Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Detecção e monitoramento de vagas disponíveis em estacionamentos abertos através de processamento de imagens

Vitor de Alencastro Lacerda

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Alexandre Zaghetto

Brasília
2015

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Coordenador: Prof. Dr. Coordenador

Banca examinadora composta por:

Prof. Dr. Alexandre Zaghetto (Orientador) — CIC/UnB
Prof. Dr. Professor I — CIC/UnB
Prof. Dr. Professor II — CIC/UnB

CIP — Catalogação Internacional na Publicação

Lacerda, Vitor de Alencastro.

Detecção e monitoramento de vagas disponíveis em estacionamentos abertos através de processamento de imagens / Vitor de Alencastro Lacerda. Brasília : UnB, 2015.
57 p. : il. ; 29,5 cm.

Monografia (Graduação) — Universidade de Brasília, Brasília, 2015.

1. Estacionamento, 2. Processamento de Imagens, 3. Vagas livres

CDU 004.4

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Detecção e monitoramento de vagas disponíveis em estacionamentos abertos através de processamento de imagens

Vitor de Alencastro Lacerda

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

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

Prof. Dr. Professor I
CIC/UnB

Prof. Dr. Professor II
CIC/UnB

Prof. Dr. Coordenador
Coordenador do Bacharelado em Ciência da Computação

Brasília, 10 de setembro de 2015

Dedicatória

Dedico a....

Agradecimentos

Agradeço a....

Abstract

A ciência...

Palavras-chave: Estacionamento, Processamento de Imagens, Vagas livres

Abstract

The science...

Keywords: Parking lot, Image processing, Free Spaces

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Atualmente, a frota de carros dos grandes centros urbanos tem fica cada vez maior. Esse aumento no número de veículos cria uma demanda por espaço de estacionamento, levando a construção de novos estacionamentos e uma dificuldade muito maior de se encontrar vagas disponíveis nos estacionamentos existentes nos estabelecimentos comerciais, áreas residenciais e centros urbanos.

De fato, todo ano, milhões de pessoas gastam milhares de horas rondando estacionamentos de supermercados, *shoppings* e prédios comerciais em busca de uma vaga de estacionamento vazia. Procurar por uma vaga em um estacionamento grande e cheio é uma tarefa cansativa e desagradável. Muitas vezes vagas desocupadas não são encontradas por motoristas que preferem parar seu carro e esperar que uma vaga seja liberada próximo a ele. Outras vezes possíveis clientes desistem de ir a um determinado estabelecimento pois sabem da dificuldade de estacionar que enfrentarão. Esse problema é comum principalmente em *shopping centers* e grandes supermercados, que possuem seus próprios estacionamentos disponíveis para os clientes, mas que estão frequentemente muito cheios.

Facilitar a tarefa da busca de vagas em estacionamentos de estabelecimentos como estes é benéfico então não só para os clientes, mas também para os donos e gerentes desses estabelecimentos. Sendo assim, esse trabalho tem como objetivo criar um sistema que seja capaz de determinar a ocupação das vagas de um estacionamento descoberto e informar quantas vagas estão disponíveis e a sua localização para os motoristas através do processamento de imagens do estacionamento. Tal solução facilitaria a procura de vagas, eliminando o tempo gasto trafegando por fileiras de vagas ocupadas.

A solução apresentada utiliza redes neurais artificais combinadas com técnicas descritoras de textura e características de crominância para diferenciar imagens de carros de imagens de vagas desocupadas. Cada região que contém vagas na imagem obtida é dividida em diversas seções verticais. Cada seção será classificada entre vaga vazia ou veículo, de forma que o programa então é capaz de determinar quantas vagas estão ocupadas nessa região e, consequentemente, quantas estão vazias. Algumas soluções propostas em trabalhos correlatos serão apresentadas no capítulo 4.

No capítulo 2 serão apresentados alguns conceitos importantes para o entendimento da solução que será apresentada posteriormente. Serão detalhados o que é uma imagem, como funciona um vídeo e outros elementos teóricos importantes como espaços de cores, classificação de padrões e redes neurais artificiais. Em seguida serão apresentadas algumas

métricas utilizadas na validação dos resultados do trabalho. Em seguida o capítulo 3 detalha conceitos relativos a redes neurais artificiais.

No capítulo 5, a solução proposta é apresentada. Nesta seção será detalhado o funcionamento do sistema criado, o fluxo de execução do programa e os elementos que permitem o funcionamento do sistema.

Em seguida, no capítulo **??** apresentará dados estatísticos sobre os resultados obtidos, acompanhados de observações. Esse capítulo também apresenta os detalhes da rede neural artificial utilizada.

O capítulo **??** possui comentários sobre a taxa de sucesso do trabalho, suas fragilidades, sua adequação para uso no estado atual e se o objetivo desejado foi alcançado. O capítulo finaliza o trabalho com uma discussão sobre possíveis trabalhos futuros para a evolução do sistema.[3]

Capítulo 2

Fundamentação Teórica

Esse capítulo tem como objetivo apresentar conceitos que facilitarão o entendimento do conteúdo do capítulo 5, detalhando técnicas e conceitos utilizados na execução do trabalho através de exemplos e imagens. Serão apresentados conceitos relacionados ao processamento de imagens estáticas, seguidos de uma discussão breve sobre técnias aplicadas em vídeos. Por fim serão detalhados as redes neurais e o conceito de classificação de padrões.

