Daniel Luis Cosmo

Detecção de Pedestres Utilizando Descritores de Orientação do Gradiente e Auto Similaridade de Cor

Brasil

2014

Daniel Luis Cosmo

Detecção de Pedestres Utilizando Descritores de Orientação do Gradiente e Auto Similaridade de Cor

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Universidade Federal do Espírito Santo – UFES

Programa de Pós Graduação em Engenharia Elétrica – PPGEE

Laboratório de Computadores e Sistemas Neurais – CISNE

Orientador: Evandro Ottoni Teatini Salles

Coorientador: Patrick Marques Ciarelli

Brasil

2014

Daniel Luis Cosmo
Detecção de Pedestres Utilizando
Descritores de Orientação do Gradiente
e Auto Similaridade de Cor/ Daniel Luis Cosmo. – Brasil, 2014-
83 p. : il. (algumas color.) ; 30 cm.

Orientador: Evandro Ottoni Teatini Salles

Dissertação (Mestrado) – Universidade Federal do Espírito Santo – UFES
Programa de Pós Graduação em Engenharia Elétrica – PPGEE
Laboratório de Computadores e Sistemas Neurais – CISNE, 2014.

1. Palavra-chave1. 2. Palavra-chave2. I. Orientador. II. Universidade xxx. III.
Faculdade de xxx. IV. Título

CDU 02:141:005.7

Daniel Luis Cosmo

Detecção de Pedestres Utilizando Descritores de Orientação do Gradiente e Auto Similaridade de Cor

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Trabalho aprovado. Brasil, 06 de novembro de 2014:

Evandro Ottoni Teatini Salles - UFES
Orientador

Patrick Marques Ciarelli - UFES
Co-orientador

Lee Luan Ling - UNICAMP
Convidado

Klaus Fabian Côco - UFES
Convidado

Brasil
2014

Dedico este trabalho à minha família, pelo ensino e exemplos dados a mim.

Agradecimentos

Aos meus pais, que me ajudaram tanto em momentos difíceis;

À minha namorada, que sempre me apoiou nas minhas decisões;

Aos meus orientadores, sempre disponíveis a solucionar minhas dúvidas;

E principalmente a Deus, que me deu forças para chegar até aqui, não deixando faltar nada em minha vida.

Resumo

Detecção de pedestres é um problema muito abordado na atualidade, possuindo diversas aplicações com potencial para melhorar a qualidade de vida da sociedade. Algumas dessas aplicações se encontram nas áreas de sistemas de auxílio ao motorista, reconhecimento de pessoas em fotos e vídeos, e vigilância. Atualmente existe um grande número de pesquisas envolvendo este assunto, trazendo muitas ramificações ao estado da arte no que diz respeito a detecção de pedestres. Esta dissertação apresenta um sistema de detecção de pedestres em ambientes não controlados baseado em janelas deslizantes. Sistemas deste tipo são compostos por dois blocos principais: um para a extração de características e outro para classificação das janelas. Duas técnicas de extração de características são usadas, sendo elas: HOG (*Histogram of Oriented Gradient*) e CSS (*Color Self Similarities*), e para classificar as janelas é usado o SVM (*Support Vector Machine*) linear. Além dessas técnicas, são também utilizadas: *mean shift* e agrupamento hierárquico, para a fusão de múltiplas detecções sobrepostas; e filtro bilateral, para pré-processamento da imagem. Os resultados obtidos sobre o banco de dados INRIA *Person Database* mostram que o sistema proposto, usando somente o descritor HOG, apresenta melhorias em relação a sistemas semelhantes, com um *log average miss rate* igual a 41,8%, contra 46% da literatura. Este resultado foi possível devido ao corte das detecções finais para melhor adequação às anotações modificadas, e também a algumas modificações feitas nos parâmetros dos descritores. A adição do descritor CSS modificado ao HOG aumenta a eficácia do sistema, levando a um *log average miss rate* igual a 36,2%, classificando separadamente cada descritor.

Palavras-chaves: Detecção de Pedestres, Janelas Deslizantes, Histograma de Gradientes Orientados, Auto Similaridade de Cores, Filtro Bilateral, Agrupamento Hierárquico, *Mean Shift*

Abstract

Pedestrian detection is a key problem in our days, having a large number of applications with potential to make better the quality of life of our society. Some of these applications can be found in driver assistance systems, people recognition in photos and videos, and surveillance. Nowadays, there is a large number of researches in this area, generating a lot of ramifications in the state of the art for pedestrian detection. This dissertation presents a pedestrian detection system in non-controlled environments based on sliding windows. Systems of this type are based on two major blocks: one for feature extraction and other for window classification. Two techniques for feature extraction are used: HOG (Histogram of Oriented Gradient) and CSS (Color Self Similarities), and to classify windows we use linear SVM (Support Vector Machines). Beyond these techniques, we use: mean shift and hierarchical clustering, to fuse multiple overlapping detections; and bilateral filter, to preprocess the image. The results obtained bu testing the dataset INRIA Person show that the proposed system, using only HOG descriptors, achieves better results over similar systems, with a log average miss rate equal to 41.8%, against 46% of the literature. These results were possible due to the cutting of the final detections to better adapt them to the modified annotations, and some modifications on the parameters of the descriptors. The addition of the modified CSS descriptor to the HOG descriptor increases the efficiency of the system, leading to a log average miss rate equal to 36.2%, when classifying each descriptor separately.

Key-words: Pedestrian Detection, Sliding Windows, Histogram of Oriented Gradient, Color Self Similarities, Bilateral Filter, Hierarchical Clustering, Mean Shift

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ADAS	*Advanced Driver Assistance Systems*
CSS	*Color Self Similarities*
ETH	*Eidgenössische Technische Hochschule*
FPPI	Falso Positivo Por Imagem
HIKSVM	*Histogram Intersection Kernel Support Vector Machines*
HOG	*Histogram of Oriented Gradients*
INRIA	*Institut National de Recherche en Informatique et en Automatique*
LDA	*Linear Discriminant Analysis*
PCA	*Principal Component Analysis*
PPS	*Pedestrian Protection System*
SIFT	*Scale Invariant Feature Transform*
SVM	*Support Vector Machines*

Sumário

1 Introdução

1.1 Motivação

Nos dias atuais, o computador se tornou um equipamento essencial no mundo inteiro. Eles realizam trabalhos repetitivos e matematicamente intensivos mais rápidos e com maior eficácia do que o ser humano. É natural estudar a possibilidade do computador poder realizar tarefas mais inteligentes automaticamente, como a detecção de objetos em cenas naturais, tarefa essa realizada com facilidade pelo ser humano. Logo, pesquisas com a finalidade de fazer o computador "enxergar" estão sendo muito desenvolvidas atualmente.

Uma capacidade que o ser humano possui, e que exerce naturalmente, é a capacidade de identificar pessoas em qualquer ambiente. O mesmo não pode ser dito a respeito de um computador, pois para ser capaz de identificar pessoas em uma imagem qualquer, são necessários inúmeros cálculos matemáticos. Devido a importância dessa tarefa para o cenário atual de pesquisas na área de visão computacional, este trabalho tem como foco a detecção de pessoas em ambientes não controladas.

A detecção automática de pessoas possui diversas aplicações com potencial para melhorar a qualidade de vida da sociedade. Uma dessas aplicações se encontra na indústria automotiva, na forma de sistemas inteligentes embarcados com a finalidade de identificar pedestres próximos ao carro. Com o aumento da popularidade dos automóveis na última década, acidentes automotivos se tornaram uma importante causa de fatalidades. Anualmente, cerca de 10 milhões de pessoas no mundo se envolvem em acidentes automotivos, e 2 a 3 milhões dessas pessoas sofrem ferimentos graves (JONES, 2002).

Estes sistemas inteligentes, denominados de forma abrangente como ADAS's (*advanced driver assistance systems*), auxiliam o motorista na tomada de decisões em situações de risco, provendo medidas de distância a obstáculos e estimativas de tempo para colisão. Um exemplo de ADAS, o PPS (*Pedestrian Protection System*) (GERONIMO et al., 2010), tem como objetivo detectar a presença de pedestres em uma área específica ao redor do veículo, tanto os que estão parados como os que estão se movendo. Tal conhecimento possibilita ao PPS avisar o motorista, executar ações de frenagem e ativar *airbags* externos, caso o impacto seja inevitável. A Figura 1 apresenta uma forma de ADAS utilizando vários sensores. No Brasil houve um aumento da importância de sistemas deste tipo, devido à enorme quantidade de fatalidades causadas por acidentes de trânsito, como pode ser visto em (ANDRADE et al., 2014).

Outra aplicação da detecção de pedestres se encontra na área de monitoramento e vigilância. Nos dias atuais, o aumento da preocupação com a segurança individual e

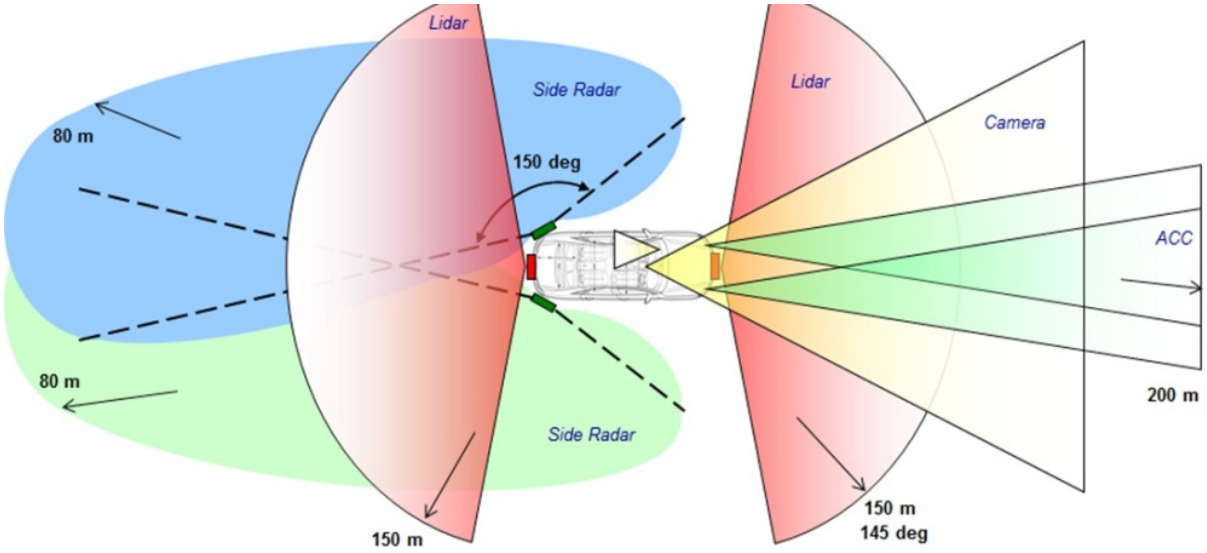


Figura 1 – Áreas de atuação de um ADAS utilizando radar, câmeras e sensores LIDAR (*Light Detection And Ranging*) (VOLKSWAGEN, 2014).

pública fez crescer a necessidade de sistemas mais sofisticados de monitoramento. Uma ideia promissora consiste no uso de um par de câmeras distribuídas em volta do ambiente de interesse, uma obtendo imagens do espectro visível e outra do espectro infravermelho (TORRESAN et al., 2004). Como não é prático ter um humano olhando as gravações em tempo real, deixa-se a cargo do computador a detecção de pessoas não autorizadas no local da gravação, que somente então serão examinadas por uma pessoa. Com isso, torna-se possível o monitoramento de várias gravações apenas por uma pessoa.

1.2 Objetivo

Este trabalho tem como objetivo a criação de um método computacional capaz de detectar pedestres em imagens não controladas, ou seja, imagens capturadas naturalmente, sem o pré conhecimento de que as mesmas serão usadas em um sistema de detecção de pedestres. Em particular, ele visa a criação de um detector de pedestres, onde o detector busca por pedestres em uma cena. Diferentemente de aplicações como ADAS, este trabalho não estuda a aquisição das imagens por meio de câmeras, nem a utilização do detector em um sistema maior.

Para uma definição mais precisa do objetivo deste trabalho, pode-se ver um detector de pedestres como a combinação de dois blocos chave: um algoritmo de extração de características que descreve regiões da imagem como vetores de características, e um classificador que usa as características computadas para realizar decisões sobre a presença ou ausência do pedestre naquela região da imagem. Para selecionar a região da imagem no qual serão aplicados esses dois blocos, é criada uma janela espacial de tamanho fixo,

que desliza sobre a imagem. Para cada posição da janela é realizada uma extração de características e uma classificação. Logo, o sistema de detecção de pedestres é baseado em janelas deslizantes. Um exemplo do procedimento básico pode ser visto na Figura 2.

