



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Contador Automático de Moedas Utilizando Redes Neurais Artificiais

Gustavo M. Meneses

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. Alexandre Zaghetto

Brasília
2016



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Contador Automático de Moedas Utilizando Redes Neurais Artificiais

Gustavo M. Meneses

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof. Dr. Alexandre Zaghetto (Orientador)
CIC/UnB

Prof. Dr. Bruno Luigi Macchiavello Espinoz
Universidade de Brasília

Prof. Dr. Camilo Chang Dorea
Universitat Politecnica de Catalunya

Prof. Dr. Ricardo Pezzuol Jacobi
Coordenador do Curso de Engenharia da Computação

Brasília, 01 de julho de 2016

Dedicatória

Dedico este trabalho a Deus e a Virgem Maria que estão sempre a minha frente me mostrando quais rumos devo seguir.

Agradecimentos

Agradeço aos meus pais, **Cícero Emanuel Vieira de Meneses** e **Ana Clébia Monteiro Meneses** por todo o apoio e amor que me foi dado em todos os momentos da minha vida.

Agradeço, também, aos meus irmãos **Rayane Monteiro Meneses** e **Henrique Monteiro Meneses** pela paciência e apoio durante a construção do presente trabalho.

Agradeço ao professor **Dr. Alexandre Zaghetto** por toda a ajuda para o bom andamento desse trabalho e todo o conhecimento a mim passado tanto de cunho profissional como também pessoal

Agradeço aos meus amigos de infância **Matheus Falcão**, **Lucas Lopes** e **Leandro Fernandes** que sempre estiveram presentes e me apoiaram independente da situação.

Resumo

O objetivo do presente trabalho é fazer a contagem do valor em moedas a partir da aquisição de uma imagem das mesmas. Apesar do crescente uso dos cartões de crédito e cédulas impressas, o mercado de alimentos ainda utiliza bastante esse meio de pagamento. Tudo será feito em um ambiente totalmente controlado onde o nível de desordem é minimizado ao máximo de forma que a disposição das moedas no anteparo de contagem é definida pelo usuário. A ideia é que o software funcione com precisão em lugares que possuam uma luminosidade relativamente boa e que a câmera de captura das imagens esteja a uma altura fixa da base de aquisição. Isso irá facilitar e agilizar o trabalho de alguns tipos de empresas que precisam fazer a contagem de uma grande quantidade de moedas, como supermercados. Por meio da utilização de técnicas em Visão Computacional, Redes neurais artificiais, o descritor de textura GLCM e Processamento de Imagens, busca-se atingir um resultado satisfatório. Este seria um processamento que permita a identificação de todas as moedas presentes na foto, com uma certa tolerância a oclusões, e o mínimo de erros ao fazer a contagem após a identificação. Tais resultados foram atingidos no projeto em questão.

Palavras-chave: Redes neurais artificiais, contagem, GLCM

Abstract

The aim of this work is to count the value of coins from the acquisition of an image of them. Despite the increasing use of credit cards and printed ballot, the food market still uses these kind of payment. Initially, everything will be done in a fully controlled environment where the level of disorder is minimized to the maximum so that the arrangement of the coins on the counting bulkhead is defined by the user. The idea is that the software works precisely in places that have a relatively good light and that the image capture camera is at a fixed height from the acquisition base. This will facilitate and expedite the work of some types of businesses that need to count a lot of coins, such as supermarkets. Through the use of techniques in Computer Vision, Artificial neural networks, the texture descriptor GLCM and Image Processing, it seeks to achieve a satisfactory result. This would be a process that allows the identification of all the coins present in the photo, with a certain occlusion tolerance, and minimal errors to make the counting after identification. These results were achieved in this project.

Keywords: Artificial neural network, count, GLCM

Sumário

1	Introdução	1
2	Processamento de Imagens	3
2.1	Imagem em Nível de Cinza	3
2.2	Filtro Gaussiano	4
2.3	Operações Lógicas	6
2.4	Descritor de textura	7
2.4.1	Matriz de Co-ocorrência de Níveis de Cinza (GLCM)	7
2.5	Espaço de Cores (YCbCr)	9
2.6	Detecção de bordas	10
2.6.1	Canny	11
3	Transformada de Hough	13
3.1	Transformada de Hough Circular	16
4	Classificação de Padrões	17
4.1	Decisão teórica	18
5	Redes Neurais Artificiais	19
5.1	Neurônio Biológico	19
5.2	Neurônio artificial	20
5.3	Funções de ativação	22
5.4	<i>Feed-forward</i>	23
6	Solução Proposta	24
6.1	Calibração de iluminação	24
6.2	Aquisição da imagem	24
6.3	Pré-processamento	25
6.4	Extração das características	27
6.5	Classificação por meio de Redes Neurais Artificiais	27

6.6	Equipamento	28
7	Resultados Experimentais	31
7.1	Descrição da base de dados	31
7.2	Organização das características	33
7.3	Treinamento	37
7.4	Experimentos	38
8	Conclusão	43
	Referências	45

Lista de Figuras

2.1	Exemplo de conversão de (a) imagem original em RGB para (b) imagem em níveis de cinza	3
2.2	Gráfico exemplo de um função gaussiana.	4
2.3	Exemplo de máscara gaussiana.	5
2.4	Aplicação do filtro gaussiano.	5
2.5	Principais operações lógicas	6
2.6	Operação lógica AND feita para extrair a região de interesse na imagem [8].	6
2.7	Exemplo da construção da matriz GLCM.	7
2.8	Exemplo do espaço de cor YCbCr. A representação dos canais Cb e Cr é feita em níveis de cinza	10
2.9	Bordas exemplificadas de forma gráfica.	11
2.10	Exemplo da aplicação da detecção de bordas pelo método de Canny.	12
3.1	Conceitos de parametrização. Apresentação do plano xy e da mudança de parâmetros para o plano ab	14
3.2	Representação das células de acumulação do plano ab	15
3.3	Representação polar de linha e células acumuladoras	15
3.4	Padrão de votação da transformada de Hough circular.	16
4.1	Diagrama de blocos de uma decisão teórica.	18
5.1	Esquema de um neurônio.	20
5.2	Neurônio simples.	21
5.3	Neurônio com um vetor de entrada.	21
5.4	Função de ativação linear.	22
5.5	Função de ativação log-sigmoide.	22
5.6	Função de ativação tangente hiperbólica.	23
5.7	Rede neural <i>feedforward</i>	23
6.1	Exemplo de moeda capturada.	25
6.2	Imagens após os dois primeiros passos do pré-processamento	25

6.3	Imagem marcada após o uso da transformada de Hough circular.	26
6.4	Passos da operação lógica	26
6.5	Moeda nos espaços de cores Cb e Cr	27
6.6	Caixa utilizada nos experimentos.	28
6.7	Estrutura de captura das imagens.	29
6.8	Circuito de controle de iluminação e câmera.	29
6.9	Esquema do circuito implementado para a realização do projeto.	30
7.1	Moedas utilizadas.	31
7.2	Grid de aquisição (4x4) para o treinamento e indicação da posição da moeda na base em relação a câmera no momento da classificação do tipo de moeda.	32
7.3	Exemplos de posições no grid utilizado para treinamento e classificação. . .	32
7.4	Histograma da distribuição dos raios de cada tipo de moeda.	34
7.5	Histograma da distribuição dos níveis de crominância azul.	34
7.6	Histograma da distribuição dos níveis de crominância vermelho.	35
7.7	Histograma da distribuição do contraste.	35
7.8	Histograma da distribuição da correlação.	36
7.9	Histograma da distribuição da homogeneidade.	36
7.10	Matriz de confusão.	38
7.11	Experimento 1 sem oclusão. Valor esperado: $R\$1,90$. Valor calculado : $R\$1,90$	39
7.12	Experimento 2 sem oclusão. Valor esperado: $R\$2,75$. Valor calculado : $R\$2,75$	39
7.13	Experimento 3 sem oclusão. Valor esperado: $R\$3,25$. Valor calculado : $R\$3,25$	39
7.14	Experimento 1 com oclusão. Valor esperado: $R\$2,15$. Valor calculado : $R\$2,15$	40
7.15	Experimento 2 com oclusão. Valor esperado: $R\$4,35$. Valor calculado : $R\$4,35$	40
7.16	Experimento 3 com oclusão. Valor esperado: $R\$4,00$. Valor calculado : $R\$4,00$	40
7.17	Experimento 1 com erro. Valor esperado: $R\$7,55$. Valor calculado : $R\$8,60$.	41
7.18	Experimento 2 com erro. Valor esperado: $R\$2,20$. Valor calculado : $R\$4,25$	41
7.19	Experimento 3 com erro. Valor esperado: $R\$7,00$. Valor calculado : $R\$8,25$	41

Capítulo 1

Introdução

Atualmente, as moedas são utilizadas por toda a população mundial. Com a criação do papel-moeda a cunhagem da moeda metálica se restringiu a valores baixos e é usualmente vista como troco.

No mercado de alimentos, é comum o uso das moedas metálicas, pois a maior parte dos produtos desse tipo de comércio possui um valor fracionado ou inferior aos valores impressos no papel-moeda. No momento em que esses estabelecimentos precisam fazer a contagem total do valor recebido em moedas, eles perdem um bom tempo fazendo a contagem manual das mesmas.

O problema em questão, a contagem manual de grande quantidade de moedas, é comumente encontrado em grandes empresas, bancos e como citado anteriormente, no mercado de alimentos. Dessa forma, este projeto tem como objetivo criar uma solução que possa agilizar a contagem de valores em moedas a partir da aquisição de uma imagem fotográfica.

Primeiramente, serão apresentados os conceitos necessários para o bom entendimento do método proposto. Dessa forma, os seguintes temas serão tratados: processamento de imagens, transformada de Hough, classificação de padrões e redes neurais artificiais.

No segundo capítulo, são apresentados os conceitos utilizados no processamento das imagens e um método descritor de textura, este sendo a matriz de coocorrência de níveis de cinza, sendo mais conhecido como GLCM.

No terceiro capítulo é feita uma explicação sobre o método matemático da transformada de Hough e a sua especialização para a detecção de objetos circulares.

O quarto capítulo mostra como são definidas as características a serem extraídas de um determinado conjunto de informações de forma a se obter um melhor resultado em classificações.

No quinto capítulo são apresentados os conceitos básicos de redes neurais artificiais e o seu funcionamento, bem como a sua importância para o desenvolvimento deste trabalho.

A solução é apresentada no sexto capítulo, mostrando como todas essas informações se integram, a justificativa do uso de uma técnica robusta como redes neurais artificiais e a importância da transformada de Hough circular para a identificação das moedas na imagem, sendo .

O sétimo capítulo apresenta os resultados obtidos, os experimentos realizados e o equipamento que foi construído para a realização de todo o projeto. É mostrado como foi a aquisição de toda a base de dados, a arquitetura da rede neural e o seu treinamento, bem como as métricas utilizadas.

Finalmente, no oitavo capítulo, é realizado o fechamento do presente trabalho e também são discutidas possibilidades futuras para o mesmo, como a implementação para dispositivos móveis e a adequação para ambientes não controlados.

Capítulo 2

Processamento de Imagens

Este capítulo irá apresentar alguns conceitos e técnicas de processamento de imagens para que haja um melhor entendimento da solução que será proposta posteriormente. Todas as técnicas serão exemplificadas para melhor compreensão.

2.1 Imagem em Nível de Cinza

Na fotografia e na computação, uma imagem em nível de cinza é uma imagem onde o valor de cada pixel carrega apenas uma informação, que é a luminosidade. O nível mais escuro é o preto que indica a ausência de luz transmitida ou refletida. O nível mais claro é o branco que indica a transmissão ou reflexão total da luz.

A quantidade de níveis de cinza que um pixel pode representar varia de acordo com a quantidade de bits que será utilizada em cada um deles. Normalmente utiliza-se 8 bits para armazenar tal informação, que possibilita 256 variações de intensidade.

