de covariâncias .....	24
Autovetores e autovalores .....	26
Analise de componentes principais .....	26
Normalização .....	27
2.2 DESCRIÇÃO .....	28
2.2.1 Propriedades de regiões .....	28
Área .....	29
Área convexa .....	29
Área Preenchida .....	29
Solidez .....	29
Extensão .....	29
Excentricidade .....	29
Tamanho do eixo principal e eixo secundário .....	30
Número de Euler .....	30
Centroide .....	30
2.2.2 Descritores de <i>Fourier</i> .....	31
2.3 CLASSIFICAÇÃO .....	33
2.3.1 Redes neurais .....	33
Treinamento de <i>perceptrons</i> .....	37
Treinamento de <i>multi-layer perceptrons</i> .....	39
2.4 CONSIDERAÇÕES .....	41
<b>3 DESENVOLVIMENTO .....</b>	<b>42</b>
3.1 SEGMENTAÇÃO: ETAPA 1 .....	43
3.2 SEGMENTAÇÃO: ETAPA 2 .....	45

3.3 DESCRIÇÃO: ETAPA 1 .....	46
3.4 CLASSIFICAÇÃO .....	47
3.5 DESCRIÇÃO: ETAPA 2 .....	49
3.6 ESTIMATIVA DE ÂNGULO .....	50
3.7 CONSIDERAÇÕES .....	53
<b>4 TESTES E ANÁLISE DE RESULTADOS .....</b>	<b>54</b>
4.1 AMBIENTE DE TESTES .....	54
4.2 TESTES .....	57
4.2.1 Primeiro teste: 8 objetos .....	57
4.2.2 Segundo teste: 16 objetos .....	60
4.3 CONSIDERAÇÕES .....	63
<b>5 CONSIDERAÇÕES FINAIS .....</b>	<b>65</b>
5.1 DISCUSSÃO DO DESENVOLVIMENTO E DOS RESULTADOS .....	65
5.2 TRABALHOS FUTUROS .....	67
<b>REFERÊNCIAS .....</b>	<b>68</b>

## 1 INTRODUÇÃO

Técnicas de visão computacional podem ser ferramentas úteis para a automação de linhas de produção em tarefas como, por exemplo, a fabricação de componentes eletrônicos (LIN, 2008, 2007) a inspeção do acabamento em objetos metálicos (ZHENG; KONG; NAHAVANDI, 2002), a produção de circuitos impressos (MAR; YARLAGADDA; FOOKES, 2011; ZENG; MA; ZHENG, 2011), entre outros.

Muitas vezes, existem etapas da produção em indústrias metalúrgicas, nas quais se necessita identificar objetos metálicos e sua respectiva orientação angular, seja para análise, separação ou mesmo somente para classificação desses objetos.

A grande quantidade de objetos distintos que podem estar em uma mesma linha de produção ou linha de montagem pode confundir os trabalhadores, estando estes portanto sujeitos a erros e falhas. A identificação automática de objetos, assim como a identificação automática de sua posição, tendem a diminuir o erro cometido na classificação ou reposicionamento manual para cada objeto. Diminuindo o erro de classificação diminui-se consequentemente o custo gerado por uma linha de montagem parada e dispensa-se a necessidade de parte dos operadores.

### 1.1 MOTIVAÇÃO E JUSTIFICATIVA

A principal motivação para o desenvolvimento deste projeto é a posterior aplicação em uma linha de produção real. Espera-se que a utilização do projeto reduza os custos operacionais da linha de produção, diminua a margem de erro e aumente a velocidade da produção. A aplicação específica do projeto não será detalhada devido ao posterior interesse comercial do autor do projeto.

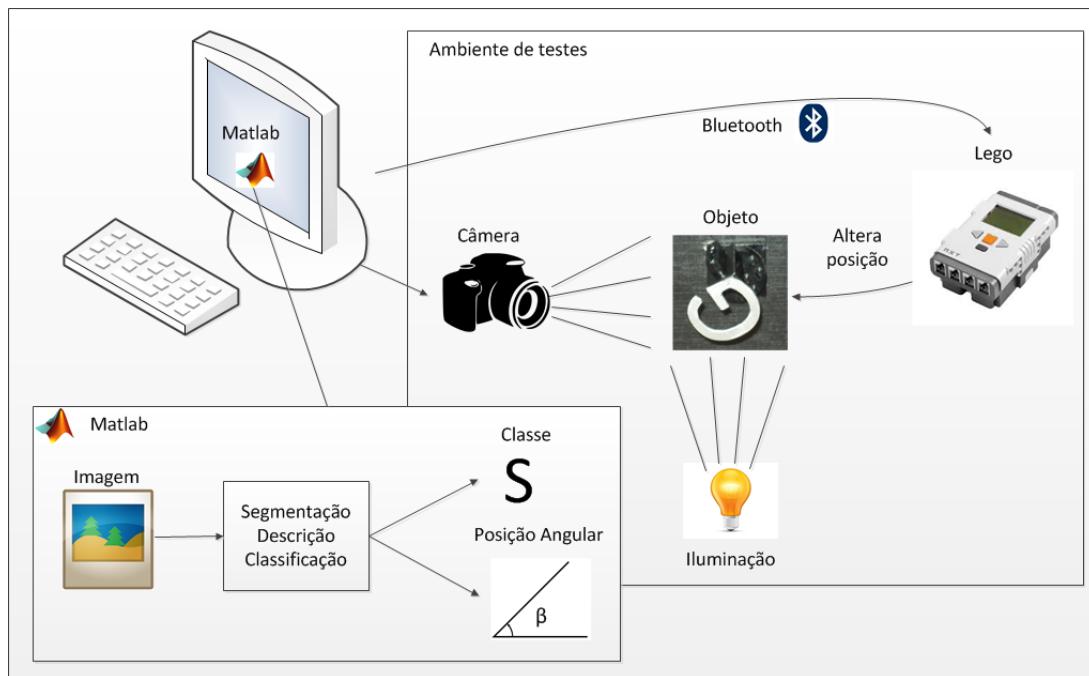
### 1.2 OBJETIVOS

O objetivo geral deste trabalho é desenvolver um sistema capaz de classificar objetos metálicos e medir seu respectivo ângulo de rotação com relação ao eixo x do plano cartesiano, sendo os ob-

jetos previamente conhecidos pelo sistema utilizando-se técnicas de aprendizagem de máquina.

Para tanto, pode-se estabelecer três objetivos específicos que possibilitarão o cumprimento do objetivo geral. O primeiro objetivo específico é a análise e aplicação de técnicas de segmentação de imagens, como algoritmos de detecção de bordas, algoritmos de limiarização, segmentação baseada em crescimento de regiões. O segundo objetivo é analisar e aplicar técnicas de descrição de imagens, como algoritmos de descrição de contornos, descrição de regiões, entre outros. O terceiro objetivo é o emprego de uma técnica de aprendizagem de máquina, como redes neurais, que possibilite a classificação e determinação do ângulo de objetos previamente conhecidos pelo sistema.

Na figura 1 mostra-se a visão geral do sistema a ser desenvolvido. Os objetivos do projeto restringem-se ao desenvolvimento do sistema em MATLAB representado no quadro “Matlab” da figura. No entanto para validação do sistema será necessário o desenvolvimento do ambiente de testes.



**Figura 1:** Visão geral do sistema.

**Fonte:** Autoria própria

### 1.3 METODOLOGIA

O desenvolvimento do projeto é dividido em três etapas. A primeira etapa consiste no estudo de técnicas de segmentação de imagem e implementação no software MATLAB. A se-

gunda etapa consiste no estudo de técnicas de descrição de imagens e também implementação no MATLAB. Finalmente, na terceira etapa é implementado no MATLAB um sistema de redes neurais capazes de classificar o objeto e medir seu ângulo. Para cada etapa do desenvolvimento do projeto adota-se a metodologia de desenvolvimento em espiral, onde a cada ciclo de desenvolvimento são agregadas novas funcionalidades ao sistema.

Para o teste e a validação do sistema é desenvolvido um ambiente de testes, onde objetos especificamente selecionados são posicionados de forma automática, possibilitando a captura de imagens do objeto em diversas posições angulares.

#### 1.4 ESTRUTURA DO DOCUMENTO

Esta monografia está dividida em cinco capítulos: o primeiro corresponde à introdução, na qual são apresentadas a motivação, os objetivos e a metodologia empregada. O segundo capítulo contém a fundamentação teórica do projeto, ou seja, o estudo das técnicas e algoritmos utilizados durante o trabalho. Nesse capítulo são abordadas técnicas de segmentação de imagens, de descrição e de classificação de imagens. O terceiro capítulo apresenta o desenvolvimento do projeto, detalhando cada parte do sistema desenvolvido. O quarto capítulo apresenta o ambiente de testes desenvolvido para validar o sistema, os testes efetuados e os resultados obtidos. Finalmente o quinto capítulo contém considerações finais sobre o projeto.

## 2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a fundamentação teórica, ou seja, o estudo das técnicas e algoritmos utilizados durante o trabalho. O capítulo divide-se em três seções.

A seção 2.1 apresenta técnicas e algoritmos que serão úteis para realizar a segmentação de imagens. A segmentação de imagens consiste em subdividir a imagem original em regiões ou objetos até que se tenha identificado a região ou objeto de interesse.

A seção 2.2 apresenta técnicas e algoritmos para realizar a descrição de imagens. Após a segmentação de uma imagem é necessário representar a imagem para que se possa processá-la. Existem duas abordagem para esse problema (GONZALEZ; WOODS, 2006): representação por meio de características externas da região, como bordas, ou representação por meio de características internas, como textura. A representação por características externas normalmente é utilizada quando se está interessado na forma da região. Já a representação por características internas é utilizada quando o foco está na cor ou textura da região. Nessa seção aborda-se principalmente a representação por meio de características externas de imagens.

Finalmente, a seção 2.3 apresenta uma visão geral de reconhecimento de padrões utilizando redes neurais. O reconhecimento de padrões é feito tomando como base descritores, que podem ser obtidos a partir de técnicas como as vistas na seção 2.2.

### 2.1 SEGMENTAÇÃO

Esta seção apresenta técnicas e algoritmos para realizar a segmentação de imagens, tais como técnicas de subtração de fundo, limiarização, operações morfológicas, esqueletonização e análise de componentes principais.

#### 2.1.1 Subtração de fundo

Técnicas de subtração de fundo são amplamente empregadas para a detecção de movimento em sequencias de imagens de vídeo. A detecção do movimento se dá pela subtração da imagem

de referência chamada “fundo”, ou em inglês *background image*, da imagem a ser segmentada. A imagem de fundo não deve possuir objetos em movimento e deve estar sempre atualizada com relação às variações de iluminação (PICCARDI, 2004).

O método mais simples de subtração de fundo consiste em subtrair a imagem de fundo da imagem a ser segmentada e em seguida aplicar um limiar no resultado. Pode-se resumir esse método na equação 1. Onde,  $D(t + 1)$  é a imagem segmentada,  $V(x, y, t + 1)$  é a imagem de entrada,  $V(x, y, t)$  é a imagem de referência, ou fundo, e  $Th$  é um limiar, que pode ser escolhido conforme os métodos apresentados na seção 2.1.2.

$$D(t + 1) = |V(x, y, t + 1) - V(x, y, t)| > Th \quad (1)$$

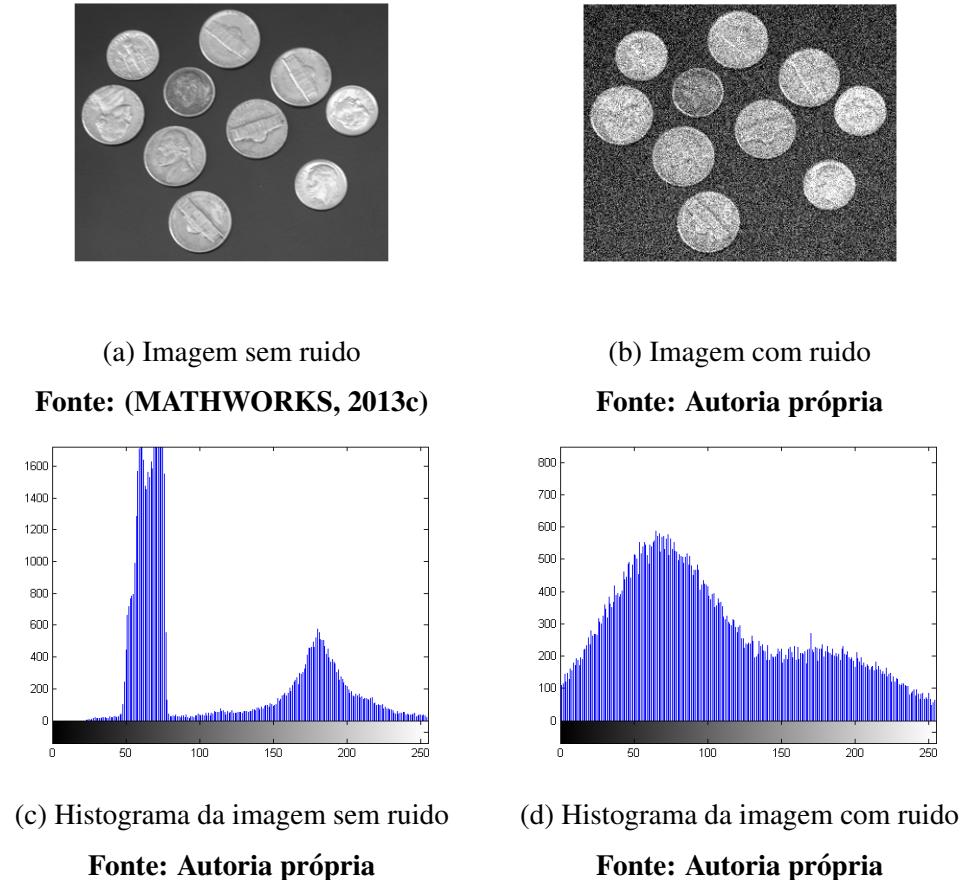
Existem outros métodos mais complexos, como *Running Gaussian average*, *Temporal median filter*, entre outros, que geram melhores resultados com fundos que não são completamente estáticos (PICCARDI, 2004), porém não são de interesse para este projeto.

### 2.1.2 Limiarização

Pela simplicidade na implementação e pela baixa complexidade computacional os algoritmos de limiarização são bastante utilizados na segmentação de imagens. Os métodos de limiarização consistem na partição da imagem baseado na intensidade dos pixels. Caso a partição seja feita com base em um único limiar de intensidade a limiarização se diz global. Caso a partição seja feita levando em conta os valores de intensidade dos pixels em uma região de vizinhança de um certo pixel, então o método é chamado de limiarização local (GONZALEZ; WOODS, 2006). A equação 2 descreve esse processo,  $g(x, y)$  é a imagem limiarizada,  $f(x, y)$  é a imagem de entrada e  $T$  é um limiar, que pode variar para diferentes valores de  $x$  e  $y$  caso a limiarização seja local.

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases} \quad (2)$$

Na figura 2 mostram-se duas imagens e seus respectivos histogramas. A figura 2c mostra o histograma para a imagem sem ruído (figura 2a) e a figura 2d mostra o histograma para a imagem com ruido gaussiano (figura 2b). Como pode-se ver na figura 2c, a escolha de um valor para o limiar  $T$  para a imagem sem ruído é uma tarefa fácil, porém a escolha do limiar para a imagem com ruído da figura 2d não é trivial. A seguir descreve-se um método para obtenção de limiar para limiarização global que pode ser aplicado nos dois casos.



**Figura 2:** Imagens com e sem ruído e seus respectivos histogramas

Limiarização global pelo método de *Otsu*

O método para determinar o valor do limiar  $T$  proposto por *Otsu* é um método ótimo que busca maximizar a variância entre as duas classes. Isso baseia-se no fato de que classes bem divididas possuem valores de intensidades distintos, e o valor ótimo para  $T$  é o valor tal que haja maior separação entre as intensidades (GONZALEZ; WOODS, 2006).

O cálculo do valor limiar pelo método de *Otsu* se dá pela equação 3.

$$\sigma_B^2(t) = \frac{[m_G P_1(t) - m(t)]^2}{P_1(t)[1 - P_1(t)]} \quad (3)$$

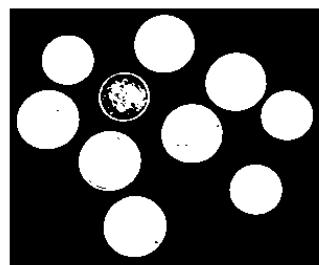
Onde:

- $\sigma_B^2(t)$  é a variância entre as classes para um valor limiar  $t$ ;
- $m_G$  é a média global de intensidades da imagem;

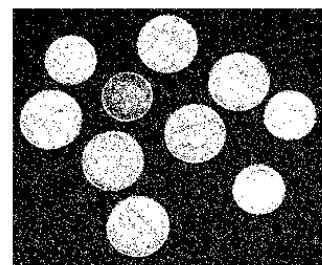
- $P_1(t)$  é a probabilidade de um pixel qualquer da imagem estar na classe 1 dado o valor limiar  $t$ .
- $m(t)$  é a média acumulada das intensidades de  $t = 0$  até a intensidade  $t$ ;

Portanto, para o cálculo do valor limiar pelo método de *Otsu* basta encontrar o valor limiar  $t$  tal que a variância entre as classes,  $\sigma_B^2(t)$ , é máximo. A dedução da equação 3 está fora do escopo deste projeto, mas pode ser encontrada em (GONZALEZ; WOODS, 2006).

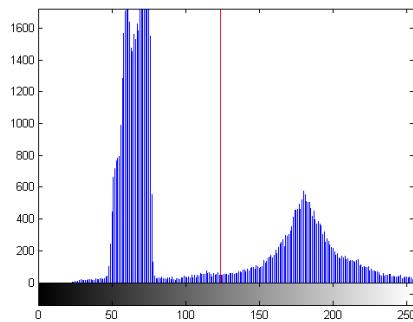
Na figura 3 mostram-se as imagens da figura 2 limiarizadas pelo método de *Otsu* e seus respectivos valores de limiarização.



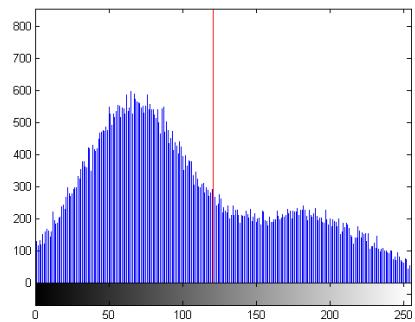
(a) Imagem sem ruido limiarizada



(b) Imagem com ruido limiarizada



(c) Histograma da imagem sem ruido com o valor do limiar T



(d) Histograma da imagem com ruido com o valor do limiar T

**Figura 3:** Imagens limiarizadas pelo método de *Otsu* e seus respectivos histogramas

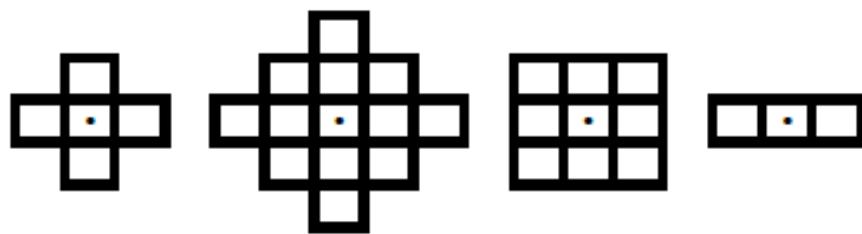
**Fonte:** Autoria própria

### 2.1.3 Operações morfológicas básicas

Operações morfológicas como abertura e fechamento podem ser úteis para filtrar imagens. Principalmente para filtrar imagens binárias onde a operação de fechamento pode ser aplicada para reduzir ruido do tipo pimenta (pixels pretos espalhados aleatoriamente pela imagem).