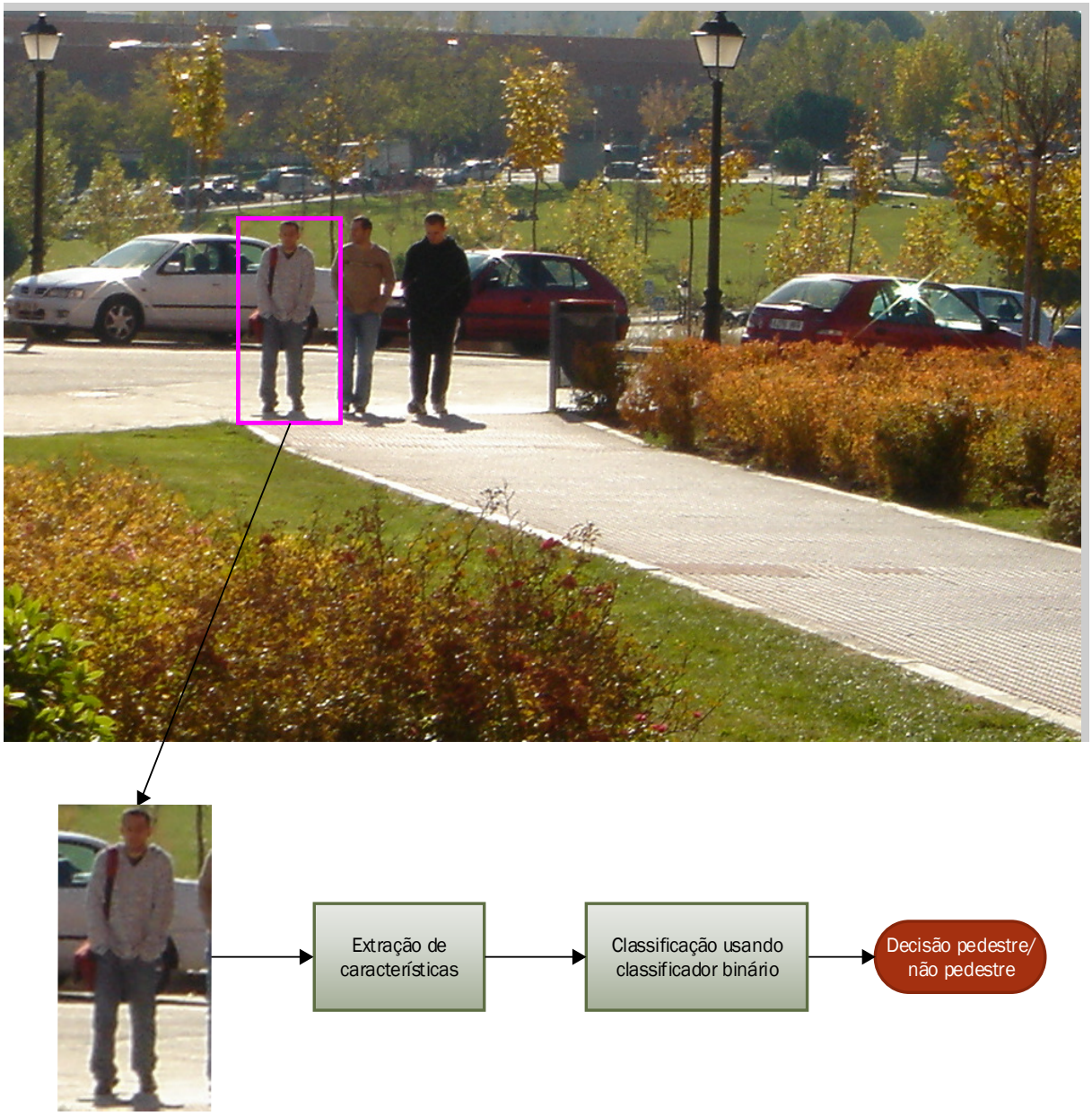


Figura 2 – Exemplo do uso dos dois blocos básicos do detector de pedestres. Uma janela é varrida na imagem, e para cada posição da janela é extraído um vetor de características e feita uma classificação da região em pedestre/não pedestre.

Para a criação do modelo usado pelo classificador, é fornecido ao mesmo um conjunto de imagens de treinamento, algumas contendo pedestres e outras não. O classificador é então treinado com essas imagens, tornando-se capaz de identificar a presença ou não de pedestres em determinada janela da imagem. Logo, o sistema de detecção de pedestres é baseado em duas fases principais: a fase de treinamento e a fase de varredura.

1.3 Caracterização do Problema

A maior dificuldade em construir um detector de pedestres robusto diz respeito a grande quantidade de variações em uma imagem. Vários fatores contribuem para isso (DALAL, 2006):

- Primeiramente, a formação de uma imagem suprime informações de profundidade 3-D e cria uma dependência em relação ao ângulo de visão, de forma que uma pequena mudança da posição ou orientação do pedestre em relação a câmera pode ocasionar uma mudança considerável na aparência do pedestre. Um assunto semelhante diz respeito a variação de escalas em que um pedestre pode ser visto. Um detector de pedestres deve ser capaz de prover invariância à mudanças no ângulo de visão e escala dos pedestres.

- Em segundo lugar, a maioria dos objetos naturais possuem uma grande variação inter classe. No caso de pessoas, tanto a aparência como a pose mudam consideravelmente entre imagens, e diferenças no vestuário aumentam mais ainda tal variação. Um detector robusto deve tentar alcançar independência em relação a essas variações.

- Em terceiro lugar, a desorganização do fundo é comum e varia de imagem para imagem. Pode-se citar como exemplos imagens obtidas em ambientes naturais e cenas de paisagens urbanas. O detector deve ser capaz de distinguir o pedestre de regiões de fundo complexo.

  As duas últimas dificuldades são conflitantes e precisam ser resolvidas simultaneamente. Um detector que é treinado especificamente para uma instância do objeto que se deseja classificar irá possuir uma pequena taxa de falsos positivos, mas em compensação irá perder qualquer objeto que apresente um pouco de variação interclasse. Já um detector treinado para uma grande variação interclasse irá perder poucos objetos daquela classe, mas em compensação irá possuir uma grande taxa de falsos positivos devido a complexidade das regiões de fundo.

- Em quarto lugar, a cor e iluminação das imagens variam consideravelmente, podendo-se citar a incidência direta do sol em um pedestre contra a sombra criada por uma nuvem. Apesar de modelos computacionais invariantes a cor e iluminação terem avançado significantemente, eles estão longe do desempenho do sistema visual humano. Logo, um detector robusto deve lidar com mudanças na cor e iluminação das imagens.

- Em último lugar, a existência de oclusões parciais ou totais do pedestre cria dificuldades para a localização do mesmo na imagem.

A Figura 3 mostra alguns exemplos ilustrando as dificuldades em relação a detecção de pedestres. Nota-se a grande variação entre as imagens, em particular a grande variação

das poses das pessoas, a variação na iluminação, a diferença entre as vestimentas e a quantidade de fundos complexos.

Figura 3 – Exemplos de imagens com pedestres.

1.4 Estado da Arte

Esta seção revisa detectores de pedestre com foco nas abordagens por janelas deslizantes. Esta abordagem é mais promissora para imagens de baixa a média resolução, na qual abordagens baseadas em segmentação (GU et al., 2009) e detecção de *keypoints* (LEIBE; SEEMANN; SCHIELE, 2005; SEEMANN et al., 2005) geralmente não apresentam bons resultados (DOLLAR et al., 2012).

1.4.1 Proposta de Papageorgiou e Poggio (2000)

Um dos primeiros sistemas baseado em janelas deslizantes foi proposto por Papageorgiou e Poggio (2000). Na fase de treinamento, o sistema recebe como entradas um conjunto de imagens de pedestres alinhadas e redimensionadas para o tamanho da janela de varredura, e um conjunto de imagens com padrões de fundo. Uma representação intermediária que capta informações importantes da classe a se detectar é calculada para cada uma dessas imagens, resultando em vetores de características. Esses vetores são

usados para treinar um classificador binário, capaz de fazer a classificação entre pedestres e fundo. Na fase de varredura, o sistema desliza uma janela de tamanho fixo pela imagem, extraindo o mesmo conjunto de características e fornecendo-as ao classificador, que realiza a decisão entre pedestre/não pedestre naquela região da imagem. Para realizar a detecção em múltiplas escalas, as imagens de teste são redimensionadas em vários escalas, e para cada escala da imagem é varrida uma janela de tamanho fixo.

A representação intermediária usada é a transformada de *Haar wavelet*, que identifica diferenças de intensidade locais e orientadas em diferentes escalas, e é eficiente computacionalmente. A transformada *Haar wavelet* transforma as imagens do espaço da intensidade para o espaço dos coeficientes da *wavelet*, resultando em um dicionário supercompleto de características que é usado para o treinamento do classificador. A principal motivação do uso de *wavelets* é que elas capturam características visualmente plausíveis do contorno e estrutura interna de objetos. O resultado é uma representação compacta onde exemplos de imagens dissimilares da mesma classe de objeto são mapeados para vetores similares.

A técnica usada para aprender e tomar decisões é uma máquina de vetores suporte (SVM). O algoritmo SVM acha uma superfície de separação, dentre todas as superfícies, que maximiza a distância entre os elementos mais próximos das duas classes. Na prática, isso é determinado resolvendo-se um problema de otimização quadrática.

Este sistema foi treinado em uma base de dados particular (não disponível publicamente). Não há dados sobre o desempenho deste sistema no conjunto de teste da base de dados INRIA.

1.4.2 Proposta de Viola e Jones (2004)

Seguindo as ideias de Papageorgiou e Poggio (2000), Viola e Jones (2004) introduziram duas novas técnicas no seu sistema: imagens integrais e *AdaBoost*.

As características usadas fazem lembrar as funções de base Haar, usadas por Papageorgiou e Poggio (2000). Mais especificamente, as características usadas correspondem às diferenças da soma de pixels presentes em retângulos distribuídos na imagem. Três características são usadas: a diferença entre dois retângulos, a diferença entre três retângulos e a diferença entre quatro retângulos. Essas características são calculadas rapidamente com o uso da técnica de imagens integrais.

AdaBoost (*Adaptive Boosting*) é um algoritmo de classificação cujo o conceito básico é utilizar classificadores fracos para formar um classificador robusto. A saída dos classificadores fracos é combinada em uma soma ponderada que representa a saída final do classificador.

Este sistema foi treinado na base de dados INRIA *Person Database*, alcançando

uma velocidade de varredura de 0,447 imagens por segundo e uma média logarítmica da taxa de erro igual a 72% no conjunto de teste da base de dados INRIA.

1.4.3 Proposta de Dalal e Triggs (2005)

Grandes ganhos foram obtidos com a adoção de características baseadas em gradientes. Inspirados pelo SIFT (*Scale Invariant Feature Transform*), Dalal e Triggs (2005) popularizaram o uso de histogramas de gradientes orientados (*Histogram of Oiented Gradients*) como características, mostrando ganhos substanciais em relação a características baseadas na intensidade dos pixels. Dollar et al. (2012) cita que nenhuma característica sozinha consegue superar o desempenho do HOG, mas características adicionais podem melhorar o desempenho do sistema.

O descritor HOG é baseado na avaliação de histogramas locais normalizados dos gradientes orientados da imagem. O vetor de características é formado pela concatenação dos histogramas normalizados dos gradientes orientados presentes em uma janela de detecção. A tarefa de decisão entre pedestre/não pedestre de cada janela é deixada a cargo de um SVM linear.

Este sistema foi treinado na base de dados INRIA *Person Database*, alcançando uma velocidade de varredura de 0,239 imagens por segundo e uma média logarítmica da taxa de erro igual a 46% no conjunto de teste da base de dados INRIA.

1.4.4 Proposta de Wojek e Schiele (2008)

Além de fazerem uma análise do estado da arte no que diz respeito a detecção de pedestres, testando diferentes técnicas de extração de características em conjunto com diferentes técnicas de classificação, Wojek e Schiele (2008) propõem a combinação de diferentes características para a criação de um detector robusto que superasse o estado da arte atual.

Para a criação desse detector, são combinadas três técnicas de extração de características: descritores HOG (DALAL; TRIGGS, 2005), características baseadas em *Haar wavelets* (PAPAGEORGIOU; POGGIO, 2000) e características baseadas em contorno, denominadas *shape context* (BELONGIE; MALIK; PUZICHA, 2002). O descritor *shape context* é baseado em bordas que são extraídas com o auxílio de um detector Canny e armazenadas em um histograma log-polar.

O sistema é baseado em janelas deslizantes, e são testadas duas técnicas de classificação: SVM linear e AdaBoost. Todos os vetores de características encontrados em uma janela são normalizados e concatenados em um único vetor, que é então fornecido à técnica de classificação para treinamento e posterior classificação das janelas em pedes-

tre/não pedestre. Também é aplicado ao sistema uma etapa de retreinamento para melhor desempenho do classificador, como mostrado em (DALAL; TRIGGS, 2005).

Este sistema foi treinado na base de dados INRIA *Person Database*, alcançando uma velocidade de varredura de 0,072 imagens por segundo e uma média logarítmica da taxa de erro igual a 36% no conjunto de teste da base de dados INRIA.

1.4.5 Proposta de Wu e Nevatia (2008)

O sistema proposto por Wu e Nevatia (2008) é baseado em um classificador em cascata (VIOLA; JONES, 2001). Nesta abordagem, cada classificador fraco corresponde a uma janela da imagem, e várias características são extraídas dessa janela. A função de classificação para cada tipo de característica é aprendida no seu próprio espaço de características, e o classificador fraco faz uma predição entre pedestre/não pedestre examinando os diferentes tipos de características uma por uma. Quando a predição baseada numa característica extraída não possui uma confiança alta o bastante, o classificador fraco olha para a próxima característica. A ordem em que as características são avaliadas é determinada por uma medida de poder de classificação que inclui o custo computacional de extração da característica. Um número de classificadores fracos é então selecionado e combinado para a formação do classificador em cascata.

As características usadas são: descritores HOG, descritores *edgeles* (WU; NEVATIA, 2005) e descritores de covariância (TUZEL; PORIKLI; MEER, 2007).

Este sistema foi treinado na base de dados MIT (disponível em (MIT, 2014)). Não há dados sobre o desempenho deste sistema no conjunto de teste da base de dados INRIA.

1.4.6 Proposta de Wang, Han e Yan (2009)

O sistema proposto por Wang, Han e Yan (2009) combina informações de borda/contorno local com informações de textura para a criação de um detector mais eficaz. São usados descritores HOG e descritores LBP (*Local Binary Pattern*) estruturados em células (AHONEN; HADID; PIETIKAINEN, 2006).

Dois tipos de detectores são usados: detectores globais que levam em conta a janela de detecção inteira e detectores por parte usados em regiões locais da imagem. Ambos são treinados usando um SVM linear.

Inicialmente, o detector global é usado na janela de detecção. Em cada janela ambígua (*scores* do SVM entre [-1, 1]) são construídos mapas de probabilidade de oclusão usando as respostas de cada bloco do descritor HOG dadas pelo detector global. Cada mapa de oclusão é considerado como uma imagem, e tais imagens são segmentadas usando a técnica *mean shift*. Ao resultado da segmentação que não é considerado oclusão é aplicado

o detector por partes, gerando assim o resultado final da detecção. As janelas não ambíguas têm sua classificação final dada pela resposta do classificador global.

Este sistema foi treinado na base de dados INRIA *Person Database*, alcançando uma velocidade de varredura de 0,062 imagens por segundo e uma média logarítmica da taxa de erro igual a 39% no conjunto de teste da base de dados INRIA.

1.4.7 Proposta de Walk et al. (2010)

Este trabalho introduz uma novo descritor baseado na auto similaridade de histogramas de cor advindos de diferentes sub-regiões dentro da janela de detecção, chamado CSS (*Color Self Similarity*). Este descritor captura características de pares de distribuições de cor espacialmente localizados, sendo assim independente da distribuição de cor em si. Segundo Walk et al. (2010), a adição do CSS aumenta significantemente o desempenho de classificadores do estado da arte, tanto para imagens estáticas como para sequências de imagem. O trabalho relata ainda que características de movimento derivadas do fluxo ótico podem trazer benefícios a detecção do pedestre.

Outra contribuição relatada é a de que o número de retreinamentos possui uma drástica influência no resultado final do sistema.