Serão utilizadas imagens em nível de cinza para a transformada de Hough e o descritor de textura GLCM, que serão apresentadas no Capítulo 3 e na seção 2.4 deste capítulo respectivamente.

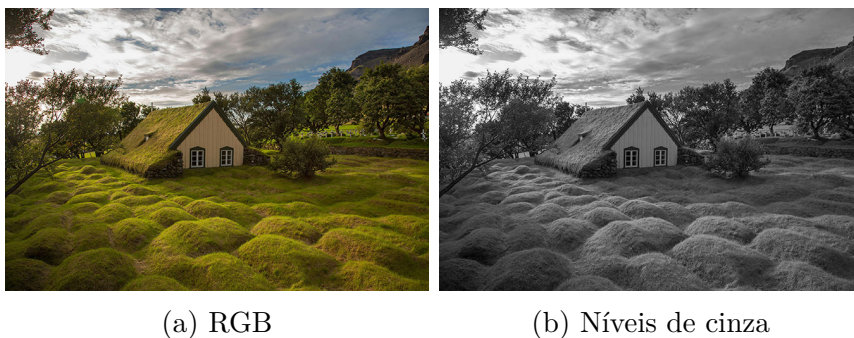


Figura 2.1: Exemplo de conversão de (a) imagem original em RGB para (b) imagem em níveis de cinza

2.2 Filtro Gaussiano

Na eletrônica e no processamento de sinais, o filtro Gaussiano passa-baixas é um operador 2D (varredura de imagem por uma máscara) usado para a suavização de imagens (mais conhecido como *blur*) e remoção de ruídos.

Em um conjunto de dados, como os pixels de uma imagem, pode-se ter valores que se repetem. A quantidade com que um valor se repete pode ser chamado de frequência. Esse conjunto de valores pode ter frequências distribuídas de acordo com uma curva conhecida como curva de Gauss.

Esta curva pode ser calculada de acordo com a equação abaixo:

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \quad (2.1)$$

onde σ é o desvio padrão.

A equação gaussiana apresenta, tipicamente, a seguinte forma:

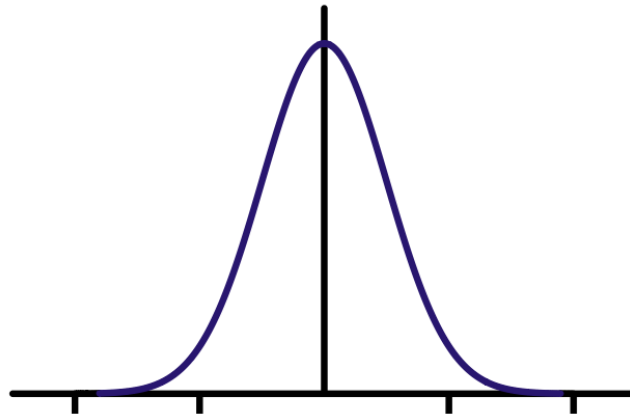


Figura 2.2: Gráfico exemplo de um função gaussiana.

É assumido que a distribuição tem média 0 (está centrada em $x=0$). O ponto mais alto da curva indica essa média. A distribuição dos outros valores varia de acordo com o desvio padrão calculado.

Em processamento de imagens, que é o foco deste capítulo, o filtro gaussiano é utilizado como uma máscara que percorre toda a imagem. Isso será exemplificado a seguir.

Vamos supor que tenhamos a seguinte máscara:

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

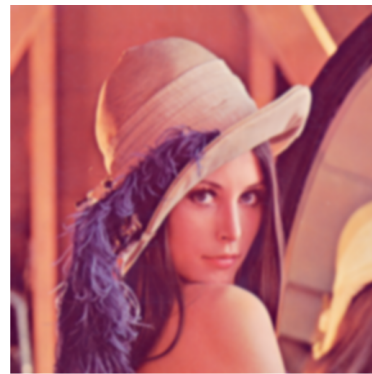
Figura 2.3: Exemplo de máscara gaussiana.

Ao se alinhar esta máscara com um pixel da imagem a ser filtrada, podemos calcular um novo valor de pixel como sendo a soma dos valores dos 25 pixels sob a máscara, ponderados pelos valores da mesma. Dessa forma, toda a vizinhança contribui para o novo valor do pixel.

O efeito da filtragem depende do desvio padrão escolhido. Veja na Figura 2.4 a aplicação do filtro com diferentes valores de σ .



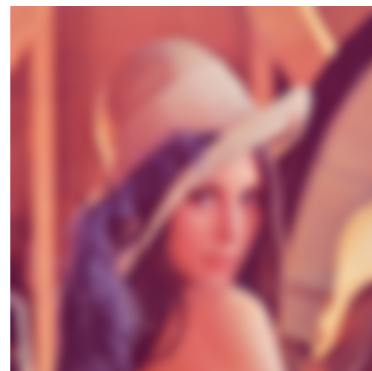
(a) Original



(b) $\sigma=2$



(c) $\sigma=4$



(d) $\sigma=8$

Figura 2.4: Aplicação do filtro gaussiano.

2.3 Operações Lógicas

O operador lógico é uma classe de operação sobre variáveis. *NOT*, *AND*, *OR*, *NAND* e *XOR* são os principais operadores lógicos que são utilizados como base para implementação de sistemas digitais, lógica proposicional e linguagem de programação. Os operadores *AND*, *OR*, *NAND* e *XOR* são binários, ou seja, precisam de dois elementos para realizar a operação, já o operador *NOT* precisa somente de um.

O funcionamento dos operadores lógicos citados é apresentado na figura 2.5:

<table><tr><td>A</td><td>X</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0	<table><tr><td>A</td><td>B</td><td>X</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1																									
A	X																																														
0	1																																														
1	0																																														
A	B	X																																													
0	0	0																																													
0	1	0																																													
1	0	0																																													
1	1	1																																													
(a) not	(b) and																																														
<table><tr><td>A</td><td>B</td><td>X</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1	<table><tr><td>A</td><td>B</td><td>X</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	1	0	1	1	1	0	1	1	1	0	<table><tr><td>A</td><td>B</td><td>X</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	0
A	B	X																																													
0	0	0																																													
0	1	1																																													
1	0	1																																													
1	1	1																																													
A	B	X																																													
0	0	1																																													
0	1	1																																													
1	0	1																																													
1	1	0																																													
A	B	X																																													
0	0	0																																													
0	1	1																																													
1	0	1																																													
1	1	0																																													
(c) or	(d) nand	(e) xor																																													

Figura 2.5: Principais operações lógicas

No presente projeto, será utilizada a operação *AND* feita bit a bit na imagem, que será exemplificada logo abaixo.

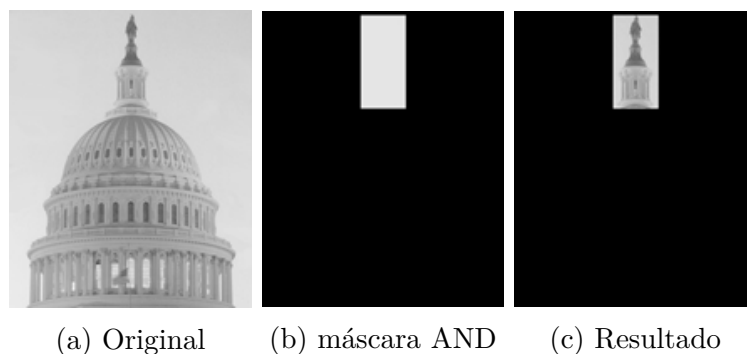


Figura 2.6: Operação lógica AND feita para extrair a região de interesse na imagem [8].

2.4 Descritor de textura

Textura é um termo bastante utilizado em aplicações com imagens. Nós reconhecemos uma textura ao vê-la, mas é difícil defini-la com precisão. Nós podemos encontrar várias definições de textura na literatura. Coggins, por exemplo, criou um catálogo com as definições de textura presentes na literatura [4]. De forma breve, podemos dizer que a textura em uma imagem é uma função da variação espacial dos valores dos pixels.

A extração de características de uma imagem que fornecem informações que possibilitam sua diferenciação entre as demais é chamada descritor de textura.

2.4.1 Matriz de Co-ocorrência de Níveis de Cinza (GLCM)

A matriz de co-ocorrência de níveis de cinza é uma estatística de segunda ordem que opera em uma função de probabilidade que por sua vez mede a probabilidade de ocorrência de um par de valores em pixel [9].

Cria-se uma matriz quadrada de tamanho M , onde M é o maior valor encontrado em um pixel na imagem, que irá guardar o grau de ocorrência de cada par de pixel dessa imagem analisada.

A matriz GLCM se comporta da seguinte maneira: se tomamos um pixel de uma determinada imagem e analisamos o pixel imediatamente a direita, o elemento da GLCM é tal que $P = [i, j]$, sendo P uma função de probabilidade que representa a quantidade de vezes que esse arranjo de dois pixels se repete na imagem, onde i é o pixel de referência e j é o pixel vizinho a sua direita [9]. Os valores de i e j são os níveis de cinza dos pixels avaliados.

Analisando a Figura 2.7, percebe-se que em $[8, 8]$ há o valor 9, que é a quantidade de vezes que um pixel de valor 8 tem um outro pixel de valor 8 a sua direita. As posições na matriz GLCM que possuem o valor 0 não influenciam no cálculo das métricas.[3]

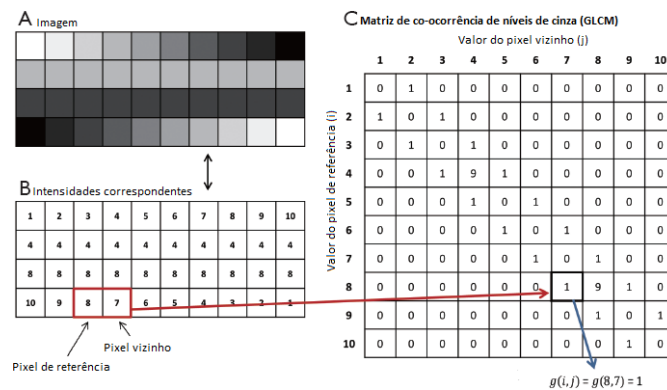


Figura 2.7: Exemplo da construção da matriz GLCM.

Uma vez que a GLCM é construída, podem-se extrair certas métricas de interesse, são elas:

- Contraste

É uma estimativa das variações locais ao quadrado dos níveis de cinza entre pares de pixels.

$$Contraste = \sum_{i,j} |i - j|^2 glcm(i, j) \quad (2.2)$$

- Correlação

É a medida da dependência linear de pixels em relação a sua vizinhança.

Fazendo uma substituição tem-se que:

$$C'(i) = glcm'(i) \quad (2.3)$$

$$C''(j) = glcm''(j) \quad (2.4)$$

Primeiro define-se a probabilidade marginal de cada pixel:

$$C'(i) = \sum_j glcm(i, j) \quad (2.5)$$

$$C''(j) = \sum_i glcm(i, j) \quad (2.6)$$

Agora calcula-se as médias das probabilidades marginais:

$$\mu_{C'} = \sum_i iC'(i) \quad (2.7)$$

$$\mu_{C''} = \sum_j jC''(j) \quad (2.8)$$

Finalmente calcula-se os desvios padrões:

$$\sigma_{C'} = \sqrt{\sum_i (i - \mu_{C'})^2 C'(i)} \quad (2.9)$$

$$\sigma_{C''} = \sqrt{\sum_j (j - \mu_{C''})^2 C''(j)} \quad (2.10)$$

$$Correlação = \sum_{i,j} \frac{(i - \mu_{C'})(j - \mu_{C''})glcm(i, j)}{\sigma_{C'}\sigma_{C''}} \quad (2.11)$$

- Energia

Mede o grau de ordem existente na imagem.

$$Energia = \sum_{i,j} glcm(i,j)^2 \quad (2.12)$$

- Homogeneidade

É uma medida que se comporta de forma inversamente proporcional ao contraste.

$$Homogeneidade = \sum_{i,j} \frac{glcm(i,j)}{1 + |i - j|} \quad (2.13)$$

Utilizando essas 4 as métricas do descritor GLCM na Figura 2.7 obtém-se os seguintes resultados:

- Contraste: 0.5000
- Correlação: 0.9547
- Energia: 0.1389
- Homogeneidade: 0.7500

2.5 Espaço de Cores (YCbCr)

O espaço de cor conhecido como YCbCr é normalmente utilizado como parte do processo de correção de imagens, como reduções de ruído, redimensionamento, correção gama, aprimoramentos e outros.