Para o estudo das operações morfológicas de abertura e fechamento precisa-se primeiramente esclarecer alguns pontos:

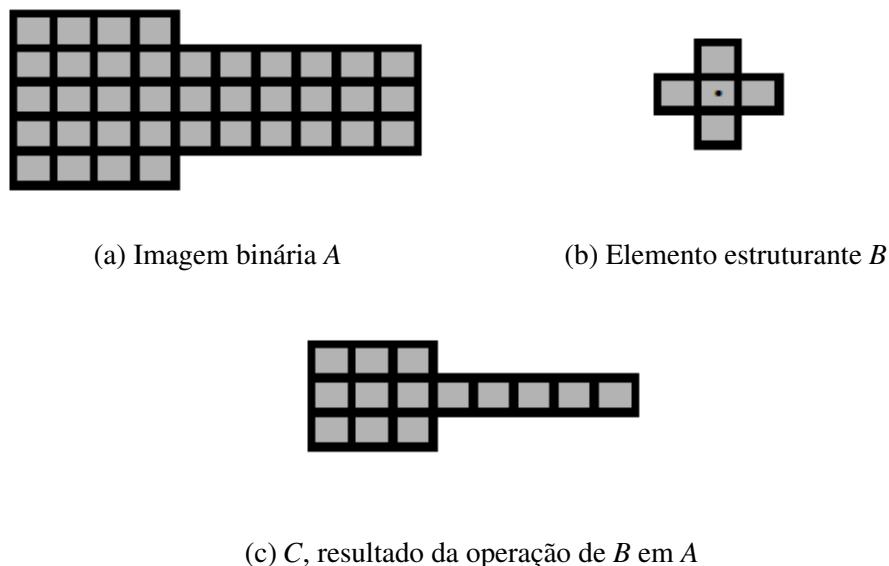
- Podemos representar imagens binárias como sendo um conjunto de pixels no espaço  $Z^2$ , onde cada pixel é representado por uma tupla  $(x, y)$  e  $x$  e  $y$  são as coordenadas do pixel.
- A translação de um conjunto  $B$  para um ponto  $z = (z_1, z_2)$ , denotado por  $(B)_z$  é o conjunto de pontos em  $B$  com as respectivas coordenadas  $(x, y)$  tendo sido substituídas pela coordenadas  $(x + z_1, y + z_2)$ .
- Elemento estruturante é um conjunto de pixels que é utilizado para fazer operações sobre uma imagem. Cada elemento estruturante possui um pixel de origem. A figura 4 mostra exemplos de elementos estruturantes e suas respectivas origens.



**Figura 4:** Exemplos de elementos estruturantes.

**Fonte:** Autoria própria

- O elemento estruturante pode ser utilizado para fazer operações sobre uma imagem binária. Considere por exemplo a operação no conjunto  $A$ , feita pelo elemento estruturante  $B$  (figuras 5a e 5b): Cria-se um novo conjunto  $C$  sobrepondo  $B$  em  $A$  da maneira que a origem de  $B$  passe por todos os pixels de  $A$ . Em cada ponto que a origem de  $B$  passa verifica-se se  $B$  está completamente contido em  $A$ . Se estiver marca-se esse ponto no conjunto  $C$ . A figura 5c mostra o resultado da operação, ou seja, o conjunto  $C$  (como será descrito adiante em mais detalhes, esta é uma operação morfológica de erosão).



**Figura 5:** Exemplo de operação morfológica: erosão.

**Fonte:** Autoria própria

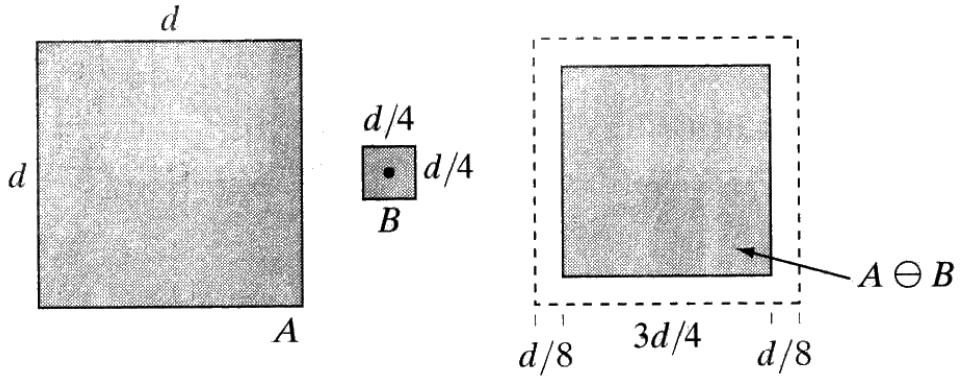
A seguir apresentam-se as operações de erosão e de dilatação, que são necessárias para as operações de abertura e fechamento que são apresentadas na sequência.

### Erosão

Sendo  $A$  e  $B$  conjuntos no espaço  $Z^2$ , a erosão do conjunto  $A$  pelo elemento estruturante  $B$ , denotada por  $A \ominus B$ , é definida na equação 4 (GONZALEZ; WOODS, 2006).

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \quad (4)$$

Ou seja, a erosão de  $A$  por  $B$  é o conjunto de todos os pontos  $z$  tal que  $B$ , transladado por  $z$ , está completamente contido em  $A$ . Pode-se ver melhor esse processo na figura 6, onde o conjunto  $A$  é erodido pelo elemento estruturante  $B$  formando o conjunto  $A \ominus B$ .



**Figura 6:** Exemplo de erosão do conjunto  $A$  por  $B$ .

Fonte: (GONZALEZ; WOODS, 2006)

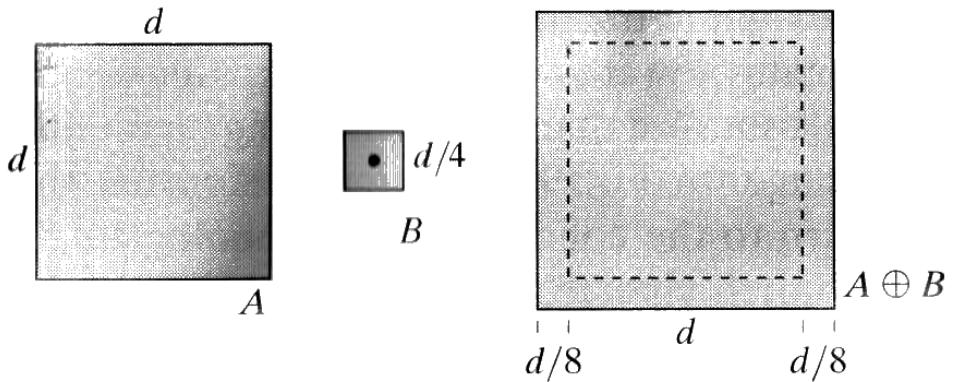
### Dilatação

Sendo  $A$  e  $B$  conjuntos no espaço  $Z^2$ , a dilatação do conjunto  $A$  pelo elemento estruturante  $B$ , denotada por  $A \oplus B$ , é definida na equação 5 (GONZALEZ; WOODS, 2006).

$$A \oplus B = \{z \mid [(\hat{B})_z \cap A] \subseteq A\} \quad (5)$$

Onde  $\hat{B}$  é o conjunto  $B$  espelhado. Quando o elemento estruturante é simétrico então  $\hat{B} = B$ .

A equação 5 estabelece que a dilatação do conjunto  $A$  pelo elemento estruturante  $B$  é o conjunto de todos os pontos  $z$  tal que se  $\hat{B}$  e  $A$  se sobrepõem em pelo menos um ponto, então  $z$  faz parte de  $A \oplus B$ . Pode-se ver melhor esse processo na figura 7, onde o conjunto  $A$  é dilatado pelo elemento estruturante  $B$  formando o conjunto  $A \oplus B$ .



**Figura 7:** Exemplo de dilatação do conjunto  $A$  por  $B$ .

Fonte: (GONZALEZ; WOODS, 2006)

## Abertura e Fechamento

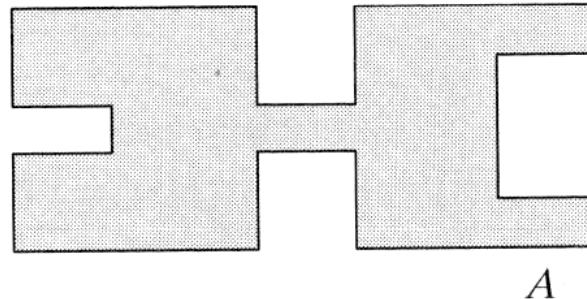
A abertura do conjunto  $A$  pelo elemento estruturante  $B$  é definida pela equação 6 (GONZALEZ; WOODS, 2006).

$$A \circ B = (A \ominus B) \oplus B \quad (6)$$

O fechamento do conjunto  $A$  pelo elemento estruturante  $B$  é definido pela equação 7 (GONZALEZ; WOODS, 2006).

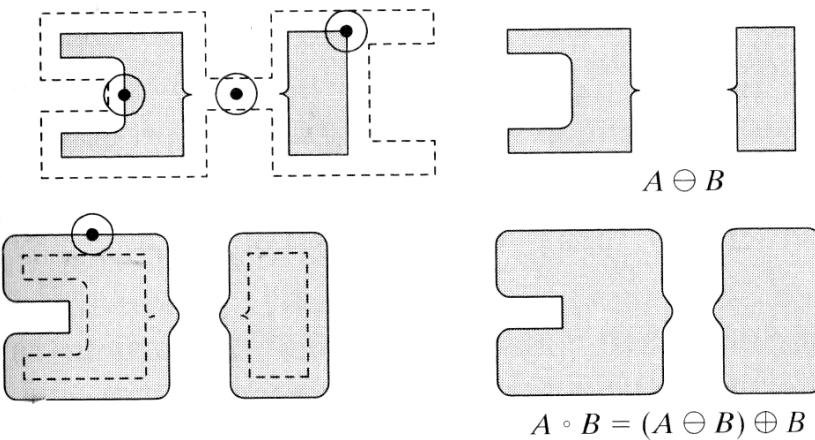
$$A \bullet B = (A \oplus B) \ominus B \quad (7)$$

Na figura 9 pode-se ver o resultado da execução da operação de abertura sobre o conjunto  $A$  da figura 8. Na figura 10 pode-se ver o resultado da execução da operação de fechamento sobre o mesmo conjunto. Como pode-se ver, a operação de abertura “abre” a imagem em pontos estreitos, já a operação de fechamento “fecha” a imagem onde existem cavidades pequenas.



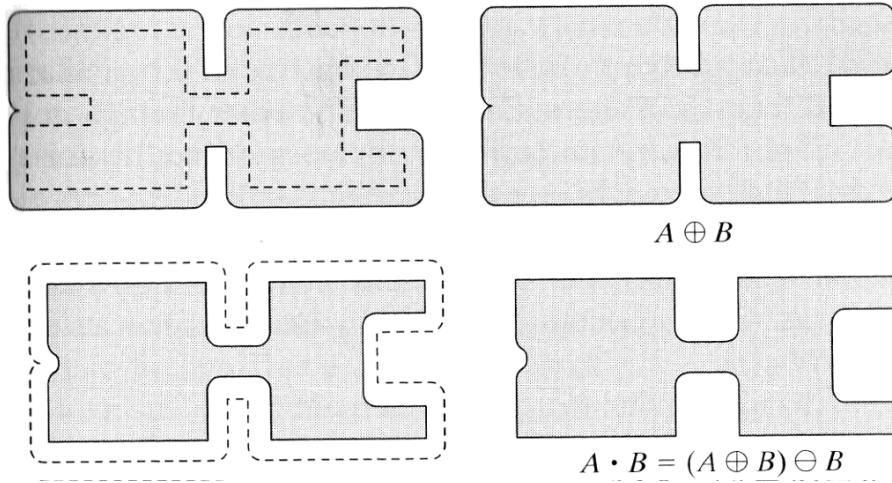
**Figura 8:** Conjunto  $A$ .

**Fonte:** (GONZALEZ; WOODS, 2006)



**Figura 9:** Conjunto  $A \circ B$ .

**Fonte:** (GONZALEZ; WOODS, 2006)



**Figura 10:** Conjunto  $A \bullet B$ .

**Fonte:** (GONZALEZ; WOODS, 2006)

#### 2.1.4 Esqueletonização

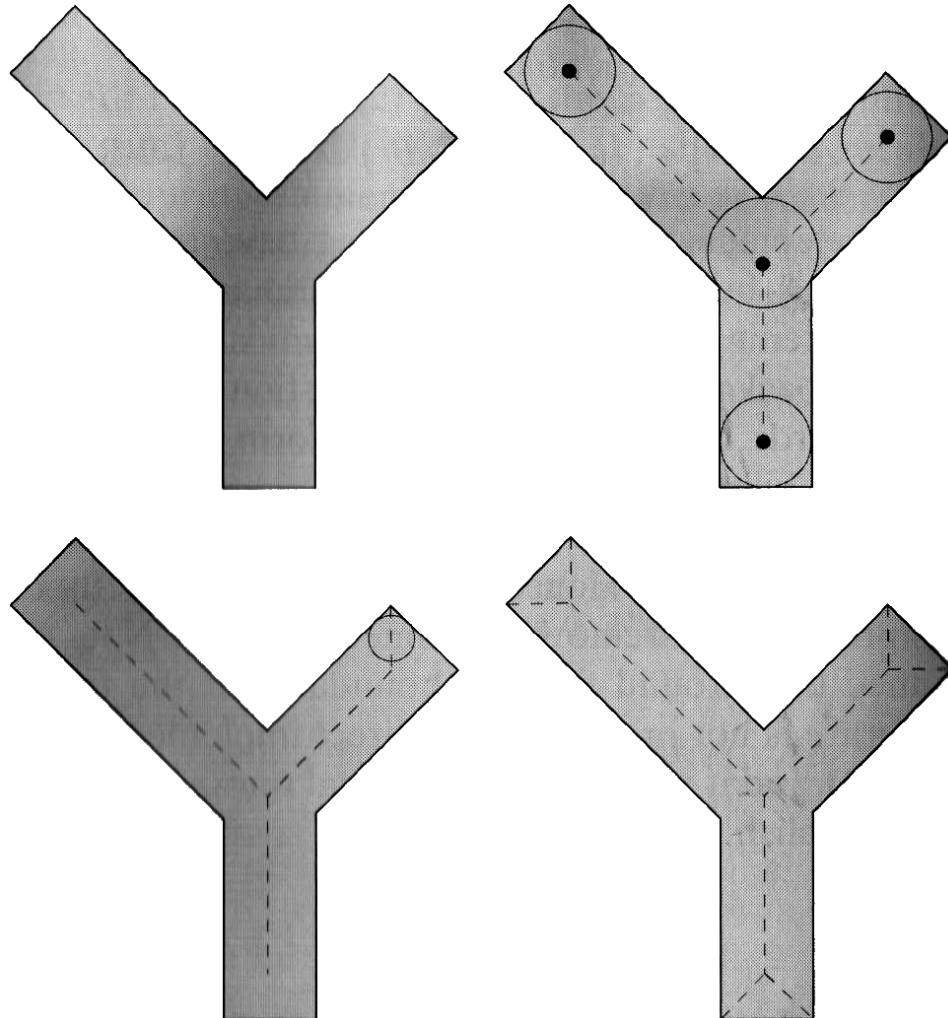
Define-se esqueleto conforme segue (GONZALEZ; WOODS, 2006).

Se  $S(A)$  é o esqueleto de  $A$  e  $(D)_z$  o maior disco centrado em  $z$  e contido em  $A$ , pode se dizer que :

- Se  $z$  faz parte de  $S(A)$  então não é possível encontrar um disco maior que  $(D)_z$ , não necessariamente centrado em  $z$ , que contenha  $(D)_z$  e que esteja completamente inclusa em  $A$ ;

- O disco  $(D)_z$  toca a borda de  $A$  em pelo menos dois pontos distintos.

Isso pode ser visto melhor na figura 11.



**Figura 11:** Conjunto  $A$ , disco máximo centrado em diversos pontos  $z$  e esqueleto representado pelas linhas tracejadas.

Fonte: (GONZALEZ; WOODS, 2006)

Pode-se definir também o esqueleto  $S(A)$  em função de erosões e aberturas sucessivas como mostrado nas equações 8, 9, 10 e 11 (GONZALEZ; WOODS, 2006).

$$S(A) = \bigcup_{k=0}^K S_k(A) \quad (8)$$

Com:

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B \quad (9)$$

Onde  $B$  é um elemento estruturante e  $(A \ominus kB)$  indica  $k$  sucessivas erosões de  $A$ :

$$(A \ominus kB) = (((\dots((A \ominus B) \ominus B) \ominus \dots) \ominus B) \ominus B) \dots \quad (10)$$

Sendo  $K$  o ultimo passo antes de  $A$  erodir para um conjunto vazio. Ou seja:

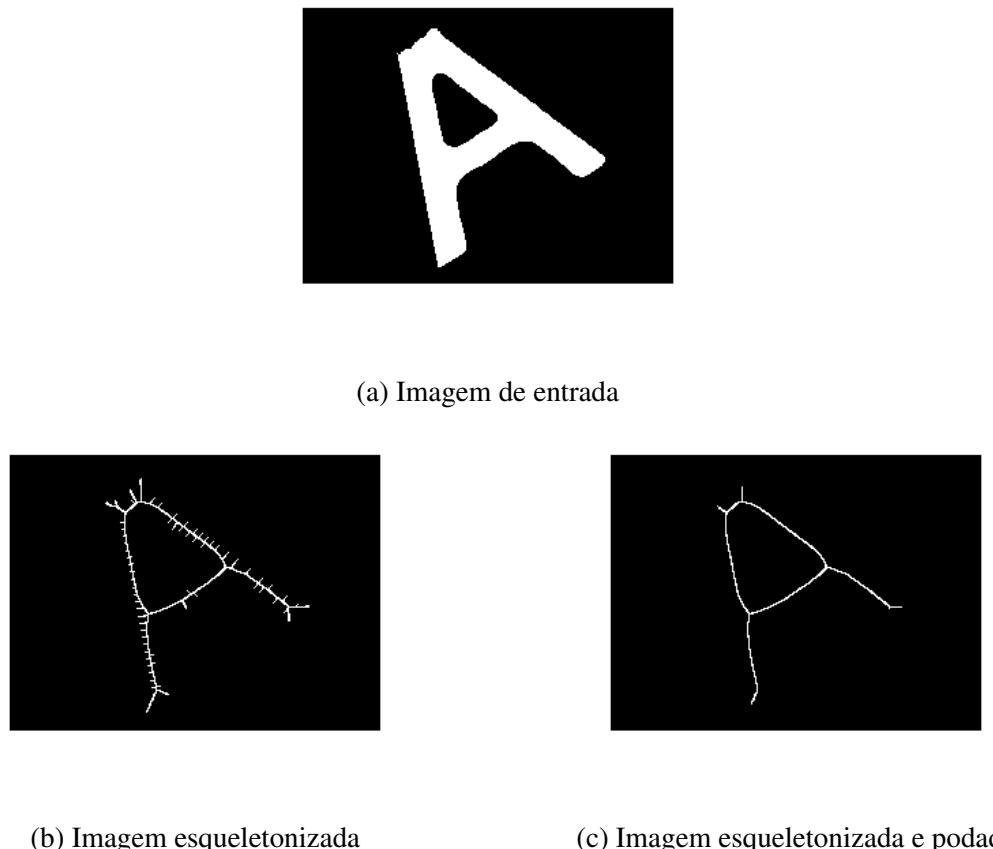
$$K = \max \{k \mid (A \ominus kB) \neq \emptyset\} \quad (11)$$

### Poda (*Pruning*)

Aplicando-se o método de esqueletonização mencionado anteriormente em figuras reais, como a figura 12a, pode-se obter resultados como a figura 12b. Devido a esse tipo de resultados aplica-se após a esqueletonização um método de poda. A poda do esqueleto consiste em avaliar o tamanho dos ramos gerados, caso sejam menores que um valor limiar  $n$  são removidos. Isso pode ser feito conforme explicado a seguir.

Considere todos os pixels finais dos ramos, ou seja, os pixels do esqueleto que possuem somente um pixel como vizinho. A partir de cada pixel final remove-se o pixel e chama-se seu vizinho de pixel final. Esse passo é executado por  $n$  vezes para cada ramo, ou até que o novo pixel final tenha mais de um pixel vizinho, não sendo mais chamado de pixel final a partir desse ponto. Em seguida os pixels finais resultantes são dilatados com um elemento estruturante quadrado de uns com tamanho  $2n + 1$ . Finalmente o conjunto resultante é interseccionado com o conjunto original do esqueleto formando o esqueleto podado com limiar  $n$ .

A figura 12c mostra o método de poda aplicado com um limiar de 20 pixels.



**Figura 12:** Exemplo de esqueletoização e poda.

**Fonte:** Autoria própria

### 2.1.5 Normalização

A análise de componentes principais pode ser empregada para normalizar imagens em função de rotação, translação e tamanho (GONZALEZ; WOODS, 2006). Explica-se a seguir alguns conceitos necessários para efetuar a normalização utilizando a análise de componentes principais e em seguida o processo de normalização utilizando componentes principais.