Este sistema foi treinado na base de dados TUD-Brussels (disponível em (WOJEK STEFAN WALK, 2014)), alcançando uma velocidade de varredura de 0,027 imagens por segundo e uma média logarítmica da taxa de erro igual a 25% no conjunto de teste da base de dados INRIA.

1.4.8 *Benchmark* criado por Dollar et al. (2012)

No seu trabalho, Dollar et al. (2012) criou um extenso *benchmark* de detectores de pedestres e comparou os sistemas do estado da arte atual, entre os quais se encontram aqueles citados anteriormente e mais alguns outros. Vários banco de dados foram usados, inclusive o INRIA Person Database, usado nesta dissertação. Tal trabalho se tornou uma referência no que diz respeito ao estado da arte na área de detecção de pedestres. Os resultados obtidos aqui nesta dissertação são comparados aos resultados obtidos no *benchmark*, disponibilizados em (DOLLAR, 2014).

1.5 Motivação da escolha das técnicas

O sistema aqui proposto tem como base dois blocos funcionais, executando as funções de extração de características e classificação. Duas técnicas de extração de características são usadas, sendo elas: HOG (*Histogram of Oriented Gradient*) e CSS (*Color Self Similarities*), e para classificar as janelas é usado o SVM (*Support Vector Machine*) linear.

Além dessas técnicas, são também utilizadas: *mean shift* e agrupamento hierárquico, para a fusão de múltiplas detecções sobrepostas; e filtro bilateral, para pré-processamento da imagem.

O descritor HOG foi escolhido devido ao desempenho alcançado, se comparado com outras técnicas. Dollar et al. (2012) mencionam que nenhuma característica sozinha consegue superar o desempenho de tal descritor. Por sua vez, o descritor CSS foi escolhido para complementar o sistema, pois foi mostrado por Walk et al. (2010) que sua aplicação aumenta o desempenho do sistema usando múltiplas características.

Duas técnicas se sobressaem no que diz respeito a classificadores em sistemas de detecção de pedestres: SVM e AdaBoost. O SVM linear foi escolhido por ser um classificador robusto e rápido, capaz de usar um grande conjunto de amostras com facilidade.

O problema da fusão de múltiplas detecções, que consiste na fusão de detecções envolvendo o mesmo pedestre, é resolvido adequadamente pela técnica *mean shift* (YU; YAO, 2009; COMANICIU; MEER, 2002; FUKUNAGA; HOSTETLER, 1975), considerando tanto a posição como a escala das detecções. Abordagens de fusão heurística, como o proposto por Viola e Jones (2004), consideram detecções sobrepostas como sendo uma só para o mesmo pedestre, não levando em conta a escala dos mesmos.

Para identificar os *clusters* formados pelo *mean shift* e acelerar o processo de fusão das múltiplas detecções sobrepostas, foi proposta neste trabalho a utilização da técnica de agrupamento hierárquico (THEODORIDIS; KOUTROUMBAS, 2008), por ser rápida, de simples implementação, e não precisar da informação sobre o número de *clusters* a priori. Tal problema é muitas vezes relevado em outros trabalhos (DOLLAR et al., 2012; WALK et al., 2010). Técnicas de agrupamento como *k-means* não são indicadas, pois as mesmas necessitam saber o número de *clusters* formados a priori.

O uso do filtro bilateral supõe que tanto o descritor HOG como o CSS podem se beneficiar de uma imagem que contenha menos informação de textura, preservando as bordas entre pedestre e fundo. Seu emprego foi considerado, pois o descritor HOG tem como principal função descrever as bordas entre pedestre e fundo, através do gradiente da imagem, e texturas muito fortes podem gerar gradientes que sejam confundidos com bordas. Já o descritor CSS se baseia na semelhança de cores entre diferentes partes do pedestre, e texturas mais uniformes, através da suavização, podem aumentar essa semelhança.

Também foi observado neste trabalho que o corte exato do tamanho das detecções finais gera um melhor resultado. O corte realizado de forma correta aumenta a área de sobreposição entre detecções e anotações, e com isso aumenta-se a precisão na localização dos pedestres.

1.6 Modificações propostas

Foram propostas algumas modificações que serão citadas aqui rapidamente, com o intuito de exemplificar o trabalho feito. Todas essas propostas serão explicadas detalhadamente no decorrer do texto.

- A primeira proposta de modificação consiste em mudar o método de cálculo do gradiente da imagem na extração do descritor HOG.
- A segunda proposta consiste em fazer a normalização local dos histogramas durante o cálculo do descritor CSS.
- A terceira proposta consiste em mudar a métrica de distância usada para calcular a distância entre histogramas na extração do descritor CSS.
- A quarta proposta consiste em utilizar a técnica de agrupamento hierárquico após o *mean shift* para identificar os *clusters* formados e acelerar o processo de fusão de múltiplas detecções.
- A quinta proposta consiste no uso da filtragem bilateral como modo de pré-processamento das imagens antes da extração de características.
- A sexta proposta consiste no corte exato das detecções finais para melhor adaptação às anotações modificadas.
- A sétima e última proposta consiste em usar classificadores separados para os descritores HOG e CSS, fazendo com que a dimensão final do descritor não fique muito elevada.

1.7 Termos usados no decorrer do trabalho

Nesta seção serão definidos alguns termos usados no decorrer deste trabalho.

- Janela deslizante ou janela de detecção: Região espacial de tamanho fixo (128×64 pixels) que é varrida em uma imagem, a uma taxa de deslocamento constante (8 pixels) tanto na horizontal como na vertical. Em cada janela ocorre a extração de características e a classificação entre pedestre/não pedestre.
- Anotações: Marcações feitas no banco de dados por um usuário que indicam a presença de um pedestre através de uma caixa delimitadora retangular.
- Varredura: Ato de deslocar a janela de detecção pela imagem, com o intuito de detectar a presença de um pedestres em qualquer posição da imagem.

- Descritores: Vetores de características usados pelo classificador para decidir entre a presença ou não de pedestres em uma janela de detecção.
- Célula: Sub região espacial (8×8 pixels) usada para o cálculo dos histogramas nos descritores HOG e CSS.
- Corte das detecções: Redução do número de pixels de uma detecção, sendo que os mesmo são jogados fora, ou seja, não é feito nenhum processo de subamostragem.

1.8 Estrutura da Dissertação

O resto deste trabalho está dividido da seguinte forma: O Capítulo 2 apresenta as técnicas de extração de características utilizadas. O Capítulo 3 apresenta as técnicas de classificação usadas. O Capítulo 4 apresenta as técnicas usadas para fundir as múltiplas detecções sobrepostas e as técnicas para o pré processamento das imagens. O Capitulo 5 apresenta o banco de dados usado e a metodologia de detecção de pedestres. O Capítulo 6 apresenta os resultados do sistema de detecção de pedestres proposto. O Capítulo 7 apresenta as conclusões do trabalho e ideias para trabalhos futuros.

2 Extração de Características

Neste capítulo serão descritas as técnicas de extração de características usadas no sistema de detecção de pedestres.

2.1 HOG (*Histogram of Oriented Gradients*)

Esta seção descreve a técnica de extração de características denominada HOG (*Histogram of Oriented Gradients*), proposta por Dalal e Triggs (2005). Histogramas de gradientes orientados têm como base o trabalho de Lowe (2004), denominado SIFT (*Scale Invariant Feature Transform*). Porém, ao contrário do SIFT, que calcula os histogramas de gradientes em volta de *keypoints* invariáveis a escala, o descritor HOG calcula os histogramas de gradientes em uma densa e sobreposta grade de células uniformemente espaçadas na imagem.

Neste trabalho, o descritor HOG foi calculado conforme os seguintes passos:

- O primeiro estágio consiste em calcular os gradientes de primeira ordem da imagem. O cálculo dos gradientes captura contornos, silhuetas e um pouco de textura da imagem. As máscaras espaciais usadas para este cálculo são dadas por: $[-1 \;\; 0 \;\; 1]$ para o cálculo do gradiente na horizontal e $[-1 \;\; 0 \;\; 1]^T$ para o cálculo do gradiente na vertical. O gradiente é calculado em cada canal de cor do modelo RGB, e o gradiente final, para cada pixel, é o gradiente com a maior magnitude entre os canais de cor.

- O segundo estágio consiste na criação dos histogramas de orientação do gradiente, que tem como objetivo produzir uma codificação que seja sensível ao conteúdo local da imagem e ao mesmo tempo resistente a pequenas variações de pose e aparência. A janela de detecção (128×64 pixels) é dividida em pequenas regiões espaciais de 8x8 pixels, denominadas células. Cada célula então é formada por 64 pixels, e cada um desses pixels é usado para criar um histograma da orientação dos gradientes, sendo um histograma por célula. Cada histograma possui 9 divisões (*bins*), igualmente espaçados entre 0 e 180 graus. A contribuição de cada pixel para os histogramas é ponderada pelo valor da magnitude do gradiente na posição do pixel. Para evitar o efeito de *aliasing*, os pesos de cada pixel para os histogramas são interpolados bilinearmente no espaço e linearmente na orientação, ou seja, cada pixel contribui para os 8 *bins* de histograma que o envolvem no espaço 3-D. Para maiores informações sobre cálculo de histogramas com interpolação, consultar o Apêndice A.

- O terceiro estágio consiste na normalização local dos histogramas em regiões espaciais

da imagem denominadas blocos, formados por um conjunto de células. A normalização introduz um aumento na invariância à iluminação, sombreamento e contraste dos contornos. Cada bloco é formado por um conjunto de 2×2 células, ou 16×16 pixels. Antes da formação dos histogramas nas células, uma máscara gaussiana do tamanho do bloco (16×16) é aplicada sobre a magnitude do gradiente dos pixels do bloco. Essa máscara possui desvio padrão σ igual a metade da largura do bloco, ou seja, $\sigma = 8$, e possui o intuito de diminuir o peso dos pixels que se encontram na borda do bloco. Isso é feito porque os pixels da borda são os que possuem a maior chance de passarem para outro bloco em qualquer outra amostra. Os blocos possuem uma sobreposição de 8 pixels entre eles, fazendo com que cada célula pertença a 4 blocos diferentes, com a exceção das células das bordas e dos cantos, que pertencem, respectivamente, a dois blocos e a um bloco. Isso faz com que cada célula seja normalizada em relação a diferentes blocos, aparecendo assim mais de uma vez no descritor final. Neste trabalho foram testados quatro tipos de normalização. Seja $\mathbf{v}$ o descritor não normalizado, $\|\mathbf{v}\|_k$ sua k-ésima norma, e ϵ uma constante pequena. As normalizações são:

  - normalização L2; $\mathbf{v} \rightarrow \mathbf{v}/\sqrt{\|\mathbf{v}\|_2^2 + \epsilon^2}$,
  - normalizaçao L2-Hys (usada em (LOWE, 2004)); normalização L2, seguida de limitação dos valores de $\mathbf{v}$ em 0,2 e renormalização L2,
  - normalizaçao L1; $\mathbf{v} \rightarrow \mathbf{v}/(\|\mathbf{v}\|_1 + \epsilon)$,
  - normalizaçao L1-Sqrt; raiz quadrada da normalização L1, $\mathbf{v} \rightarrow \sqrt{\mathbf{v}/(\|\mathbf{v}\|_1 + \epsilon)}$,

  onde ϵ é um valor pequeno usado para evitar uma possível divisão por zero em locais da imagem onde não existe a variação da orientação do gradiente.

- O passo final consiste em concatenar todos os blocos presentes na janela de detecção, após a normalização, num único vetor, criando assim o descritor final. Na janela de detecção usada nesse sistema, de tamanho 128x64 pixels, existem 15x7 blocos. Cada bloco é formado por 2x2 células, que por sua vez possuem um histograma de 9 *bins*. Logo, o descritor final possui uma dimensão igual a $15 \times 7 \times 2 \times 2 \times 9 = 3780$.

As principais vantagens do descritor HOG são: a captura de informações de contorno locais através da codificação da orientação do gradiente em histogramas, redução da variância espacial através do acúmulo local desses histogramas sobre regiões da imagem e uma redução da variância à iluminação através da normalização local desses histogramas.

A ideia básica por traz da utilização do descritor HOG é a de que o contorno e aparência locais de objetos pode muitas vezes ser bem caracterizada pela distribuição local das orientações do gradiente da imagem, mesmo sem o conhecimento exato da posição do gradiente. Na prática, isto é implementado dividindo a imagem em pequenas regiões

espaciais, denominadas células, e acumulando, para cada célula, um histograma 1-D de orientações do gradiente de cada pixel presente na célula. A combinação dos histogramas forma a representação dos contornos e aparências locais. Para diminuir a variância à iluminação e sombreamento, é útil que se faça uma normalização local desses histogramas. Essa normalização é feita acumulando uma medida da energia local dos histogramas sobre regiões maiores, denominadas blocos, formadas por uma quantidade específica de células, e usando essa medida para normalizar as células dentro de um determinado bloco. O descritor HOG é formado pela concatenação dos diversos blocos presentes em uma janela de detecção varrida sobre a imagem. Foi mostrado em Dalal e Triggs (2005) que a sobreposição dos blocos, fazendo com que a mesma célula seja normalizada em relação a mais de um bloco, aumenta o desempenho do descritor, mas traz a desvantagem do aumento da dimensão do mesmo. O fluxograma exemplificando o cálculo do descritor pode ser visto na Figura 4.

Um exemplo do cálculo do descritor HOG original pode ser visto na Figura 5.