O canal Y representa a componente de luminância da imagem, mais conhecida como brilho. Os canais Cb e Cr representam as componentes de cromaticidade azul e vermelho respectivamente, ou seja, as informações de cor da imagem [7].

Esse espaço de cor é definido por uma transformação matemática de coordenadas. Essa transformação do espaço de cor RGB para YCbCr se dá pelas seguintes equações:

$$Y = 0,299R + 0,587G + 0,114B \quad (2.14)$$

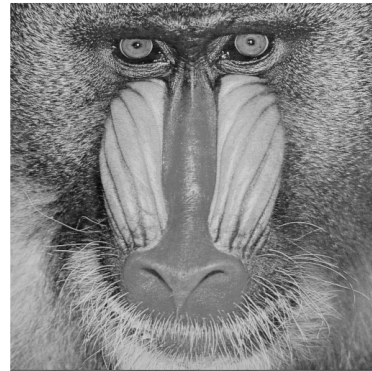
$$Cb = 0,564(B - Y) \quad (2.15)$$

$$Cr = 0,713(R - Y) \quad (2.16)$$

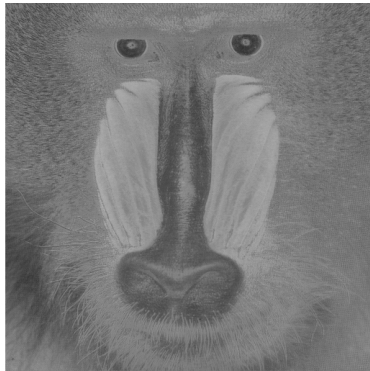
Para um melhor entendimento do resultado prático dessas transformações, é mostrado um exemplo logo abaixo:



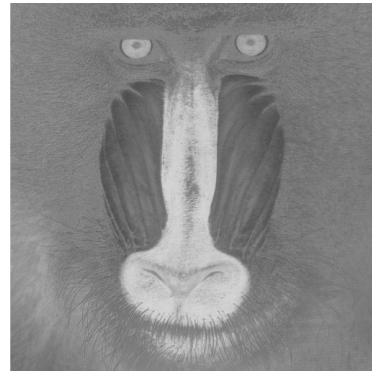
(a) Original



(b) Canal Y



(c) Canal Cb



(d) Canal Cr

Figura 2.8: Exemplo do espaço de cor YCbCr. A representação dos canais Cb e Cr é feita em níveis de cinza.

2.6 Detecção de bordas

Detectar pontos e linhas é certamente muito importante em qualquer trabalho que tenha como objetivo a segmentação. A detecção de bordas, por sua vez, é a abordagem mais comum para tal, pois ela permite encontrar descontinuidades significativas em nível de cinza.

Intuitivamente, uma borda é um conjunto de pixels conectados que se encontram na fronteira entre duas regiões. A borda é um conceito local, enquanto a fronteira é uma ideia mais global sobre o limite entre regiões. Na prática, uma definição razoável de borda é aquela que consegue medir a transição entre níveis de cinza de forma significativa.[8]

Uma borda ideal possui as propriedades da Figura 2.9a, mas isso dificilmente acontece na prática[12].

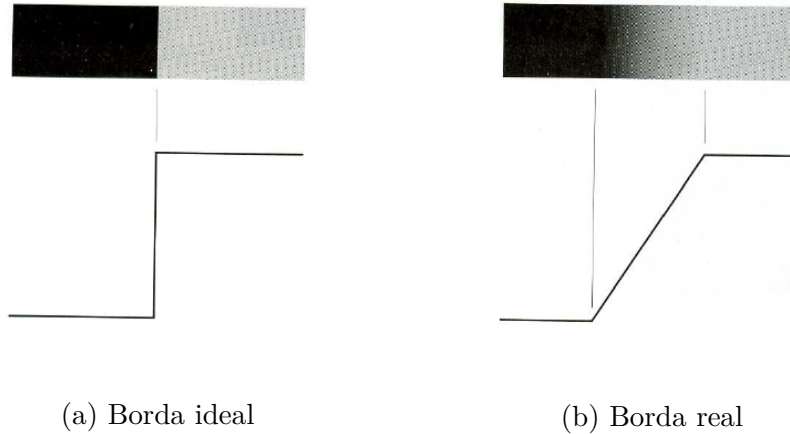


Figura 2.9: Bordas exemplificadas de forma gráfica.

2.6.1 Canny

O detector de bordas de Canny garante uma boa tolerância a ruídos e ao mesmo tempo identifica as bordas com o mínimo de erros. Canny otimizou o processo de detecção da seguinte forma[7]:

1. Maximizando a relação sinal-ruído do gradiente.
2. Um fator de localização da borda, o que procura garantir que a borda será detectada com uma boa precisão.
3. Minimizando múltiplas respostas para uma mesma borda.

A taxa do sinal-ruído do gradiente é maximizada quando bordas verdadeiras são encontradas e bordas falsas são evitadas. Dessa forma, descartando as respostas falsas quando há múltiplas respostas para uma mesma borda, o ruído é removido. Nesse método, a imagem é primeiro convoluída com um filtro de suavização gaussiano com desvio padrão σ . Esta operação é seguida da computação do gradiente na imagem suavizada resultante.

Algumas técnicas são essenciais para o bom funcionamento do algoritmo de Canny:

Supressão não máxima : o detector de bordas de Canny produz bordas com a espessura maior do que um pixel. Essa operação de supressão diminui as grandes magnitudes do gradiente de bordas.

Limiar duplo : A imagem gradiente obtida após o uso da supressão não máxima ainda pode conter muitas bordas falsas. Para removê-las, um limiar apropriado é escolhido onde todos os pontos que possuem uma magnitude superior ao limiar escolhido serão preservados como bordas verdadeiras. Há um problema nessa primeira abordagem, pois se o limiar for muito pequeno, bordas falsas serão classificadas como bordas verdadeiras. Esse problema é resolvido com o uso de dois limiares. Os limiares são definidos de forma empírica de acordo com o problema em questão.

Seleção do limiar da borda : A detecção das bordas é baseada na comparação das mesmas com um limiar. Esse limiar irá variar de acordo com o ruído presente na imagem em questão.

O resultado de uma imagem gerada a partir do detector de Canny é mostrado abaixo:



(a) Original



(b) Canny

Figura 2.10: Exemplo da aplicação da detecção de bordas pelo método de Canny.

Capítulo 3

Transformada de Hough

A transformada de Hough é uma técnica para extração de características usada em análise de imagens, visão computacional e processamento de imagens. O propósito da técnica é encontrar características imperfeitas de objetos dentro de uma determinada classe de formas por meio de um processo de votação, ou seja, encontrar a localização de linhas em imagens. Esse processo envolve um espaço de parâmetros (conjunto de todas as possíveis combinações de valores para todos os diferentes parâmetros) onde cada objeto candidato é obtido como um máximo local em um espaço acumulador construído pelo algoritmo em si. É possível conseguir bons resultados mesmo que haja ruídos e oclusões na imagem analisada [15].

Em sua concepção, a transformada de Hough era utilizada para a identificação somente de retas[11]. Mais tarde ela foi estendida para a identificação de linhas em imagens e outras formas como o círculo e a elipse[6].

Partindo do pressuposto de que se tem uma imagem que passou por um processo de detecção de borda como discutido na seção 2.6, a transformada de Hough pode ser aplicada.

Assim, tem-se N pixels que podem parcialmente descrever os contornos de alguns objetos. É preciso encontrar conjuntos de pixels que formam uma linha. Considere um ponto (x_i, y_i) e uma linha reta definida pela seguinte equação:

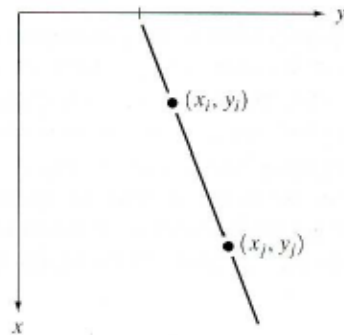
$$y_i = ax_i + b \quad (3.1)$$

Existem infinitas linhas que passam pelo ponto (x_i, y_i) , comum a elas e que satisfazem a equação para um conjunto de parâmetros (a, b) .

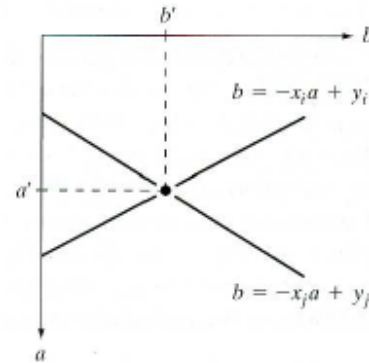
Essa equação pode ser reescrita como:

$$b = -x_i a + y_i \quad (3.2)$$

Agora considera-se x e y como parâmetros e a e b como variáveis. Essa é uma linha no plano (a,b) parametrizado por x e y . Então, um único ponto no plano xy gera uma linha no espaço ab .



(a) Plano xy



(b) Plano ab

Figura 3.1: Conceitos de parametrização. Apresentação do plano xy e da mudança de parâmetros para o plano ab

Dois pontos (x,y) e (z,k) definem uma linha no plano xy . Esses dois pontos dão origem a duas linhas diferentes no plano ab . Nesse plano ab , essas linhas irão se interceptar em um ponto (a',b') onde a' é a inclinação e b' a intersecção da linha definida por (x,y) e (z,k) no plano xy .

Todos os pontos na linha definida por (x,y) e (z,k) no plano xy irão parametrizar linhas que se interceptam em (a',b') no plano ab . Pontos que coincidem com essa linha irão formar um conjunto de intersecção no plano ab .

O algoritmo funciona da seguinte forma:

1. Quantização do plano ab , ou seja, divisão em células. Essa quantização do plano é comumente referida como células de acumulação. A Figura 3.2 deixará esse ponto mais claro.
2. Contagem do número de vezes que uma linha intercepta uma determinada célula.
 - Para cada ponto (x,y) com o valor 1 em uma imagem binária, achar os valores de (a,b) no intervalo $[[a_{min}, a_{max}], [b_{min}, b_{max}]]$ definindo a linha correspondente a esse ponto.
 - Aumentar o valor do acumulador para esse ponto (a',b') .
 - Proceder para o próximo ponto na imagem.

3. Células que recebem a menor quantidade de votos são assumidas como linhas correspondentes no plano xy .

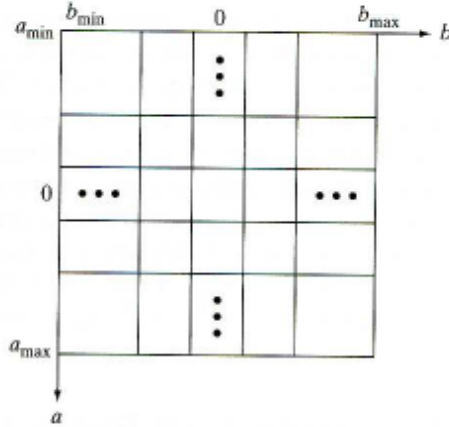
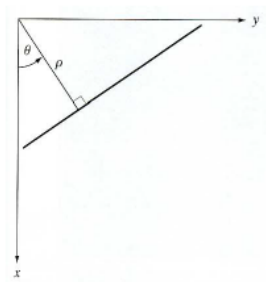


Figura 3.2: Representação das células de acumulação do plano ab .