#### Matriz de covariâncias

Considere as imagens da figura 13. Os pontos (pixels) da imagem, podem ser considerados como sendo vetores  $x_k = (x_1, x_2)^T$ , onde  $x_1$  e  $x_2$  são as coordenadas do ponto  $x_k$ , e  $T$  indica a matriz transposta. Os pontos  $x$  portanto formam uma população de vetores no espaço  $Z^2$ . Sendo assim pode-se calcular a matriz de covariâncias  $C_x$  e o vetor da média  $m_x$  para a população de vetores.

Para  $K$  amostras de uma população aleatória pode-se aproximar a média  $m_x$  pela equação 12 (GONZALEZ; WOODS, 2006).

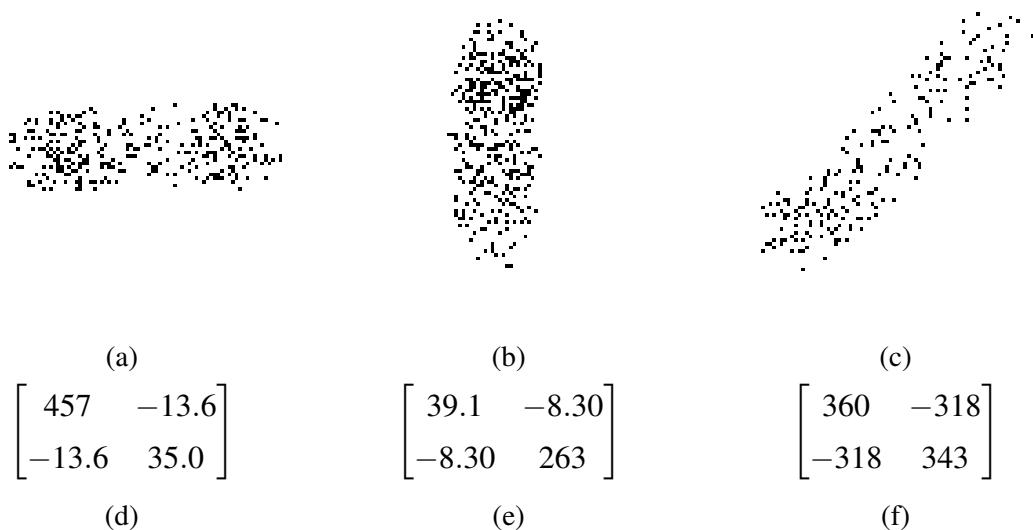
$$m_x = \frac{1}{K} \sum_{k=1}^K x_k \quad (12)$$

Agora utilizando a média  $m_x$ , pode-se aproximar a matriz de covariâncias  $C_x$  pela equação 13 (GONZALEZ; WOODS, 2006).

$$C_x = \frac{1}{K} \sum_{k=1}^K (x_k x_k^T) - m_x m_x^T \quad (13)$$

Na matriz de covariâncias os elementos  $c_{ii}$  de  $C_x$  representam a variância entre os valores na coordenada  $x_i$ . Os elementos  $c_{ij}$  representam a covariância entre os valores na coordenada  $x_i$  com os valores da coordenada  $x_j$ . Portanto, a matriz de covariância possui informações que dizem respeito à distribuição dos pixels da imagem.

Pode-se ver isso melhor nas matrizes de covariância apresentadas nas figuras 13d, 13e e 13f que foram calculadas para as imagens das figuras 13a, 13b e 13c respectivamente. Percebe-se que na figura 13d a maior variância ocorre entre os elementos da coordenada  $x_1$ , visto que  $c_{11}$  possui o maior valor na matriz de covariâncias. Semelhantemente percebe-se que na figura 13e a maior variância ocorre entre os elementos da coordenada  $x_2$ . A terceira imagem, figura 13f, mostra uma distribuição mais continua em  $x_1$  e  $x_2$ , mas valores maiores para as covariâncias entre os elementos das coordenadas  $x_1$  e  $x_2$ . A terceira imagem mostra portanto o relacionamento entre as duas coordenadas da imagem: quando  $x_1$  aumenta,  $x_2$  tende a aumentar também, e vice-versa.



**Figura 13:** Imagens de exemplo e suas respectivas matrizes de covariância.

**Fonte:** Autoria própria

## Autovetores e autovalores

Um autovetor de uma matriz  $A$  é um vetor não nulo  $v$ , tal que quando a matriz é multiplicada por  $v$  tem-se um vetor que é múltiplo de  $v$  por uma constante  $\lambda$  denominada autovalor. Ou seja, a equação 14 é verdadeira (LEON, 1998).

$$A\gamma \equiv \lambda\gamma \quad (14)$$

### Algumas propriedades de autovetores:

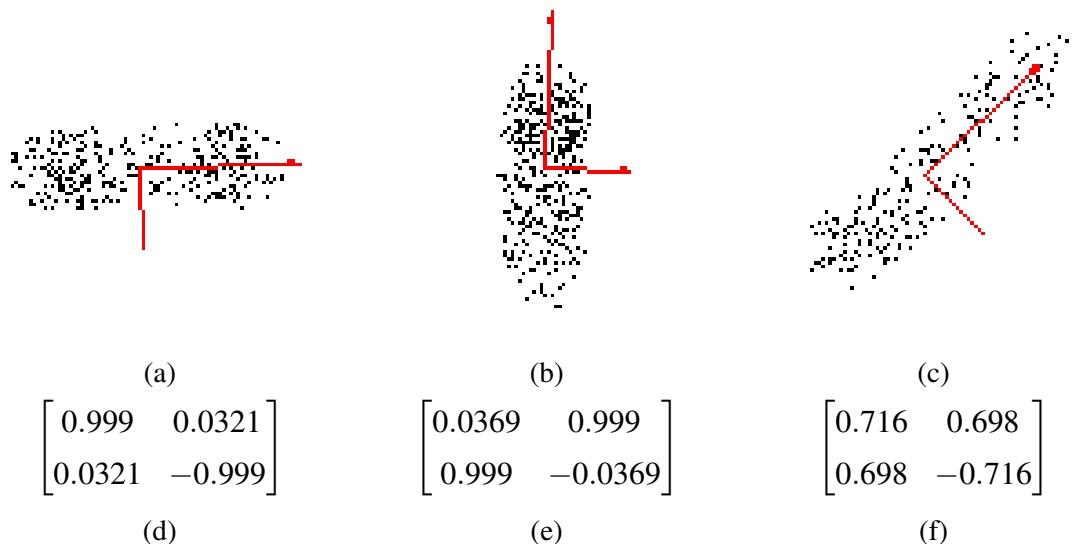
- Os autovetores podem ser calculados apenas para matrizes quadradas (LEON, 1998);
  - Para uma matriz  $n \times n$  existem  $n$  autovetores (LEON, 1998);
  - Todos os autovetores são ortogonais entre si (GONZALEZ; WOODS, 2006).

## Analise de componentes principais

Pode-se demonstrar que o autovetor associado ao maior autovalor de uma matriz de co-variâncias aponta na direção de maior variação dos dados (GHAOUI, 2013). Pode-se demonstrar também que o segundo autovetor, associado ao segundo maior autovalor, aponta na segunda

direção de maior variação dos dados. A esses autovetores se dá o nome de componentes principais da imagem.

Considere por exemplo as imagens da figura 14. Calculamos os autovetores para cada uma das imagens tomando como base as matrizes de covariância apresentadas nas figuras 13d, 13e e 13f. Os autovetores podem ser vistos nas figuras 14d, 14e e 14f. O autovetor associado ao maior autovalor está na primeira linha de cada matriz e o segundo autovetor está na segunda linha. Os autovetores foram representados sobre as respectivas imagens nas figuras 15a, 15b e 15c.



**Figura 14:** Imagens de exemplo e seus respectivos autovetores.

**Fonte:** Autoria própria

Vê-se portanto que é possível definir um eixo absoluto para cada imagem com base na distribuição dos pixels sobre a imagem.

### Normalização

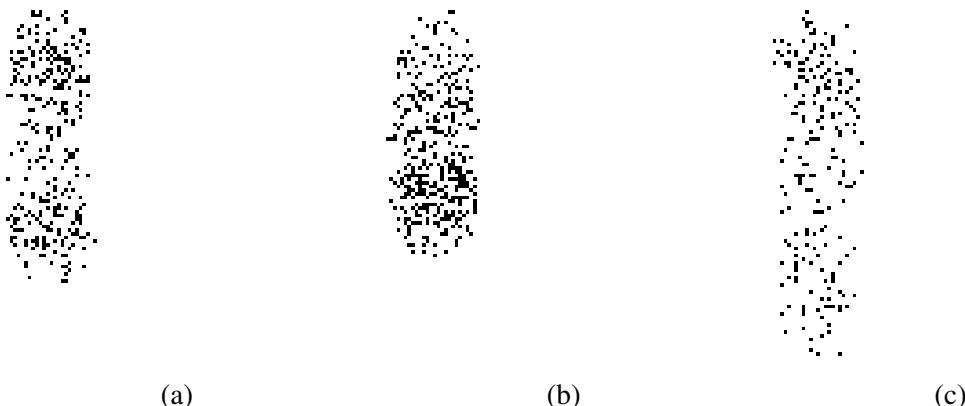
A normalização das imagens pode ser feita por meio de uma matriz de transformação que mapeia valores do sistema de coordenadas original da imagem para o sistema de coordenadas formado pelos autovetores da imagem (GONZALEZ; WOODS, 2006). A transformação em questão pode ser expressa pela equação 15.

$$y = A(x - m_x) \quad (15)$$

Onde:

- $y$  é o valor no novo sistema de coordenadas formado pelos autovetores.
- $x$  o valor da coordenada no sistema original.
- $A$  a matriz de autovetores.
- $m_x$  o vetor da média.

As imagens da figura 13 podem ser vistas normalizadas na figura 15.



**Figura 15:** Imagens normalizadas.

**Fonte:** Autoria própria

Caso se deseje normalizar a imagem com relação ao tamanho pode-se dividir o valor das coordenadas  $y$  pelos autovalores correspondentes (GONZALEZ; WOODS, 2006).

## 2.2 DESCRIÇÃO

Esta seção apresenta técnicas e algoritmos para realizar a descrição de imagens, tais como extração de propriedades básicas de regiões e descritores de *Fourier*.

### 2.2.1 Propriedades de regiões

Propriedades básicas de regiões, como área e excentricidade, podem ser usadas para descrever imagens. A seguir descreve-se algumas propriedades básicas de regiões, principalmente aquelas que possuem como saída um valor escalar, visto que são facilmente adicionados à um vetor descritor.

## Área

A área de uma região é o número de pixels da região (MATHWORKS, 2013g).

## Área convexa

A área convexa é dada pela área do menor polígono capaz de conter a região (MATHWORKS, 2013g).

## Área Preenchida

Área preenchida corresponde a área da região cujas cavidades foram todas preenchidas (MATHWORKS, 2013g).

## Solidez

A solidez de uma região é dada pela razão Área / Área convexa (MATHWORKS, 2013g).

## Extensão

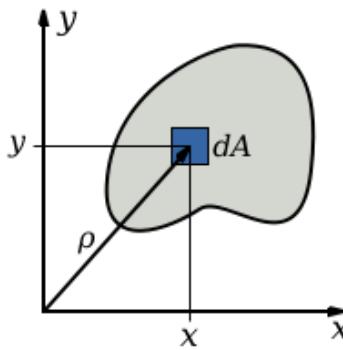
Semelhantemente à área convexa, a extensão é dada pela razão entre a área da região (definida anteriormente) e a área do menor retângulo capaz de conter a região (MATHWORKS, 2013g).

## Excentricidade

Para a obtenção da excentricidade de uma região deve-se primeiramente obter o momento de inércia de área da região (MATHWORKS, 2013g). O momento de inércia de área para uma região qualquer no sistema de coordenadas polares é definido pela equação 16:

$$J_{BB} = \int_A \rho^2 dA \quad (16)$$

Onde  $A$  é a área da região sobre a qual se deseja calcular o momento de inércia de área e  $\rho$  é a distância do centro do sistema de coordenadas até  $dA$ . Uma representação da área pode ser vista na figura 16.



**Figura 16:** Representação da área para cálculo do momento de inércia.

**Fonte:** (WIKIPEDIA, 2013)

A excentricidade da região é então a excentricidade da elipse que possui o mesmo momento de inércia que a área da região. A excentricidade para uma elipse com semi-eixo maior  $a$  e semi-eixo menor  $b$  é dada pela equação 17:

$$e = \sqrt{1 - \frac{b^2}{a^2}} \quad (17)$$

#### Tamanho do eixo principal e eixo secundário

O tamanho do eixo principal de uma região diz respeito ao tamanho do maior eixo da elipse que possui o mesmo momento de inércia de área que a área da região, conforme visto anteriormente na seção Excentricidade (MATHWORKS, 2013g). Da mesma forma o tamanho do eixo secundário diz respeito ao tamanho do menor eixo da elipse que possui o mesmo momento de inércia de área que a área da região.

#### Número de Euler

O número de Euler é dado pelo total de objetos desconexos da região subtraindo o total de cavidades da região (MATHWORKS, 2013g).

#### Centroide

O centroide de uma imagem é dado pelo centro de massa da imagem (MATHWORKS, 2013g).

### 2.2.2 Descritores de *Fourier*

Considere a figura 17a. Os pixels com valor 1 da imagem binária podem ser representados como sendo pares de coordenadas  $(x_0, y_0), (x_1, y_1), (x_2, y_2), \dots, (x_{K-1}, y_{K-1})$ , onde  $K$  é o número de pixels na imagem. Os pixels também podem ser representados como uma sequência  $x(k) = x_k$  e  $y(k) = y_k$  ou como  $s(k) = [x(k), y(k)]$  para  $k = 0, 1, 2, \dots, K - 1$ . Além disso pode-se tratar as coordenadas dos pixels como números complexos:  $s(k) = x(k) + jy(k)$ .

Tem-se que a transformada discreta de *Fourier* de um sinal  $s(k)$  é dado pela equação 18 (GONZALEZ; WOODS, 2006):

$$a(u) = \sum_{k=0}^{K-1} s(k) e^{-j2\pi uk/K} \quad , u = 0, 1, 2, \dots, K - 1 \quad (18)$$

Os coeficientes complexos  $a(u)$  da transformada discreta de *Fourier* são chamados de descritores de *Fourier* (GONZALEZ; WOODS, 2006).

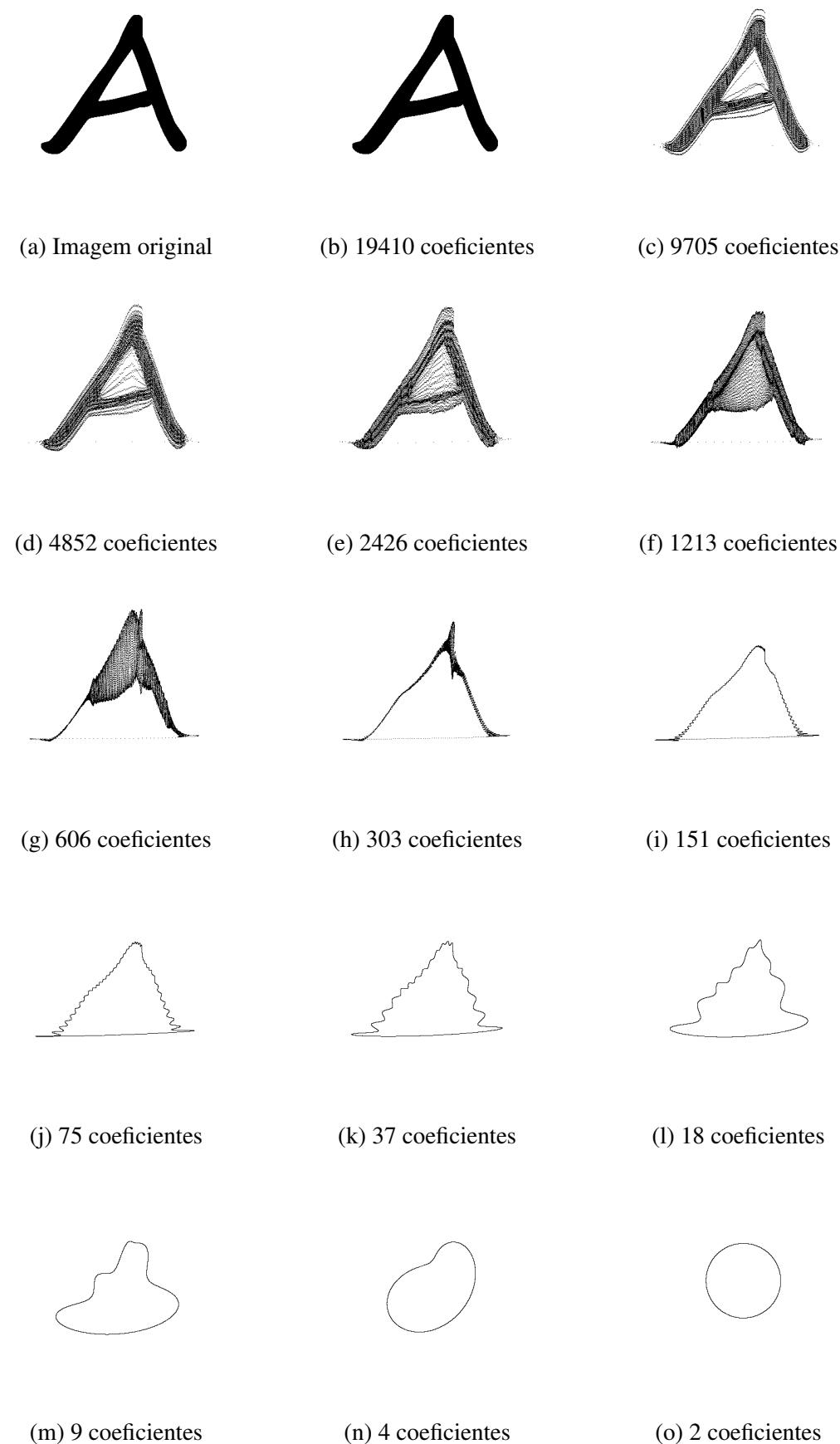
Da mesma forma que se aplicou a transformada discreta de *Fourier* para obter os descritores de *Fourier* pode-se aplicar a transformada discreta inversa de *Fourier* nos coeficientes complexos para obter novamente os números complexos representando as coordenadas dos pixels (equação 19).

$$s(k) = \frac{1}{K} \sum_{u=0}^{K-1} a(u) e^{j2\pi uk/K} \quad , k = 0, 1, 2, \dots, K - 1 \quad (19)$$

Suponha agora que ao invés de utilizar todos os descritores de *Fourier* utiliza-se apenas os  $P$  primeiros descritores para reconstituir  $s(k)$ . Tem-se portanto uma aproximação para  $s(k)$  dada por  $\hat{s}(k)$  na equação 20.

$$\hat{s}(k) = \frac{1}{P} \sum_{u=0}^{P-1} a(u) e^{j2\pi uk/P} \quad , k = 0, 1, 2, \dots, K - 1 \quad (20)$$

Quando os  $P$  coeficientes de menor frequência são escolhidos para reconstituir a imagem tem-se uma imagem aproximada da imagem original porém que mantém as principais características. Pode-se ver esse comportamento nas imagens da figura 17. A imagem da figura 17b foi reconstruída utilizando todos os 19410 coeficientes da imagem original, a imagem 17c foi reconstruída com 9705 coeficientes, e assim por diante, até apenas 2 coeficientes na figura 17o.



**Figura 17:** Imagem original e imagens reconstituídas utilizando  $n$  descritores de *Fourier*.

**Fonte:** Autoria própria

## 2.3 CLASSIFICAÇÃO

Esta seção apresenta uma visão geral de reconhecimento de padrões utilizando redes neurais.