2.1.1 Modificações propostas no descritor HOG

Foi proposto neste trabalho fazer o cálculo dos gradientes tratando cada pixel como um vetor com componentes R, G e B, como explicado em (GONZALEZ; WOODS; EDDINS, 2009). Para tal cálculo, primeiro se define os vetores

$$\mathbf{u} = \frac{\partial R}{\partial x}\mathbf{r} + \frac{\partial G}{\partial x}\mathbf{g} + \frac{\partial B}{\partial x}\mathbf{b} \tag{2.1}$$

e

$$\mathbf{v} = \frac{\partial R}{\partial y}\mathbf{r} + \frac{\partial G}{\partial y}\mathbf{g} + \frac{\partial B}{\partial y}\mathbf{b}, \tag{2.2}$$

sendo que $\mathbf{r}$, $\mathbf{g}$ e $\mathbf{b}$ são vetores unitários ao longo dos eixos R, G e B, e os valores x e y indicam as coordenadas espaciais da imagem. As derivadas parciais das equações (2.1) e (2.2) são calculadas com as máscaras $[-1 \quad 0 \quad 1]$ e $[-1 \quad 0 \quad 1]^T$, iguais as usadas no cálculo do gradiente no método original. Através do produto escalar dos vetores $\mathbf{u}$ e $\mathbf{v}$, se calculam as quantidades

$$g_{xx} = \mathbf{u}.\mathbf{u} = \mathbf{u}^\mathbf{t}\mathbf{u} = \left|\frac{\partial R}{\partial x}\right|^2 + \left|\frac{\partial G}{\partial x}\right|^2 + \left|\frac{\partial B}{\partial x}\right|^2, \tag{2.3}$$

$$g_{yy} = \mathbf{v}.\mathbf{v} = \mathbf{v}^\mathbf{t}\mathbf{v} = \left|\frac{\partial R}{\partial y}\right|^2 + \left|\frac{\partial G}{\partial y}\right|^2 + \left|\frac{\partial B}{\partial y}\right|^2, \tag{2.4}$$

$$g_{xy} = \mathbf{u}.\mathbf{v} = \mathbf{u}^\mathbf{t}\mathbf{v} = \frac{\partial R}{\partial x}\frac{\partial R}{\partial y} + \frac{\partial G}{\partial x}\frac{\partial G}{\partial y} + \frac{\partial B}{\partial x}\frac{\partial B}{\partial y}. \tag{2.5}$$

Utilizando esta notação, pode-se demonstrar (ZENZO, 1986) que a direção da taxa máxima de variação no ponto (x, y) é dada pelo ângulo

$$\theta(x, y) = \frac{1}{2}tg^{-1}\left[\frac{2g_{xy}}{g_{xx} - g_{yy}}\right] \tag{2.6}$$

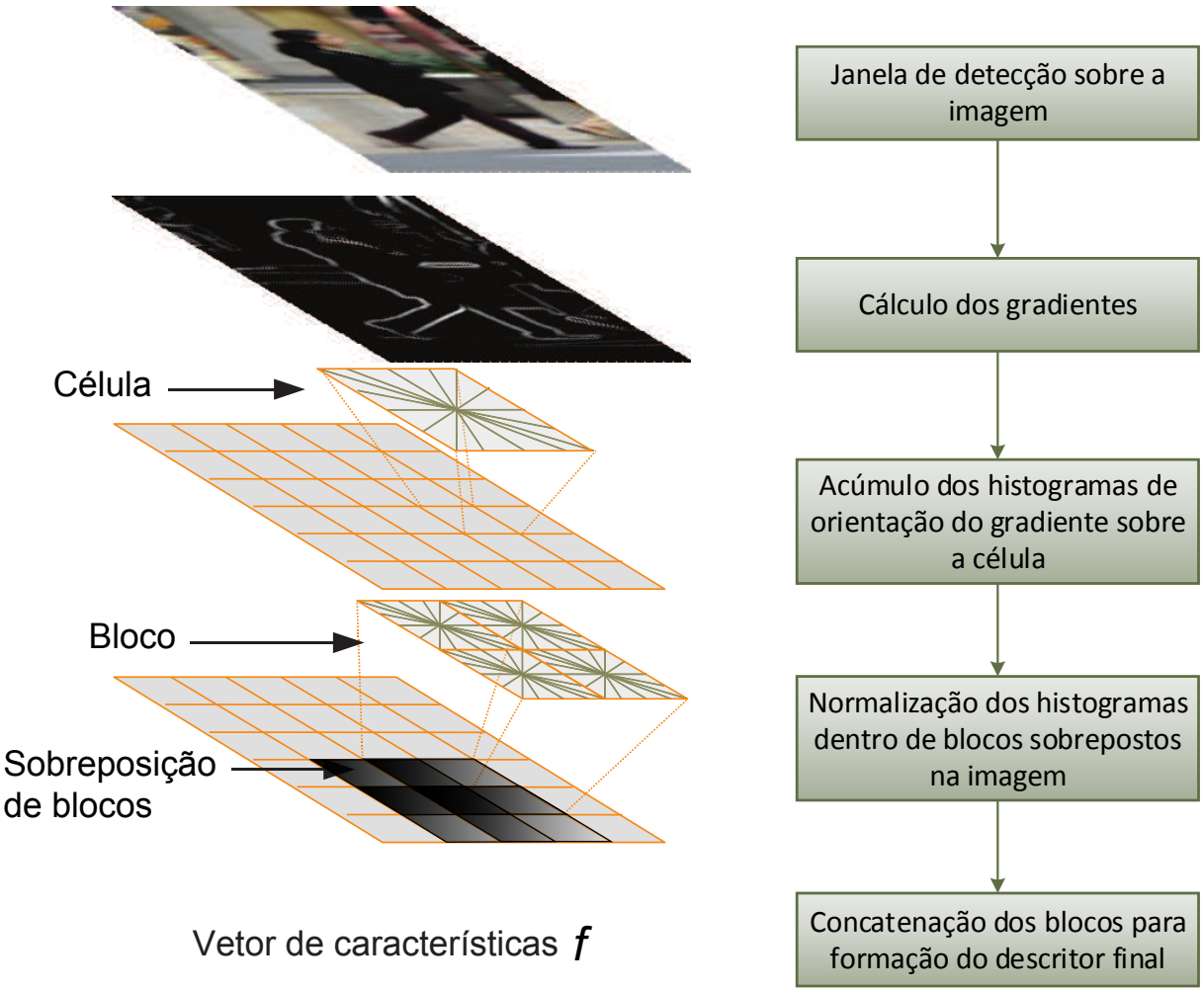


Figura 4 – Uma visão global do cálculo do descritor HOG. A janela de detecção é preenchida com uma grade de blocos sobrepostos. Cada bloco é formado por um conjunto de células, nas quais são calculados histogramas da orientação do gradiente de cada pixel presente nas mesmas. Os histogramas são normalizados localmente em relação aos blocos e concatenados para a criação do vetor de características (DALAL, 2006).

e que o valor da taxa de variação no mesmo ponto é

$$F_\theta(x,y) = \left\{ \frac{1}{2} \left[(g_{xx} + g_{yy}) + (g_{xx} - g_{yy}) + cos2\theta(x,y) + 2g_{xy} sen2\theta(x,y) \right] \right\}^{\frac{1}{2}} \tag{2.7}$$

Tendo em vista que $tg(\theta) = tg(\theta \pm \pi)$, se θ_0 for uma solução para a Equação (2.6), o mesmo se aplica a $\theta_0 \pm \frac{\pi}{2}$. Além disso, $F_\theta = F_{\theta+\pi}$, de forma que F deve ser calculado apenas para valores de θ no intervalo semiaberto [0,π[. O fato de a Equação (2.6) proporcionar dois valores com 90 graus de diferença significa que essa equação associa a cada ponto (x,y) um par de direções ortogonais. Em uma dessas direções se tem o F máximo, e na outra direção o mínimo. Logo deve-se calcular o valor de F para as duas direções e através do resultado determinar a direção da máxima taxa de variação. A comparação dos resultados usando este método e o método original podem ser vistos no Capítulo 6.

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

Figura 5 – Cálculo do descritor HOG sobre uma janela de detecção. (a) - Janela de detecção de 128×64 pixels, (b) - magnitude do gradiente, (c) - orientação dos gradientes da imagem, (d) visualização dos histogramas de cada célula.

2.2 CSS (*Color Self Similarities*)

Esta seção descreve a técnica de extração de características denominada CSS(*Color Self Similarities*), proposta em (WALK et al., 2010). Este descritor é baseado em auto similaridades de características de baixo nível, sendo tais características os histogramas de cor de diferentes regiões espaciais em uma janela de detecção sobreposta à imagem. O CSS captura estatísticas de distribuição de cor entre pares de regiões espaciais locais, sendo assim independente da cor atual de um exemplo específico. A auto similaridade consegue representar propriedades como a semelhança da distribuição de cores em diferentes partes do corpo como, por exemplo, o ombro esquerdo e direito de uma pessoa, sem se importar com a atual distribuição de cor, que pode variar de pessoa para pessoa. O fluxograma exemplificando o cálculo do descritor pode ser visto na Figura 6.

Neste trabalho, o descritor CSS foi calculado conforme os seguintes passos.

- O primeiro estágio consiste na criação dos histogramas locais de cor. O modelo de cor usado no cálculo deste descritor é o modelo HSV (*Hue, Saturation, Value*). Os histogramas de cor são os objetos que traduzem os padrões de cor locais da imagem. Para a criação dos histogramas locais, a janela de detecção é dividida em pequenas regiões espaciais de 8x8 pixels, denominadas células, logo cada célula é formada por 64 pixels. Cada pixel presente na célula, com determinados valores de H, S e V, é usado para a criação de um histograma 3-D de cor. Cada dimensão do histograma possui 3 divisões (*bins*), igualmente espaçados entre 0 e 1, formando assim um histograma de $3^3 = 27$ *bins* por célula. Para evitar o efeito de *aliasing*, os pesos de cada pixel para os histogramas são interpolados bilinearmente no espaço

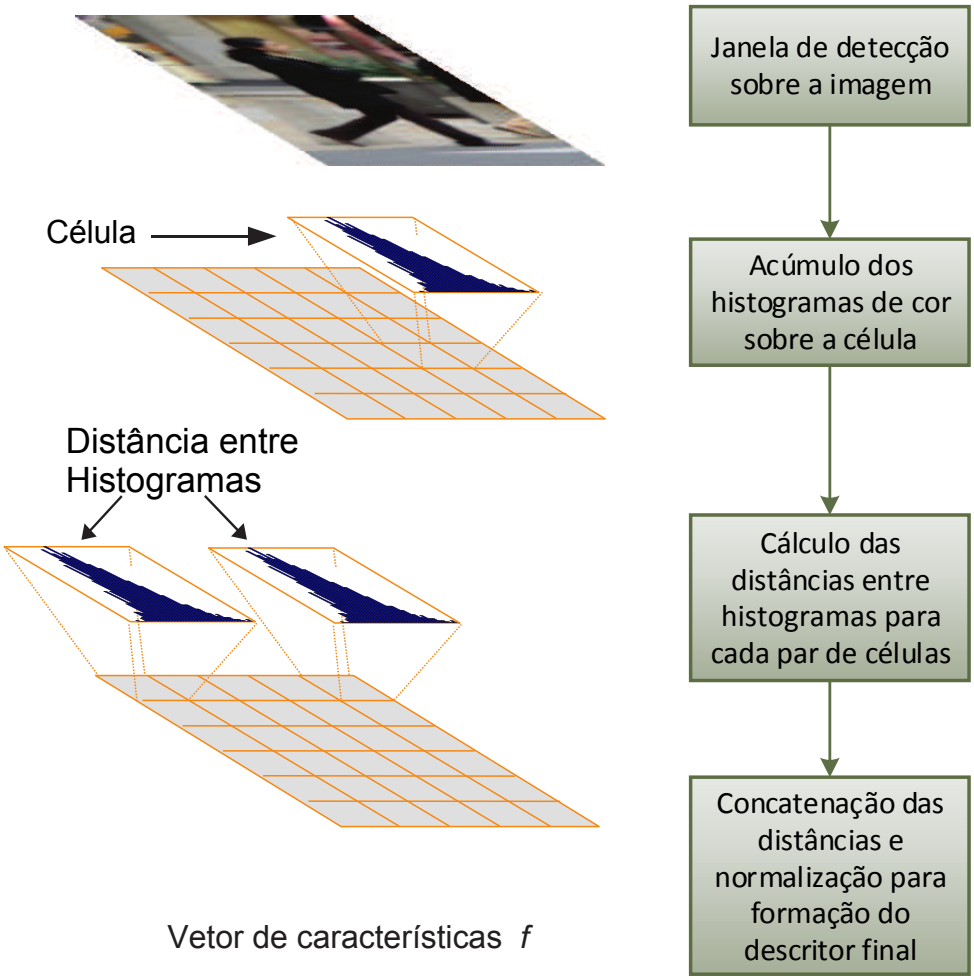


Figura 6 – Uma visão global do cálculo do descritor CSS. A janela de detecção é preenchida com uma grade de células. Em cada célula são calculados histogramas de cor. O conceito da auto similaridade de cores é criado a partir do cálculo das distâncias entre histogramas.

e trilinearmente no espaço de cor, ou seja, cada pixel contribui para os 32 *bins* de histograma que o envolvem no espaço 5-D. Para maiores informações sobre cálculo de histogramas com interpolação, consultar o Apêndice A.

- O segundo estágio consiste no cálculo da distância entre pares de histogramas de cor locais. Este passo é o responsável por criar as medidas de auto similaridade de cores, fundamento do descritor CSS. Quanto menor a distância entre dois histogramas locais, maior é a semelhança entre os padrões de cores dessas duas células. A medida de distância usada para o cálculo da distância entre os histogramas é a interseção de histogramas, dada pela equação

$$K_{\cap}(a,b) = \sum_{i=1}^{n} min(a_i, b_i). \tag{2.8}$$

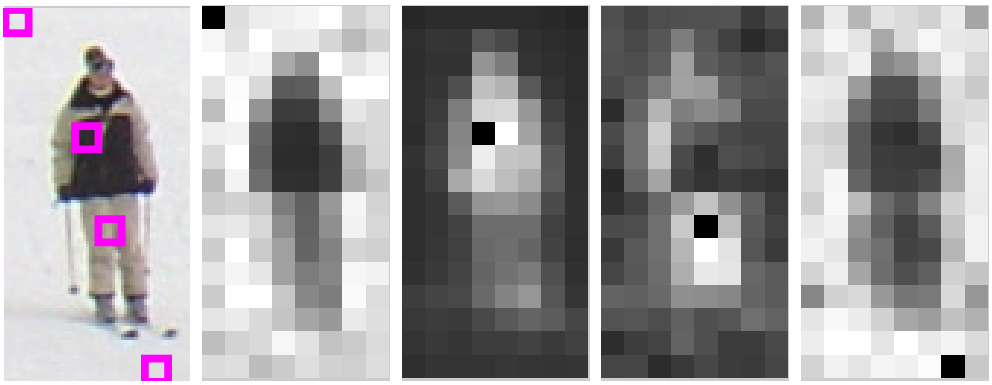
Essa distância é calculada para todos os pares de histogramas. Em uma janela de detecção de tamanho 128x64 pixels existem $16 \times 8 = 128$ células de 8x8 pixels. Logo,

a quantidade de distâncias calculadas é igua a $(128 \times (128-1)/2) = 8.128$. Esse é o tamanho do vetor de características final.