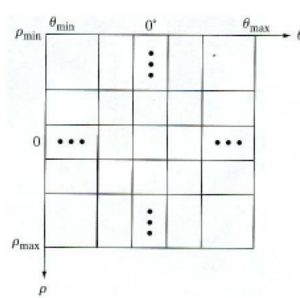
Normalmente não se usa a equação $y = ax + b$ para representar linhas e sim a forma polar de representação:

$$x \cos(\theta) + y \sin(\theta) = \rho \quad (3.3)$$

Cada ponto (x_i, y_i) no plano xy gera uma senoide no plano $\rho\theta$. M pontos colineares que coincidem com a linha $x_i \cos \theta + y_i \sin \theta = \rho$ irá gerar M curvas que se interceptam em (ρ_i, θ_j) no plano $\rho\theta$.



(a) Plano $\rho\theta$



(b) Subdivisao plano $\rho\theta$

Figura 3.3: Representação polar de linha e células acumuladoras

O algoritmo de Hough para a forma polar é semelhante ao descrito anteriormente com apenas algumas adaptações.

3.1 Transformada de Hough Circular

A transformada de Hough circular é uma especialização da transformada proposta por Hough.

Em um espaço de duas dimensões um círculo pode ser descrito pela seguinte equação:

$$(x - a)^2 + (y - b)^2 = r^2 \quad (3.4)$$

Onde (a,b) é o centro do círculo e r é o raio.

A transformada pode ser calculada de forma semelhante a apresentada anteriormente. A diferença básica é a presença de três parâmetros $(a, b \text{ e } r)$, o que resulta em um plano tridimensional. Dessa forma, as células acumuladoras serão da forma $A(i,j,k)$. O procedimento é fixar um valor para o terceiro parâmetro (r) , resolver para as variações de (a,b) que satisfaçam a equação 3.4 e atualizar a célula acumuladora correspondente com a tripla (a,b,r) .

A complexidade e o custo computacional serão proporcionais a quantidade de coordenadas e coeficientes da função que descreve o objeto a ser analisado.

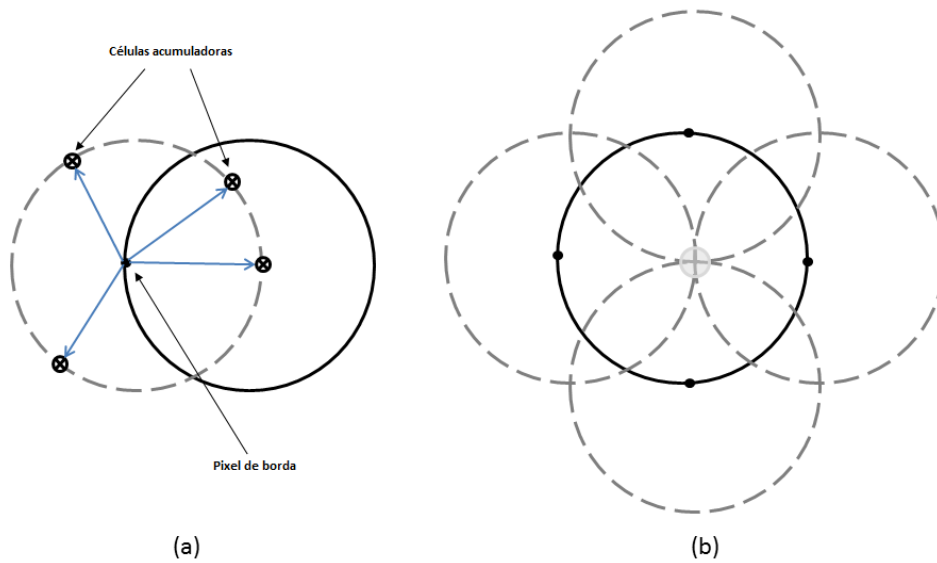


Figura 3.4: Padrão de votação da transformada de Hough circular.

Capítulo 4

Classificação de Padrões

Uma vez que se tem uma imagem segmentada, ou seja, foram identificadas e extraídas as regiões de interesse na imagem, o próximo passo é reconhecer os objetos e regiões segmentadas. Por isso, o objetivo do reconhecimento de padrões é reconhecer objetos de uma imagem a partir de um conjunto de medidas desses objetos. Cada objeto é uma padrão e os valores medidos são características desse padrão. Um conjunto de objetos que possuem mais ou menos características idênticas são ditos pertencentes a uma mesma classe[7].

Existem vários tipos de características e cada uma delas tem uma técnica específica para ser medida. Além disso, características de maior ordem de grandeza são formadas pela combinação ou distribuição de conjuntos mais simples de características. Por exemplo, cada letra no alfabeto norte-americano é composto por um conjunto de características como linhas horizontais, verticais e inclinadas bem como alguns segmentos de curva. Enquanto a letra *A* é descrita por duas linhas inclinadas e uma linha horizontal, a letra *B* possui uma linha vertical com dois segmentos de curva, unidos por uma estrutura específica. Algumas características de um objeto de duas ou três dimensões são a área, volume, perímetro e superfície que podem ser medidos pela contagem de pixels. Alguns dos atributos para caracterizar padrões de formas de objetos são o descritor de Fourier [8], momento invariante, eixo e outros.

A cor de um objeto é uma característica extremamente importante, que pode ser descrita em vários espaços de cor. Além disso, vários tipos de atributos de textura caracterizam a superfície de um objeto. As técnicas para medir as características são conhecidas como técnicas de extração de características. Padrões podem ser descritos por um conjunto de características, todos os quais podem não ter capacidade discriminatória suficiente para diferenciar uma classe de padrões de outra [7]. A seleção e extração das características apropriadas são o maior problema para a classificação de padrões.

4.1 Decisão teórica

Um grande número de técnicas de classificação já foi utilizada para reconhecer padrões. Algumas dessas técnicas são conhecidas como técnicas de decisão teórica, onde a classificação de um padrão desconhecido é feita com base em algo determinístico ou estatístico [7].

A Figura 4.1 mostra um diagrama de blocos de uma decisão teórica classificadora de padrões.

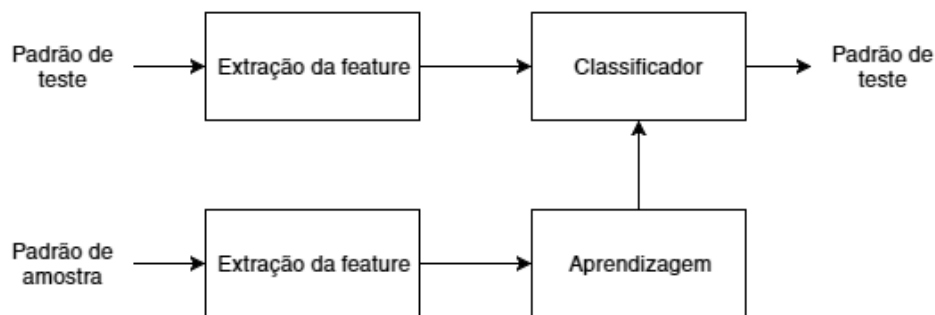


Figura 4.1: Diagrama de blocos de uma decisão teórica.

As decisões teóricas de reconhecimento de padrões são principalmente de dois tipos[14],[5]:

1. Métodos de classificação baseados em aprendizagem supervisionada.
2. Métodos de classificação que usam técnicas não supervisionadas.

Os algoritmos de classificação supervisionados podem ser classificados como:

- Classificadores paramétricos
- Classificadores não paramétricos

Na classificação supervisionada paramétrica, que é o foco deste trabalho, o classificador é treinado com um grande conjunto de amostras, onde esse conjunto de padrões pertencentes a cada classe já é conhecido, com o intuito de estimar estatisticamente parâmetros de cada classe de padrões como a média, variância e outros.

Por outro lado, os parâmetros não são levados em consideração nas técnicas de classificação supervisionadas não paramétricas.

No caso não supervisionado, a máquina particiona todo o conjunto de dados baseada em algum critério de similaridade. Isso resulta em um conjunto de grupos, onde cada grupo de padrões pertence a uma classe específica[7].

Capítulo 5

Redes Neurais Artificiais

Redes Neurais Artificiais são compostas por simples elementos operando em paralelo. Esses elementos procuram imitar a estrutura básica do sistema nervoso biológico, o neurônio. Como na natureza, a funcionalidade da rede é determinada pelas conexões entre estas pequenas centrais de processamento (modelo conexionista). Pode-se treinar uma rede neural para realizar uma função particular, ajustando os valores das conexões entre os neurônios, ou seja, ajustando as eficiências sinápticas. Dessa maneira, o conhecimento fica armazenado nessas conexões e simplesmente emerge como consequência da ação em paralelo dessa coletividade[10].

As redes neurais vem sendo treinadas para realizar funções complexas em vários campos do conhecimento humano, incluindo reconhecimento de padrões, identificação, classificação, processamento de voz e imagem, sistemas de controle e etc [10].

Hoje em dia, as redes neurais são utilizadas para solucionar problemas difíceis de serem resolvidos pela computação convencional, mas que são resolvidos extraordinariamente bem pelos seres humanos. Problemas de classificação e otimização têm sido um desafio para a computação convencional, enquanto tais funções são realizadas com sucesso pelo cérebro humano.

O estudo de redes neurais tem pouco mais de cinco décadas de história, mas encontrou um campo de aplicações sólidas apenas nos últimos quinze ou vinte anos e continua se desenvolvendo muito rapidamente.

Existe uma variedade muito grande de arquiteturas e métodos de treinamento que abrem ao projetista um vasto leque de possibilidades[16].

5.1 Neurônio Biológico

O neurônio é a célula que sofreu o mais alto grau de especialização. Devido a este fato, é uma célula permanente com precária capacidade de regeneração [13].

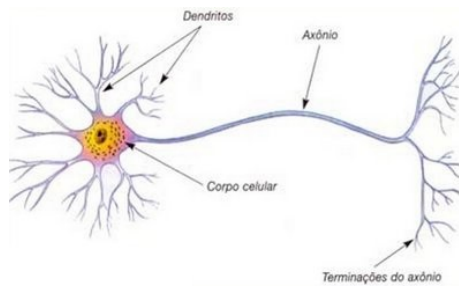


Figura 5.1: Esquema de um neurônio.

Seus principais componentes são:

- **Corpo celular ou soma:** é a parte do neurônio de onde partem todas as outras estruturas. Responsável pela maior parte da nutrição da célula.
- **Dendritos:** são prolongamentos que partem do corpo celular, constituindo as partes receptoras principais do neurônio. A maior parte dos sinais transmitidos a um neurônio chega a ele através dos dendritos, embora alguns sinais cheguem diretamente pelo corpo celular.
- **Axônio:** é o prolongamento único destinado à condução dos impulsos nervosos para células nervosas vizinhas, músculos e glândulas. Seu comprimento varia de milímetros, como ocorre no caso dos neurônios pequenos do cérebro, até aproximadamente um metro, como é o caso dos axônios que levam impulsos da medula até os músculos dos pés.
- **Terminais axônicos:** ramificações que surgem próximo à extremidade dos axônios. Tais ramificações permitem a ligação de um neurônio a diversos outros neurônios dispostos em diferentes regiões do corpo humano. Nas extremidades destas ramificações aparecem estruturas denominadas botões sinápticos, responsáveis por transmitir, através de um processo químico, o impulso elétrico de um neurônio a outro.