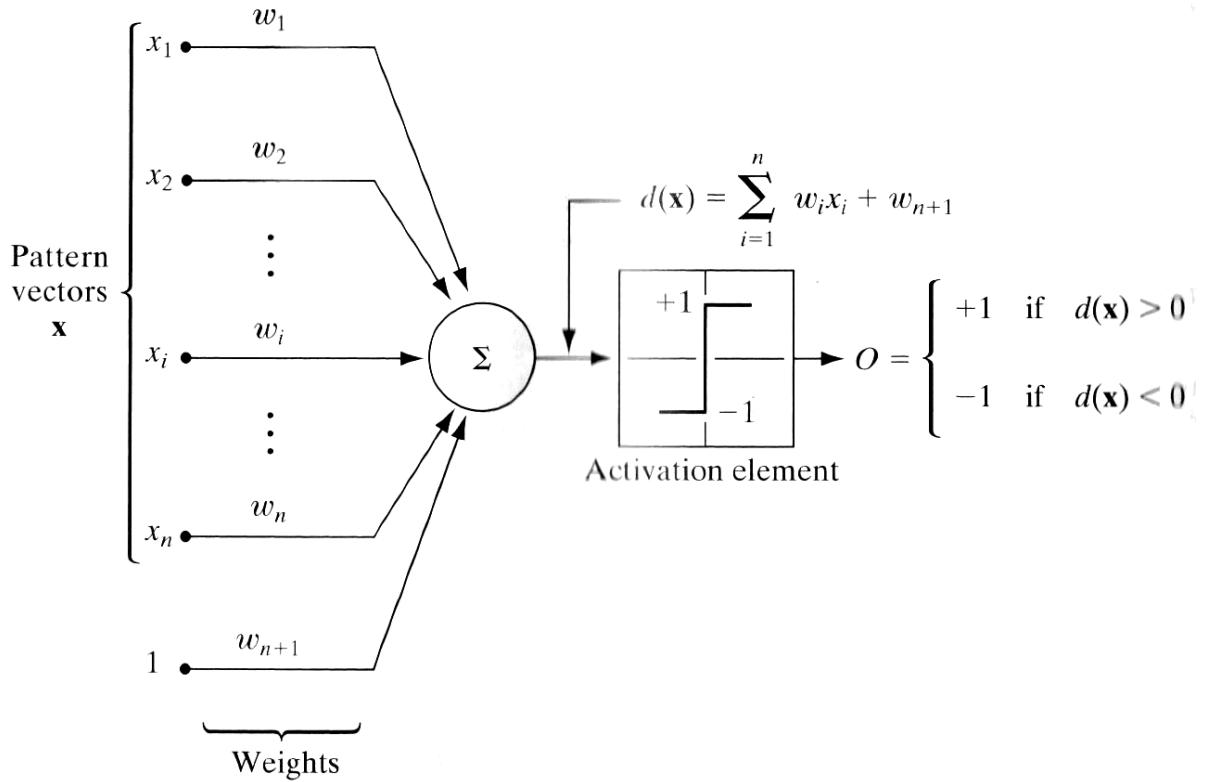
### 2.3.1 Redes neurais

Redes neurais são elementos não lineares (chamados neurônios) interconectados da mesma maneira que se acredita que os neurônios estão conectados no cérebro. Usam-se essas redes para tomadas de decisão após seus coeficientes terem sido ajustados por meio de apresentações sucessivas de conjuntos de dados de treinamento (GONZALEZ; WOODS, 2006).

Na forma mais simples no entanto a rede é um elemento linear e é composta por um único neurônio, também chamado *perceptron*. Esse neurônio então aprende uma função linear para separar dois conjuntos de dados de treinamento linearmente separáveis. A figura 18 apresenta o modelo básico para um *perceptron*. A resposta desse *perceptron* é baseada na soma ponderada das entradas, dada pela equação 21.

$$d(x) = \sum_{i=1}^n (\omega_i x_i) + \omega_{n+1} \quad (21)$$

Os coeficientes  $\omega_i, i = 1, 2, \dots, n, n+1$ , são chamados de pesos e modificam a soma antes que os valores de entrada passem pelo elemento de ativação, também chamado de função de ativação. Os pesos podem ser comparados às sinapses que ocorrem no cérebro humano. No caso particular da figura 18 a saída do *perceptron* será  $+1$  caso a entrada pertença à classe  $\omega_1$  e  $-1$  caso a entrada pertença à classe  $\omega_2$ .



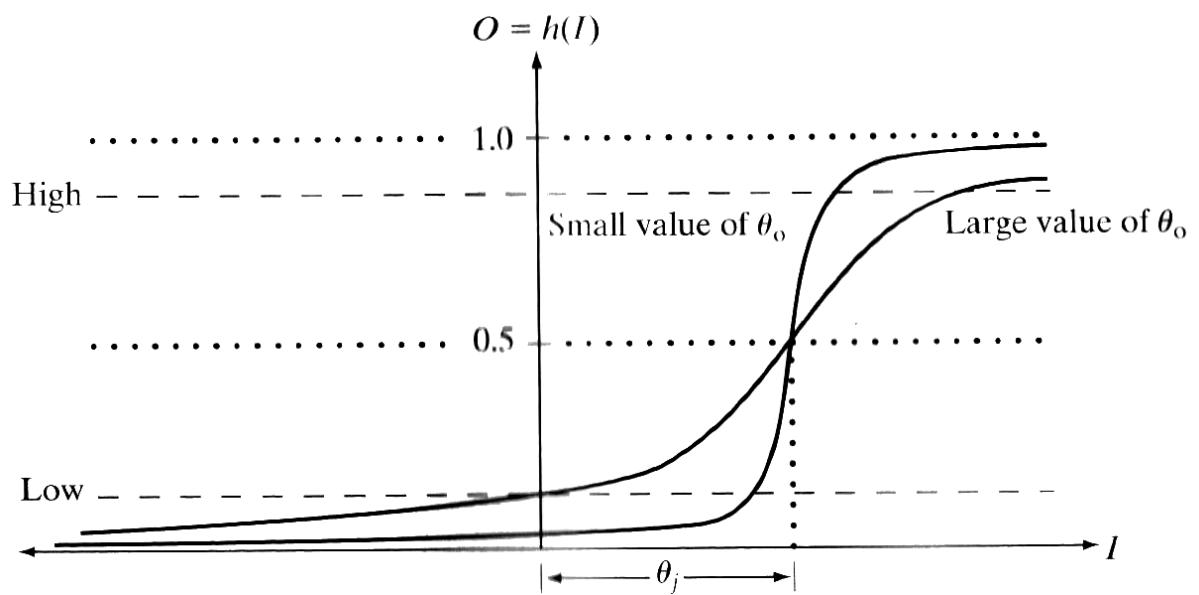
**Figura 18:** Modelo de *perceptron*.

**Fonte:** (GONZALEZ; WOODS, 2006)

Esse tipo de abordagem formada por um único *perceptron* não é capaz de separar mais do que duas classes e também não é capaz de separar classes que não são linearmente separáveis. Isso se dá devido ao fato de que a equação 21, quando expandida, define um hiperplano em um espaço  $n$ -dimensional.

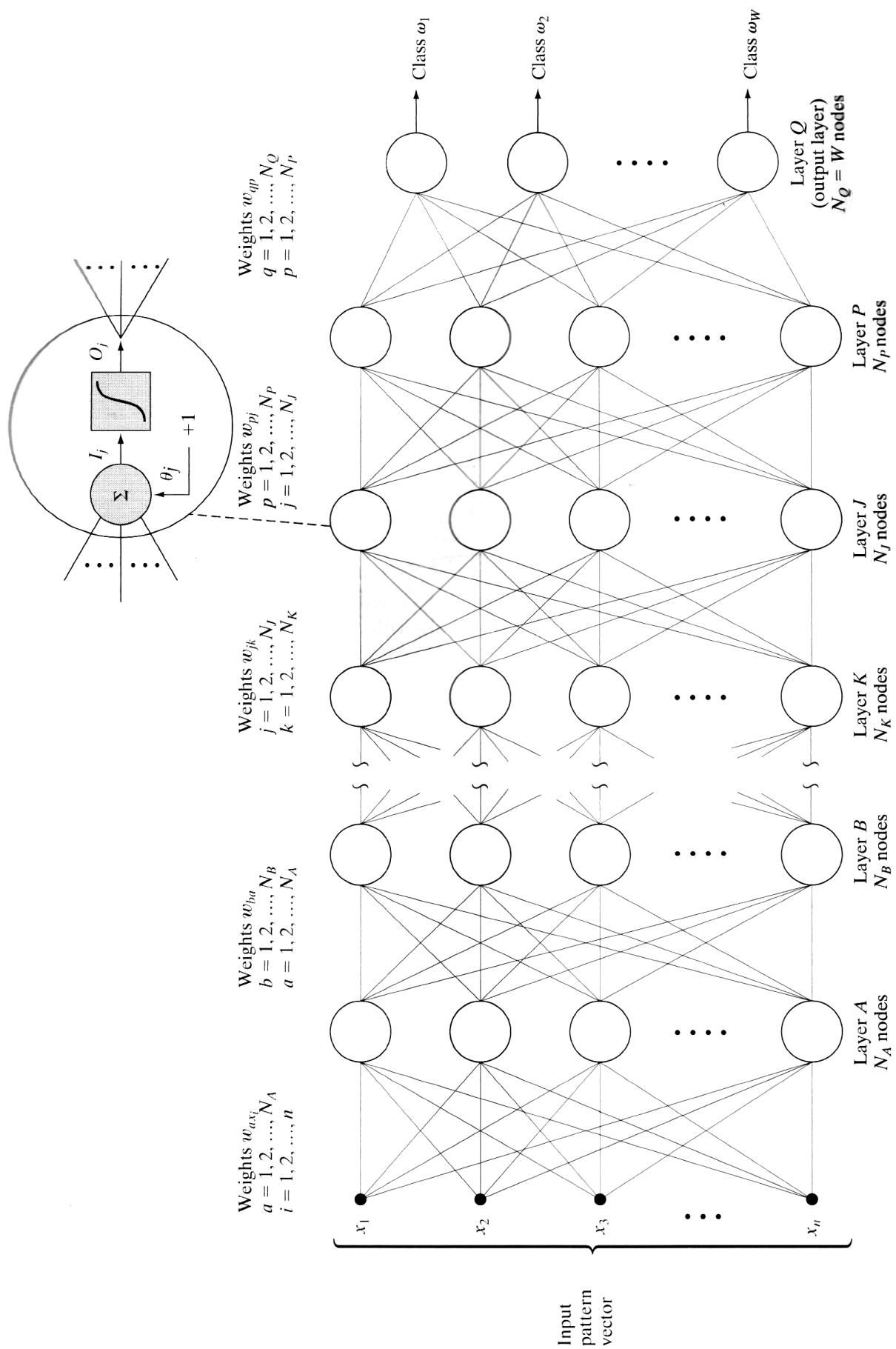
Utilizam-se portanto para problemas mais complexos redes de neurônios, também chamada de *multi-layer perceptron* ou *multi-layer feedforward network*. Cada neurônio é formado por uma estrutura do mesmo tipo da estrutura do *perceptron*, com a diferença de a função de ativação ser geralmente uma função sigmoide ao invés de um simples limiar. A função sigmoide pode ser vista na equação 22 e na figura 19. As saídas dos neurônios estão interconectadas com as entradas de outros neurônios. Essa estrutura pode ser vista na figura 20.

$$h_j(I_j) = \frac{1}{1 + e^{-(I_j + \theta_j)/\theta_o}} \quad (22)$$



**Figura 19:** Gráfico da função de ativação sigmoide.

Fonte: (GONZALEZ; WOODS, 2006)



**Figura 20:** Modelo de *multi-layer feedforward network*.

Fonte: (GONZALEZ; WOODS, 2006)

Nesse tipo de rede a primeira camada,  $A$ , possui o mesmo número de neurônios que a dimensão do vetor descriptor da entrada,  $N_A = n$ . Semelhantemente o número de neurônios da camada de saída,  $Q$ , é igual ao número de classes que a rede deve reconhecer,  $N_Q = W$ . A rede reconhece uma entrada como sendo parte de uma classe quando a saída da classe está próxima de um e as demais saídas estão próximas de zero.

Pode-se demonstrar que redes desse tipo são capazes de separar regiões convexas no espaço  $n$ -dimensional de entradas utilizando duas camadas de neurônios. São também capazes de separar classes em regiões arbitrárias utilizando três camadas de neurônios (GONZALEZ; WOODS, 2006). A complexidade das regiões separadas por uma rede de três camadas se dá pelo número de neurônios em cada camada. Isso está resumido na figura 21.

Network structure	Type of decision region	Solution to exclusive-OR problem	Classes with meshed regions	Most general decision surface shapes
Single layer	Single hyperplane			
Two layers	Open or closed convex regions			
Three layers	Arbitrary (complexity limited by the number of nodes)			

**Figura 21:** Estruturas de redes neurais e sua respectiva capacidade de decisão.

Fonte: (GONZALEZ; WOODS, 2006)

### Treinamento de *perceptrons*

O treinamento de *perceptrons* consiste em determinar os valores adequados de  $\omega_i$  (da equação 21) para que a soma ponderada das entradas forme a saída adequada para a função de ativação. Existem diversos métodos de treinamento para determinar os valores de  $\omega_i$  para

um único perceptron. Apresenta-se a seguir o método denominado *delta rule* ou *least-mean-square*.

A principal característica do método *delta rule* é que ele minimiza o erro entre a saída desejada e a saída real do *perceptron* em cada passo de treinamento. Considere a função da equação 23.

$$J(w) = \frac{1}{2}(r - w^T y)^2 \quad (23)$$

$J(w)$  representa o erro quadrático para um dado conjunto de pesos  $w$ . O erro é calculado como sendo a diferença entre a saída desejada do *perceptron*  $r$  e a saída real dada pelo produto do vetor de pesos transposto  $w^T$  com o valor de entrada do conjunto de treinamento  $y$ .

O objetivo do treinamento pelo método *delta rule* é minimizar  $J(w)$  ajustando os valores dos pesos  $w$ .

Se  $w(k)$  representa o vetor de pesos no passo de treinamento  $k$ , então pode-se calcular  $w(k+1)$  de forma a reduzir o valor de  $J(w)$  utilizando a equação 24 (GONZALEZ; WOODS, 2006).

$$w(k+1) = w(k) - \alpha \left[ \frac{\partial J(w)}{\partial w} \right]_{w=w(k)} \quad (24)$$

Onde  $\alpha > 0$  é uma constante escolhida de acordo com o grau de correção que se deseja a cada passo. Da equação 23 tem se que:

$$\frac{\partial J(w)}{\partial w} = -(r - w^T y)y \quad (25)$$

Substituindo a equação 25 na equação 24 resulta na equação 26 que pode ser utilizada para o cálculo dos coeficientes  $w(k+1)$  dados os coeficientes  $w(k)$ , a saída desejada  $r$  e o valor de entrada do conjunto de treinamento  $y$ .

$$w(k+1) = w(k) + \alpha [r(k) - w^T(k)y(k)]y(k) \quad (26)$$

Pode-se demonstrar que utilizando esse método para calcular os coeficientes, tem-se a redução do erro do *perceptron* em um fator de  $\alpha y(k)^T y(k)$  a cada ciclo de treinamento (GONZALEZ; WOODS, 2006).

### Treinamento de *multi-layer perceptrons*

O treinamento de redes multicamadas pode ser efetuado de maneira semelhante ao treinamento de um único *perceptron*. Considere a equação 27. O erro quadrático  $E_Q$  obtido na camada de saída é dado pela soma dos erros quadráticos em todos os *perceptrons* da camada de saída (*perceptrons* de 1 a  $N_Q$ ). O erro em cada *perceptron* é dado pela diferença entre a saída desejada  $r_q$ , no *perceptron*  $q$ , e a saída obtida  $O_q$ .

$$E_Q = \frac{1}{2} \sum_{q=1}^{N_Q} (r_q - O_q)^2 \quad (27)$$

Novamente deseja-se minimizar o erro  $E_Q$  em cada passo de treinamento. Minimizando a equação 27 chega-se a dois casos. O primeiro é quando o *perceptron* está na última camada da rede. Nesse caso sabe-se o valor esperando na saída do *perceptron*, que é dado por  $O_q$ . Por outro lado quando o *perceptron* está em uma camada intermediária é necessário que primeiro se defina o erro em função de parâmetros conhecidos na rede (GONZALEZ; WOODS, 2006).

Para *perceptrons* na camada de saída  $Q$  os pesos  $w(k+1)$  podem ser calculados pelas equações 28 e 29.  $K$  é a camada que precede a camada de saída  $Q$ .

$$w_q(k+1) = w_q(k) + \alpha \delta_q O_K \quad (28)$$

$$\delta_q = (r_q - O_q) h'_q(I_q) \quad (29)$$

Onde:

- $w_q(k+1)$  são os pesos atualizados do *perceptron*  $q$ ;
- $w_q(k)$  são os pesos antigos do *perceptron*  $q$ ;
- $\alpha > 0$  é uma constante escolhida de acordo com o grau de correção que se deseja a cada passo;
- $O_K$  são as saídas dos *perceptrons* da camada  $K$ ;
- $r_q$  é a saída esperada do *perceptron*, dada pelo conjunto de treinamento;
- $O_q$  é a saída atual no *perceptron*  $q$ ;
- $h'_q(I_q)$  é a derivada da função de ativação do *perceptron*  $q$  para a entrada  $I_q$ .

Para *perceptrons* em uma camada intermediária  $J$ , onde a camada seguinte é a camada  $P$  e a camada antecedente é a camada  $K$ , os pesos  $w(k+1)$  podem ser calculados pelas equações 30 e 31.

$$w_j(k+1) = w_j(k) + \alpha \delta_j O_K \quad (30)$$

$$\delta_j = h'_j(I_j) \sum_{p=1}^{N_P} \delta_p \omega_{jp} \quad (31)$$

Onde:

- $w_j(k+1)$  são os pesos atualizados do *perceptron*  $j$ ;
- $w_j(k)$  são os pesos antigos do *perceptron*  $j$ ;
- $\alpha > 0$  é uma constante escolhida de acordo com o grau de correção que se deseja a cada passo;
- $O_K$  são as saídas dos *perceptrons* da camada  $K$ ;
- $h'_j(I_j)$  é a derivada da função de ativação para a entrada  $I_j$ ;
- $p$  até  $N_P$  são todos os *perceptrons* da camada  $P$ ;
- $\delta_p$  é o valor de  $\delta$  que havia sido calculado para o *perceptron*  $p$  da camada  $P$  quando os pesos da camada  $P$  foram atualizados.
- $\omega_{jp}$  é o valor do peso  $\omega$  que havia sido calculado para o *perceptron*  $j$  em cada *perceptron* da camada  $P$ .

Caso a função de ativação escolhida seja a função sigmoide da equação 22, com  $\theta_O = 1$ , pode-se demonstrar que  $h'_j(I_j)$  assume o valor da equação 32 e semelhantemente  $h'_q(I_q)$  assume o valor da equação 33 (GONZALEZ; WOODS, 2006).

$$h'_j(I_j) = O_j(1 - O_j); \quad (32)$$

Onde:

- $O_j$  é a saída atual no *perceptron*  $j$ ;

$$h'_q(I_q) = O_q(1 - O_q); \quad (33)$$

Onde:

- $O_q$  é a saída atual no *perceptron*  $q$ ;

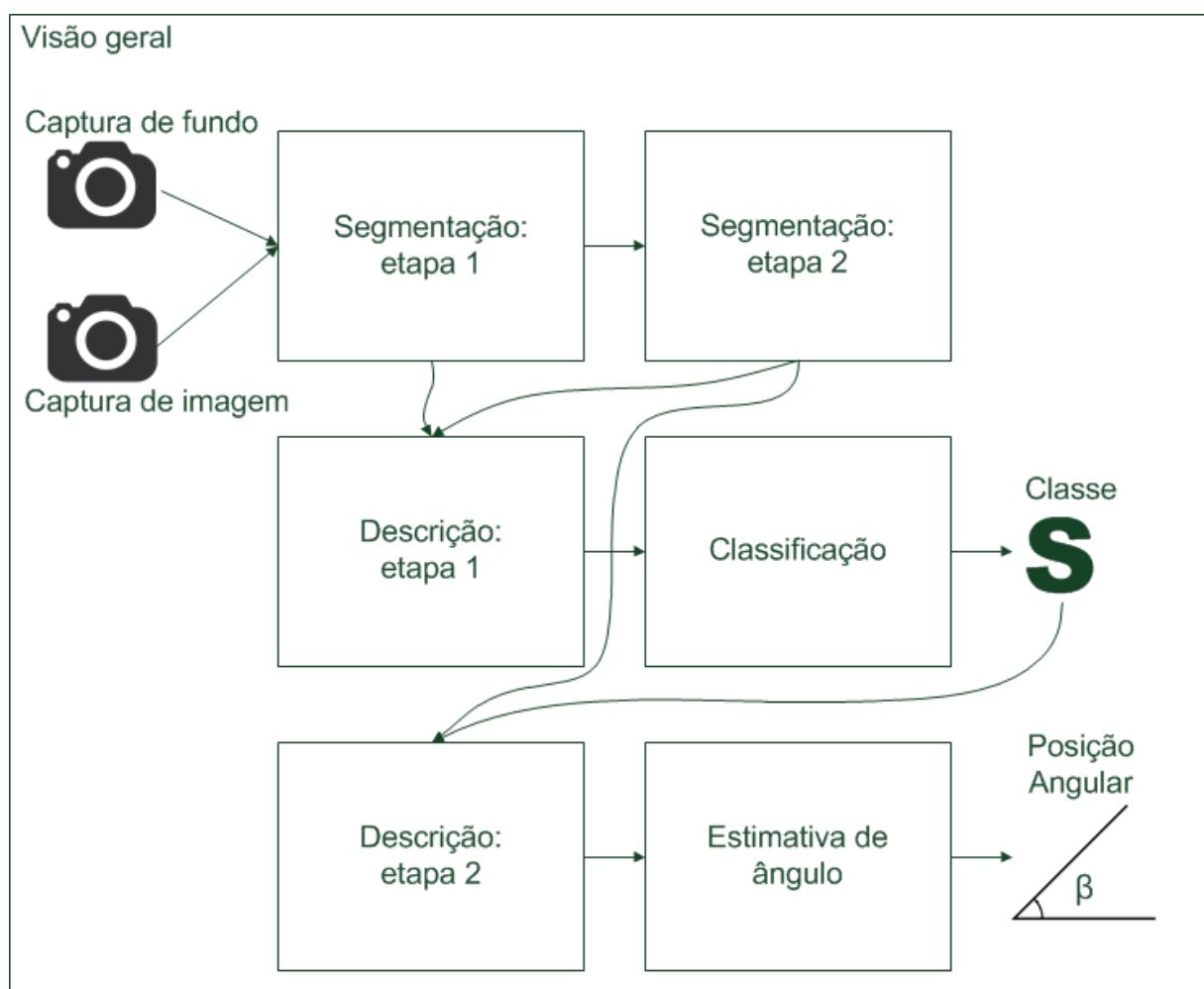
Como se pode ver é necessário iniciar o processo de atualização dos pesos da rede pela camada de saída. A camada de saída da rede é o único lugar onde se sabe qual é a saída que cada *perceptron* deveria ter. Para as camadas antecedentes a atualização deve ser feita em função dessas saídas. Portanto cada camada depende de parâmetros da camada seguinte na rede. Esse processo de atualização dos pesos iniciando pela ultima camada e voltando até a primeira recebe o nome de *back propagation*. O método mencionado anteriormente recebe o nome de *delta rule*, no entanto existem diversos métodos diferentes mas que utilizam a mesma estratégia de propagação das correções na rede. Todos esses métodos são classificados como métodos de *back propagation*.