2.1 Processamento de Imagens

2.1.1 Imagens em nível de cinza

Para que um computador seja capaz de operar sobre uma imagem, é preciso que seja utilizado um modelo de representação que traduza o que os nossos sentidos conseguem perceber em informações que podem ser interpretadas por uma máquina que não é dotada de visão. A forma mais simples de se representar uma imagem são as imagens em nível de cinza. Podemos definir imagens como uma função fx,y onde x e y são coordenadas espaciais e o valor de fx,y é a luminosidade, ou nível de cinza, da imagem naquele ponto. Quando esses valores são todos discretos, chamamos essa imagem de uma imagem digital.[6] Cada elemento individual dessa imagem, cada valor em cada coordenada, pode ser chamado de um *picture element* ou mais comumente *pixel*. Em imagens digitais em nível de cinza, cada *pixel* possui 1 *bit* de informação, ou seja, pode assumir valores entre 0 e 255, onde o valor 0 representa o preto e 255 representa o branco.

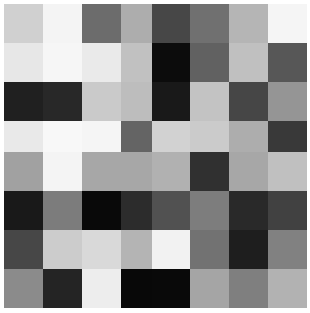
Uma imagem então, pode ser interpretada por um programa de computador como uma matriz MxN de elementos de 1 *bit*, onde M é a largura da imagem e N é a sua altura. Cada elemento $p_{i,j}$ da matriz possui a informação de luminosidade do pixel correspondente e o computador é capaz de exibir e interpretar esses valores apropriadamente. A figura 2.1 exemplifica esse processo.

2.1.2 Espaços de cores

Para que um sistema de processamento de imagens possa interpretar e processar cores, é preciso criar modelos apropriados para representá-las. Chamamos os modelos de representação das cores em uma imagem um espaços de cor. Existem diversos espaços de

208	244	108	173	71	112	181	245
231	246	234	193	12	97	192	87
32	40	202	189	25	195	70	149
233	248	245	100	210	203	173	57
161	244	167	167	177	48	167	192
25	124	9	44	81	125	41	65
71	204	217	180	242	114	30	129
139	36	238	8	9	165	127	178

(a)



(b)

Figura 2.1: (a) Uma matriz com níveis de cinza e (b) a imagem correspondente

cor, cada um importante para a realisação de tarefas e necessidades diferentes. Para a compreensão deste trabalho, é necessário, porém, entender apenas dois: o espaço RGB e o espaço YCbCr.

2.1.3 O espaço RGB

RGB é um acrônimo para *Red, Green, Blue*, vermelho, verde e azul em inglês. Pesquisas mostram que um grande gama de cores pode ser formado através de combinações aditivas das cores vermelho, verde e azul. Essas cores são consideradas então cores primárias aditivas.[9]. Nesse modelo, a cor de um *pixel* de uma imagem é representada através de um três coeficientes que definem a influência de cada cor primária na combinação. Uma imagem no modelo RGB é então representada por três matrizes de níveis de cinza de dimensões iguais, chamadas canais, onde cada valor representado na matriz, representa o valor do coeficiente da cor correspondente na imagem final. Isto é, a imagem pode ser representada por uma matriz $MxNx3$ onde a cor final $C_{i,j}$ de cada *pixel* da imagem é definida pela equação 2.1. A imagem 2.2a mostra os canais de uma imagem RGB separadamente.

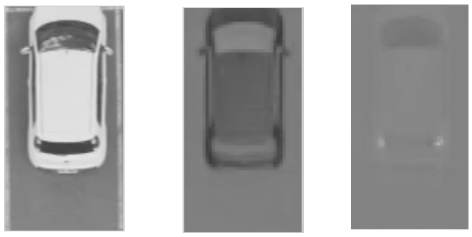
Esse espaço de cor é o mais comumente encontrado no cotidiano, uma vez que é semelhante visão humana e portanto é utilizado pelos dispositivos multimídia mais comuns.

$$C_{i,j} = p_{i,j,1}.R + p_{i,j,2}.G + p_{i,j,3}.B, (0 < i < M, 0 < j < N) \tag{2.1}$$

2.1.4 Espaço YCbCr

Assim como o espaço de cor RGB, o espaço YCbCr também representa uma imagem através de três matrizes, porém, seus canais contém informações diferentes do modelo RGB. Esse modelo, é muito utilizado para o armazenamento de vídeos, uma vez que o modelo tira vantagem de alguns aspectos da visão humana para poder armazenar menos dados, sem perda significativa de informação visual. Por exemplo, humanos são mais sensíveis à detalhes e variações em níveis de cinza do que detalhes em imagens coloridas. Além disso, o olho humano é mais sensível ao verde do que qualquer outra cor[5]. Com esse conhecimento, o espaço YCbCr representa as cores de uma imagem através de sua luminosidade(Y) e os valores de crominância azul(Cb) e crominância vermelha(Cr). Cb e Cr são sinais de diferença de cor e são definidos pela subtração do valor de luminosidade

(a)



(b)

Figura 2.2: (a) Uma imagem e cada um de seus canais RGB ordenados da esquerda para a direita e (b) os canais YCbCr da imagem ordenados da esquerda para a direita.

do canal azul e vermelho da imagem, respectivamente. A luminosidade é definida pela equação 2.2[14] que reflete a sensibilidade maior a cor verde da visão humana.

Na figura 2.2b estão exemplificados os canais de uma imagem no espaço YCbCr.

$$Y = 0,299.R + 0,587.G + 0,114.B \tag{2.2}$$

2.1.5 Descritores de textura

Descritores de textura são algoritmos que procuram fazer o que o olho humano faz com facilidade: distinguir entre tipos diferentes de objetos apenas por algumas de suas características visuais. Estes algoritmos são utilizados quando a abordagem de processamento *pixel-a-pixel* se mostra insuficiente. Eles observam e analisam características pertinentes a imagem inteira e são capazes de identificar diferenças mais sutis entre imagens diferentes. Existem diversas técnicas de descrição de textura que são utilizadas com sucesso em aplicações de processamento de imagem. Para esse trabalho, a técnica escolhida foi a conhecida como GLCM(*gray-level co-ocurrence matrix*) descrita na seção 2.1.6, por ter execução rápida e ser invariante quanto a escala de cinza.