5.2 Neurônio artificial

A parte fundamental de uma rede neural é o neurônio com apenas uma entrada.[2] Há três operações distintas que ocorrem nesse tipo de neurônio. Em primeiro lugar, a entrada escalar x é multiplicada pelo peso escalar w para formar o produto $w.x$. Depois, a entrada ponderada $w.x$ é somada ao bias b para formar a entrada da rede n . Finalmente, a entrada da rede passa por uma função de ativação f que gera uma saída escalar v . Essas operações dão origem a seguinte equação:

$$v = f(wx + b) \quad (5.1)$$

O nome dado para esses três processos são função de peso, função de entrada da rede e função de ativação. Isso é exemplificado na figura abaixo:

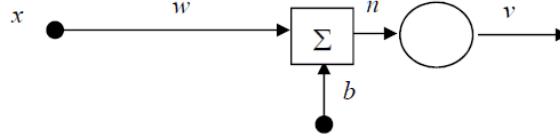


Figura 5.2: Neurônio simples.

O neurônio simples pode ser estendido para manipular entradas vetoriais.

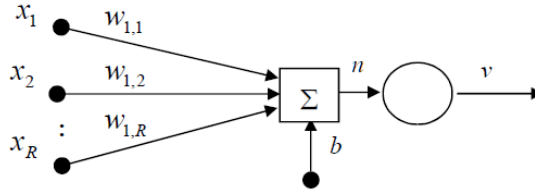


Figura 5.3: Neurônio com um vetor de entrada.

O neurônio possui um peso b , que é somado com as entradas ponderadas para formar a rede de saída n . A entrada da rede n é o argumento para a função de ativação f .

$$n = w_{1,1}x_1 + w_{1,2}x_2 + \dots + w_{1,R}x_R + b \quad (5.2)$$

Em aprendizagem de máquina, o perceptron é um modelo de neurônio para aprendizagem supervisionada de classificadores binários, funções que podem decidir se uma entrada pertence a uma classe ou outra.

O perceptron é definido como uma função que mapeia uma entrada x para um valor de saída $f(x)$:

$$f(x) = \begin{cases} 1, & \text{se } w * x + b > 0 \\ 0, & \text{caso contrário} \end{cases} \quad (5.3)$$

onde w é um vetor de pesos, $w * x$ é o produto escalar $\sum_{i=0}^m w_i x_i$, m é a quantidade de entradas para o perceptron e b é o peso.

O valor de $f(x)$ é usado para classificar x como um valor positivo ou negativo, no caso de um problema de classificar binário.

O perceptron é um neurônio artificial que utiliza a função degrau como uma função de ativação.

5.3 Funções de ativação

As funções de ativação são responsáveis por definir como o neurônio irá tratar as entradas. As duas funções mais comuns são mostradas logo abaixo:

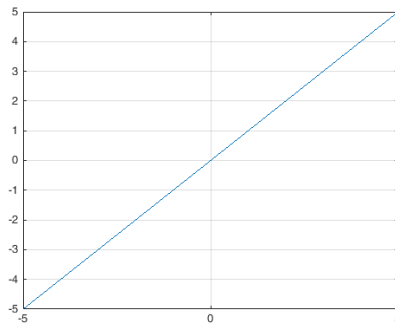


Figura 5.4: Função de ativação linear.

Esta função de ativação simplesmente faz com que a saída seja igual ao argumento da entrada.

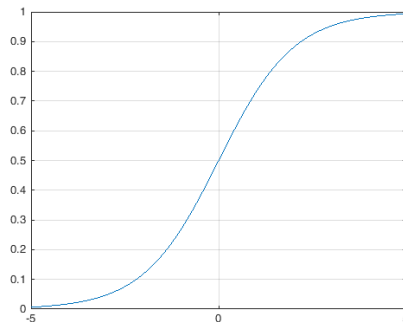


Figura 5.5: Função de ativação log-sigmoide.

A função de ativação sigmoide mostrada acima recebe os valores de entrada, que podem ter qualquer valor entre mais e menos infinito, e normaliza a saída dentro do intervalo de 0 a 1. Essa função é comumente utilizada nas camadas escondidas de redes *backpropagation* pelo fato de ser diferenciável.

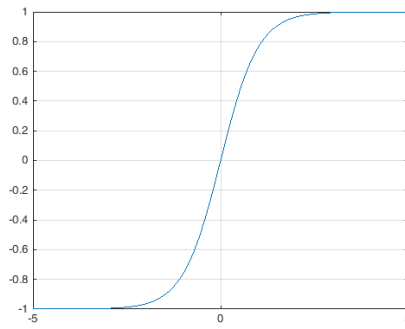


Figura 5.6: Função de ativação tangente hiperbólica.

A sigmoide acima recebe entradas que podem variar de menos infinitos até mais infinito, e limita a saída entre -1 e 1. Também é muito utilizada em redes *backpropagation*.

5.4 *Feed-forward*

Uma rede neural *feedforward* é uma rede neural artificial onde as conexões entre os neurônios não formam um ciclo, ou seja, cada camada conecta a próxima camada e não há caminho de volta. Ela foi a primeira e mais simples neural inventada. Nessa rede, a informação se move em apenas uma direção, para frente, dos nós de entrada, pelos nós das camadas escondidas até os nós de saída.

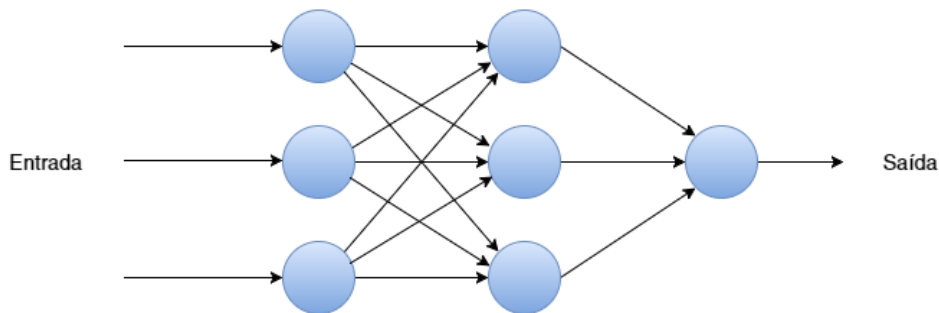


Figura 5.7: Rede neural *feedforward*.

A rede mostrada na Figura 5.7 é uma rede de 3 camadas, onde a primeira camada é chamada camada de entrada, a última de camada de saída, e a intermediária de camada escondida. É importante ressaltar que qualquer camada que não seja a de entrada ou a de saída é chamada camada escondida, independente da quantidade de camadas intermediárias[1].

Para o escopo deste trabalho, apenas a rede neural artificial multicamadas será apresentada, que é uma rede que possui pelo menos uma camada intermediária.

Capítulo 6

Solução Proposta

Uma vez que se tem todo o conhecimento do conteúdo teórico necessário para o bom entendimento da solução proposta, a mesma será apresentada a seguir.

A solução desenvolvida consiste em um método que possibilita a contagem de valor em moedas a partir da aquisição de uma imagem das mesmas. Ela é tolerante a oclusões parciais de moedas. A implementação foi feita em um ambiente controlado com uma altura fixa da câmera de captura da imagem.

Esse trabalho se preocupa em ter um alto grau de confiabilidade uma vez que fazer a contagem de valor em dinheiro não tolera falhas.

O algoritmo recebe como entrada uma imagem e fornece como saída um valor que indica um valor em dinheiro.

6.1 Calibração de iluminação

Inicialmente é feita a calibração da iluminação do ambiente no qual a imagem será capturada. Essa iluminação foi definida empiricamente de forma que as moedas pudessem ser visualizadas pela câmera e ao mesmo tempo não refletissem a luz ao ponto de perder a informação de cor.

Após a calibração manual de luz no ambiente, foram capturadas algumas imagens para que a iluminação desejada pudesse ser reproduzida. Foi utilizado o espaço de cor YCbCr onde o canal Y possui a informação de luminância.

Foi definido a média e o desvio padrão da luminância a partir do canal Y da imagem.

6.2 Aquisição da imagem

Com uma câmera na posição central do ambiente de aquisição a imagem é capturada para ser analisada posteriormente.

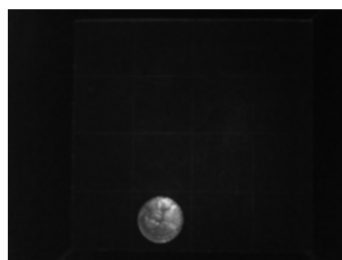


Figura 6.1: Exemplo de moeda capturada.

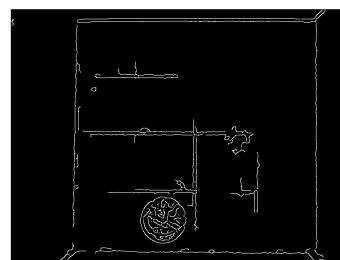
6.3 Pré-processamento

O pré-processamento consiste na manipulação da imagem de entrada para que ela esteja preparada para as etapas seguintes. As técnicas utilizadas no presente trabalho são as seguintes:

1. Filtro gaussiano: as imagens capturadas possuem ruídos que são suavizados com a utilização desse filtro.
2. Detecção de bordas (Canny): essa técnica é necessária para que a transformada de Hough seja aplicada.



(a) Filtro gaussiano



(b) Canny

Figura 6.2: Imagens após os dois primeiros passos do pré-processamento

Após esses dois primeiros passos, é aplicada a transformada de Hough circular para identificação da moeda que terá suas características extraídas posteriormente. É importante ressaltar que a transformada já fornece a localização do centro da moeda bem como

o valor do seu raio, informações essenciais para a segmentação feita a seguir. Todas essas medidas são dadas em pixels.

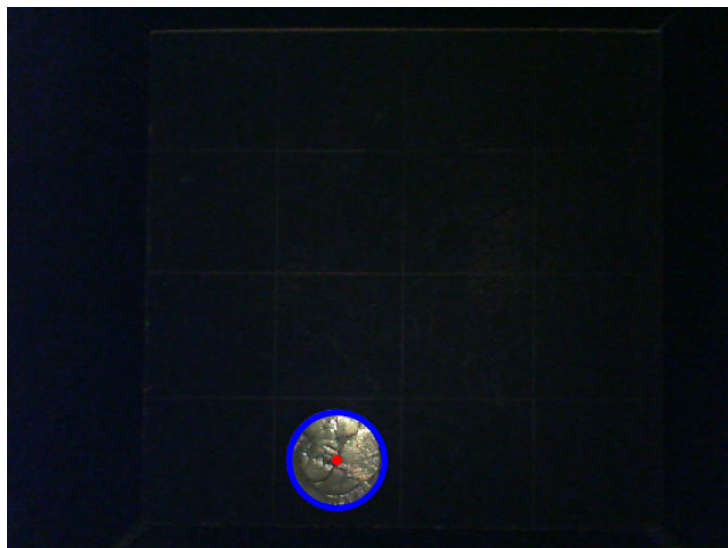
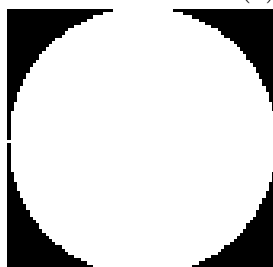


Figura 6.3: Imagem marcada após o uso da transformada de Hough circular.

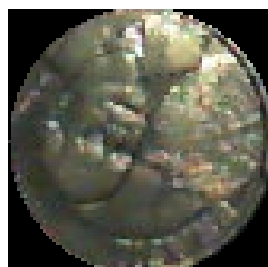
Uma vez que a moeda é localizada na imagem, é feita uma operação lógica de AND para segmentação da região a ser analisada, ou seja, a moeda em si. Para cada moeda é criada uma máscara diferente devido a variação de tamanho.