## 2.4 CONSIDERAÇÕES

Apresentou-se nesse capítulo embasamento teórico para o desenvolvimento do projeto, abrangendo as três áreas fundamentais de sistemas de processamento de imagens: segmentação, descrição e classificação. Foram estudadas técnicas de segmentação de imagens que permitem, através da combinação de diversas delas, a identificação de regiões de interesse. Além da identificação da região de interesse foi apresentada a técnica de normalização de imagens utilizando a análise de componentes principais. Foi fornecido embasamento teórico para efetuar a descrição de imagens utilizando propriedades de regiões, como área, solidez, posição do centroide, entre outros além dos conceitos para efetuar a descrição utilizando descritores de *Fourier*. Por fim apresentou-se uma visão geral de redes neurais e foram apresentadas noções sobre treinamento de redes utilizando algoritmos do tipo *back propagation*.

### 3 DESENVOLVIMENTO

Este capítulo apresenta detalhes a respeito do desenvolvimento do projeto. A descrição apresentada a seguir referencia os conceitos apresentados no capítulo 2. O sistema foi implementado no software MATLAB (MATHWORKS, 2013e) permitindo a execução de diversos testes para validação conforme será visto no capítulo 4. A figura 22 apresenta uma visão geral do sistema desenvolvido.



**Figura 22:** Visão geral do sistema desenvolvido.

**Fonte:** Autoria própria

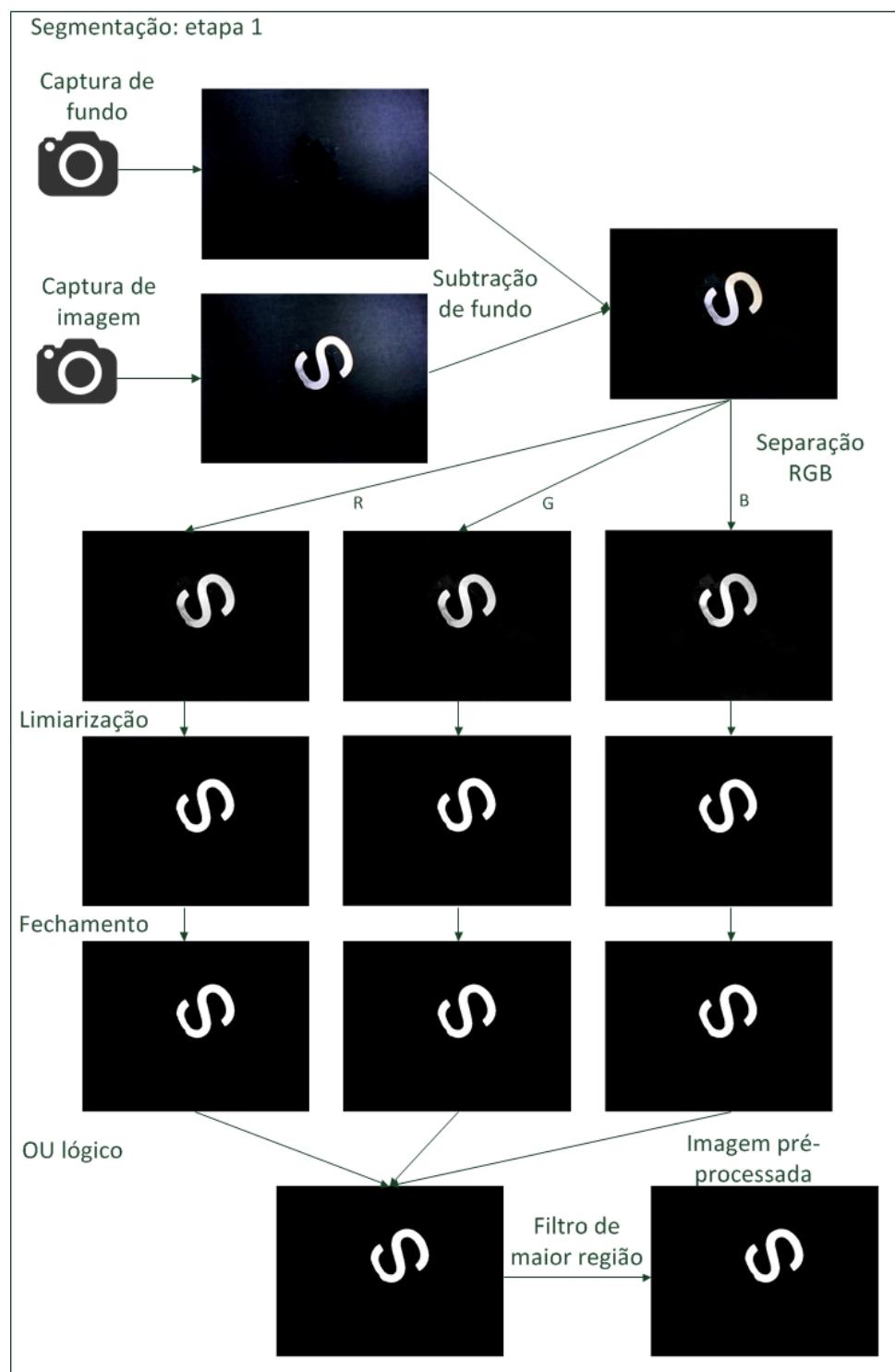
O capítulo divide-se em sete seções. As primeiras seis seções apresentam com detalhes os blocos da figura 22: a seção 3.1 e a seção 3.2 mostram como os conceitos da seção 2.1 foram aplicados para segmentar a imagem capturada. A seção 3.3 e a seção 3.5 apresentam como os descritores de *Fourier* e as propriedades estudadas na seção 2.2.1 foram aplicados para formar um descritor para as imagens. As seções 3.4 e 3.6 apresentam como os descritores foram utilizados para classificar e determinar o ângulo das imagens. Finalmente, na seção 3.7, algumas considerações sobre o desenvolvimento do projeto são apresentadas.

### 3.1 SEGMENTAÇÃO: ETAPA 1

A primeira etapa da segmentação consiste em uma série de limiarizações e filtros para localizar a região de interesse na imagem. A saída da primeira etapa é uma imagem binária contendo a região de interesse. O processo da primeira etapa pode ser visto na figura 23.

O processo inicia pela subtração da imagem RGB de fundo da imagem RGB com o objeto. Como resultado tem-se uma imagem RGB contendo as diferenças inseridas pelo objeto na imagem. Para não perder informações intrínsecas da imagem RGB no processo de conversão para escala de cinza dividem-se os canais R, G e B da imagem. Cada canal é processado separadamente. Primeiramente os canais são limiarizados utilizando o método global de *Otsu*, discutido na seção 2.1.2. Em seguida os canais são submetidos à operação morfológica de fechamento, vista na seção 2.1.3. A operação de fechamento é aplicado para reduzir o ruido do tipo pimenta (pixels pretos espalhados aleatoriamente pela imagem). Na sequência os três canais processados separadamente são unidos através da operação lógica OU para formar uma única imagem. Finalmente a maior região conexa da imagem é selecionada para ser a região de interesse.

Embora não se possa constatar visualmente nenhum progresso significativo em alguns passos da sequência de imagens da figura 23, a primeira etapa da segmentação necessita de todos os passos apresentados. Todos eles garantem a robustez do sistema, fornecendo a região de interesse para a etapa seguinte.

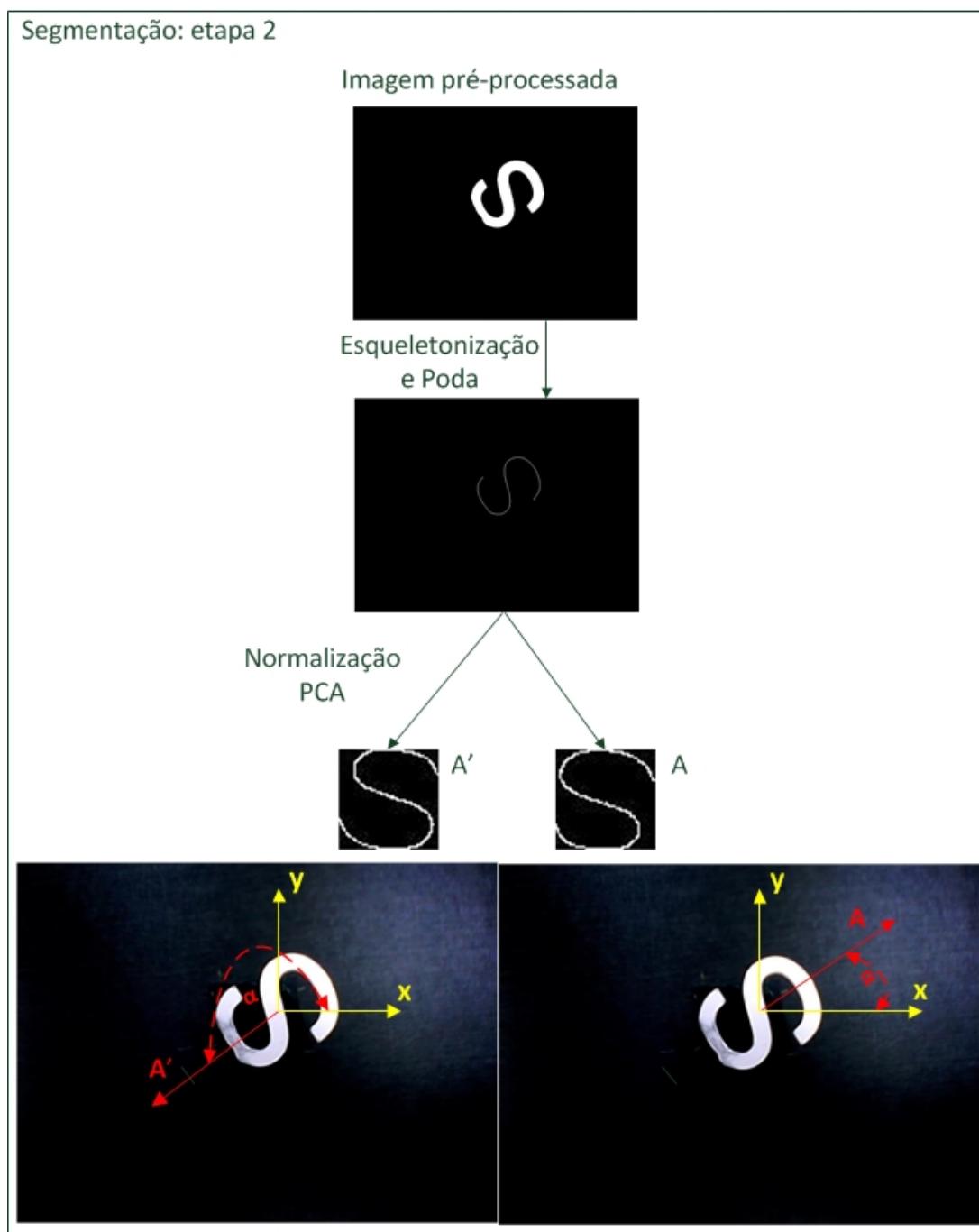


**Figura 23:** Primeira etapa da segmentação.

**Fonte:** Autoria própria

### 3.2 SEGMENTAÇÃO: ETAPA 2

A segunda etapa da segmentação é responsável por normalizar a imagem. Esta etapa recebe como entrada a região de interesse e apresenta como saída duas opções de normalização, além de um autovetor e seu oposto. O processo pode ser visto na figura 24.



**Figura 24:** Segunda etapa da segmentação.

**Fonte:** Autoria própria

A região de interesse obtida da primeira etapa de segmentação é esqueletonizada e podada pelos processos descritos na seção 2.1.4. Em seguida o processo de normalização da seção 2.1.5 é aplicado.

Como dito na seção 2.1.5, o autovetor associado ao maior autovalor da matriz de covariâncias dos pixels da imagem aponta na direção de maior variação dos dados.

O autovetor porém não aponta no sentido de maior variação dos dados. Como pode ser visto na figura 24, normalizando o “S” com o autovetor oposto  $A' = -A$  obtém-se a normalização correta para o “S”. Contudo se a normalização utilizar o autovetor  $A$  obtém-se uma imagem com o “S” normalizado rotacionado em  $180^0$ .

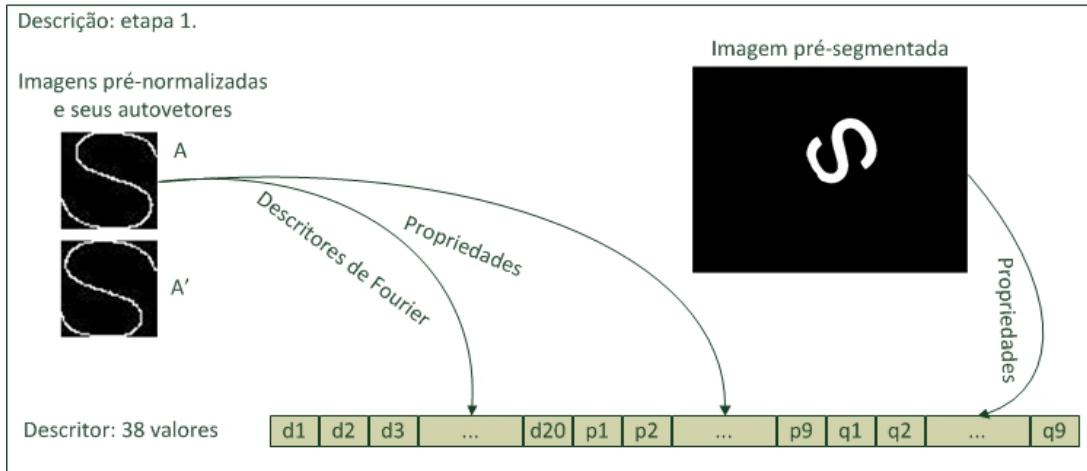
Empiricamente percebe-se que o autovetor aponta sempre no sentido positivo dos eixos da imagem. Esse comportamento leva a duas possíveis normalizações para cada imagem. A normalização obtida utilizando o autovetor  $A$  e a normalização pelo oposto do autovetor,  $A'$ .

Essas duas opções de normalização são então apresentadas à etapa seguinte juntamente com os respectivos autovetores.

### 3.3 DESCRIÇÃO: ETAPA 1

Essa etapa é responsável por gerar um descritor para o objeto. O descritor é formado a partir da imagem normalizada utilizando o autovetor  $A$  (chamada a partir daqui de imagem pré-normalizada) e utilizando também a imagem pré-segmentada da etapa 1 de segmentação. A imagem normalizada utilizando o autovetor  $A'$  não é utilizada. A figura 25 resume o processo dessa etapa.

O descritor é composto por 20 descritores de *Fourier*,  $d1, d2, \dots, d20$ , concatenados com 9 propriedades básicas da imagem pré-normalizada,  $p1, p2, \dots, p9$  e 9 propriedades da imagem pré-segmentada,  $q1, q2, \dots, q9$ . A tabela 1 detalha o descritor gerado.

**Figura 25:** Primeira etapa da descrição.**Fonte:** Autoria própria**Tabela 1:** Descritor para primeira etapa.

Descriptor	Descrição
$d_1 \dots d_{20}$	20 descritores de <i>Fourier</i>
$p_1$ e $q_1$	Área
$p_2$ e $q_2$	Área convexa
$p_3$ e $q_3$	Excentricidade
$p_4$ e $q_4$	Número de Euler
$p_5$ e $q_5$	Extensão
$p_6$ e $q_6$	Área preenchida
$p_7$ e $q_7$	Tamanho do eixo principal
$p_8$ e $q_8$	Tamanho do eixo secundário
$p_9$ e $q_9$	Solidez

**Fonte:** Autoria própria

### 3.4 CLASSIFICAÇÃO

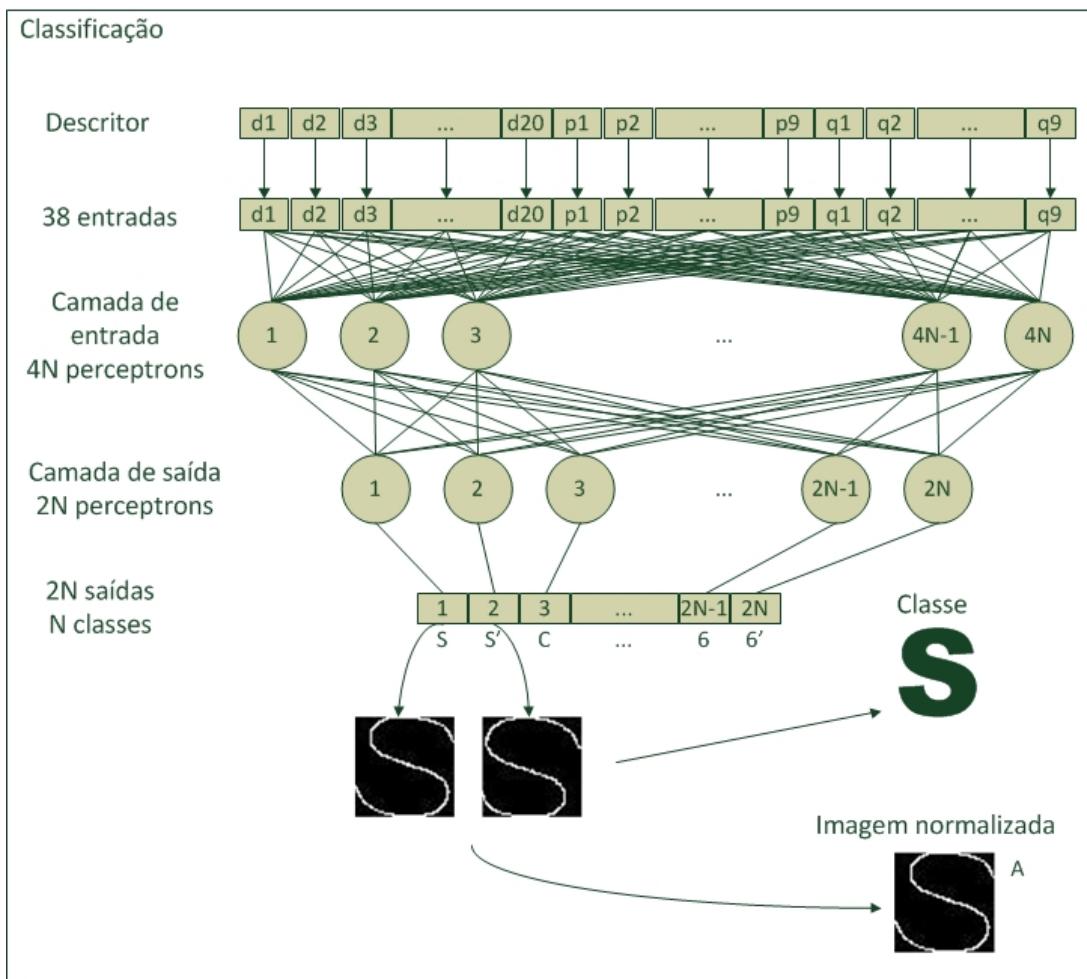
A função da etapa de classificação é tomar como entrada o descritor gerado na etapa anterior e fornecer como saída a classe a qual o objeto pertence. Além disso é responsável por dizer qual a normalização correta para a imagem. Ou seja, se a imagem normalizada corretamente utiliza o autovetor  $A$  ou o oposto  $A'$ . Esse processo é mostrado na figura 26.