- O passo final consiste na concatenação de todas as distâncias calculadas na janela de detecção, e a normalização das mesmas. As distâncias são normalizadas em relação à janela de detecção inteira, e as distâncias normalizadas formam o descritor final. Três tipos de normalização foram testados. Seja $\mathbf{v}$ o descritor não normalizado, $\|\mathbf{v}\|_k$ sua k-ésima norma, e ϵ uma constante pequena. As normalizações são:
  - normalização L2; $\mathbf{v} \rightarrow \mathbf{v}/\sqrt{\|\mathbf{v}\|_2^2 + \epsilon^2}$,
  - normalizaçao L1; $\mathbf{v} \rightarrow \mathbf{v}/(\|\mathbf{v}\|_1 + \epsilon)$,
  - normalizaçao L1-Sqrt; raiz quadrada da normalização L1, $\mathbf{v} \rightarrow \sqrt{\mathbf{v}/(\|\mathbf{v}\|_1 + \epsilon)}$,

  onde ϵ é um valor pequeno usado para evitar uma possível divisão por zero em locais da imagem onde os histogramas de cor sejam iguais.

Um exemplo da auto similaridade de cores pode ser visto na Figura 7.



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

Figura 7 – Exemplo da auto similaridade de cores. (a) - Janela de detecção, (b) (c) (d) (e) - cada uma dessa imagens mostra a distâncias entre as células marcadas e todas as outras células, onde os níveis mais claros indicam maior semelhança.

2.2.1 Modificações propostas no descritor CSS

Foi proposto neste trabalho a normalização local dos histogramas, seguindo a mesmo procedimento usado por Dalal e Triggs (2005) no cálculo do descritor HOG. Antes do cálculo das distâncias entre histogramas, a imagem é dividida em blocos de 2x2 células com sobreposição de 8 pixels entre eles, assim cada célula pertence a 4 blocos diferentes. As células então têm seus histogramas normalizados em relação aos blocos compostos por ela e mais três células adjacentes. Apesar de cada célula ter seu histograma normalizado em relação a 4 blocos diferentes, apenas um valor será usado para o cálculo das distâncias,

diferentemente do descritor HOG, onde os 4 valores são usados na formação do descritor final. O valor usado neste trabalho é dado pela média aritmética das 4 normalizações. Isto foi feito para evitar o aumento do tamanho do vetor de características final. Se forem usados os quatro valores das normalizações, existirão $4 \times 128 = 512$ valores de histogramas na janela de detecção, levando a um cálculo de $(512 \times (512 - 1)/2) = 130.816$ distâncias entre histogramas. Os tipos de normalização testados aqui foram os mesmo testados na normalização por janela da Seção 2.2.

Foi proposto também a mudança da métrica de distância usada no cálculo das distâncias entre histogramas. Como visto anteriormente, a métrica original usada por Walk et al. (2010) é a interseção de histogramas. Dados dois vetores x_s e x_t de tamanho $1 \times n$, representando dois histogramas diferentes, as métricas testadas neste trabalho foram:

- Distância Euclidiana:
$$d = \sqrt{(x_s - x_t)(x_s - x_t)^T}. \tag{2.9}$$

- Distância de Mahalanobis:
$$d = \sqrt{(x_s - x_t)C^{-1}(x_s - x_t)^T}, \tag{2.10}$$
onde C é a matriz de covariância entre x_s e x_t.

- Distância *city block*:
$$d = \sum_{j=1}^{n} |x_{sj} - x_{tj}|. \tag{2.11}$$

- Distância Chebychev:
$$d = \max_j(|x_{sj} - x_{tj}|). \tag{2.12}$$

- Distância cosseno:
$$d = 1 - \frac{x_s x_t^T}{\sqrt{(x_s x_s^T)(x_t x_t^T)}}. \tag{2.13}$$

- Distância correlação:
$$d_{st} = 1 - \frac{(x_s - \bar{x}_s)(x_t - \bar{x}_t)^T}{\sqrt{(x_s - \bar{x}_s)(x_s - \bar{x}_s)^T}\sqrt{(x_t - \bar{x}_t)(x_t - \bar{x}_t)^T}}, \tag{2.14}$$
onde $\bar{x}_s = \frac{1}{n}\sum_{i=1}^{j} x_{si}$ e $\bar{x}_t = \frac{1}{n}\sum_{i=1}^{j} x_{ti}$.

Os resultados da detecção de pedestres usando essas mudanças podem ser vistos no Capítulo 6.

3 Classificação

Neste capítulo serão descritas as técnicas de classificação usadas no sistema de detecção de pedestres.

3.1 SVM

O SVM (*Support Vector Machine*) (THEODORIDIS; KOUTROUMBAS, 2008) é um classificador linear, modelado independentemente da distribuição que origina os dados de treinamento. Por motivos didáticos, a modelagem do SVM será primeiramente explicada para classes linearmente separáveis. O caso de classes não linearmente separáveis (usado neste trabalho), será explicado em seguida.

3.1.1 Classes linearmente separáveis

Sejam $\mathbf{x}_i$, $i = 1, 2, \ldots, N$ os vetores da base de treinamento, pertencentes a uma das duas classes c_1, c_2. O objetivo dos classificadores lineares, incluindo o SVM, é construir um hiperplano

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = 0 \tag{3.1}$$

que classifique corretamente todos os vetores de treinamento. Porém, normalmente existem vários hiperplanos capazes de cumprir tal tarefa, como pode ser visto na Figura 8.

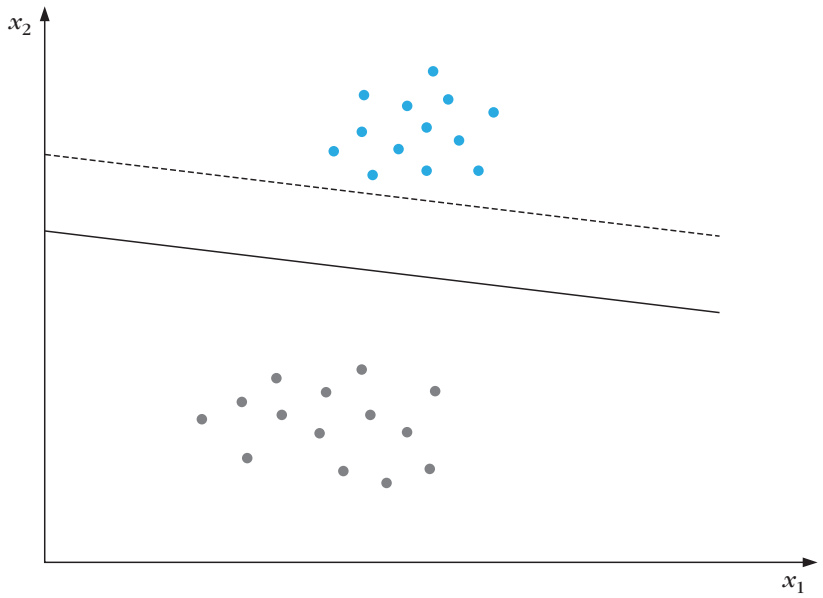


Figura 8 – Exemplo de um problema de duas classes linearmente separáveis, resolvido por dois hiperplanos diferentes. (THEODORIDIS; KOUTROUMBAS, 2008)

A escolha do hiperplano definitivo deve ser feita no sentido de aumentar a generalização do classificador, dada pela capacidade de identificar dados desconhecidos, diferentes dos dados de treinamento. Baseado nessa ideia, o hiperplano modelado pelo SVM é o hiperplano que cria a maior margem em relação as duas classes. Como não se deseja que nenhuma classe tenha preferência, é razoável que se escolha o hiperplano que tenha a mesma distância em relação aos pontos mais próximos das duas classes. Um exemplo dessa escolha pode ser visto na Figura 9. O hiperplano da direção 1 possui uma margem $2z_1$ enquanto o da direção 2 possui uma margem $2z_2$. Claramente, pode-se ver que o hiperplano na direção 2 possui uma margem maior, logo deve ser o escolhido, pois tem uma maior capacidade de generalização.

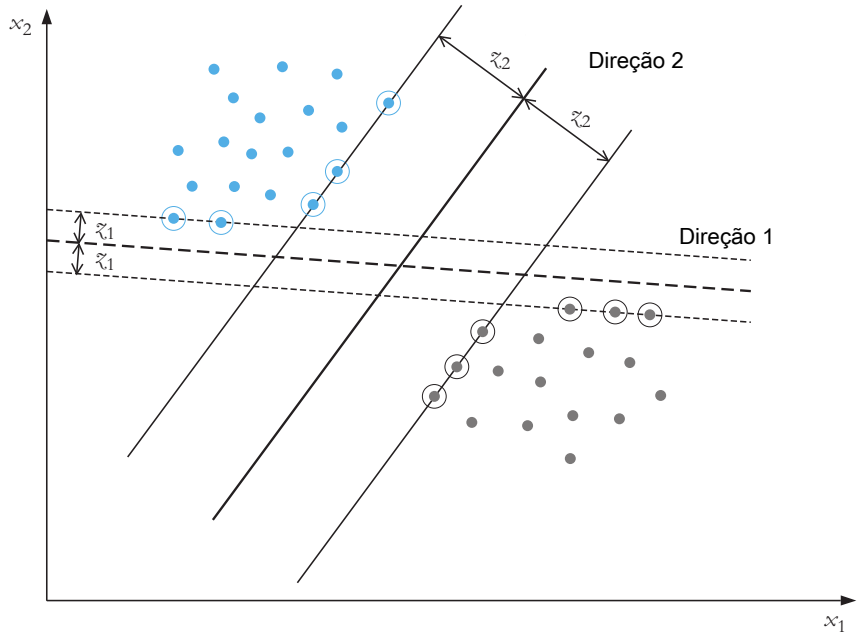


Figura 9 – Dois hiperplanos com margens iguais em relação as duas classes. (THEODORIDIS; KOUTROUMBAS, 2008)

A distância entre um ponto $\mathbf{x}$ e o hiperplano $g(\mathbf{x})$ é dada por

$$z = \frac{|g(\mathbf{x})|}{\|\mathbf{w}\|}. \tag{3.2}$$

Normalizando os hiperplanos para que o valor de $g(\mathbf{x})$ nos pontos mais próximos das classes seja 1, para a classe c_1, e -1, para a classe c_2, equivale a:

1. Possuir uma margem igual a $\frac{1}{\|\mathbf{w}\|} + \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$.
2. Satisfazer

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \geq 1 \qquad \forall\ \mathbf{x} \in c_1, \tag{3.3}$$

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \leq -1 \qquad \forall\ \mathbf{x} \in c_2. \tag{3.4}$$

Associando um y_i para cada x_i, de modo que $y_i = 1$ para $x_i \in c_1$ e $y_i = -1$ para $x_i \in c_2$, o objetivo do SVM pode ser sumarizado como escolher o hiperplano $\mathbf{w}$ que minimize a energia, sujeito à restrição de margem. Este objetivo é representado pela equação abaixo:

$$\textit{minimizar} \qquad J(\mathbf{w}, w_0) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{3.5}$$

$$\textit{sujeito a} \qquad y_i(\mathbf{w}^T\mathbf{x_i} + w_0) \geq 1,\ i = 1, 2, \ldots, N \tag{3.6}$$

onde $J(\mathbf{w}, w_0)$ é uma função de custo a ser minimizada, de modo a fazer com que a margem seja maximizada. As equações (3.5) e (3.6) formam um problema de otimização quadrática não linear sujeita a um conjunto de desigualdades, problema este que deve satisfazer as condições de Karush-Kuhn-Tucker (THEODORIDIS; KOUTROUMBAS, 2008)

$$\frac{\partial}{\partial \mathbf{w}} \pounds(\mathbf{w}, w_0, \lambda) = \mathbf{0}; \tag{3.7}$$

$$\frac{\partial}{\partial w_0} \pounds(\mathbf{w}, w_0, \lambda) = 0; \tag{3.8}$$

$$\lambda_i \geq 0, \quad i = 1, 2, \ldots, N; \tag{3.9}$$

$$\lambda_i[y_i(\mathbf{w}^T\mathbf{x_i} + w_0) - 1] = 0, \quad i = 1, 2, \ldots, N \tag{3.10}$$

onde λ é um vetor de multiplicadores de Lagrange e $\pounds(\mathbf{w}, w_0, \lambda)$ é a função de Lagrange, definida como

$$\pounds(\mathbf{w}, w_0, \lambda) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i-1}^{N} \lambda_i[y_i(\mathbf{w}^T\mathbf{x_i} + w_0) - 1]. \tag{3.11}$$

A norma L1 poderia ser usada na função de custo, mas como ela não possui derivada em todo seu domínio, outros métodos de otimização deveriam ser empregados.

A combinação da equação (3.11) com as equações (3.7) e (3.8) resulta em

$$\mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i; \tag{3.12}$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0. \tag{3.13}$$

A equação (3.12) define o valor do vetor de pesos $\mathbf{w}$ a partir de uma combinação linear das amostras $\mathbf{x}_i$ associadas aos multiplicadores de lagrange λ_i. Como os multiplicadores de Lagrange podem ser iguais a zero, apenas as amostras associadas aos multiplicadores diferentes de zero serão usadas no cálculo do vetor de pesos. Tais amostras são denominadas vetores suporte (*Support Vector*) e o hiperplano criado através do vetor de pesos

é denominado máquina de vetor suporte (*Support Vector Machine* - SVM). Os vetores suporte estão sobre um dos dois hiperplanos definidos por

$$\mathbf{w}^T\mathbf{x} + w_0 = \pm 1, \tag{3.14}$$

ou seja, eles são os pontos do conjunto de amostras que se encontram mais perto do hiperplano de separação. Esses pontos críticos podem ser vistos circulados na Figura 9, e são encontrados através do cálculo dos multiplicadores de Lagrange λ_i.

A equação (3.12) é usada para calcular apenas o vetor $\mathbf{w}$. A componente de translação do hiperplano, w_0, pode ser encontrada através das igualdades dadas na equação (3.10), apenas para os valores de $\lambda_i \neq 0$. Na prática, w_0 é dado pela média dos valores achados através de todas as igualdades referentes a multiplicadores de Lagrange positivos.

Para se achar o hiperplano de separação entre as classes, se faz necessário calcular os N multiplicadores de Lagrange referentes as amostras de treinamento. Para cumprir tal tarefa, usa-se a dualidade de Lagrange. A representação dual do problema exposto nas equações (3.5) e (3.6) é dada por

$$\begin{aligned} &\textit{maximizar} && \pounds(\mathbf{w}, w_0, \lambda), && (3.15)\\ &\textit{sujeito a} && \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i, && (3.16)\\ & && \sum_{i=1}^{N} \lambda_i y_i = 0, && (3.17)\\ & && \lambda \geq \mathbf{0}. && (3.18) \end{aligned}$$

Substituindo as equações (3.16) e (3.17) na equação (3.15), obtêm-se a tarefa de otimização dada por

$$\max_{\lambda} \left(\sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{ij} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right) \tag{3.19}$$

$$\begin{aligned} &\textit{sujeito a} && \sum_{i=1}^{N} \lambda_i y_i = 0 && (3.20)\\ & && \lambda \geq \mathbf{0} && (3.21) \end{aligned}$$

Este problema normalmente é resolvido por programas de otimização quadrática.