(a) Moeda extraída



(b) Máscara



(c) Moeda segmentada

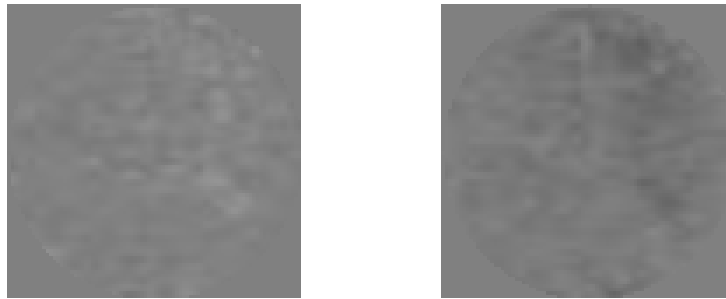
Figura 6.4: Passos da operação lógica

6.4 Extração das características

A partir desse momento, a imagem não sofre mais alterações, apenas mudança de espaço de cor. A região de interesse da imagem já foi devidamente segmentada. Nesta etapa as características da moeda serão extraídas.

O algoritmo utiliza o descritor de textura GLCM que extrai quatro informações da moeda, são elas: contraste, correlação, energia e homogeneidade.

Após o cálculo das métricas da GLCM, é feita uma mudança no espaço de cor da moeda, de RGB para YCbCr. Dessa forma, calcula-se o nível de cromaticidade azul e vermelho da imagem.



(a) Crominância azul

(b) Crominância vermelha

Figura 6.5: Moeda nos espaços de cores Cb e Cr

Tendo todas essas informações da moeda, o vetor de características que será avaliado pela rede neural possui a seguinte forma:

$$características = [r \ Cb \ Cr \ grid \ c_1 \ c_2 \ c_3 \ c_4] \quad (6.1)$$

Onde r é o raio, Cb é a cromaticidade azul, Cr é a cromaticidade vermelha, $grid$ é a posição (x,y) na base da moeda em relação a câmera, c_1 , c_2 , c_3 e c_4 são, respectivamente, contraste, correlação, energia e homogeneidade. Os valores utilizados de Cb e Cr foram a média dos valores de toda a imagem.

6.5 Classificação por meio de Redes Neurais Artificiais

Ao final de todas essas etapas, é possível classificar qual o tipo de moeda, uma vez que se é capaz de descrever a textura e algumas características da mesma. A justificativa para o

uso de algo tão robusto se dá pelo fato de que de acordo com a posição da moeda na base e conseqüentemente em relação a câmera, o círculo da moeda começa a sofrer uma certa deformação que se torna um grande problema no momento da classificação.

O vetor de características possui várias informações e a melhor forma encontrada para a classificação a partir desse vetor foi a utilização de redes neurais.

De acordo com o Capítulo 5 a rede neural do tipo *feedforward* com multicamadas consegue atender as necessidades de resolução do problema.

A rede é composta por 3 camadas. A função de ativação dos neurônios é uma tangente hiperbólica sigmoide. A camada de saída possui 5 neurônios. Os valores calculados por essa saída são divididos em 5 classes: "moeda de um real", "moeda de cinquenta centavos", "moeda de vinte e cinco centavos", "moeda de dez centavos" e "moeda de cinco centavos", onde cada neurônio identifica uma classe.

6.6 Equipamento

Para a aquisição das imagens e desenvolvimento do algoritmo exposto anteriormente, foram utilizados alguns equipamentos. Vale lembrar que a câmera foi fixada a uma altura de 15 centímetros em relação a base.



Figura 6.6: Caixa utilizada nos experimentos.

Foi construído um pequeno circuito com leds para o controle da iluminação dentro da caixa. O circuito é composto por 4 fitas de led branco, um potenciômetro de 10 quilo ohms e uma fonte de 12 volts.



Figura 6.7: Estrutura de captura das imagens.

A câmera utilizada na captura foi uma *Microsoft LifeCam Cinema*, que possui a resolução de 720p e foco automático.



Figura 6.8: Circuito de controle de iluminação e câmera.

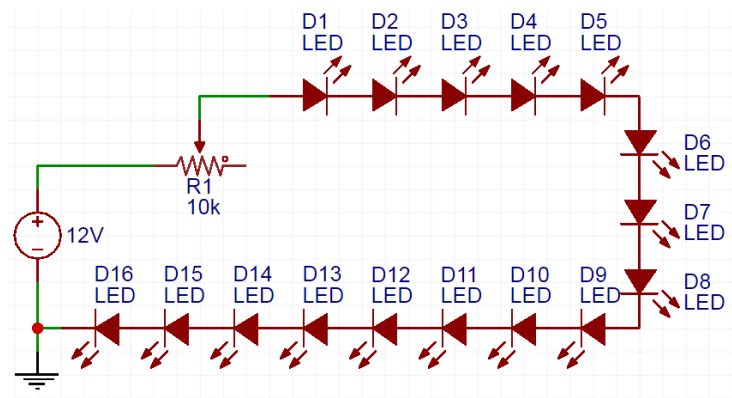


Figura 6.9: Esquema do circuito implementado para a realização do projeto.

Capítulo 7

Resultados Experimentais

Com o intuito de avaliar a precisão e robustez dos software e hardware desenvolvidos, foram realizados vários testes e experimentos que serão explicitados a seguir.

7.1 Descrição da base de dados

De forma a criar uma base de dados relativamente segura, foram utilizadas 5 moedas de cada tipo para a aquisição das imagens. São elas:



Figura 7.1: Moedas utilizadas.

Cada moeda foi submetida a 160 aquisições, sendo 80 da parte da frente e 80 da parte de trás. Foi criado um grid de 16 posições (4x4) na base, onde cada posição em que a moeda se encontrava foi capturada 5 vezes consecutivas.

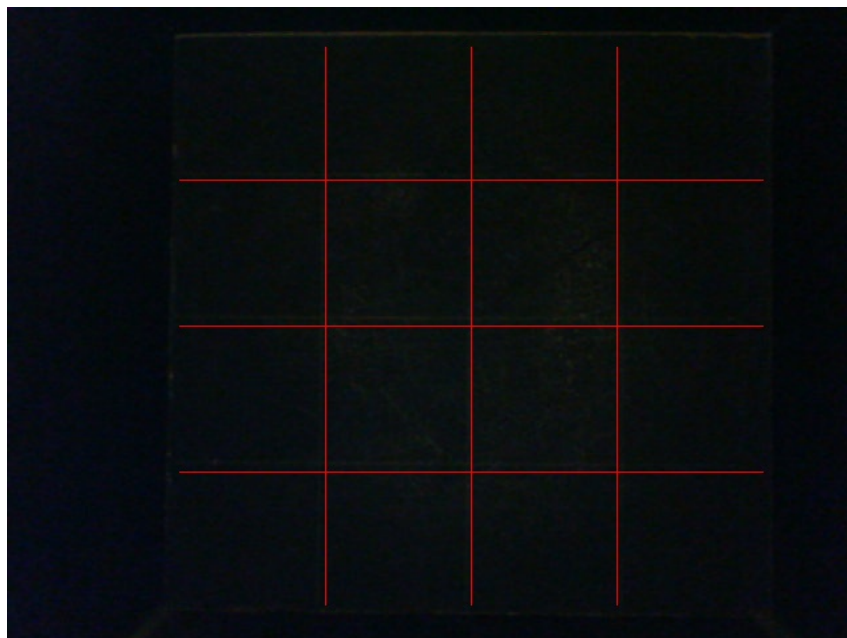


Figura 7.2: Grid de aquisição (4x4) para o treinamento e indicação da posição da moeda na base em relação a câmera no momento da classificação do tipo de moeda.

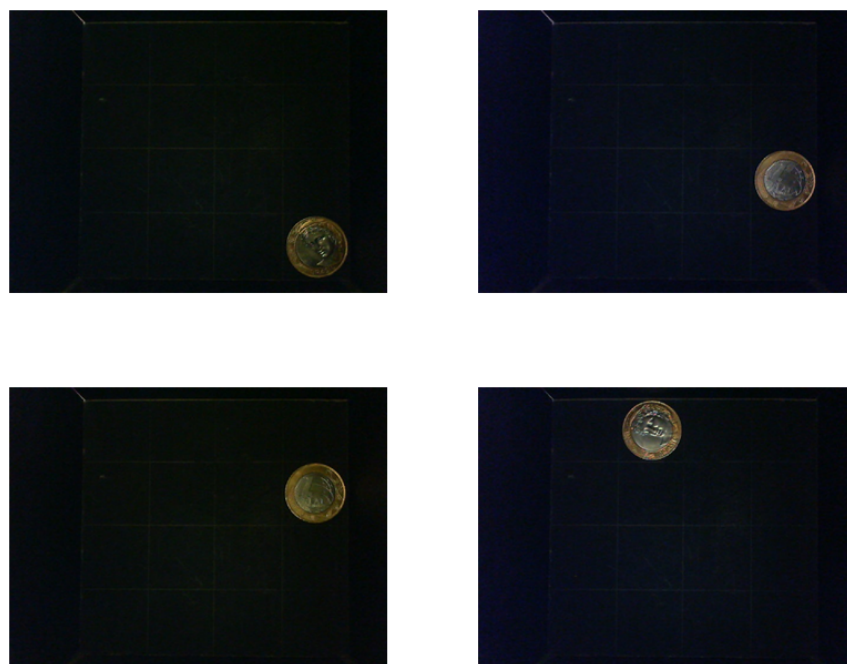


Figura 7.3: Exemplos de posições no grid utilizado para treinamento e classificação.

Todas as capturas foram feitas sem variação de luminosidade e com a altura da câmera fixa. Estas foram definidas empiricamente de forma que a câmera tivesse uma visão de toda a base onde as moedas iriam se encontrar e a iluminação fosse suficiente para que elas pudessem ser visualizadas e ao mesmo tempo não fossem ofuscadas pela luz.

Ao final das aquisições, conseguiu-se obter uma base de dados com 4000 imagens de moedas sendo 800 para cada tipo.

7.2 Organização das características

Tendo em vista que fazer a contagem de valor em moedas deve ser algo robusto, foram utilizadas 8 métricas para a classificação de cada tipo. São elas:

- Raio: distância do centro da moeda até a extremidade da borda.
- Cb: nível de azul.
- Cr: nível de vermelho.
- Grid: posição na base em relação a câmera.
- Contraste: estimativa das variações locais ao quadrado dos níveis de cinza entre pares de pixels.
- Correlação: medida da dependência linear de pixels em relação a sua vizinhança.
- Energia: mede o grau de ordem existente na imagem.
- Homogeneidade: medida que se comporta de forma inversamente proporcional ao contraste.

Para cada classe de moeda foi atribuído um código correspondente para que este fosse utilizado posteriormente no treinamento da rede neural.

Foram feitos histogramas para avaliar os valores médios e desvio padrão das características extraídas, bem como o grau de diferenciação entre cada uma delas. Seguem abaixo as imagens dos gráficos:

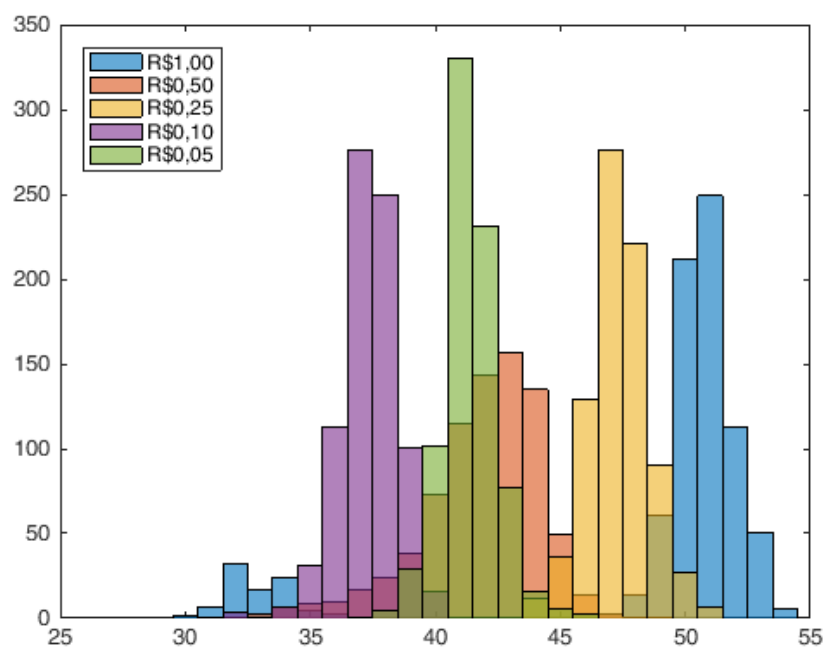


Figura 7.4: Histograma da distribuição dos raios de cada tipo de moeda.