Para essa tarefa foi utilizada uma rede neural pois ela fornece um método de classificação baseado em treinamento no qual não é necessário que se tenha conhecimento prévio das propriedades estatísticas de cada classe de padrões (GONZALEZ; WOODS, 2006). A rede utilizada é formada por duas camadas, 38 entradas e  $2N$  saídas. Onde  $N$  é o número de classes a serem

identificadas pelo sistema. Quando o objeto a ser identificado é simétrico, de tal forma que a rotação do objeto por um ângulo  $0^0 < \alpha < 360^0$  resulta no próprio objeto, não há necessidade de duas saídas para a mesma classe pois tanto a normalização utilizando  $A$  quanto a normalização utilizando  $A'$  retornam o mesmo resultado. Nesse caso o número de saídas é menor. A primeira camada possui  $4N$  perceptrons e a camada de saída possui  $2N$  perceptrons. O número de perceptrons na primeira camada foi escolhido empiricamente. O número de saídas é  $2N$  pois a rede deve ser capaz de identificar imagens normalizadas pelo autovetor  $A$  e imagens normalizadas pelo oposto  $A'$ . A função de ativação para as duas camadas é a função sigmoide vista na seção 2.3.1. A função sigmoide é uma boa escolha para redes de classificação pois possui uma transição rápida entre 0 e 1 para entradas de  $-\infty$  a  $+\infty$  (MATHWORKS, 2013f).

Para formar o conjunto de treinamento da rede é necessário que se obtenha amostras de imagens com as respectivas classes previamente conhecidas. Em seguida deve-se executar todos os passos até obter as duas possibilidades de normalização de cada imagem, e os descritores relativos à normalização dada pelo autovetor  $A$ . O descritor é então fornecido como entrada da rede e a saída da rede é dada pelo autovetor que leva a normalização correta da imagem. Por exemplo, se a normalização correta da classe  $n$  é dada pelo autovetor  $A$ , a posição  $2n$  da saída é um e o restante zero. Se a normalização correta é dada pelo oposto do autovetor,  $A'$ , então a posição  $2n + 1$  é um e o restante zero. Esse último passo necessita ser executado manualmente para que se possa “ensinar” a rede qual a normalização correta para cada classe.

A rede é treinada utilizando um algoritmo de *back propagation* conforme visto na seção 2.3.1. O algoritmo utilizado para esta rede em específico utiliza o método chamado *Resilient Backpropagation*. Esse método não apresenta bons resultados para redes de aproximação de funções, no entanto, é indicado para redes de classificação grandes com centenas de pesos por possuir baixo consumo de memória (MATHWORKS, 2013a).



**Figura 26:** Etapa de classificação.

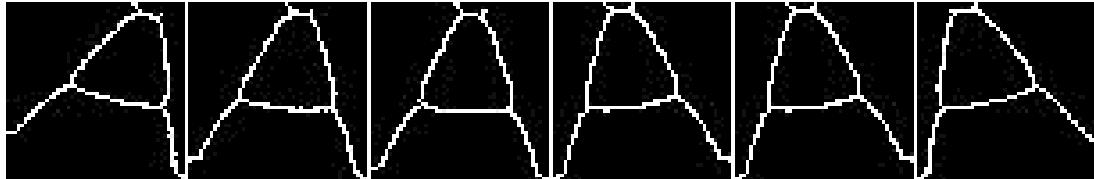
**Fonte:** Autoria própria

### 3.5 DESCRIÇÃO: ETAPA 2

A segunda etapa da descrição recebe como entrada a imagem normalizada pelo autovetor correto. A função desta etapa é gerar um descritor para a imagem normalizada, que auxiliará na etapa de estimativa do ângulo do objeto.

Tanto esta etapa, quanto a estimativa de ângulo descrita na próxima seção, seriam desnecessárias em ambientes ideais. Veja por exemplo as imagens da figura 27. Todas as imagens foram obtidas para um mesmo objeto, sujeito às mesmas condições gerais, mas em rotações diferentes. Em ambientes ideais todas as imagens seriam iguais e poderia-se calcular o ângulo de rotação da imagem diretamente a partir do ângulo do autovetor, somando-se uma constante ao seu valor. No ambiente real no entanto constatou-se que mudanças pequenas, como ruído ou diferenças na iluminação, influenciam a direção do autovetor calculado produzindo resultados

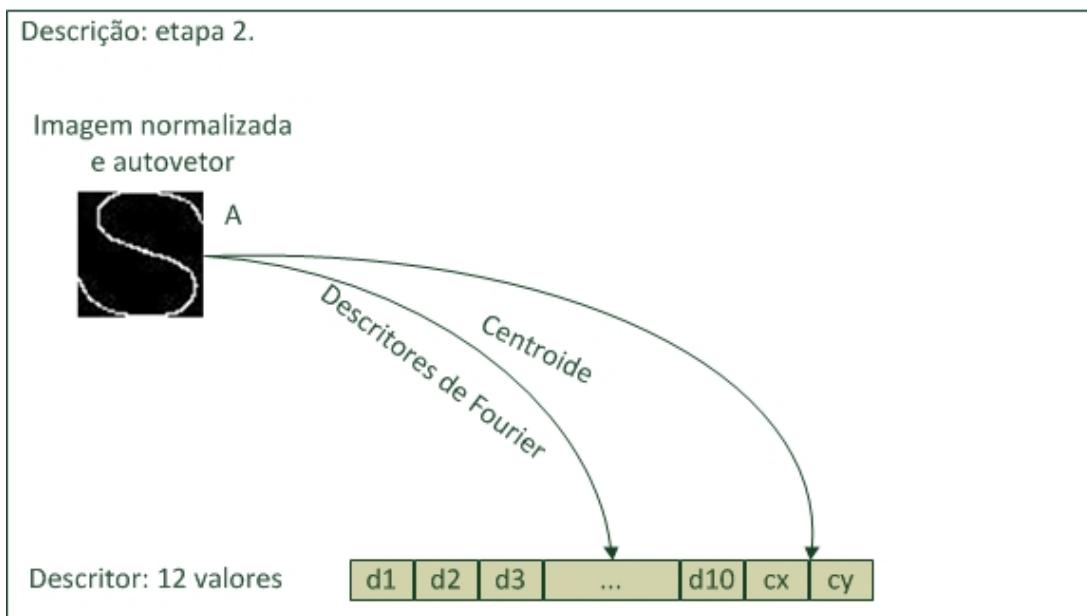
diferentes.



**Figura 27:** Imagens normalizadas para diferentes entradas do mesmo objeto.

**Fonte:** Autoria própria

A diferença entre o autovetor calculado para cada caso é tratada na etapa de estimativa de ângulo que recebe como entrada o descritor calculado aqui. O descritor portanto deve conter informações relacionadas à posição do objeto na imagem normalizada. Utilizou-se para essa tarefa um descritor contendo 10 descritores de *Fourier* juntamente com as coordenadas do centroide da imagem. O descritor pode ser visto na figura 28.



**Figura 28:** Segunda etapa da descrição.

**Fonte:** Autoria própria

### 3.6 ESTIMATIVA DE ÂNGULO

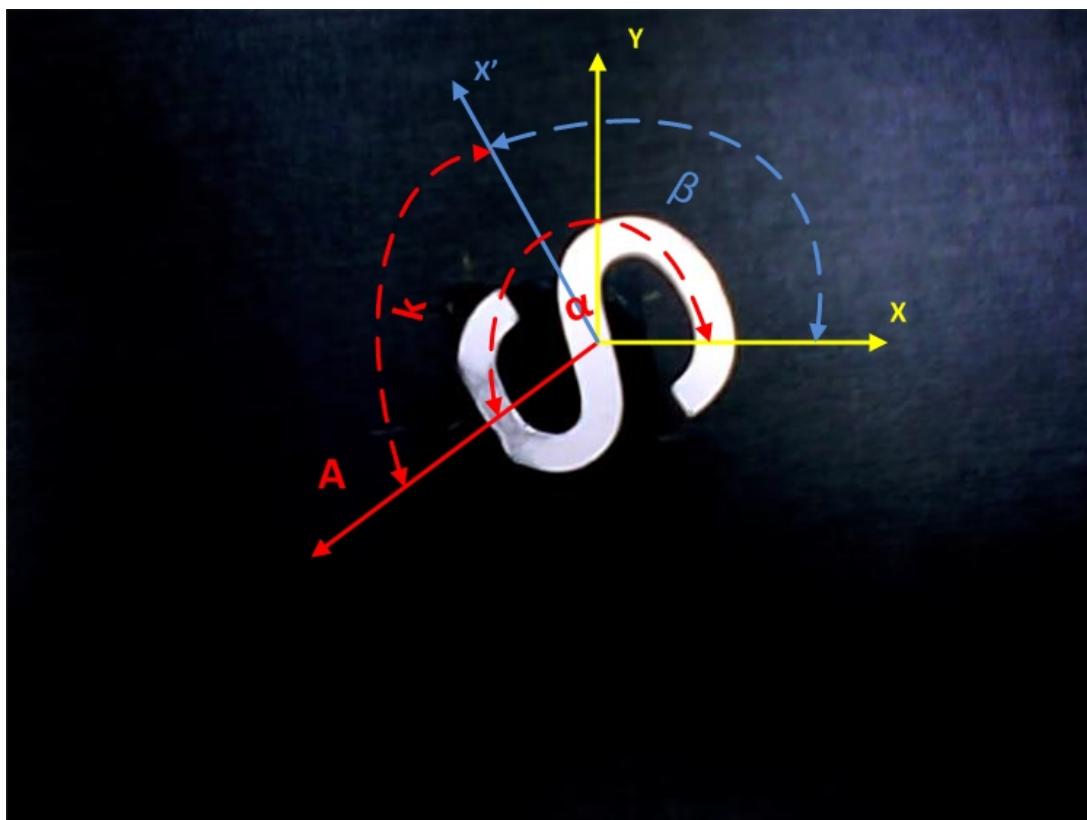
Finalmente a última etapa recebe o descritor da imagem normalizada (figura 28), a classe a qual pertence o objeto e o autovetor utilizado na normalização. O objetivo dessa etapa é fornecer o ângulo de rotação do objeto na imagem.

Considere a figura 29. Os eixos  $x, y$  são as coordenadas da imagem. O eixo  $x'$  é o eixo que foi assumido como sendo o zero para o objeto da figura.  $A$  é o autovetor calculado na etapa de normalização. Pela figura tem-se então a equação 34.

$$\beta = \alpha - k \quad (34)$$

Onde:

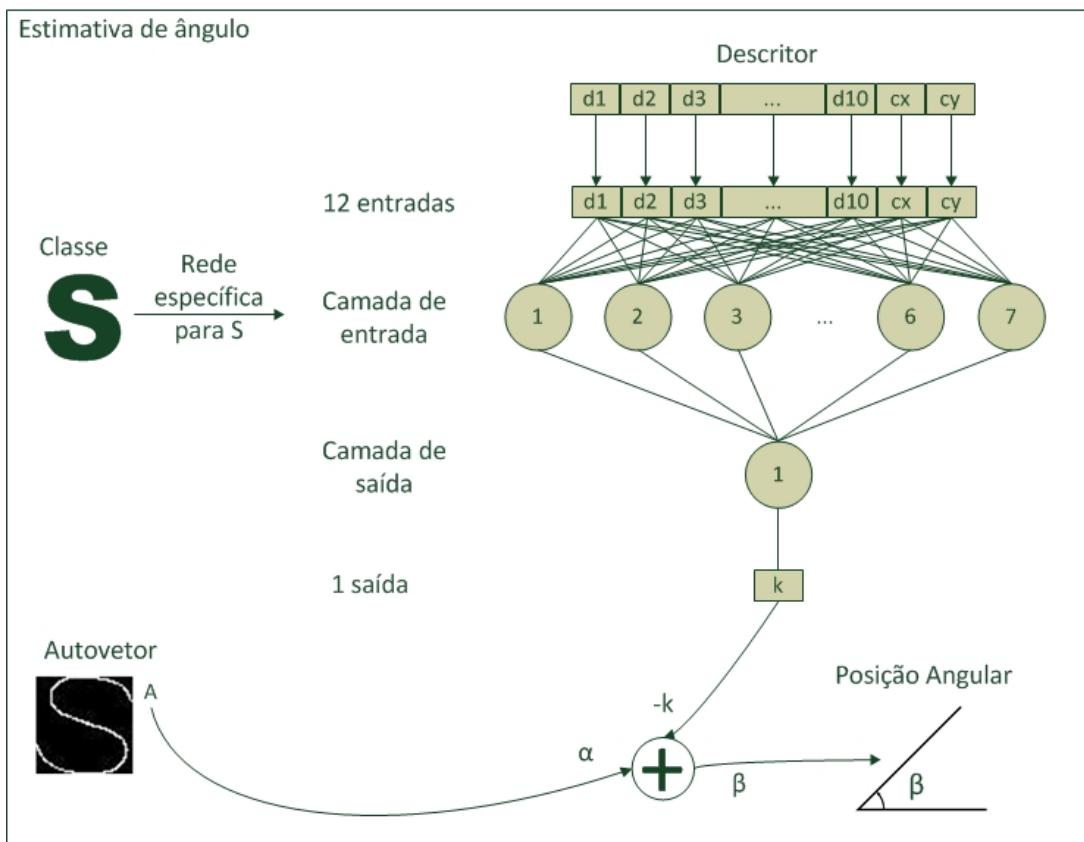
- $\beta$  é o ângulo entre as coordenadas da imagem e o zero do objeto. Ou seja, o valor que se deseja obter;
- $\alpha$  é o ângulo dado pelo autovetor;
- $k$  é a diferença entre o ângulo do autovetor e o zero do objeto. Este valor pode variar de normalização para normalização, conforme visto na figura 27.



**Figura 29:** Representação do autovetor (vermelho), dos eixos da imagem (amarelo) e do eixo do objeto (azul).

**Fonte:** Autoria própria

É necessário portanto determinar o valor de  $k$  para cada imagem normalizada. Utilizou-se para essa tarefa uma segunda rede neural para cada objeto. Essa rede recebe como entrada o descritor da imagem normalizada e informa na saída o valor de  $k$ . Para cada classe é gerada uma rede diferente, visto que as variações são diferentes de classe para classe. Após obter o valor  $k$  da rede basta utilizar a equação 34 para determinar o ângulo  $\beta$  do objeto. Esse processo está resumido na figura 30.



**Figura 30:** Segunda etapa da classificação.

**Fonte:** Autoria própria

Sabendo-se que redes neurais mostram bons resultados para aproximação de funções (MATHWORKS, 2013b), supôs-se que também gerariam bons resultados para o problema exposto, o que justifica a escolha dessa técnica de aprendizagem de máquina. A rede utilizada nessa etapa possui duas camadas, 12 entradas e uma saída. A primeira camada é formada por 7 *perceptrons*. Essa quantidade de perceptrons foi determinada empiricamente. A saída da rede possui um único perceptron pois a única saída da rede é o valor de  $k$ . A função de ativação da primeira camada da rede é a sigmoide pois permite o aprendizado de não linearidades. A segunda camada da rede possui uma função linear,  $y = x$ , favorecendo uma ampla faixa de valores de saída para  $k$  (MATHWORKS, 2013f).

Para o processo de treinamento da rede são utilizadas as mesmas amostras que foram utilizadas para o treinamento da rede responsável pela classificação do objeto. Em seguida são executadas as etapas descritas nas seções anteriores até a obtenção do descriptor de entrada da rede. Além disso é necessário o conhecimento do ângulo em que o objeto se encontra para o cálculo de  $k$  pela equação 34. Uma vez tendo o valor de  $k$  e o descriptor de entrada é possível treinar a rede.

A rede é treinada utilizando um algoritmo de *back propagation* conforme visto na seção 2.3.1. O algoritmo utilizado para esta rede em específico utiliza a otimização de *Levenberg-Marquardt*. A escolha foi baseada na velocidade de convergência do treinamento e nos bons resultados apresentados por esse método para o problema de aproximação de funções (MATHWORKS, 2013d).

### 3.7 CONSIDERAÇÕES

Expôs-se nesse capítulo a arquitetura do sistema desenvolvido. Inicialmente foi mostrado como a imagem de entrada foi segmentada utilizando técnicas de subtração de fundo, limiarização, filtros morfológicos, operações lógicas e seleção de maior região. Foi mostrado também como a imagem foi descrita utilizando propriedades de regiões e descritores de *Fourier*. E finalmente como os descritores foram aplicados nas redes neurais obtendo assim a classe dos objetos e a respectiva orientação.

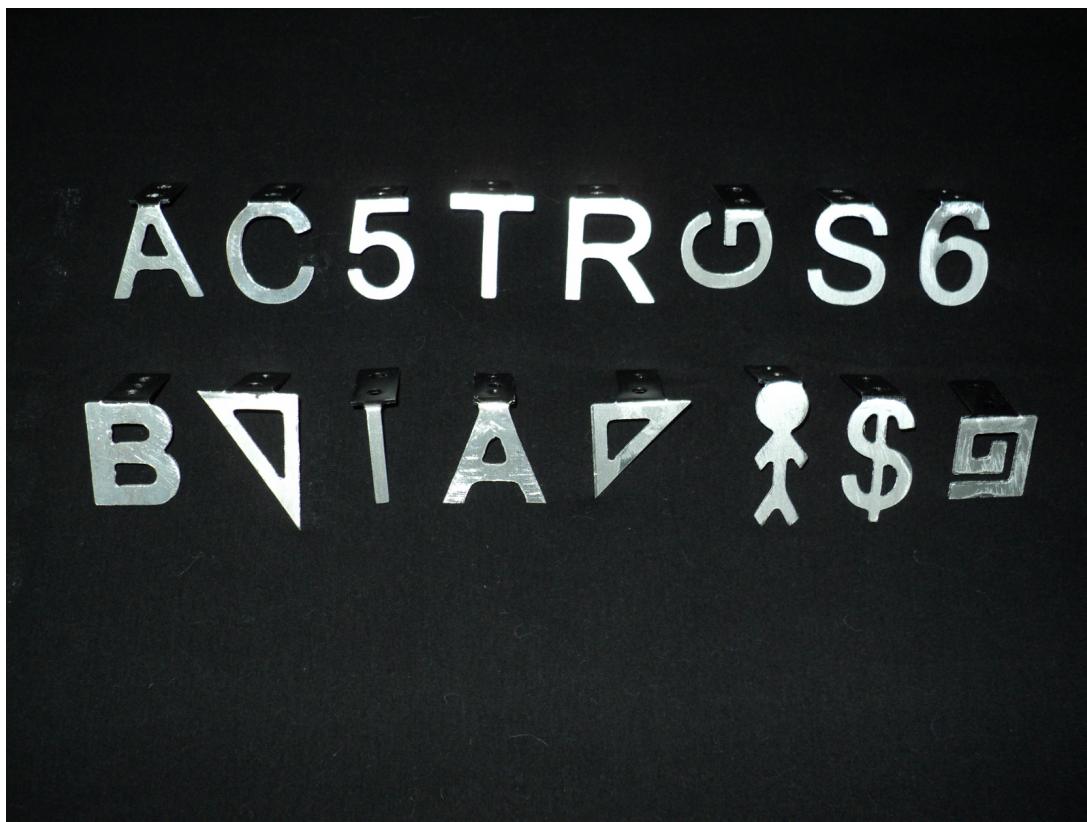
Embora tenha sido apresentado apenas a arquitetura final do sistema, ao longo do desenvolvimento do projeto diversas arquiteturas foram utilizadas. A cada ciclo de desenvolvimento da metodologia em espiral novos algoritmos foram acrescentados, substituídos ou otimizados.