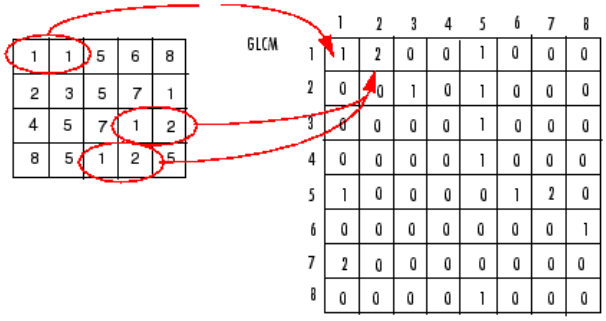


Figura 2.3: Exemplo da elaboração da GLCM. Extraída de https://www.mathworks.com/help/images/ref/graycomatrix.html

2.1.6 GLCM

GLCM, ou matriz de co-ocorrência de nível de cinza, é uma técnica de descrição de textura que visa extrair medidas estatíticas da imagem sendo analisada. Para tanto, é criada uma matriz quadrada de tamanho MxM onde M é a quantidade de níveis de cinza possíveis na imagem em questão. Essa matriz armazena a probabilidade de dois *pixels* se relacionarem através de uma certa relação espacial[12].

Para esse trabalho, a relação observada é a da vizinhança entre dois *pixels*, sempre analisando o valor a direita de um determinado *pixel* da imagem. Os 255 valores de intensidade possíveis são divididos em 8 níveis. Cada elemento $P_{i,j}$ na GLCM G conta a quantidade de vezes que um *pixel* com valor de intensidade no nível j apareceu à direita de um *pixel* com valor no nível i. A figura 2.3 exemplifica a criação da GLCM a partir de uma imagem com 8 níveis possíveis.

Uma vez criada a matriz de co-ocorrência, diversas medidas podem ser tiradas. Para esse trabalho são extraídas 4 características definidas abaixo. Nas equações apresentadas $P_{(i,j)}$ representa o valor do elemento na posição (i, j) da GLCM, μ_i e μ_j representam a média dos valores de i e j respectivamente e σ_i e σ_j os desvios padrão destes valores.

- **Contraste**: mede o contraste entre um *pixel* e seu vizinho na imagem. Deve ser 0 para uma imagem completamente homogênea. Definido por:

$$C = \sum_{i,j}(i-j)^2 P_{(i,j)} \tag{2.3}$$

- **Correlação**: mede a taxa de correlação de um *pixel* e seu vizinho na imagem inteira. Definida por:

$$Co = \sum_{i,j} \frac{(i-\mu_i)(j-\mu_j)P_{(i,j)}}{\sigma_i \sigma_j} \tag{2.4}$$

- **Energia**: soma do quadrado dos elementos da imagem. Definida por:

$$E = \sum_{i,j} P^2_{(i,j)} \tag{2.5}$$

(a)

(b)

Figura 2.4: (a) Representação dos vetores de velocidade estimado pelo fluxo óptico. (b) Imagem em níveis de cinza onde a intensidade *pixel* representa a magnitude do seu vetor de velocidade.

- **Homogeneidade**: mede a proximidade da distribuição dos elementos à diagonal da matriz. Definida por:

$$H = \sum_{i,j} \frac{P_{(i,j)}}{1 + |i - j|} \tag{2.6}$$

2.2 Vídeos

Quando observamos uma cena no mundo real, ela raramente é estática. Com o passar do tempo, os objetos presentes nela se movem continuamente pelo espaço, criando infinitas imagens diferentes em nossas retinas. Podemos simular essa sensação de movimento na nossa visão através da exibição rápida de imagens que contém valores discretos. Esse é o conceito de um vídeo digital. Um vídeo digital é formado através da amostragem de imagens em três eixos: horizontal, vertical e temporal[14]. A cada intervalo fixo de tempo, uma imagem representada por uma matriz bidimensional de valores de intensidade luminosa e cor é amostrada. Cada uma dessas imagens, ou quadros, representa o estado da cena naquele momento no tempo.

2.2.1 Fluxo Óptico

O movimento é um dos elementos principais para a extração de informações de um vídeo. O fluxo óptico, ou *optical flow*, é uma das técnicas para a estimativa de movimento em vídeos digitais. A técnica consiste em medir a projeção 2D no plano da imagem de um movimento real 3D[**?**]. Essa projeção 2D mede a velocidade de cada *pixel* da imagem. Esses vetores de velocidade da imagem podem ser utilizados para a realização de diversas tarefas. Por exemplo, é possível extrair a magnitude dos vetores e criar uma nova imagem, aonde os valores de intensidade representam o movimento realizado pelo *pixel*. Nesse trabalho, essa técnica é utilizada para a segmentação de objetos em movimento contra um fundo estático, detalhada no capítulo 5. A figura 2.4 mostra os vetores de movimento calculados e uma imagem em níveis de cinza que representa as magnitudes das velocidades de cada pixel.

Para calcular o fluxo óptico, começamos assumindo que no intervalo entre dois quadros a intensidade de cada *pixel* não muda de forma significativa. Chamamos então a intensidade de cada *pixel* na posição (x,y) em um determinado momento t de $I(x, y, t)$ e portanto, podemos dizer que:

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{2.7}$$

onde dx e dy são o deslocamento que o *pixel* no intervalo dt entre dois quadros.