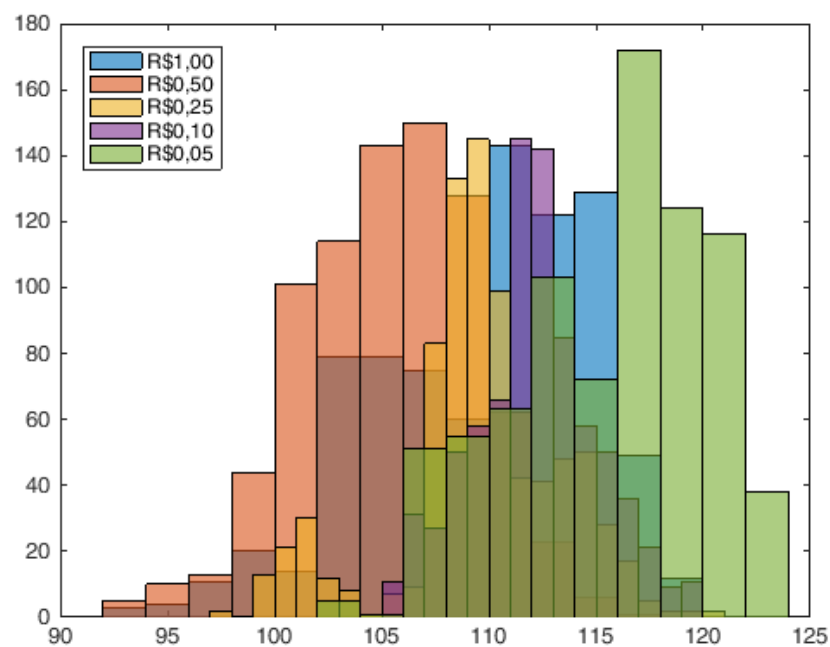


Figura 7.5: Histograma da distribuição dos níveis de crominância azul.

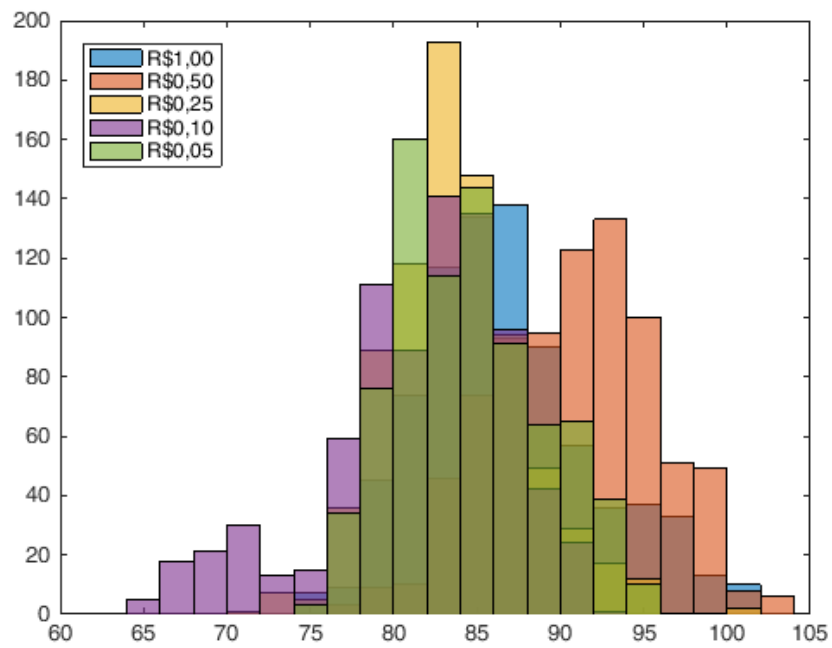


Figura 7.6: Histograma da distribuição dos níveis de crominância vermelho.

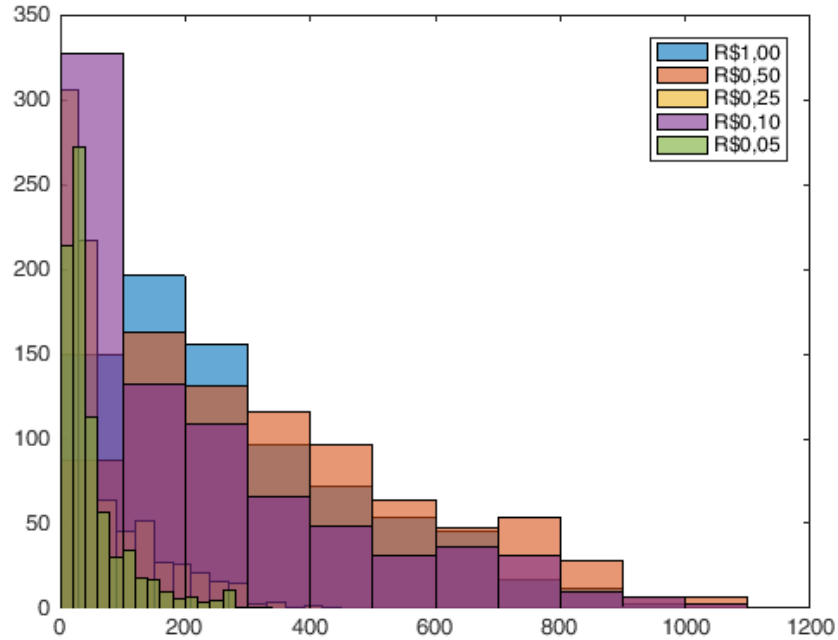


Figura 7.7: Histograma da distribuição do contraste.

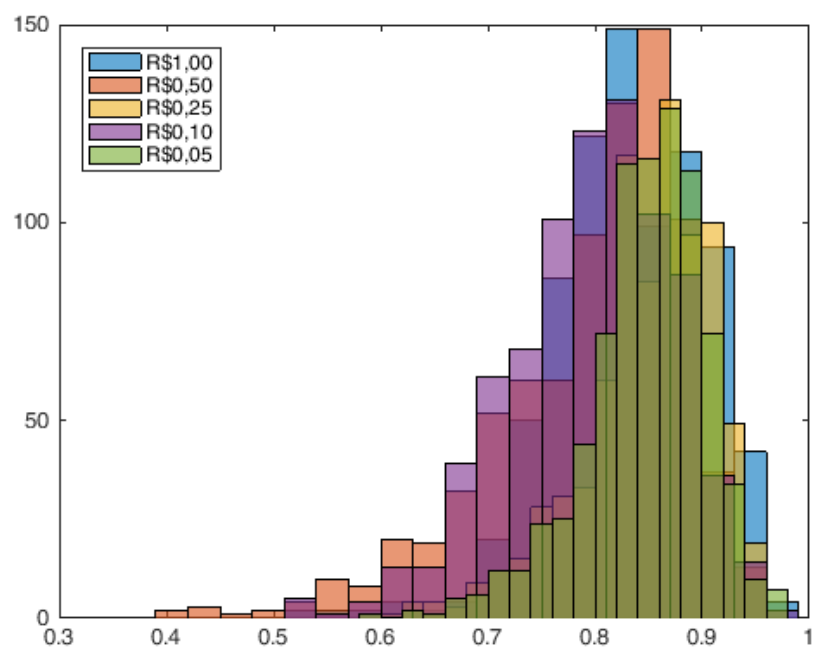


Figura 7.8: Histograma da distribuição da correlação.

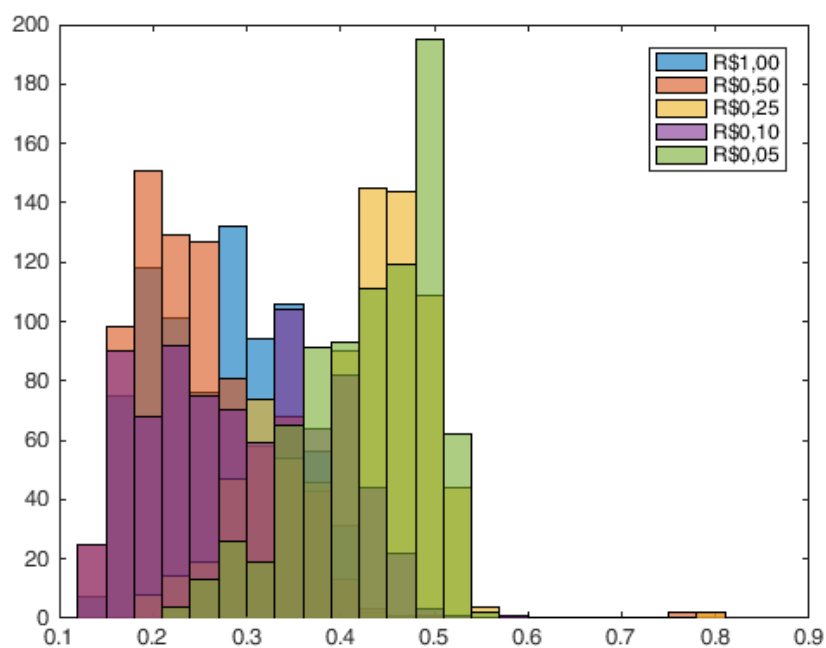


Figura 7.9: Histograma da distribuição da homogeneidade.

7.3 Treinamento

O treinamento da rede neural foi feito de forma supervisionada, o método utilizado foi o *backpropagation*, a quantidade de épocas foi de 100 e sua criação segue o algoritmo abaixo:

DEFINIÇÃO DA REDE NEURAL

```
1   $T = 0$  // Melhor desempenho
2   $i = 5$  // neurônios na segunda camada.
3  while  $i \leq 65$ 
4      CREATE( $net, i$ )
5      CONFIGURE( $net, inTrain, tgtTrain$ )
           //  $inTrain$  : Matriz de entradas
           //  $tgtTrain$  : Matriz de alvos
6       $cont = 0$ 
7       $contMax = 50$ 
8      while  $cont \leq contMax$ 
9           $cont = cont + 1$ 
10         INITIALIZE( $net$ )
11         TRAIN( $net, inTrain, tgtTrain$ )
12         SIMULATE( $net, inTest, outTest$ )
           //  $inTest$  : Matriz de teste
           //  $outTest$  : Matriz de predição
13         CONFMAT( $tgtTest, outTest, mat$ )
           //  $tgtTest$  : Conjunto de alvos teste
           //  $mat$  : Matriz de confusão
14         if  $mat(allMainDiagElements) > T$ 
15              $T = mat(allMainDiagElements)$ 
16             SAVE( $net, i$ )
17
18          $i = i + 5$ 
```

A função *create* cria a rede com a configuração de camadas definida pelos parâmetros, no caso, 3 camadas. A função *configure* configura a rede informando para ela as possíveis entradas e os alvos. A função *initialize* inicializa a rede para que ela possa ser treinada. A função *train* treina a rede com uma certa quantidade de amostras da base de dados. A função *simulate* simula a rede gerada pela função *train* e compara com os alvos. A função *confmat* gera a matriz de confusão de acordo com o erros e acertos da função *simulate*. A função do objeto *mat* verifica o grau de desempenho da rede gerada.