Observa-se na equação (3.19) que as amostras de treinamento aparecem na forma de um produto interno, logo a função a ser maximizada não depende explicitamente da dimensão do vetor de características, trazendo uma grande vantagem para a representação dual. Após o cálculo dos multiplicadores de Lagrange, dado pela equação (3.19), o hiperplano, dado por $\mathbf{w}$ e w_0, pode ser calculado através das equações (3.12) e (3.10).

3.1.2 Classes não linearmente separáveis

Quando as classes não são linearmente separáveis, o método explicado na seção anterior não é válido. A Figura 10 ilustra um caso de duas classes não linearmente separáveis. Qualquer tentativa de se traçar um hiperplano não irá resultar em uma separação perfeita entre as classes.

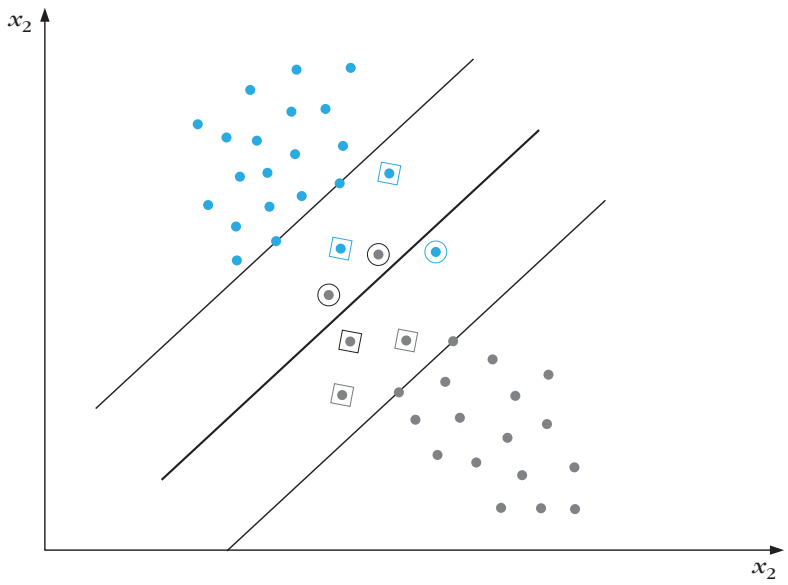


Figura 10 – Um caso de classes não linearmente separáveis. (THEODORIDIS; KOUTROUMBAS, 2008)

Como visto na seção 3.1.1, a margem é definida pela distância entre os hiperplanos que satisfazem a equação

$$\mathbf{w}^T\mathbf{x} + w_0 = \pm 1. \tag{3.22}$$

Os vetores de treinamento agora pertencem a uma de três categorias:

- Vetores localizados do lado de fora da margem que são corretamente classificados. Esses vetores se adequam a inequação (3.6).
- Vetores localizados dentro da margem que são corretamente classificados. Tais vetores são representados por quadrados na Figura 10 e satisfazem a inequação

$$0 \leq y_i(\mathbf{w}^T\mathbf{x_i} + w_0) < 1. \tag{3.23}$$

- Vetores que são erroneamente classificados. Eles são representados por círculos na Figura 10 e satisfazem a inequação

$$y_i(\mathbf{w}^T\mathbf{x_i} + w_0) < 0. \tag{3.24}$$

Todos os três casos podem ser acomodados numa mesma restrição se for adicionado à equação um novo conjunto de variáveis,

$$y_i(\mathbf{w}^T\mathbf{x_i} + w_0) \geq 1 - \xi_i. \tag{3.25}$$

A primeira categoria de vetores corresponde a $\xi_i = 0$, a segunda categoria corresponde a $0 < \xi_i \leq 1$ e a terceira categoria corresponde a $\xi_i > 1$. As variáveis ξ_i são conhecidas como "*slack variables*". O objetivo da tarefa de otimização agora é fazer com que a margem seja a maior possível mantendo o número de pontos com $\xi_i > 0$ no menor nível possível. Neste caso, a função de custo a ser minimizada é expressa como

$$J(\mathbf{w}, w_0, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} I(\xi_i) \tag{3.26}$$

onde ξ é o vetor de parâmetro ξ_i e

$$I(\xi_i) = \begin{cases} 1 & \xi_i > 0 \\ 0 & \xi_i = 0 \end{cases} \tag{3.27}$$

O parâmetro C é uma constante positiva que controla a influência das amostras classificadas erroneamente na função de custo. Otimizar a equação (3.26) não é viável, pois a mesma possui uma função descontínua $I(\cdot)$. Escolhe-se então minimizar uma equação semelhante, dada por

$$\textit{minimizar} \qquad J(\mathbf{w}, w_0, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i \tag{3.28}$$

$$\textit{sujeito a} \qquad y_i(\mathbf{w}^T\mathbf{x_i} + w_0) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, N \tag{3.29}$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, N \tag{3.30}$$

A função de Lagrange deste problema de otimização é

$$\pounds(\mathbf{w}, w_0, \xi, \lambda, \mu) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \mu_i\xi_i - \sum_{i=1}^{N} \lambda_i[y_i(\mathbf{w}^T\mathbf{x_i} + w_0) - 1 + \xi_i]. \tag{3.31}$$

As condições de Karush-Kunh-Tucker correspondentes são

$$\frac{\delta\pounds}{\delta\mathbf{w}} = \mathbf{0} \quad => \quad \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i; \tag{3.32}$$

$$\frac{\delta\pounds}{\delta w_0} = 0 \quad => \quad \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i; \tag{3.33}$$

$$\frac{\delta\pounds}{\delta\xi_i} = 0 \quad => \quad C - \mu_i - \lambda_i = 0, \quad i = 1, 2, \ldots, N; \tag{3.34}$$

$$\lambda_i[y_i(\mathbf{w}^T\mathbf{x_i} + w_0) - 1 + \xi_i] = 0, \quad i = 1, 2, \ldots, N; \tag{3.35}$$

$$\mu_i\xi_i = 0, \quad i = 1, 2, \ldots, N; \tag{3.36}$$

$$\mu_i \geq 0, \quad i = 1, 2, \ldots, N; \tag{3.37}$$

$$\lambda_i \geq 0, \quad i = 1, 2, \ldots, N. \tag{3.38}$$

A representação dual do problema de otimização é

$$\textit{maximizar} \qquad \pounds(\mathbf{w}, w_0, \lambda, \xi, \mu); \tag{3.39}$$

$$\textit{sujeito a} \qquad \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i; \tag{3.40}$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0; \tag{3.41}$$

$$C - \mu_i - \lambda_i = 0, \quad i = 1, 2, \ldots, N; \tag{3.42}$$

$$\lambda_i \geq \mathbf{0} \quad i = 1, 2, \ldots, N; \tag{3.43}$$

$$\mu_i \geq \mathbf{0} \quad i = 1, 2, \ldots, N. \tag{3.44}$$

Substituindo as equações (3.40), (3.41) e (3.42) na equação (3.39), o problema dual de otimização se transforma em

$$\max_{\lambda} \left(\sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{ij} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right) \tag{3.45}$$

$$\textit{sujeito a} \qquad \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.46}$$

$$0 \leq \lambda_i \leq C, \quad i = 1, 2, \ldots, N. \tag{3.47}$$

Nota-se que o problema de se maximizar a equação (3.45) para se achar os multiplicadores de Lagrange é o mesmo problema do caso em que as classes são linearmente separáveis. Porém, todos os pontos que estão dentro da margem, sejam corretamente classificados ou não, possuem $\xi_i > 0$ e $\lambda_i \neq 0$. De fato, verificando-se as condições de Karush-Kunh-Tucker, os vetores com $\xi_i \neq 0$ possuem $\mu_i = 0$, levando a $\lambda_i = C$, que é o máximo valor possível de λ_i. Logo, os vetores que se encontram dentro da margem são os que mais contribuem para a formação do vetor de pesos $\mathbf{w}$.

4 Fusão de múltiplas detecções sobrepostas e pré processamento

Em um sistema de detecção de pedestres baseado em janelas deslizantes, técnicas como o SVM são usadas para detectar a presença ou não de pedestres em cada região da imagem. Entretanto, pode acontecer de o sistema gerar múltiplas detecções sobrepostas, as quais devem ser fundidas em uma única janela por pedestre, para não aumentar a quantidade de falsos positivos no resultado final. A solução adotada para se fazer esta fusão é baseada em duas premissas (DALAL, 2006), descritas abaixo:

- Se o classificador é robusto, o mesmo deve atribuir um valor alto de confiança para janelas que se encontram ligeiramente fora de posição (no espaço ou na escala) em relação ao pedestre na imagem.
- O mesmo classificador não vai disparar com a mesma frequência nem com a mesma confiança em janelas que não possuam pedestres (falso positivos).

A primeira premissa assume que a resposta de um detector decai ligeiramente para pequenas mudanças na posição ou na escala de um pedestre, mas a resposta máxima acontece somente na localização exata do mesmo. A segunda premissa assume que falsos positivos ocorrem somente quando a janela de detecção está sobre um alinhamento específico na imagem, logo a chance de ocorrerem falsos positivos em posições e escalas adjacentes é muito pequena. A técnica ideal usada para a fusão de múltiplas detecções sobrepostas deve considerar que, ocorrendo a primeira premissa, de fato ali encontra-se um pedestre. Já para a segunda premissa, a técnica deve identificar a presença de um falso positivo.

Sumarizando, a técnica usada para a fusão de múltiplas detecções sobrepostas deve possuir as seguintes características:

- Quanto maior a confiança dada pelo classificador à detecção, maior a chance de ali se encontrar um pedestre.
- Quanto maior a quantidade de detecções sobrepostas que se encontrarem na vizinhança de uma região da imagem, maior a chance de se encontrar um pedestre nesta região.
- Detecções sobrepostas que se encontram bem perto uma da outra espacialmente, mas que se encontram em escalas muito diferentes, não devem ser fundidas.

As duas primeiras características são bem claras; expressam que se houver múltiplas detecções em uma área da imagem, de preferência com *scores* altos dado pelo classificador, a probabilidade de existir um pedestre ali é grande, logo as detecções devem ser fundidas para que haja somente uma detecção por pedestre, evitando assim o aumento de falsos positivos provenientes das detecções sobrepostas.

A terceira característica é baseada na observação de que as detecções possuem uma margem em volta dos pedestres, logo pode existir sobreposição de detecções pertencentes a pedestres diferentes, principalmente quando os pedestres se encontram na mesma região espacial, mas em escalas diferentes. Métodos heurísticos de fusão, como o proposto por (VIOLA; JONES, 2004), dividem as detecções em conjuntos formados por detecções sobrepostas, e cada conjunto gera uma detecção final. Tais métodos não são ideais para o caso proposto aqui, pois não levam em consideração a escala dos pedestres.

Neste trabalho, as detecções são representadas por meio de densidades estimadas através de *kernels* (DUDA; HART; STORK, 2001), no espaço 3-D constituído pela posição e escala das detecções. Estimação de densidade por meio de *kernels* é um processo onde densidades contínuas são estimadas por meio da aplicação de *kernels* com larguras de banda determinadas para cada amostra dada. Os máximos locais da distribuição estimada correspondem ao resultado da fusão das detecções. Um exemplo do uso desta técnica pode ser visto na Figura 11.

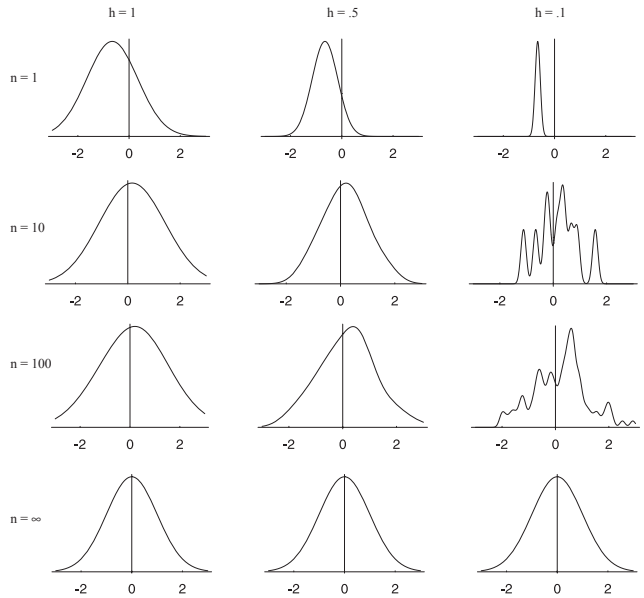


Figura 11 – Estimação de uma densidade normal por janela de Parzen. O parâmetro n é a quantidade de amostras, e o parâmetro h é a largura do kernel, que também possui uma distribuição normal (DUDA; HART; STORK, 2001).

A largura de banda do *kernel* define a densidade na vizinhança de cada amostra. O *score* de confiança dado pelo classificador é incorporado à cada amosta através de pesos

que são atrelados aos *kernels* correspondentes. Com isso, as duas primeiras características expostas acima são atendidas.

A terceira característica também é atendida, pois detecções sobrepostas que se encontram em escalas muito diferentes irão ficar separadas no espaço 3-D, fazendo com que elas contribuam para máximos locais diferentes e consequentemente sejam fundidas separadamente.

Um exemplo da fusão de múltiplas detecções pode ser visto na Figura 12. Nota-se que em volta dos pedestres existem muitas detecções, em várias posições e escalas, que foram fundidas, como pode ser visto na segunda imagem. Porém, as detecções que ocorreram em volta do painel na parte superior da imagem foram fundidas juntamente com as detecções dos pedestres, por serem em pouca quantidade e possivelmente com *scores* baixos.

Duas técnicas são usadas em sequência para realizar a fusão de múltiplas detecções sobrepostas, sendo elas: *Mean shift* e agrupamento hierárquico.

4.1 *Mean shift*

A técnica *mean shift* foi proposta por (FUKUNAGA; HOSTETLER, 1975), e tem como objetivo achar a moda de uma distribuição estimada por *kernel*, mas com a vantagem de não precisar calcular a distribuição em si. Mais tarde, ela foi estudada por (COMANICIU; MEER, 2002), que propôs o uso do *mean shift* com *kernels* de largura de banda variados em (COMANICIU, 2003). No contexto da detecção de pedestres, o *mean shift* é usado para resolver o problema da fusão de múltiplas detecções sobrepostas (YU; YAO, 2009).