Queremos então encontrar a velocidade $v = (\frac{dx}{dt}, \frac{dy}{dt})$ do *pixel*. Usando a expansão por série de Taylor e manipulações matemáticas na equação 2.7, obtemos a equação 2.8[?].

$$\nabla I v + I_t = 0 \tag{2.8}$$

onde ∇I é o gradiente da função intensidade. Essa equação descreve o chamado problema de Restrição do Fluxo Óptico[?]. Porém, há um problema na equação. A partir dela só é possível obter um dos componentes da velocidade. A solução completa da velocidade depende de um cálculo na vizinhança de cada ponto. Diversos métodos foram criados para a fazer a estimativa da velocidade, com propriedades diversas. Para esse trabalho foi escolhido o método desenvolvido por Lucas e Kanade, detalhado em [?][?] e [?]. Este método se utiliza de características de uma vizinhança local e visa minimizar o erro de mínimos quadrados. Foi escolhido por ser resistente a ruído e ter mostrado um resultado mais adequado para as exigências do trabalho.

Capítulo 3

Redes Neurais Artificiais

Comparado com os computadores mais avançados que existem hoje, o cérebro humano ainda se mostra muito mais poderoso e eficaz do que as máquinas. O cérebro é capaz de processar informação a uma velocidade muito maior do qualquer computador convencional e realiza com facilidade tarefas como o reconhecimento de padrões e classificação de objetos, enquanto algoritmos tradicionais falham.

Redes neurais artificiais foram criadas com inspiração no funcionamento e na estrutura do cérebro, como uma alternativa poderosa para resolver problemas que as arquiteturas tradicionais não eram capazes de resolver eficientemente. Essas redes emulam a estrutura cerebral natural, se utilizando de elementos distintos de processamento (neurônios) que se comunicam para realizar a tarefa desejada.

Simon Haykin[8] define uma rede neural como "...um processador distribuído massivamente paralelo composto por unidades simples de processamento, que possui uma propensidade natural a armazenar conhecimento experimental e torná-lo disponível para uso."

As redes neurais artificiais são utilizadas para resolver problemas como ajuste de funções, classificação de objetos e reconhecimento de padrões. Nesse capítulo, serão explicados conceitos importantes para o entendimento das redes artificiais, começando pelo seu elemento mais básico, o neurônio.

3.1 Neurônios

O neurônio é a estrutura básica do sistema nervoso central. É uma célula composta principalmente de três partes distintas: o corpo celular, os dendritos e o axônio. O corpo celular é a estrutra central do célula aonde está contido o núcleo do neurônio e são executados as suas funções vitais. Os dendritos são prolongamentos do corpo celular responsáveis por receber sinais e o axônio é uma extensão maior do corpo celular, responsável por enviar sinais a outros neurônios. Dois neurônios interagem em apenas pontos de contato chamados sinapses. Nessas sinapses o axônio de um neurônio envia sinais para os dendritos de um segundo neurônio. A figura ?? ilustra a estrutura básica dos neurônios.

O neurônio artificial também possui três estruturas básicas semelhantes àquelas do neurônio natural. Seus três elementos básicos são: as entradas(análogas aos dendritos), a sáida(análoga ao axônio) e um núcleo de processamento.

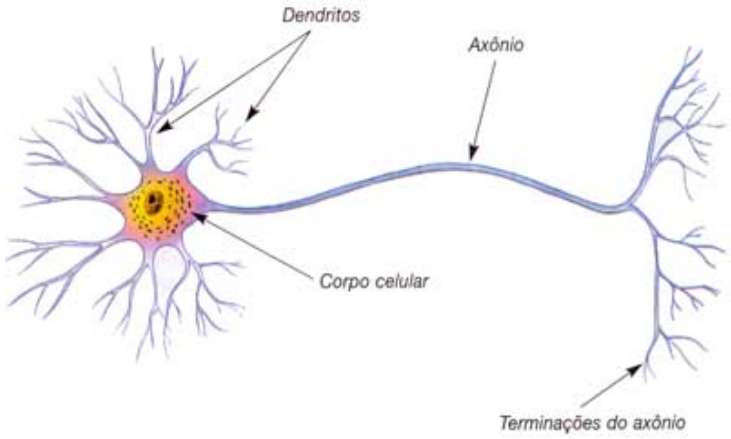


Figura 3.1: A estrutura básica de um neurônio humano

O neurônio pode ter uma ou mais conexões de entrada caracterizadas por um peso wi e que recebe um valor xi. No núcleo do neurônio artificial há um somador que calcula o valor u agregado das entradas, ponderadas pelo peso correspondente. O resultado dessa soma ponderada é então deslocado de um valor escalar e em seguida submetido a uma função chamada de função de ativação, que determina a saída final do neurônio. Em suma podemos descrever a saída de um neurônio n que possui i entradas através das seguintes equações:

$$u_n = \sum_{j=1}^{i} x_j w_j \tag{3.1}$$

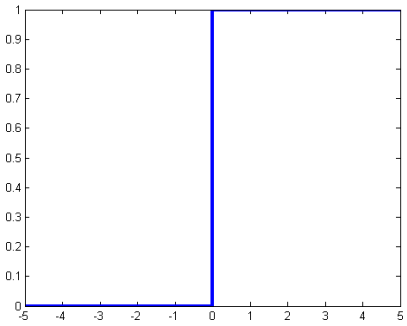
$$s(n) = f(u_n + b) \tag{3.2}$$

Aonde u_n é o valor agregado das entradas, x_j é a j-ésima entrada e w_j o peso da conexão associada, $s(n)$ é a saída do neurônio, f é a função de ativação e b o deslocamento, ou *bias* do neurônio n. A figura **??** ilustra a estrutura básica de um neurônio artificial.