A organização dos dados para o treinamento da rede neural foi feito da seguinte forma:

1. Inicialmente foi feita uma normalização dos dados para que todos ficassem dentro do intervalo de 0 a 1.
2. Uma vez que cada classe possui 800 amostras, essas foram divididos em dois grupos. 400 amostras para a matriz de treinamento e as outras 400 amostras para a matriz de teste. Essas amostras não foram escolhidas de forma ordenada, foi utilizado um algoritmo de embaralhamento.

Segue abaixo a matriz de confusão gerada pela melhor rede neural:

	R\$1,00	R\$0,50	R\$0,25	R\$0,10	R\$0,05
R\$1,00	393	3	3	1	0
R\$0,50	2	397	0	1	0
R\$0,25	0	0	393	0	7
R\$0,10	2	3	0	390	5
R\$0,05	0	1	4	0	395

Figura 7.10: Matriz de confusão.

A rede neural que gerou essa matriz de confusão teve um desempenho de 98,4%. Foi composta de 3 camadas, sendo a camada de entrada com 8 neurônios, a camada intermediária com 55 neurônios e a camada de saída com 5 neurônios.

7.4 Experimentos

Após uma serie de experimentos, eles foram separados em três grupos:

1. Experimentos sem oclusão. As moedas estão espalhadas.
2. Experimentos com oclusão. Algumas moedas estão com uma parte sobreposta por outra.
3. Experimentos que apresentaram erros de classificação ou detecção.



(a) Captura

(b) Detecção

Figura 7.11: Experimento 1 sem oclusão. Valor esperado: $R\$1,90$. Valor calculado : $R\$1,90$.



(a) Captura

(b) Detecção

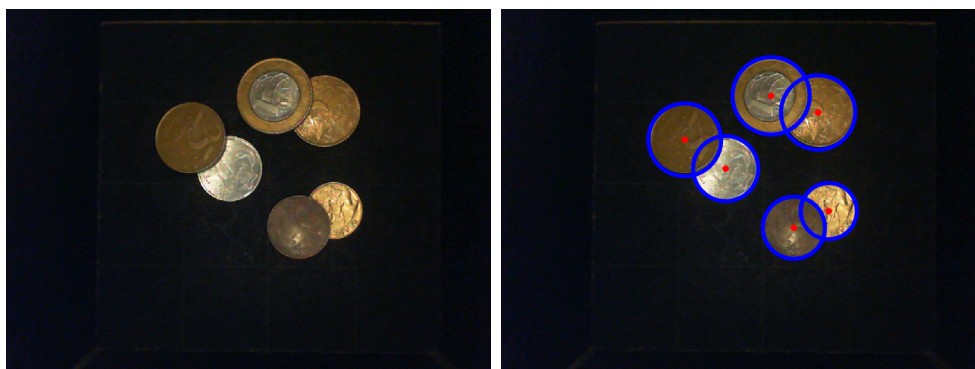
Figura 7.12: Experimento 2 sem oclusão. Valor esperado: $R\$2,75$. Valor calculado : $R\$2,75$.



(a) Captura

(b) Detecção

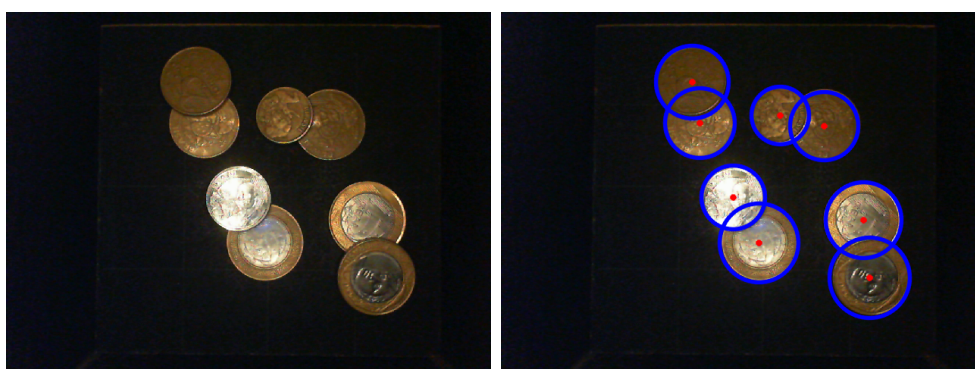
Figura 7.13: Experimento 3 sem oclusão. Valor esperado: $R\$3,25$. Valor calculado : $R\$3,25$.



(a) Captura

(b) Detecção

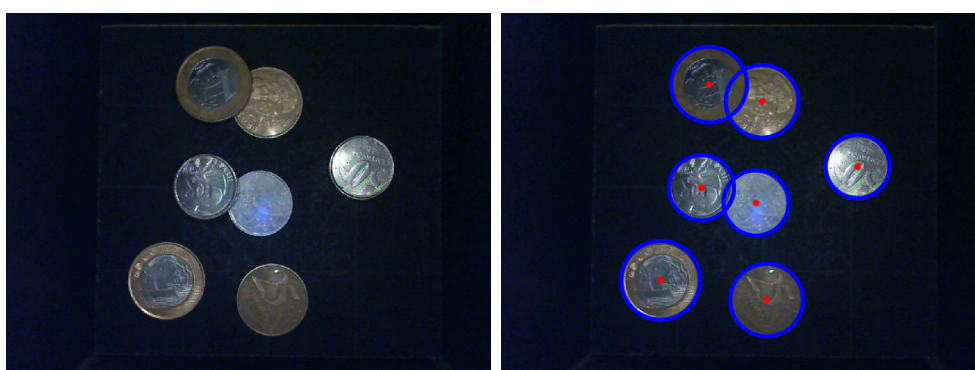
Figura 7.14: Experimento 1 com oclusão. Valor esperado: $R\$2,15$. Valor calculado : $R\$2,15$.



(a) Captura

(b) Detecção

Figura 7.15: Experimento 2 com oclusão. Valor esperado: $R\$4,35$. Valor calculado : $R\$4,35$.



(a) Captura

(b) Detecção

Figura 7.16: Experimento 3 com oclusão. Valor esperado: $R\$4,00$. Valor calculado : $R\$4,00$.



(a) Captura

(b) Detecção

Figura 7.17: Experimento 1 com erro. Valor esperado: $R\$7,55$. Valor calculado : $R\$8,60$.

A imagem acima teve um erro de classificação. Todas as moedas foram detectadas.



(a) Captura

(b) Detecção

Figura 7.18: Experimento 2 com erro. Valor esperado: $R\$2,20$. Valor calculado : $R\$4,25$

A imagem acima teve um erro de detecção o que gera um erro de classificação.



(a) Captura

(b) Detecção

Figura 7.19: Experimento 3 com erro. Valor esperado: $R\$7,00$. Valor calculado : $R\$8,25$

A imagem acima teve um erro de detecção, ou seja, a transformada de Hough não conseguiu segmentar as moedas corretamente, provavelmente ocasionado pela reflexão da iluminação.

Capítulo 8

Conclusão

A partir da análise dos resultados obtidos com o algoritmo é possível perceber que o mesmo consegue discriminar moedas com um alto grau de precisão. As técnicas utilizadas para atingir o objetivo são robustas, por isso possuem uma boa tolerância a ruídos.

A motivação inicial do trabalho era desenvolver uma aplicação capaz de fazer a contagem de moedas em qualquer ambiente, independente da iluminação, da altura da câmera em relação as moedas bem como a superfície que elas fossem capturadas. Não foi possível chegar a tal ponto devido a complexidade do problema.

Os dois principais métodos utilizados para a detecção e descrição das moedas no presente trabalho, que foram a transformada de Hough circular e o descritor de textura GLCM, tiveram um bom desempenho com relação aos acertos esperados e velocidade de processamento. Contudo, eles podem ter um alto custo computacional dependendo da quantidade de moedas a ser contada e da resolução da imagem.

O uso de redes neurais artificiais para classificação das moedas foi um fator de extrema importância para chegar aos ótimos resultados que foram apresentados. Ainda é possível melhorar a taxa de acertos por meio da inserção de outras características das moedas e também com o aumento da base de dados.

Um dos fatores limitantes encontrados foi a iluminação do ambiente utilizado para captura que poderia ser contornado por meio da utilização de uma iluminação indireta e com uma maior quantidade de amostras com variações de luz para o treinamento da rede. Um segundo fator, foi a distância entre a câmera e as moedas, pois com a variação da mesma o tamanho das moedas varia proporcionalmente, com o uso de apenas uma câmera a definição dessa distância se torna uma tarefa não trivial. Um terceiro fator limitante foi a oclusão parcial das moedas no momento da captura, apesar do algoritmo ter uma tolerância razoável a essa limitação, a porcentagem de erros ainda foi relativamente alta.

Apesar das dificuldades encontradas, o algoritmo utilizado dentro do ambiente controlado e sem oclusões de moedas teve uma taxa de acerto de 98,4%, que foi um resultado

que excedeu as expectativas.

Ainda é possível deixar o algoritmo mais robusto inserindo técnicas que possibilitem a variação da altura da câmera e da iluminação. O aumento da base de dados também teria um efeito bastante positivo.

Em trabalhos futuros serão incluídas técnicas que possibilitem a contagem de moedas independente da iluminação do ambiente, a altura da câmera de captura bem como a superfície em que as moedas se encontram.

Referências

- [1] Augusto Cesar de Magalhães Chaves Natália Figueiredo Moreira e Flávio de Barros Vidal Alexandre Zaghetto, Manoel Neto. Classificação de madeiras amazônicas utilizando o módulo de young e redes neurais artificiais. *Journal of LATEX class files*, VOL. 6, NO. 1, January, 2007. 23
- [2] Mark Hudson Beale, Martin T. Hagan, e Howard B. Demuth. *Neural Network ToolboxTM User's Guide*. The MathWorks, Inc., 2016. 20
- [3] Matjaž Bevk e Igor Kononenko. A statistical approach to texture description of medical images: A preliminary study. *Computer-Based Medical Systems*, pages 239–244, 2002. 7
- [4] J.M Coggins. A framework for texture analysis on spatial filtering. *Computer Science Department, Michigan State University, East Lansing, Michigan*, 1982. 7
- [5] Li Deng e Dong Yu. *Deep Learning Methods and Applications*. Now, The Essence of Knowledge, 2014. 18
- [6] R. O. Duda e P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Magazine Communications of the ACM*, Volume 15:11–15, 1, Jan. 1972. 13
- [7] Tinku Acharya e Ajoy K. Ray. *Image Processing: principles and applications*. John Wiley Sons, Inc., Hoboken, New Jersey, 2005. 9, 11, 17, 18
- [8] Rafael C. Gonzalez e Richard E. Woods. *Digital Image Processing*. Prentice Hall, second edition, 2002. x, 6, 10, 17
- [9] Robert M Haralick, K Shanmugam, e Its'hak Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(06):610–621, 1973. 7
- [10] Simon Haykin. *Neural Networks: a comprehensive foundation*. Pearson Education (Singapore) Pte. Ltd., Indian Branch, 482 F.I.E. Patparganj, Delhi 110 092, India, 1999. 19
- [11] P.V.C. Hough. Method and means for recognizing complex patterns. *U.S. Patent 3,069,654*, Dec. 18, 1962. 13
- [12] William K. Pratt. *Digital Image Processing*. John Wiley Sons, Inc., Hoboken, New Jersey, 2007. 11

- [13] Dale Purves. *Neurociências*. Porto Alegre: Artmed, 2010. 19
- [14] Stuart J. Russell e Peter Norvig. *Artificial Intelligence: a modern approach*. Pearson Education, Inc., Upper Saddle River; New Jersey 07458, 1995. 18
- [15] Linda Shapiro e George Stockman. *Computer Vision*. Prentice-Hall, Inc., 2001. 13
- [16] A Zaghetto. Codificação de imagens médicas usando imposição da razão sinal/ruído local mínima. *Dissertação de Mestrado, Publicação 184/04, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 111 p*, Junho, 2004. 19