4.1.1 *Mean shift* de largura de banda variada

Seja $\mathbf{x}_i$, $i = 1, 2, \ldots, n$ um conjunto de pontos no espaço R^d e assuma que, para cada ponto, existe uma matriz $d \times d$ simétrica e definida positiva $\mathbf{H}_i$, relacionada à largura de banda do *kernel*. A matriz $\mathbf{H}_i$ quantifica a incerteza associada ao ponto x_i na criação da densidade estimada. A densidade estimada com *kernel* gaussiano de dimensão d, calculada no ponto $\mathbf{x}$, é dada por

$$\hat{f}_v(\mathbf{x}) = \frac{1}{n(2\pi)^{d/2}} \sum_{i=1}^{n} \frac{1}{|\mathbf{H}_i|^{1/2}} \exp\left(\frac{1}{2} D^2(\mathbf{x}, \mathbf{x}_i, \mathbf{H}_i)\right), \tag{4.1}$$

onde

$$D^2(\mathbf{x}, \mathbf{x}_i, \mathbf{H}_i) = (\mathbf{x} - \mathbf{x}_i)^T \mathbf{H}_i^{-1} (\mathbf{x} - \mathbf{x}_i) \tag{4.2}$$

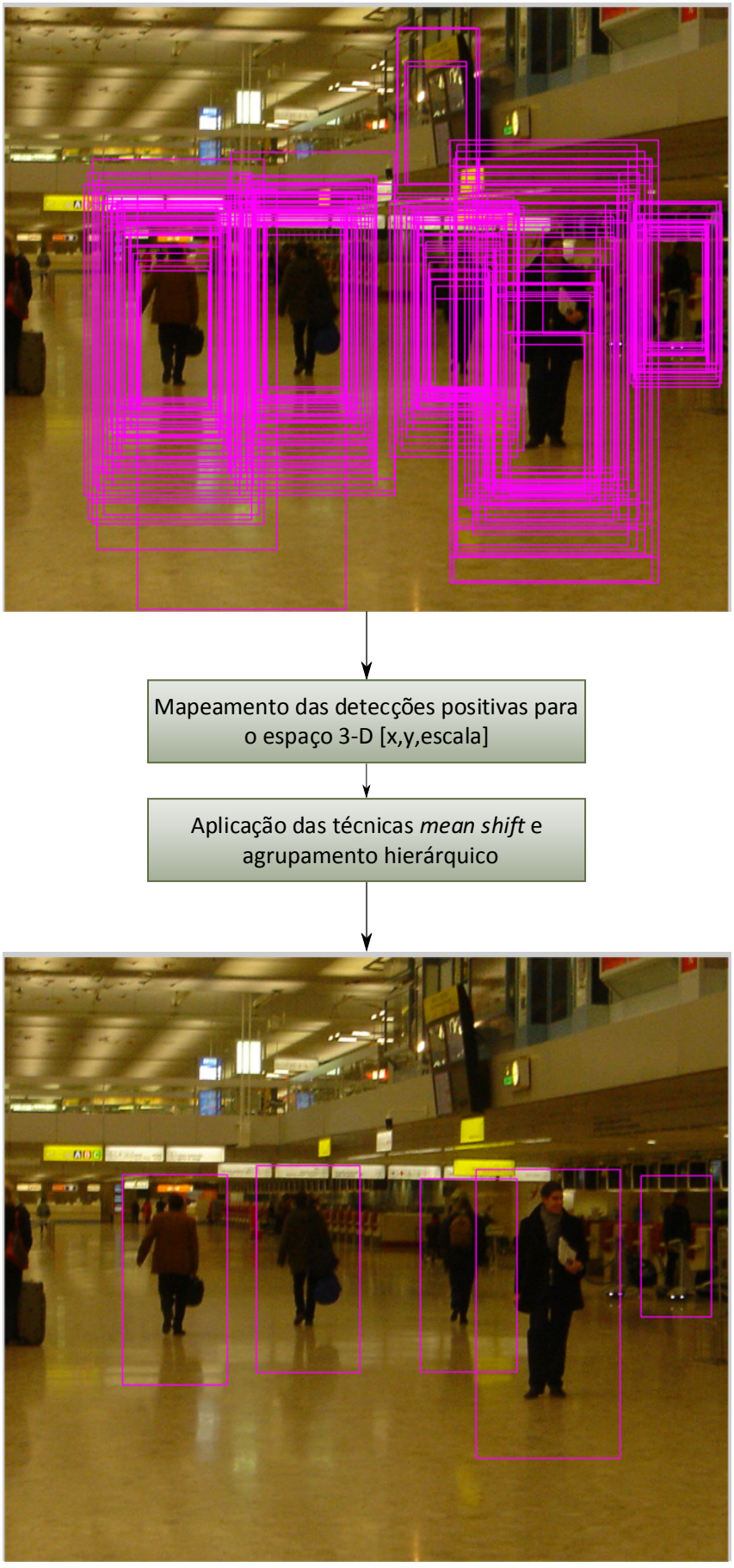


Figura 12 – Fluxograma exemplificando a fusão de múltiplas detecções sobrepostas.

é a distância de Mahalanobis entre $\mathbf{x}$ e $\mathbf{x}_i$. Seja $\mathbf{H}_h$ a média harmônica ponderada das matrizes de incerteza $\mathbf{H}_i$, computada em $\mathbf{x}$

$$\mathbf{H}_h^{-1}(\mathbf{x}) = \sum_{i=1}^{n} \varpi_i(\mathbf{x}) \mathbf{H}_i^{-1} \tag{4.3}$$

onde os pesos

$$\varpi_i(\mathbf{x}) = \frac{\frac{1}{|\mathbf{H}_i|^{1/2}} exp(-D^2[\mathbf{x}, \mathbf{x}_i, \mathbf{H}_i]/2)}{\sum_{i=1}^{n} \frac{1}{|\mathbf{H}_i|^{1/2}} exp(-D^2[\mathbf{x}, \mathbf{x}_i, \mathbf{H}_i]/2)} \tag{4.4}$$

satisfazem $\sum_{i=1}^{n} \varpi_i(\mathbf{x}) = 1$. Um estimador do gradiente da verdadeira densidade é o gradiente da densidade estimada $\hat{f}_v(\mathbf{x})$

$$\hat{\nabla} f_v(\mathbf{x}) = \nabla \hat{f}_v(\mathbf{x}) = \frac{1}{n(2\pi)^{d/2}} \sum_{i=1}^{n} \frac{\mathbf{H}_i^{-1}(\mathbf{x}_i - \mathbf{x})}{|\mathbf{H}_i|^{1/2}} \exp\left(\frac{1}{2} D^2(\mathbf{x}, \mathbf{x}_i, \mathbf{H}_i)\right). \tag{4.5}$$

Multiplicando a equação (4.5) por $\mathbf{H}_h(x)$ pelo lado esquerdo e usando a equação (4.1), obtêm-se

$$\mathbf{H}_h(\mathbf{x}) \hat{\nabla} f_v(\mathbf{x}) = \hat{f}_v(\mathbf{x}) \mathbf{m}_v(\mathbf{x}), \tag{4.6}$$

onde

$$\mathbf{m}_v(\mathbf{x}) = \mathbf{H}_h(\mathbf{x}) \sum_{i=1}^{n} \left[\varpi_i(\mathbf{x}) \mathbf{H}_i^{-1} \mathbf{x}_i\right] - \mathbf{x} \tag{4.7}$$

é o vetor *mean shift* de largura de banda variada. Através da equação (4.6), o vetor *mean shift* pode ser representado da forma

$$\mathbf{m}_v(\mathbf{x}) = \mathbf{H}_h(\mathbf{x}) \frac{\hat{\nabla} f_v(\mathbf{x})}{\hat{f}_v(\mathbf{x})}, \tag{4.8}$$

mostrando que o vetor *mean shift* é um estimador do gradiente normalizado da verdadeira distribuição.

No local da moda da distribuição, o gradiente $\nabla \hat{f}_v(\mathbf{x}) = 0$, fazendo com que $\mathbf{m}_v(\mathbf{x}) = 0$. A equação 4.7 sugere que a moda da distribuição pode ser iterativamente estimada computando

$$\mathbf{x}_{m+1} = \mathbf{H}_h(\mathbf{x}_m) \sum_{i=1}^{n} \left[\varpi_i(\mathbf{x}_m) \mathbf{H}_i^{-1} \mathbf{x}_i\right] \tag{4.9}$$

onde $\mathbf{x}_m$ é iniciado com o valor de uma amostra $\mathbf{x}_i$, e as iterações ocorrem até que $\mathbf{x}_m = \mathbf{x}_{m+1}$. A Figura 13 demonstra este processo iterativo espacialmente, onde a cada iteração a janela de densidade estimada se move para a amostra com maior agrupamento ao seu redor.

É mostrado em (COMANICIU; MEER, 2002) que o vetor *mean shift* aponta na direção do máximo aumento da distribuição estimada. Logo, o ponto $\mathbf{x}_m$ sempre se move na direção da moda da densidade. A prova da convergência do algoritmo *mean shift* também pode ser encontrada em (COMANICIU; MEER, 2002).

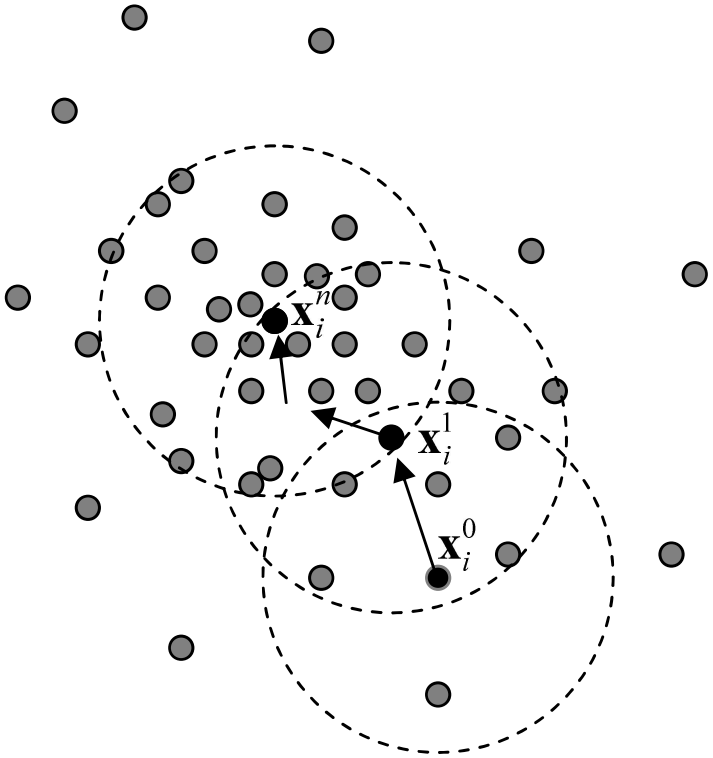


Figura 13 – Exemplo espacial do algoritmo *mean shift* (DERPANIS, 2005). Superescritos indicam a iteração, os pontos sombreados e negros indicam a amostra inicial e os centros das janelas, respectivamente, e os círculos tracejados indicam as janelas de estimação de densidade.

4.1.2 *Mean shift* na detecção de pedestres

Para utilizar o algoritmo *mean shift* na fusão de múltiplas detecções sobrepostas, algumas observações devem ser feitas. O uso desta técnica em detecção de pedestres foi proposto em (YU; YAO, 2009).

Foi discutido na seção 4.1 que a técnica usada deveria levar em conta o *score* de confiança dado pelo classificador. Esse *score* é incorporado à densidade estimada através de um fator multiplicativo w_i, fazendo com que

$$\hat{f}_v(\mathbf{x}) = \frac{1}{n(2\pi)^{d/2}} \sum_{i=1}^{n} \frac{1}{|\mathbf{H}_i|^{1/2}} w_i \exp\left(\frac{1}{2} D^2(\mathbf{x}, \mathbf{x}_i, \mathbf{H}_i)\right). \tag{4.10}$$

Desse modo, quanto maior a confiança que o classificador fornece à detecção positiva, maior será sua influência na densidade estimada.

Cada amostra $\mathbf{x}_i$ é composta por três dimensões $[x_i, y_i, s_i]$, onde x_i e y_i correspondem a posição espacial das detecções positivas, e o valor $s_i = ln(escala)$. O logaritmo da escala garante a homogeneidade no espaço 3-D, já que a escala usada para criação das imagens redimensionadas segue uma progressão geométrica.

Um ponto importante é a quantidade de incerteza, dada pela matriz $\mathbf{H}_i$, a ser associada a cada detecção positiva. A técnica *mean shift* usada é aquela de largura de

banda variada, como mencionado na seção 4.1.1. Desse modo, cada detecção positiva contribuirá para a densidade estimada com uma janela gaussiana 3-D com matriz de covariância $\mathbf{H}_i$. Yu e Yao (2009) propuseram o uso de matrizes de covariância diagonal

$$\mathbf{H}_i = \begin{pmatrix} (\exp(s_i)\sigma_x)^2 & 0 & 0 \\ 0 & (\exp(s_i)\sigma_y)^2 & 0 \\ 0 & 0 & \sigma_s^2 \end{pmatrix}. \tag{4.11}$$

onde σ_x, σ_y e σ_s são parâmetros de suavização especificados pelo usuário. Multiplicar os parâmetros σ_x e σ_y por $\exp(s_i)$ aumenta a incerteza espacial da detecção. Este deve ser o comportamento natural, pois quanto maior a escala de detecção maior deve ser a incerteza associada aquele ponto. O exponencial é usado pois s_i está em escala logarítmica.

Yu e Yao (2009) mostraram que os valores de suavização σ_x e σ_y são dependentes da razão de escala da janela e do intervalo espacial de varredura da mesma. Neste trabalho, a janela espacial possui um tamanho de 128x64 pixels, tendo assim uma razão de aspecto de 0,5. O intervalo espacial entre janelas é de 8 pixels. Logo, os valores de suavização escolhidos foram $\sigma_x = 8$ e $\sigma_y = 16$. Foi mostrado, através de experimentos em (YU; YAO, 2009), que valores de suavização σ_s entre $\ln(1,2)$ e $\ln(1,5)$ dão bons resultados na tarefa de detecção de pedestres. Neste trabalho, foi usado o valor $\sigma_s = \ln(1,3)$.