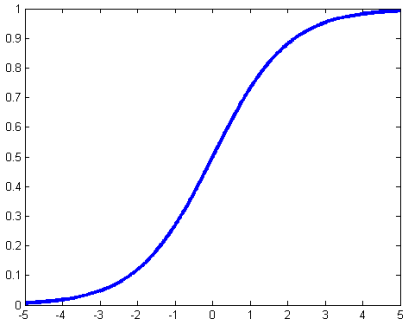
Claramente, a função de ativação do neurônio é o grande fator que define o seu comportamento e consequentemente sua funcionalidade. Existem dois tipos básicos de funções de ativação[8]: a função degrau e a função sigmóide.

O neurônio mais simples possível tem sua saída regida por uma função degrau e saída binária. Isto é, se o valor agregado u de suas entradas for maior que um determinado limiar, a sua saída será 1 e a saída será 0 caso contrário. Apesar de ser capaz de resolver alguns problemas, esse tipo de neurônio tem algumas desvantagens. Por vezes, alterações sutis podem representar a diferença entre os dois valores possíveis da sáida. Além disso, um alcance limitado de valores na saída prejudica o treinamento.

A escolha mais comum de de função de ativação para a construção de redes neurais é a família de funções sigmóides. Essas funções, que tem um gráfico com formato de S, assumem valores contínuos entre 0 e 1, o que faz com que alterações sutis na entrada representem alterações mais sutis na saída, aumentando a qualidade da informação gerada pelo neurônio. A mais comum das funções sigmóides utilizadas é a função logística, definida pela equação 3.3, onde a é uma constante que mede a declividade da curva.



(a)



(b)

Figura 3.2: (a) Gráfico da função degrau com limiar 0. (b) Gráfico da função logística com a = 1

$$f(u) = \frac{1}{1 + \exp -au} \tag{3.3}$$

Um dos motivos que tornou a função logística uma escolha popular para os neurônios artificiais foi a sua derivação simples, uma vez que algoritmos de treinamento muitas vezes usam a derivada da função de ativação[10].

É interessante reparar que quando a se aproxima do infinito, a função logística se comporta da mesma forma que a função degrau.

3.2 Redes Neurais Feed-Forward

Apesar de neurônios serem capazes de resolver alguns problemas sozinhos, o verdadeiro poder das redes neurais vem da interconexão entre os neurônios de forma a criar ligações semelhantes às sinapses dos neurônios naturais.

Existem várias arquiteturas para a formação destas redes de neurônios, porém neste trabalho a discussão será limitada àquela utilizada na implementação do programa: as chamadas redes neurais *feed-forward*.

São necessárias ao menos três camadas para a construção da rede neste modelo: a primeira camada, chamada de camada de entrada, uma ou mais camadas intermediárias(ou ocultas) e a camada de saída. O papel das camadas ocultas é intermediar entre as entra-

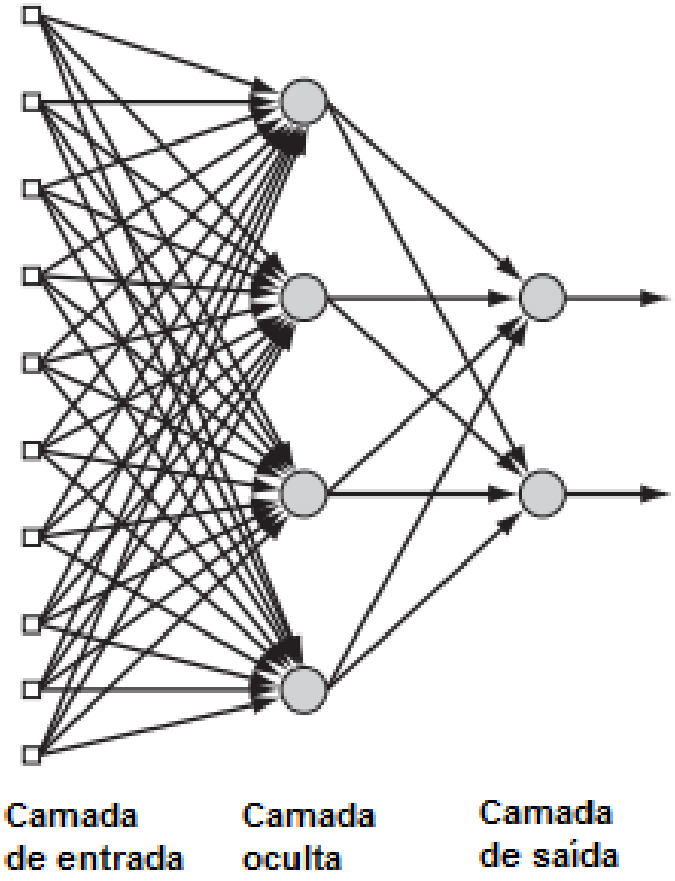


Figura 3.3: A arquitetura feed-forward. Cada neurônio se comunica com os neurônios da camada seguinte até que a saída final seja produzida. Adaptada de [8]

das externas e a saída da rede, de forma a possibilitar que a rede extraia dados estatíticos mais significativos da sua entrada. Uma camada é composta de um número de neurônios que agem de forma paralela. Cada neurônio de uma camada que não é a de saída está conectado apenas ao neurônios da camada seguinte, de forma que não há comunicação entre uma camada e camadas anteriores a ela. Isto é, a informação flui na rede apenas no sentido entrada-saída, como ilustrado na figura ??.

O vetor de entrada da rede é alimentado aos neurônios da primeira camada, cuja saída serve de entrada para a segunda camada e assim por diante, até que a última camada seja alimentada pela saída da camada anterior e produza a saída final da rede.