## 4 TESTES E ANÁLISE DE RESULTADOS

Este capítulo apresenta o ambiente de testes desenvolvido para validar o sistema, os testes efetuados e os resultados obtidos. A seção 4.1 apresenta um ambiente de testes, que possibilita a captura de imagens de forma automatizada e os objetos utilizados para validação do sistema. Em seguida, na seção 4.2, são apresentados os testes aos quais o sistema foi submetido e os respectivos resultados obtidos. Ao final, na seção 4.3, são apresentadas algumas considerações.

### 4.1 AMBIENTE DE TESTES

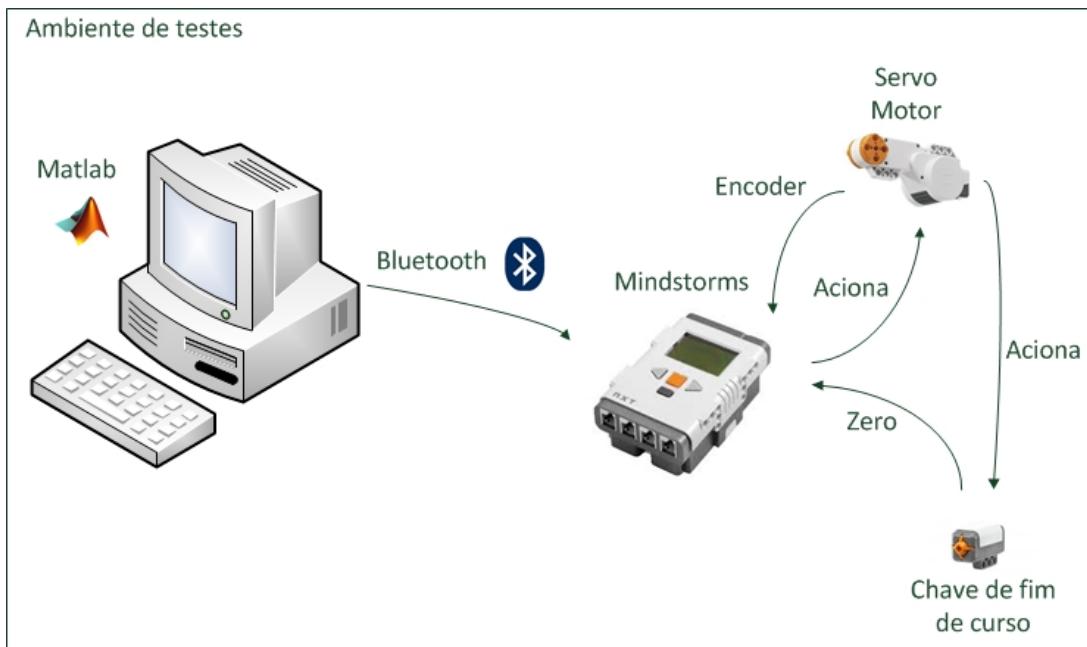
Inicialmente para validar o sistema proposto é necessária a definição dos objetos a serem reconhecidos. Com o objetivo de possibilitar a validação do sistema porém sem comprometer a aplicação futuramente pretendida para o projeto selecionou-se os objetos dispostos na figura 31. Os objetos são caracteres alfanuméricos, formas geométricas básicas e algumas formas aleatórias. Os objetos foram selecionados com o objetivo de testar a robustez do sistema. Portanto foram utilizadas formas semelhantes, como o caractere “S” e o número “5”, o caractere “A” e sua forma em negrito “A”, um triangulo retângulo e um triângulo isósceles. Além disso existem algumas peculiaridades em alguns caracteres. O caractere “S”, por exemplo, possui a parte superior ligeiramente menor que a inferior, avaliando a robustez e precisão da estimativa de ângulo. Para posteriores referências nomeou-se os objetos da figura 31 como segue, da esquerda para a direita e de cima para baixo tem-se: *A, C, 5, T, R, G, S, 6, Bb, tr, l, Ab, ti, bn, \$, sn*.



**Figura 31:** Objetos de teste.

**Fonte:** Autoria própria

Para facilitar a execução de testes o ambiente deve permitir a captura de imagens dos objetos em diversas orientações angulares, e possibilitar também o reposicionamento automático dos objetos para outros ângulos. Um ambiente que atende a esses requisitos foi montado utilizando o kit de montagem de robôs da LEGO chamado *Mindstorms* (LEGO, 2013). O ambiente foi montado conforme mostra a figura 32.



**Figura 32:** Montagem do ambiente de testes.

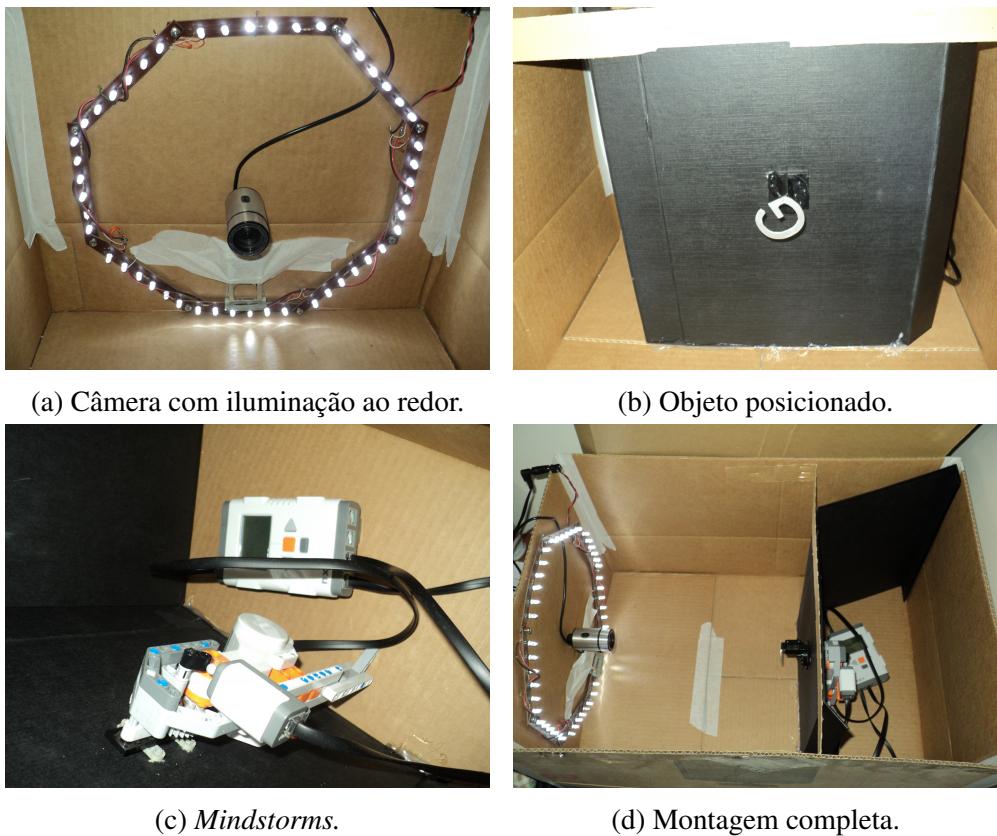
**Fonte:** Autoria própria

O sistema desenvolvido em MATLAB comunica-se com a unidade programável do *Mindstorms*, também chamada de *brick*, através de uma conexão *bluetooth*. A comunicação é unidirecional do ponto de vista do MATLAB. O *brick* comunica-se com um servo-motor que fornece realimentação por meio de um *encoder*. Além da realimentação do *encoder* foi adicionada uma malha de realimentação externa para fornecer um referencial absoluto ao sistema por meio de uma chave de fim de curso.

As funções implementadas dentro do *brick* são as seguintes:

- Reset: Gira o objeto no sentido anti-horário até atingir o zero absoluto definido pelo acionamento da chave de fim de curso.
- Step: Gira o objeto no sentido anti-horário até um ângulo especificado por meio de um parâmetro. A posição do servo-motor é obtida pela realimentação do *encoder*.

Para completar o ambiente de testes foi utilizado uma *webcam* acessada pelo MATLAB para captura de imagens e foi montado um sistema de iluminação. O sistema de iluminação foi montado ao redor da câmera de maneira a não formar sombras visíveis a partir do ponto de vista da *webcam*. A montagem completa pode ser vista nas imagens da figura 33.



**Figura 33:** Montagem do ambiente de testes.

**Fonte:** Autoria própria

## 4.2 TESTES

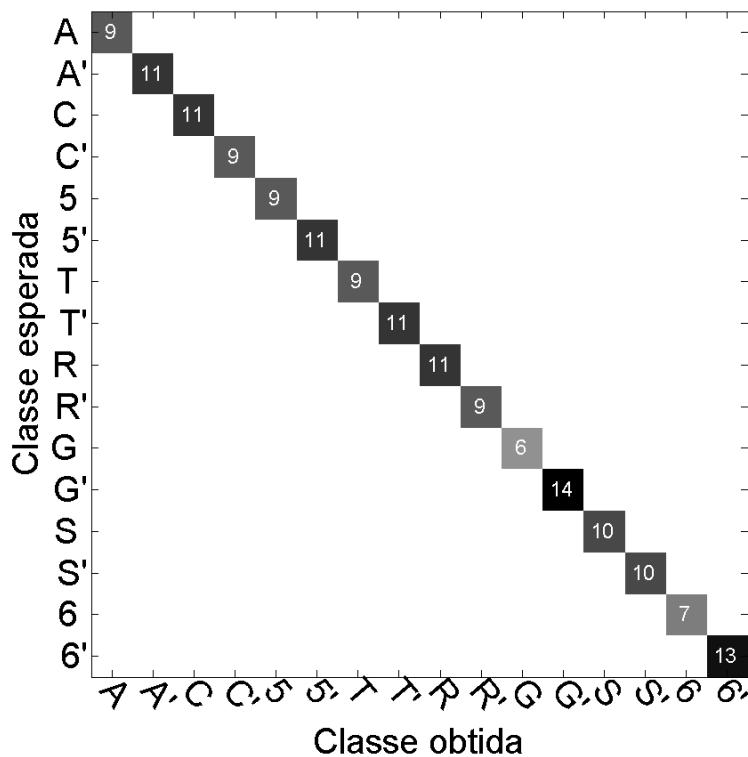
O sistema foi testado com duas configurações diferentes. A primeira configuração foi formada pelos oito primeiros objetos da figura 31. Ou seja: *A, C, 5, T, R, G, S, 6*. A segunda configuração foi formada com todos os objetos apresentados na figura, ou seja: *A, C, 5, T, R, G, S, 6, Bb, tr, l, Ab, ti, bn, \$, sn*.

### 4.2.1 Primeiro teste: 8 objetos

Inicialmente cada objeto foi posicionado no ambiente de testes e foram efetuadas 360 capturas. Uma captura a cada  $1^\circ$ , tanto para o fundo, quanto para o objeto. As imagens foram salvas e a classe e ângulo de captura armazenados. Após capturar as imagens para todos os objetos os descritores foram calculados conforme exposto no capítulo 3. Tanto a rede de classificação quanto as redes de estimativa de ângulo foram treinadas com os descritores das 360 capturas.

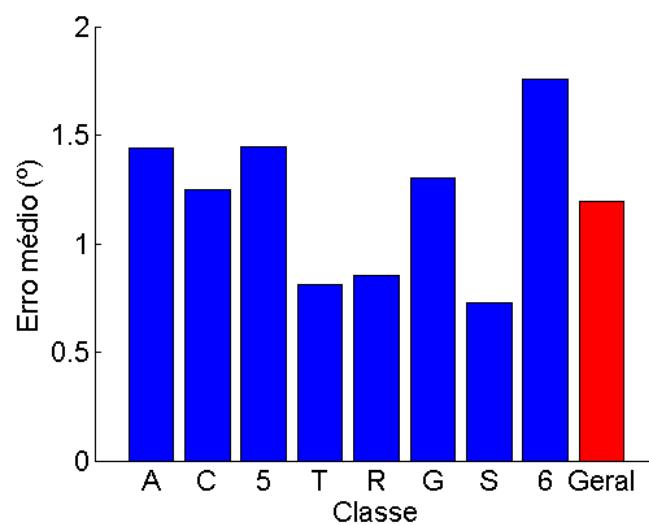
Na sequência foram capturadas 20 imagens em rotações aleatórias de cada objeto e as res-

pectivas imagens de fundo ( $8 \text{ objetos} \times 20 \text{ ângulos} = 160 \text{ imagens}$ ). Novamente os dados foram armazenados. O sistema foi executado para as 160 imagens de teste capturadas e o resultado foi comparado com os valores esperados. A comparação entre valores de classe esperados e obtidos pode ser visto na matriz de confusão apresentada na figura 34. Para cada objeto existem duas classes na matriz de confusão. Uma para a normalização utilizando o autovetor  $A$  e outra para o autovetor  $A' = -A$ . O valor esperado na diagonal da matriz de confusão é de  $A + A' = 20$ , pois foram capturadas 20 imagens que podem ter a normalização correta utilizando tanto  $A$  quanto  $A'$ . Para a estimativa do ângulo o resultado pode ser visto nas figuras 35, 36 e 37.



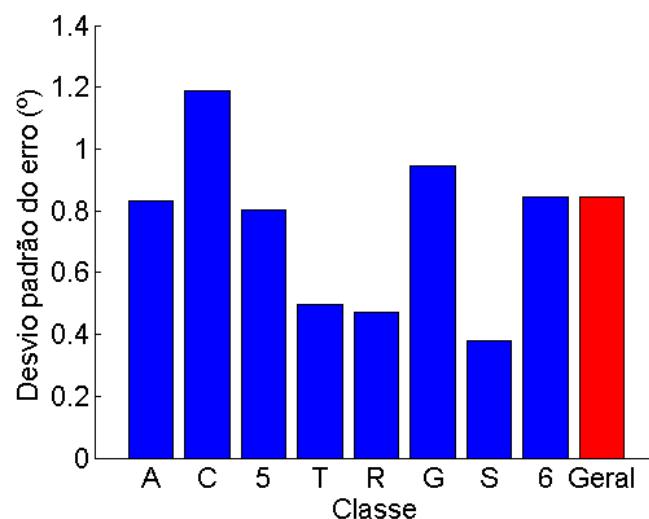
**Figura 34:** Matriz de confusão para 8 objetos (formando 16 classes).

**Fonte:** Autoria própria



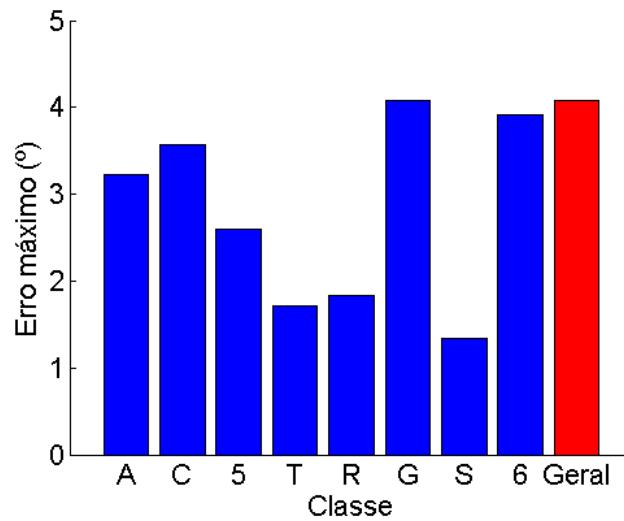
**Figura 35:** Erro médio da estimativa do ângulo para 8 objetos.

**Fonte:** Autoria própria



**Figura 36:** Desvio padrão do erro da estimativa do ângulo para 8 objetos.

**Fonte:** Autoria própria



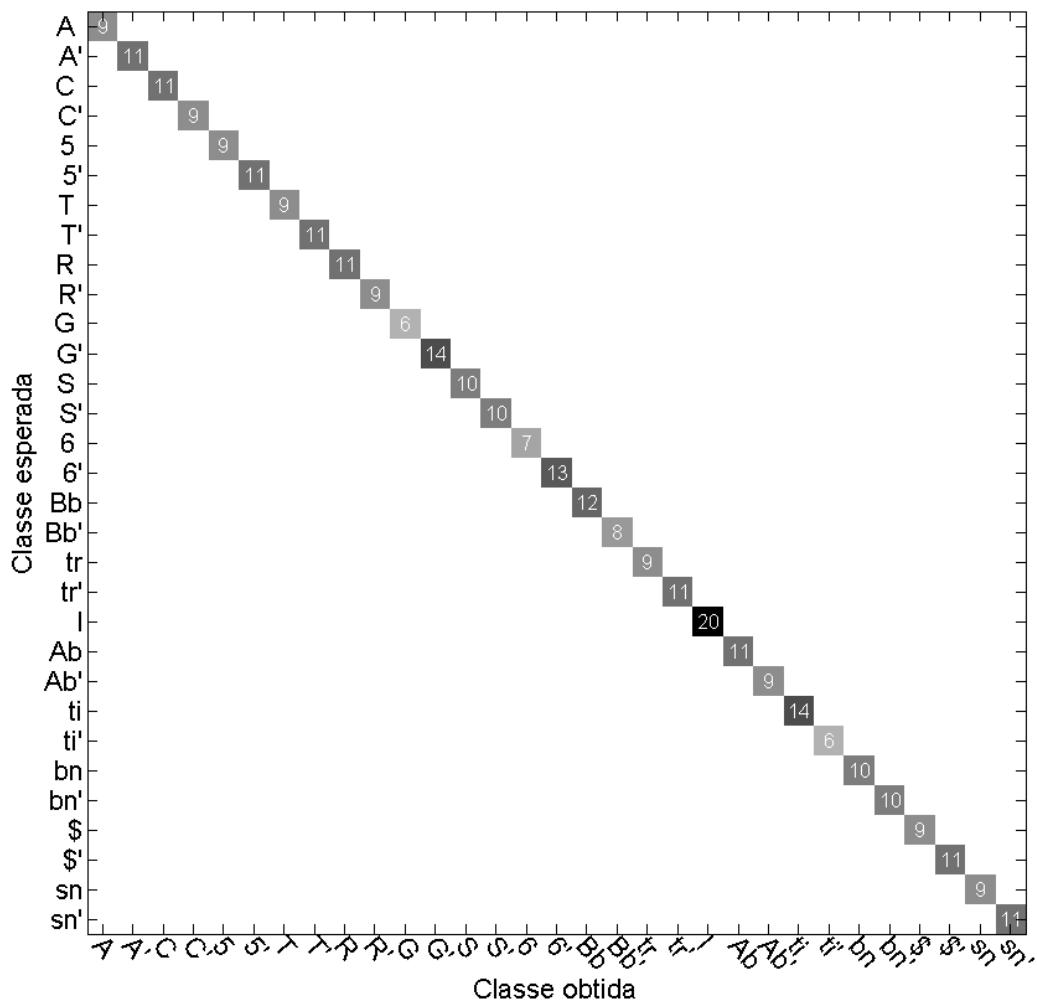
**Figura 37:** Erro máximo da estimativa do ângulo para 8 objetos.

**Fonte: Autoria própria**

#### 4.2.2 Segundo teste: 16 objetos

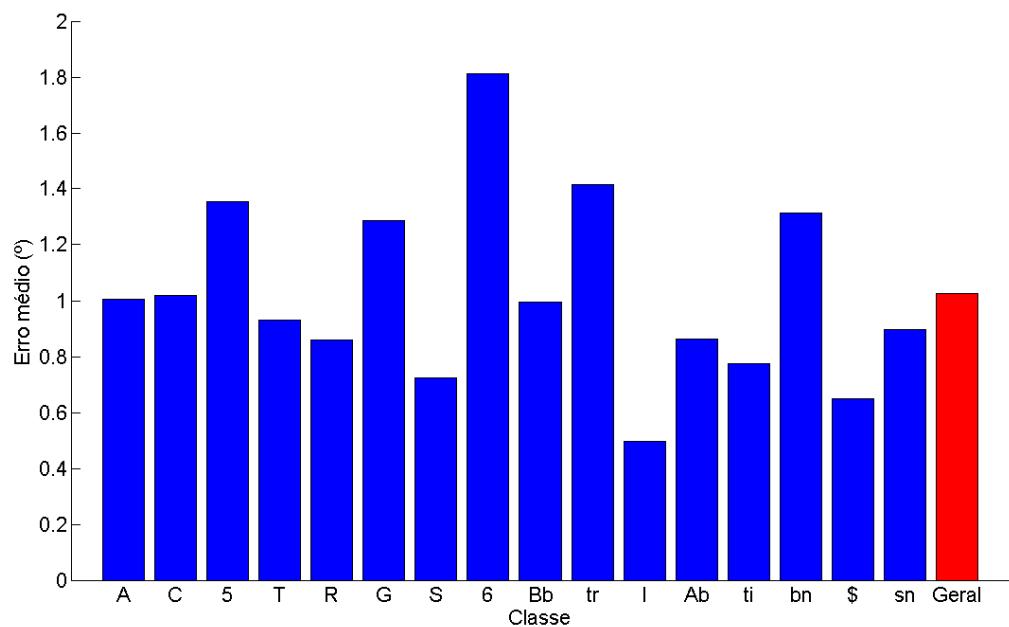
O mesmo procedimento executado para o primeiro teste foi executado para o segundo, no entanto com algumas diferenças. Foram utilizados todos os 16 objetos mostrados na figura 31. Para o objeto *l* foram efetuadas apenas 180 capturas de 0 a  $179^\circ$  por se tratar de um objeto simétrico. Para os demais objetos foram efetuadas 360 capturas. Após a captura as redes de classificação e estimativa de ângulo foram treinadas.