3.3 Treinamento

Para que uma rede neural artificical possa ser utilizada, ela precisa antes aprender a realizar a tarefa para a qual foi criada. Para isso a rede precisa ter duas abilidades importantes: de aprendizado e de generalização. Aprendizado é a capacidade da rede de aproximar o comportamento das entradas fornecidas durante o treinamento, enquanto

generalização é a sua capacidade de prever e operar sobre dados além do conjunto com a qual foi treinada[16].

É necessário então que a rede passe por um processo de treinamento, onde os valores dos pesos wi e os deslocamentos b de cada neurônio são definidos de forma que o funcionamento da rede seja ótimo para a tarefa que se deseja realizar. Para realizar o treinamento de uma rede, o primeiro passo é definir 3 conjuntos distintos de entradas:

- **Conjunto de testes:** A rede é submetida às entradas deste conjunto para que seja feita a calibração dos valores da matriz de pesos W e do vetor de deslocamentos B.
- **Conjunto de validação:** Após uma etapa de treinamento, a rede computa os dados deste conjunto de entradas para validar os valores da matriz W e do vetor B escolhidos até então.
- **Conjunto de testes:** Por fim, após o treinamento, a rede é submetida aos dados deste conjunto, a fim de verificar o seu funcionamento correto.

Isto é, a cada ciclo de treinamento, a rede ajusta os seus pesos e deslocamentos de acordo com o conjunto de treinamento, e em seguida processa os dados do conjunto de validação. Se for determinado que a rede teve sucesso no processo de validação, ela fica pronta pra passar pelo conjunto de testes, aonde as sua capacidade de generalização é testada. A matriz W de pesos gerada durante este processo é uma forma de representação do "conhecimento"da rede. Assim, o aprendizado não é uma característica de cada neurônio, mas sim um processo que ocorre na rede inteira como resultado do treinamento[**?**].

Existem duas principais maneiras de se executar o treinamento de uma rede.

- **Treinamento supervisionado:** Neste paradigma os conjunto de treinamento e de validação consistem em um grupo de vetores de entrada x e um gabarito de vetores de saída desejado y. O treinamento acontece até que o vetor de saída da rede para uma determinada entrada x_i seja suficientemente próximo do vetor gabarito y_i correspondete.
- **Treinamento não-supervisionado:** Neste paradigma a rede apenas recebe um conjunto x de entradas e aprende características intrínsicas do dados apresentados a ela.

Uma vez definido o paradigma de treinamento, é necessário definir uma técnica para a análise do erro e alteração dos valores da matriz de pesos W e do vetor B de deslocamentos. Uma das técnicas mais utilizadas é a chamada de *backpropagation*[7, 16]. A técnica consiste de uma análise do erro apresentado na saída da rede, que então é utilizada para se percorrer a rede no sentido contrário dos dados modificando os valores de W e B de forma a diminuir o erro na próxima iteração. Essa técnica é muito utilizada para sistemas de treinamento supervisionado, onde o erro é avaliado através de uma comparação da saída da rede com o gabarito fornecido. Uma vez que a os valores de W e B param de mudar, diz-se que a rede atingiu um estado de convergência[**?**] e o treinamento finaliza.

3.4 Classificação de padrões

Um problema muito comum que pode ser solucionado com redes neurais artificiais é o da classificação de padrões. Isto é, analisar um padrão ou um conjunto de características de um conjunto de dados e designá-lo a uma de várias classes pré-determinadas. Os nossos sentidos são muitas vezes capazes de fazer isso com extrema facilidade em uma fração de segundo. Por exemplo, se estamos andando pela rua, somos capazes de dizer que um som mais agudo e melódico é o canto de um pássaro, enquanto um som mais grave e contínuo provavelmente pertence ao motor de um carro passando. Essa habilidade é adquirida através de um aprendizado que ocorre durante as nossas vidas, e da mesma forma podemos ensinar uma rede neural artificial a reconhecer e classificar padrões.

Normalmente, uma rede que será utilizada para classificar padrões passa por um treinamento supervisionado, onde ela é alimentada com um vetor de entradas que contém características, ou *features*, que descrevem os padrões junto com um gabarito que determina a classe a qual cada entrada pertence. Espera-se que depois do treinamento, ao ser apresentada a um novo padrão, a rede seja capaz de determinar a qual classe esse padrão pertence. Para que a rede demonstre boa generalização, é importante que ela não aprenda uma representação exata do conjunto de treinamento, mas que ela construa um modelo estatístico do processo que gera os dados [1].

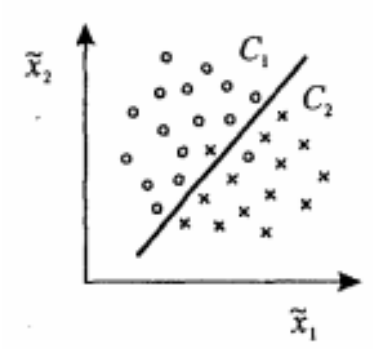


Figura 3.4: Um gráfico que mostra duas características $\tilde{x}_1$ e $\tilde{x}_2$ de um problema de classificação hipotético. Círculos são padrões da classe C_1, enquanto os 'x' são padrões da classe C_2. O limiar de decisão, representado pela linha, consegue separar bem as duas classes quando as *features* são consideradas como um par. Porém haveria grande sobreposição se os valores fossem considerados separadamente. Extraída de [1]

A escolha dos *features* a serem utilizados para descrever os padrões não pode ser aleatória. Deve se escolher um número de características suficiente para descrever todos os padrões que podem ser encontrados. Além disso é importante que os *features* sejam suficientemente discriminantes. Isto é, deve haver um mínimo de superposição entre as características de classes distintas. Esses *features* podem ser escolhidos a mão, com base em um conhecimento prévio dos dados a serem classificados, ou extraídos automaticamente por um algoritmo desenvolvido para tal. Normalmente as *features* são processadas em conjunto e não consideradas individualmente. A figura ?? mostra como isso é benéfico. Se as características hipotéticas $\tilde{x}_1$ e $\tilde{x}_2$ forem consideradas separadamente, há significativa superposição, de forma que é difícil separar com convicção as duas classes C_1 e C_2, porém, quando consideradas juntas, é possível traçar uma linha que as separa facilmente, apesar

de alguns poucos elementos se confundirem. O papel da rede é determinar a função que traça essa linha através do treinamento, e retornar uma probabilidade de que um conjunto de *features* alimentado a ela faça parte de uma determinada classe.