Em seguida foram capturadas 20 imagens em rotações aleatórias para cada objeto, totalizando 320 imagens de teste. O sistema foi executado e os resultados foram comparados com os valores esperados conforme a matriz de confusão apresentada na figura 38. A matriz de confusão apresentada mostra apenas 31 classes ao invés de 32 como esperado. Isso se dá pelo fato do objetos *l* ser simétrico, resultando sempre em um autovetor correto para a normalização. Para a estimativa do ângulo o resultado pode ser visto nas figuras 39, 40 e 41.



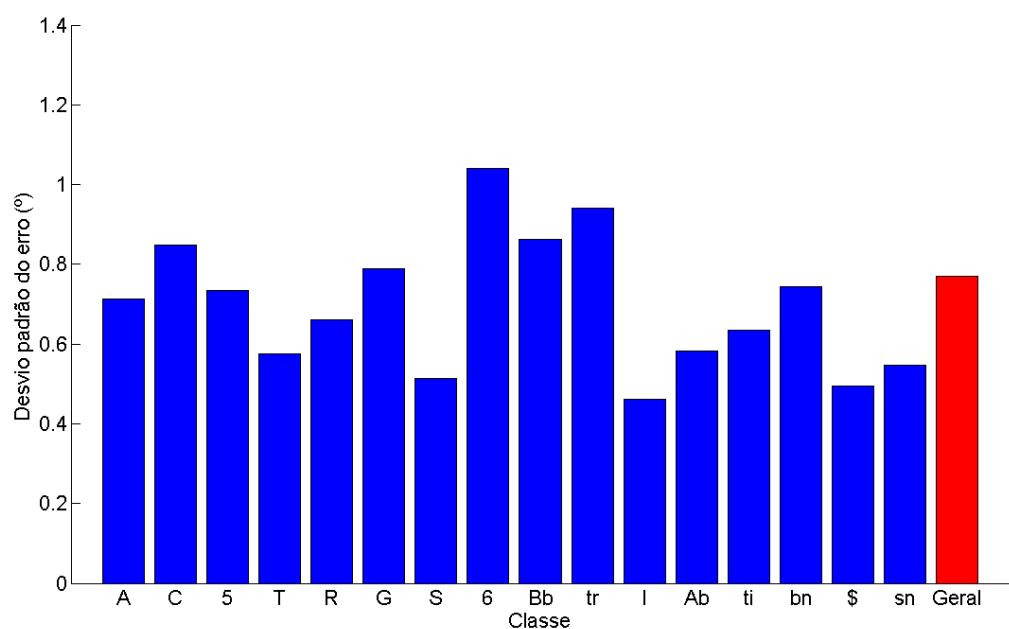
**Figura 38:** Matriz de confusão para 16 objetos (formando 31 classes).

**Fonte:** Autoria própria



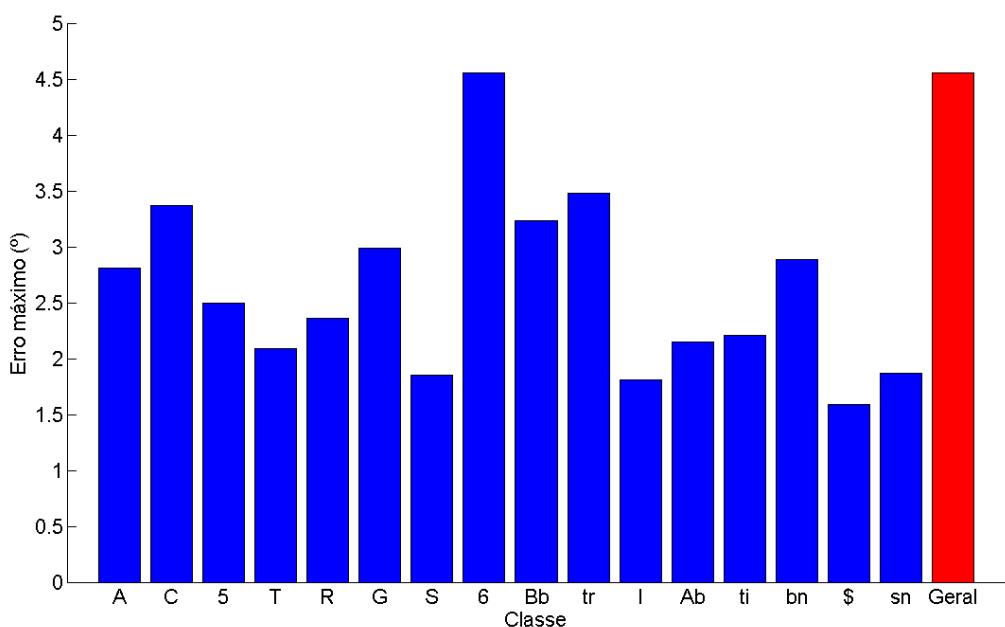
**Figura 39:** Erro médio da estimativa do ângulo para 16 objetos.

**Fonte:** Autoria própria



**Figura 40:** Desvio padrão do erro da estimativa do ângulo para 16 objetos.

**Fonte:** Autoria própria



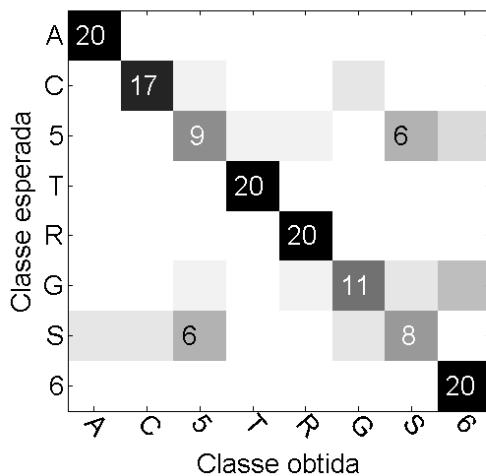
**Figura 41:** Erro máximo da estimativa do ângulo para 16 objetos.

**Fonte:** Autoria própria

#### 4.3 CONSIDERAÇÕES

Nesse capítulo apresentou-se um ambiente de testes automatizado que possibilitou a validação do sistema apresentado no capítulo 3. Utilizando o ambiente foram executados dois grupos de testes. O primeiro com 8 objetos e o segundo com 16 objetos.

O sistema mostrou-se capaz de classificar todos os objetos corretamente com 100% de acerto. Isso se deu devido à qualidade dos descritores calculados, que permitiram resumir as principais características de cada classe. Além disso a topologia da rede de classificação teve papel fundamental para a convergência do treinamento e para a qualidade de decisão. Inicialmente havia sido utilizada uma rede com apenas uma saída, onde cada classe estava associada a um intervalo de valores dentro da faixa dinâmica de saída da rede (0 até 1). O resultado dessa abordagem inicial está exposto na figura 42. Como se pode ver, a taxa de acerto foi de apenas 78%, o que é inaceitável para o problema, visto que um erro na classificação automaticamente resultará em falha na estimativa do ângulo.



**Figura 42:** Matriz de confusão para 8 objetos - rede com 1 saída.

**Fonte:** Autoria própria

Na estimativa do ângulo obteve-se a precisão de  $\pm 4.1^\circ$  para o teste com 8 objetos e  $\pm 4.5^\circ$  para o teste com 16 objetos. O erro médio manteve-se em  $1.3^\circ$ , com desvio de  $0.85^\circ$  para o primeiro teste e  $1.1^\circ$ , com desvio de  $0.8^\circ$  para o segundo teste. O erro médio geral ficou portanto em  $1.2^\circ$ . O erro na estimativa do ângulo é bastante influenciado pelo sucesso na etapa de segmentação e normalização da imagem. Sendo assim, para obter bons resultados é necessário controlar ao máximo as condições externas de captura das imagens. Condições boas de iluminação por exemplo tendem a melhorar o resultado da etapa de segmentação.

Pôde-se observar também que o erro médio na estimativa do ângulo não aumentou com a inserção de mais objetos no sistema. Isso se deu devido a estimativa ser feita com uma rede neural específica para cada objeto. O erro máximo aumentou devido aos pesos iniciais das redes neurais que são escolhidos aleatoriamente antes do treinamento. Essa aleatoriedade pode influenciar o resultado entre redes treinadas e testadas com o mesmo conjunto de treinamento e teste.

Os testes mostraram portanto que o sistema foi capaz de identificar os objetos corretamente e estimar a posição angular com precisão de  $\pm 4.5^\circ$  no pior caso.

## 5 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as considerações finais da monografia. Aqui são discutidos os objetivos do projeto e os resultados alcançados no sistema desenvolvido, são apresentadas considerações sobre a metodologia utilizada, além de sugestões para trabalhos futuros.

### 5.1 DISCUSSÃO DO DESENVOLVIMENTO E DOS RESULTADOS

Conforme apresentado nos capítulos anteriores, desenvolveu-se um sistema capaz de ser integrado em uma linha de produção minimizando falhas e erros de classificação de objetos metálicos. Os objetos utilizados na validação do sistema foram caracteres e símbolos de alumínio. O sistema também é capaz de estimar a orientação angular de cada objeto para posterior reposicionamento caso necessário. Atingiu-se portanto o objetivo geral do trabalho visto que através da utilização de um sistema de redes neurais foi possível a classificação dos objetos testados com 100% de acerto e mediu-se o ângulo de rotação com erro médio de  $1.2^\circ$ . Como pôde ser visto nas seções 3.1 e 3.2 foram analisadas e aplicadas as seguintes técnicas de segmentação de imagens: subtração de fundo, limiarização, filtros morfológicos, esqueletonização e normalização utilizando a análise de componentes principais. A utilização dessas técnicas permitiram a identificação da região de interesse, atingindo portanto o primeiro objetivo específico. O segundo objetivo específico foi cumprido pela formação de descritores contendo informações suficientes para classificar os objetos com sucesso. Para formação dos descritores foram utilizados descritores de *Fourier* e propriedades de regiões como área, excentricidade, número de Euler, tamanho do eixo principal, entre outras. O terceiro objetivo foi contemplado pelo emprego de redes neurais para classificação dos objetos e estimativa de ângulo.

O desenvolvimento do projeto foi dividido em três etapas conforme proposto na metodologia. Porém foi necessária a volta à etapa de descrição para ajustes durante o desenvolvimento da terceira etapa. Cada etapa foi executada utilizando a metodologia de desenvolvimento em espiral, onde a cada ciclo de desenvolvimento foram agregadas novas funcionalidades ao sistema.

As atividades desenvolvidas em cada ciclo estão dispostas na tabela 2.

**Tabela 2:** Etapas do desenvolvimento em espiral.

Etapa-Ciclo	Descrição
1-1	Implementação da subtração de fundo.
1-2	Implementação da limiarização utilizando aproximações sucessivas e separação dos canais RGB.
1-3	Implementação da limiarização pelo método de Otsu.
1-4	Implementação de filtro morfológico e separação dos canais RGB.
1-5	Implementação do filtro de maior região conexa.
1-6	Implementação de crescimento de regiões baseado em características locais.
1-7	Remoção de crescimento de regiões e implementação de esqueletização e poda.
1-8	Implementação de normalização utilizando PCA.
2-1	Implementação de descrição utilizando descritores de <i>Fourier</i> .
2-2	Implementação de descrição utilizando propriedades de regiões.
3-1	Implementação de classificação utilizando redes neurais com uma única saída.
3-2	Adaptação da estrutura da rede para uma saída para cada classe.
3-3	Utilização de duas saídas da rede para cada classe. Uma para normalização utilizando o autovetor e outra para o autovetor oposto.
2-3	Adição de propriedades da imagem pré-normalizada ao descritor de classificação.
3-4	Implementação da estimativa de ângulo utilizando redes neurais com uma única saída.

**Fonte: Autoria própria**

No capítulo 2 apresentou-se o embasamento teórico para o desenvolvimento do projeto, abrangendo as três áreas fundamentais de sistemas de processamento de imagens: segmentação, descrição e classificação. Foram estudadas as técnicas de subtração de fundo, limiarização, filtros morfológicos e esqueletização que permitiram através de sua combinação a identificação de regiões de interesse. Além da identificação da região de interesse foi apresentada a técnica de normalização de imagens utilizando a análise de componentes principais. Foi fornecido embasamento teórico para efetuar a descrição de imagens utilizando as seguintes propriedades de regiões: área, área convexa, excentricidade, número de Euler, extensão, área preenchida, tamanho do eixo principal, tamanho do eixo secundário e solidez. Além disso foram abordados os conceitos para efetuar a descrição utilizando descritores de *Fourier*. Por fim apresentou-se uma visão geral de redes neurais e foram apresentadas noções sobre treinamento de redes utilizando algoritmos do tipo *back propagation*.

No capítulo 3 expôs-se a arquitetura do sistema desenvolvido. Inicialmente mostrou-se como a imagem de entrada foi segmentada utilizando técnicas de subtração de fundo, limiarização, filtros morfológicos, operações lógicas e seleção de maior região. Mostrou-se também como a

imagem foi descrita utilizando propriedades de regiões e descritores de *Fourier*. E finalmente como os descritores foram aplicados nas redes neurais obtendo assim a classe dos objetos e a respectiva orientação.

Finalmente no capítulo 4 apresentou-se um ambiente de testes automatizado que possibilitou a validação do sistema apresentado no capítulo 3. Utilizando o ambiente, foram executados dois grupos de testes: o primeiro com 8 objetos e o segundo com 16 objetos.

O sistema apresentou-se capaz de classificar todos os objetos testados corretamente com 100% de acerto. Isso se deu devido à qualidade dos descritores calculados, que permitiram capturar as principais características de cada classe. Além disso a topologia da rede de classificação teve papel fundamental para a convergência do treinamento e para a qualidade de decisão.

Na estimativa do ângulo, obteve-se a precisão de  $\pm 4.1^\circ$  para o teste com 8 objetos e  $\pm 4.5^\circ$  para o teste com 16 objetos. O erro médio manteve-se em  $1.3^\circ$ , com desvio de  $0.85^\circ$  para o primeiro teste e  $1.1^\circ$ , com desvio de  $0.8^\circ$  para o segundo teste. O erro médio geral ficou portanto em  $1.2^\circ$ . O erro na estimativa do ângulo é bastante influenciado pelo sucesso na etapa de segmentação e normalização da imagem. Sendo assim, para obter bons resultados é necessário controlar ao máximo as condições externas de captura das imagens. Condições boas de iluminação que não formem sombras, por exemplo, tendem a melhorar o resultado da etapa de segmentação. Pôde-se observar também que o erro médio na estimativa do ângulo não aumentou com a inserção de mais objetos no sistema. Isso se deu devido a estimativa ser feita com uma rede neural específica para cada objeto. O erro máximo aumentou devido aos pesos iniciais das redes neurais que são escolhidos aleatoriamente antes do treinamento. Essa aleatoriedade pode influenciar o resultado entre redes treinadas e testadas com o mesmo conjunto de treinamento e teste.

Os testes mostraram que o sistema foi capaz de classificar os objetos corretamente e estimar a posição angular com precisão de  $\pm 4.5^\circ$  no pior caso. Portanto, os objetivos do projeto foram alcançados e a metodologia proposta foi seguida com algumas adaptações.

## 5.2 TRABALHOS FUTUROS

Futuramente o projeto que foi escrito em MATLAB deverá ser implementado em uma linguagem de programação convencional como C++ ou JAVA. Isso faz se necessário para implantação do sistema em ambiente industrial real, onde ele deverá fazer parte de uma linha de produção. Mais detalhes da futura aplicação do projeto não podem ser apresentados devido ao interesse comercial no projeto.

## REFERÊNCIAS

- GHAOUI, L. E. **Hyper-Textbook: Optimization Models and Applications**. 2013. Disponível em: <<https://inst.eecs.berkeley.edu/ee127a/book/login/index.html>>. Acesso em: 13 de novembro de 2013.
- GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing (3rd Edition)**. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. ISBN 013168728X.
- LEGO. **Mindstorms**. 2013. Disponível em: <<http://mindstorms.lego.com>>.
- LEON, S. **Linear algebra with applications**. [S.l.]: Prentice Hall, 1998. ISBN 9780138493080.
- LIN, H.-D. Computer-aided visual inspection of surface defects in ceramic capacitor chips. **Journal of Materials Processing Technology**, v. 189, p. 19 – 25, 2007. ISSN 0924-0136. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0924013607000052>>.
- LIN, H.-D. Tiny surface defect inspection of electronic passive components using discrete cosine transform decomposition and cumulative sum techniques. **Image and Vision Computing**, v. 26, n. 5, p. 603 – 621, 2008. ISSN 0262-8856. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0262885607001151>>.
- MAR, N.; YARLAGADDA, P.; FOOKE, C. Design and development of automatic visual inspection system for {PCB} manufacturing. **Robotics and Computer-Integrated Manufacturing**, v. 27, n. 5, p. 949 – 962, 2011. ISSN 0736-5845. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0736584511000457>>.
- MATHWORKS. **Choose a Multilayer Neural Network Training Function**. 2013. Disponível em: <<http://www.mathworks.com/help/nnet/ug/choose-a-multilayer-neural-network-training-function.html>>. Acesso em: 25 de novembro de 2013.
- MATHWORKS. **Fit Data with a Neural Network**. 2013. Disponível em: <<http://www.mathworks.com/help/nnet/gs/fit-data-with-a-neural-network.html>>. Acesso em: 25 de novembro de 2013.
- MATHWORKS. **Image Enhancement and Analysis**. 2013. Disponível em: <<http://www.mathworks.com/help/images/image-enhancement-and-analysis.html>>. Acesso em: 06 de Março de 2012.
- MATHWORKS. **Levenberg-Marquardt backpropagation**. 2013. Disponível em: <<http://www.mathworks.com/help/nnet/ref/trainlm.html>>. Acesso em: 25 de novembro de 2013.
- MATHWORKS. **Matlab**. 2013. Disponível em: <<http://www.mathworks.com/products/matlab/>>.

MATHWORKS. **Multilayer Neural Network Architecture.** 2013. Disponível em: <<http://www.mathworks.com/help/nnet/ug/multilayer-neural-network-architecture.html>>. Acesso em: 17 de novembro de 2013.

MATHWORKS. **regionprops.** 2013. Disponível em: <<http://www.mathworks.com/help/images/ref/regionprops.html>>. Acesso em: 16 de novembro de 2013.

PICCARDI, M. Background subtraction techniques: a review. In: **Systems, Man and Cybernetics, 2004 IEEE International Conference on.** [S.l.: s.n.], 2004. v. 4, p. 3099–3104 vol.4. ISSN 1062-922X.

WIKIPEDIA. **Second moment of area.** 2013. Disponível em: <[http://en.wikipedia.org/wiki/Second\\_moment\\_of\\_area](http://en.wikipedia.org/wiki/Second_moment_of_area)>. Acesso em: 15 de novembro de 2013.

ZENG, Z.; MA, L.; ZHENG, Z. Automated extraction of pcb components based on specularity using layered illumination. **Journal of Intelligent Manufacturing**, Springer US, v. 22, n. 6, p. 919–932, 2011. ISSN 0956-5515. Disponível em: <<http://dx.doi.org/10.1007/s10845-009-0367-6>>.

ZHENG, H.; KONG, L.; NAHAVANDI, S. Automatic inspection of metallic surface defects using genetic algorithms. **Journal of Materials Processing Technology**, v. 125-126, n. 0, p. 427 – 433, 2002. ISSN 0924-0136. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0924013602002947>>.