Um exemplo: se queremos criar uma rede que diferencia imagens de peixes palhaço de imagens de tubarões, um conjunto adequado de *features* provavelmente envolveria a cor c predominante do animal, o tamanho do animal t e o tamanho b das suas barbatanas. A rede então seria submetida a entradas na forma de um vetor $(c, t, b)^T$, valores numéricos que caracterizam uma imagem a ser analisada. Podemos facilmente separar as classe através dessas características. Peixes com cor mais próxima do laranja e barbatanas pequenas são muito provavelmente peixes palhaços, enquanto aqueles que têm a barbatana grande seriam classificados como tubarões.

Capítulo 4

Trabalhos Correlatos

Nesse capítulo serão expostas soluções para o problema apresentado no capítulo 1, acompanhadas de uma discussão sobre vantagens e desvantagens, e as razões pela escolha da solução proposta no capítulo 5.

Podemos separar os métodos de detecção de vagas em duas grandes categorias: métodos intrusivos e não-intrusivos. Métodos intrusivos são caracterizados por exigirem instalações mais complexas, envolvendo a inserção de equipamento no asfalto do estacionamento ou em um estrutura de concreto. Os métodos não intrusivos normalmente se utilizam de equipamentos externos, que não exigem obras para a sua instalação.

Um exemplo comum de método itrusivo é o uso de sensores individuais em cada vaga do estacionamento. Diversos tipos de sensores podem ser utilizados, cada um com vantagens e desvantagens próprias [13]. Sensores infravermelhos versáteis e de simples instalação, mas podem ser afetados por condições ambientais. Tubos pneumáticos e sensores de peso sob o asfalto têm alta cofiabilidade, mas exigem grandes obras para a instalação.

Alguns tipos de sensores podem ser instalados de forma não intrusiva. Esses sensores podem ser afixados no teto ou em uma estrutura próxima a vaga. Sensores ultrassônicos por exemplo, pertencem a essa categoria. Eles transmitem ondas sonoras de baixa frequência e usam a energia refletida para determinar a ocupação das vagas [11].

Intrusivos ou não, os métodos de detecção que utilizam sensores compartilham duas desvantagens: é necessário a instalação do sensor em cada uma das vagas, aumentando o custo do sistema e diminuindo sua escalabilidade, e eles não são adequados para estacionamentos descobertos, onde é impossível instalar sensores no teto e obras no asfalto rodoviário podem danificá-lo permanentemente [13].

Uma solução que se mostrou adequada para os estacionamentos descobertos foi o uso de *softwares* de visão computacional, que analisam imagens de câmeras de vídeo para determinar a ocupação das vagas do estacionamento. Essa opção tem um custo baixo e exige apenas que sejam instaladas câmeras em pontos estratégicos do estacionamento. Uma vez que as câmeras foram instaladas e as imagens estão sendo capturadas, resta que seja executado um programa que as analise.

Para a escolha da abordagem para este trabalhos foram escolhidos os seguintes critérios principais:

- O sistema deve poder ser instalado e começar sua execução em um estacionamento em qualquer estado, sem a necessidade de iniciar com todas as vagas descocupadas,

possibilitando uma instalação a qualquer momento, sem interrupção das operações do estacionamento.

- O sistema deve necessitar de interação humana mínima, se limitando apenas uma rápida calibração no início de sua execução a fim de evitar erro humano.
- O sistema deve ser capaz de identificar a posição das vagas do estacionamento após um certo tempo de execução, e não através de entrada manual, para compensar por erros na calibração.
- O sistema deve, além de determinar o número de vagas livres no estacionamento, ser capaz de indicar a posição aproximada de tais vagas de forma a facilitar ainda mais a busca por uma vaga desocupada.

Em [4], Delibaltov *et al* descrevem um método de detecção que estima um volume tridimensional para cada vaga marcada na imagem e depois se utiliza de redes neurais para determinar *pixels* da imagem pertencentes a veículos. Em seguida a sistema computa a probabilidade de que uma certa vaga esteja ocupada baseado em uma função de probabilidade e os *pixels* de veículo na região de interesse de cada vaga. Esse método se mostra preciso e eficaz, mas exige que o estacionamento esteja vazio e que cada vaga seja marcada individualmente no momento de execução, de forma que ele não é capaz de mapear as vagas do estacionamento ou detectar veículos estacionados irregularmente.

Bong [2] propõe um sistema que se utiliza de subtração de imagens e o uso de detecção automática de coordenadas aproximadas das vagas para resolver este problema. Porém esse sistema exige que seja armazenada uma imagem do estacionamento completamente vazio, além de que seja criada uma imagem que marca cada vaga para o processo de inicialização do sistema.

Um sistema que se utiliza de uma técnica de classificação de histogramas de crominância em áreas de interesse e um algoritmo de que encontra pontos com *features* relevantes foi proposto em [15]. Esse sistema tem a vantagem de ser relativamente invariante quanto a iluminação da imagem, por utilizar os canais de crominância para a análise. Também é bastante capaz de resolver o problema de oclusão de veículos atrás de outros na imagem de câmeras com uma angulação maior.

A solução proposta neste trabalho procura compensar alguns dos problemas presentes nos trabalhos mencionados, enquanto apresenta uma nova abordagem a ser considerada e aprimorada por trabalhos futuros.

Capítulo 5

Solução Proposta

Uma vez que o conhecimento teórico necessário para o entendimento completo do trabalho já foi apresentado, neste capítulo será definida a solução proposta.

A solução desenvolvida consiste em um algoritmo analisa imagens de vídeo de um estacionamento descoberto e determina o número de vagas livres na imagem, além da sua localização aproximada. O sistema funciona bem em imagens de menor qualidade, mas é necessário que o vídeo adquirido seja em cores.

O trabalho se preocupa em cumprir os critérios definidos no capítulo 4. Além disso, o sistema foi desenvolvido de forma que pudesse processar as imagens adquiridas de forma mais próxima possível do tempo real, causando o mínimo de atrasos para o processamento de quadros subsequentes do vídeo.

O sistema recebe como entrada um vídeo em cores capturado em um certo ângulo e um vetor de regiões de interesse no vídeo. A saída do programa a cada quadro é o número de vagas livres em cada região de interesse no vídeo.

A imagem **??** contém um fluxograma que mostra as etapas do processamento de cada quadro do vídeo adquirido. No decorrer deste capítulo cada um desses passo será discutido com mais detalhes.

5.1 Aquisição

Uma câmera montada em um poste de luz ou outro ponto similar captura as imagens utilizadas pelo programa. A câmera é montada de forma que o seu campo de visão contenha o máximo de vagas possível, porém que ainda seja possível visualizar o asfalto das vagas desocupadas e ocorra o mínimo de oclusão de veículos. A figura ?? mostra um quadro de uma aquisição em ângulo ideal.

Diversas câmeras podem ser instaladas para aumentar a cobertura do estacionamento. Neste caso, cada vídeo é processado por uma cópia diferente do sistema. Por isso, neste capítulo a discussão será focada apenas no processamento do vídeo de uma câmera.

5.2 Regiões de Interesse

No momento da instalação do programa, é necessário definir um número qualquer de regiões de interesse(ROIs). Essas regiões determinam a área da imagem onde existem

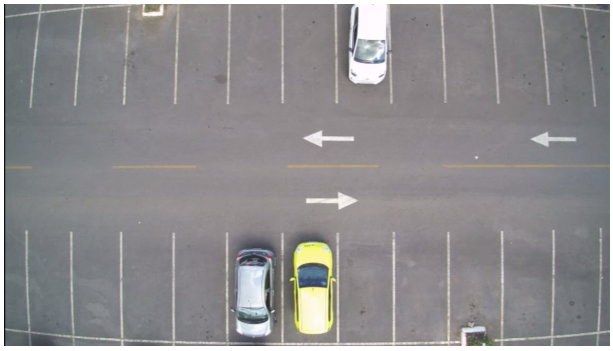


Figura 5.1: Um quadro de um vídeo adquirido por uma câmera do sistema

vagas. Além de determinar as regiões, deve-se informar ao programa o número de vagas existente em cada região de interesse.

As ROIs devem ser retangulares e determinadas de forma a não haver interseção entre elas, como exemplificado na figura **??** que mostra as regiões determinadas para o quadro da figura ??.

5.3 Seções Verticais

Referências

[1] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995. vi, 14

[2] DBL Bong, KC Ting, and KC Lai. Integrated approach in the design of car park occupancy information system (coins). *IAENG International Journal of Computer Science*, 35(1):7–14, 2008. 17

[3] José Eustáquio Rangel de Queiroz and Herman Martins Gomes. Introdução ao processamento digital de imagens. *RITA*, 13(2):11–42, 2006. 2

[4] Diana Delibaltov, Wencheng Wu, Robert P Loce, Edgar Bernal, et al. Parking lot occupancy determination from lamp-post camera images. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 2387–2392. IEEE, 2013. 17

[5] R Gaunt. Color spaces in digital video. Technical report, Lawrence Livermore National Lab., CA (United States), 1997. 4

[6] Rafael C Gonzalez. *Digital image processing*. Pearson Education India, 2009. 3

[7] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016. 13

[8] Simon Haykin. *Neural Networks*. Prentice Hall, 2 edition, 1999. vi, 9, 10, 12

[9] IBGE. *Introdução ao processamento digital de imagens*. IBGE, 2000. 4

[10] Nikola K Kasabov. *Foundations of neural networks, fuzzy systems, and knowledge engineering*. Marcel Alencar, 1996. 11

[11] Amin Kianpisheh, Norlia Mustaffa, Pakapan Limtrairut, and Pantea Keikhosrokiani. Smart parking system (sps) architecture using ultrasonic detector. *International Journal of Software Engineering and Its Applications*, 6(3):55–58, 2012. 16

[12] Jefferson Gustavo Martins, YMG Costa, D Bertolini, and LS Oliveira. Uso de descritores de textura extraıdos de glcm para o reconhecimento de padroes em diferentes domınios de aplicaçao. In *XXXVII Conferencia Latinoamericana de Informática*, pages 637–652, 2011. 6

[13] Idris M.Y.I, Leng Y.Y, et al. Car park system: A review of smart parking system and its technology. *Information Technology Journal*, 8(8):101–113, June 2009. 16

[14] Charles A. Poynton. *A Technical Introduction to Digital Video*. John Wiley & Sons, Inc., New York, NY, USA, 1996. 5, 7

[15] Nicholas True. Vacant parking space detection in static images. *University of California, San Diego*, 2007. 17

[16] Guoqiang Peter Zhang. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(4):451–462, 2000. 13