Para que ocorra a parada das iterações, é necessário que o vetor *mean shift* convirja. Na teoria, essa convergência acontece quando $\mathbf{x}_{m+1} - \mathbf{x}_m = 0$, como explicado na seção 4.1.1. Porém, para que essa convergência aconteça, o vetor deve ser calculado muitas vezes, fazendo com que o tempo de execução do algoritmo seja muito grande. Para diminuir tal tempo, optou-se neste trabalho por usar um limiar de parada $\|\mathbf{x}_{m+1} - \mathbf{x}_m\| \leq 0,1$, fazendo com que cada detecção positiva convirja para o máximo da densidade estimada em uma média de 3 iterações. Desse modo, as detecções não convergem exatamente para o mesmo ponto, mas o algoritmo de agrupamento hierárquico, explicado adiante, resolve este problema.

4.2 Clusterização Hierárquica

O uso da técnica de agrupamento hierárquico (THEODORIDIS; KOUTROUMBAS, 2008) foi proposto neste trabalho para identificar os *clusters* formados pelo *mean shift* e acelerar o processo de fusão das múltiplas detecções sobrepostas. Após o *mean shift*, as detecções ficam bem próximas umas das outras, sempre perto do máximo local da distribuição estimada pelas mesmas. Para que todas as detecções convirjam exatamente para o máximo local, o limiar de parada das iterações do *mean shift* deve tender a zero, ou seja, $\mathbf{x}_{m+1} - \mathbf{x}_m = 0$. Na prática, usar um limiar muito próximo de zero faz com que o algoritmo *mean shift* demore muito para convergir. Logo, foi proposto o uso do algoritmo

de agrupamento hierárquico no intuito de identificar e extrair o representante de cada *cluster* formado pelo *mean shift*.

Os algoritmos de agrupamento hierárquico não produzem um único agrupamento. Como o próprio nome diz, eles produzem uma hierarquia de *clusters*. Seja

$$\mathbf{X} = \{\mathbf{x}_i,\ i = 1, 2, \ldots, N\} \tag{4.12}$$

um conjunto de N vetores no espaço R^d a serem agrupados, e

$$\Re = \{C_j,\ j = 1, 2, \ldots, m\} \tag{4.13}$$

o resultado de uma etapa de agrupamento, sendo que $C_j \subseteq \mathbf{X}$. Um agrupamento $\Re_1$ contendo k *clusters* é dito estar aninhado a um agrupamento $\Re_2$, contendo $r(< k)$ *clusters*, se cada *cluster* de $\Re_1$ é um subconjunto de um *cluster* em $\Re_2$. Nesse caso, escreve-se $\Re_1 \sqsubset \Re_2$. Algoritmos de agrupamento hierárquico produzem uma hierarquia de *clusters* aninhados. Mais especificamente, esses algoritmos envolvem N passos, igual a quantidade de vetores a serem agrupados. A cada passo t, um novo agrupamento é obtida, baseada no agrupamento do passo $t - 1$. Esses algoritmos possuem duas categorias, os algoritmos aglomerativos (categoria usada neste trabalho) e os algoritmos divisivos.

O agrupamento inicial $\Re_0$ para os algoritmos aglomerativos consiste em N *clusters*, cada um contendo um único elemento de $\mathbf{X}$. No primeiro passo, o agrupamento $\Re_1$ é produzido. Ela contêm $N - 1$ *clusters*, de modo que $\Re_0 \sqsubset \Re_1$. Este procedimento continua até que o agrupamento final $\Re_{N-1}$ seja produzido, contendo um único *cluster* formado pelo conjunto de dados de entrada $\mathbf{X}$. A hierarquia final do algoritmo aglomerativo é dada por

$$\Re_0 \sqsubset \Re_1 \sqsubset \ldots \sqsubset \Re_{N-1}. \tag{4.14}$$

Seja $g(C_i, C_j)$ uma função definida para todos os pares de *clusters* de $\mathbf{X}$, que mede a similaridade ou dissimilaridade entre os *clusters* C_i e C_j, e seja t o nível de hierarquia do agrupamento. O algoritmo de agrupamento aglomerativo é apresentado no Algoritmo 1.

Neste trabalho, a função $g(.)$ corresponde a distância euclidiana entre os pontos médios dos *clusters*, logo, é uma medida de dissimilaridade.

Fica claro que algoritmo de agrupamento aglomerativo cria uma hierarquia com N agrupamentos, de modo que cada uma fica aninhada a todos os agrupamentos sucessivos, ou seja, $\Re_{t_1} \sqsubset \Re_{t_2}$ para $t_1 < t_2$.

Em cada nível hierárquico t, existem $N - t$ *clusters*. Para determinar quais *cluster* serão unidos no nível $t + 1$, um total de $\binom{N-t}{2} = \frac{(N-t)(N-t-1)}{2}$ *clusters* devem ser analisados. Logo, o número total de pares de *clusters* que devem ser analisados no processo

Algoritmo 1: Algoritmo de Agrupamento Hierárquico Aglomerativo

Inicialização: Escolher $\Re_0 = \{C_i = \{\mathbf{x}_i\}, i = 1, 2, \ldots, N\}$ como o agrupamento inicial ;
$t = 0$;
while $t \leq N - 1$ **do**
  $t = t + 1$;
  Entre todos os possíveis pares de *clusters* (C_r, C_s) pertencentes a $\Re_{t-1}$, achar o par (C_i, C_j) de modo que

$$g(C_i, C_j) = \begin{cases} \min_{r,s} g(C_r, C_s), & \text{se } g(.) \text{ for uma medida de dissimilaridade} \\ \max_{r,s} g(C_r, C_s), & \text{se } g(.) \text{ for uma medida de similaridade} \end{cases}$$

  Definir $C_q = C_i \cup C_j$ e produzir o novo agrupamento $\Re_t = (\Re_{t-1} - \{C_i, C_j\}) \cup \{C_q\}$;
end

de agrupamento hierárquico aglomerativo é

$$\sum_{t=0}^{N-1} \binom{N-t}{2} = \sum_{k=1}^{N} \binom{k}{2} = \frac{(N-1)N(N+1)}{6} \tag{4.15}$$

fazendo com que o processo seja proporcional a N^3. Porém, a complexidade exata do algoritmo está relacionada a função $g(.)$.

Um dendrograma é uma ferramenta efetiva para a representação de uma sequência de agrupamentos produzidos por um algoritmo aglomerativo. Um exemplo pode ser visto na Figura 14. Cada passo do algoritmo de agrupamento aglomerativo corresponde a um nível do dendrograma. Cada novo nível criado corresponde a um aumento da medida de dissimilaridade. Cortar o dendrograma em algum nível dessa medida corresponde a escolher o agrupamento imediatamente superior a esse nível.

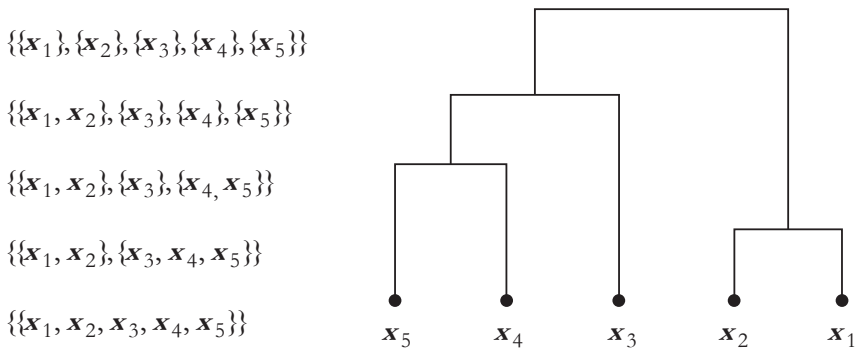
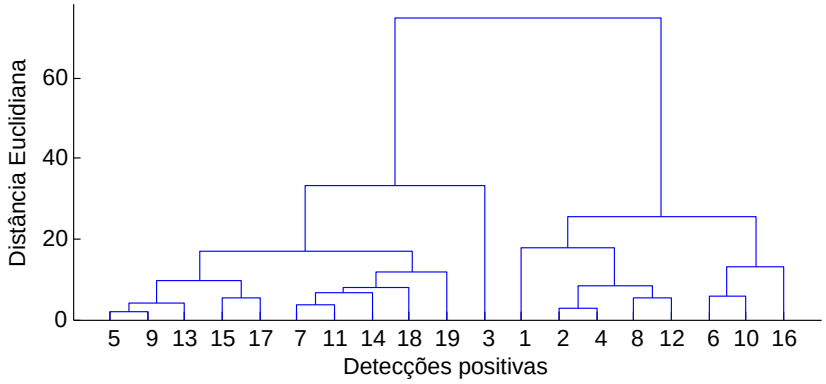


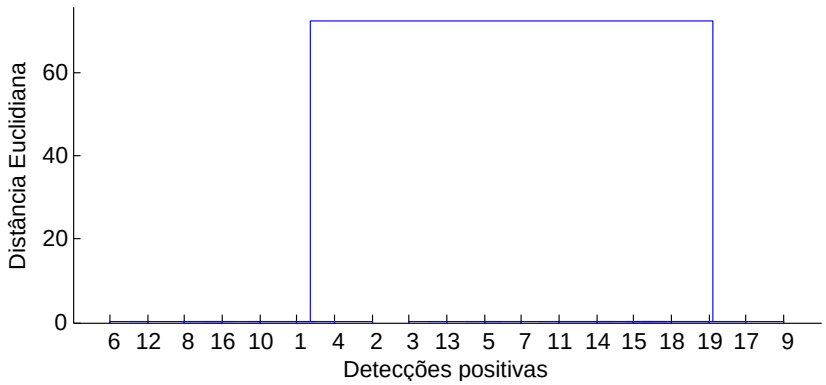
Figura 14 – Dendrograma exemplificativo de um agrupamento envolvendo cinco vetores (THEODORIDIS; KOUTROUMBAS, 2008).

Neste trabalho, há uma certa liberdade sobre o nível certo a ser escolhido para o corte, porque após o *mean shift* as detecções positivas ja estão bem agrupadas em torno do

máximo local da densidade estimada. Este fato pode ser constatado pela Figura 15, que representa o dendrograma referente às detecções de uma imagem, antes e depois do *mean shift*. As detecções referentes ao mesmo pedestre tendem a ser unidas nos primeiros níveis do dendrograma, representando distâncias muito pequenas se comparadas as distâncias dos níveis superiores. Logo, a distância de corte escolhida para este trabalho é igual a 5. É importante lembrar que o *mean shift* sozinho pode agrupar as detecções, mas para isso o limiar dele deve ser bem pequeno, aumentando assim o número de iterações, e com isso, o tempo de execução. Relaxando o limiar e aplicando o agrupamento hierárquico, ganha-se em tempo.



(a)



(b)

Figura 15 – Dendrograma referente a uma imagem deste trabalho (a) - antes do *mean shift*, (b) - depois do *mean shift*.

4.3 Filtro bilateral

Proposto por Tomasi e Manduchi (1998), o filtro bilateral, usado no pré processamento das imagens, suaviza as imagens enquanto preserva as bordas, por meio de uma combinação não linear de valores próximos da imagem. O método combina níveis de intensidade de cinza ou cor baseado tanto na proximidade geométrica como na similari-

dade fotométrica. Em contraste com filtros que operam nas três bandas do canal de cor separadamente, o filtro bilateral consegue reforçar a percepção métrica presente no espaço de cores CIE-Lab, suavizando a imagem e preservando as bordas de um modo focado na percepção humana.

O funcionamento do filtro bilateral se baseia no fato de que somente os pixels que possuem um nível de intensidade similar ao pixel central são usados para o cálculo do valor do mesmo. Logo, pixels do outro lado de uma borda não são levados em consideração. A ideia por traz do filtro bilateral é a combinação de uma filtragem espacial e outra na intensidade.

Seja $\mathbf{x}$ a posição espacial do pixel, $f(\mathbf{x})$ o valor de intensidade da imagem na posição $\mathbf{x}$ e $h(\mathbf{x})$ a imagem filtrada, a filtragem bilateral em imagens preto e branco é definida pela equação

$$h(\mathbf{x}) = \frac{1}{W_s} \sum_{\mathbf{x}_s \in S} G_{\sigma_s}(\|\mathbf{x} - \mathbf{x}_s\|) G_{\sigma_r}(|f(\mathbf{x}) - f(\mathbf{x}_s)|) f(\mathbf{x}_s) \tag{4.16}$$

onde S é a janela de filtragem quadrada, G_{σ_s} é o *kernel* no domínio espacial, G_{σ_r} é o *kernel* no domínio da intensidade e W_s é um fator de normalização. Neste trabalho, tanto o *kernel* espacial como o *kernel* da intensidade são gaussianos

$$G_\sigma(\mathbf{x}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\mathbf{x}^2}{2\sigma^2}\right) \tag{4.17}$$

com desvio padrão σ_s e σ_r, respectivamente.

A Figura 16 apresenta o exemplo de uma filtragem bilateral numa função degrau 2-D corrompida por ruído. Observa-se na imagem que o resultado da filtragem foi suavizada mas manteve suas bordas preservadas.

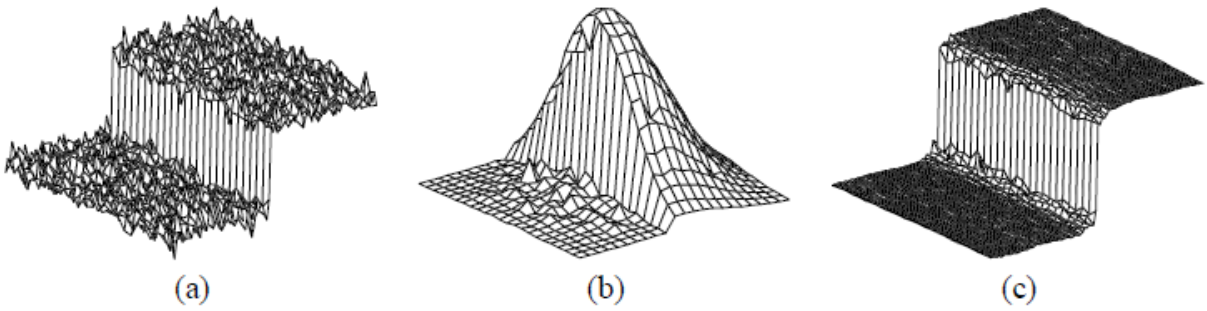


Figura 16 – (a) - Sinal degrau 2-D corrompido com ruído, (b) - máscara bilateral para o cálculo do valor de intensidade de um pixel que se encontra na parte de cima do degrau, (c) - sinal suavizado com as bordas preservadas. (TOMASI; MANDUCHI, 1998)

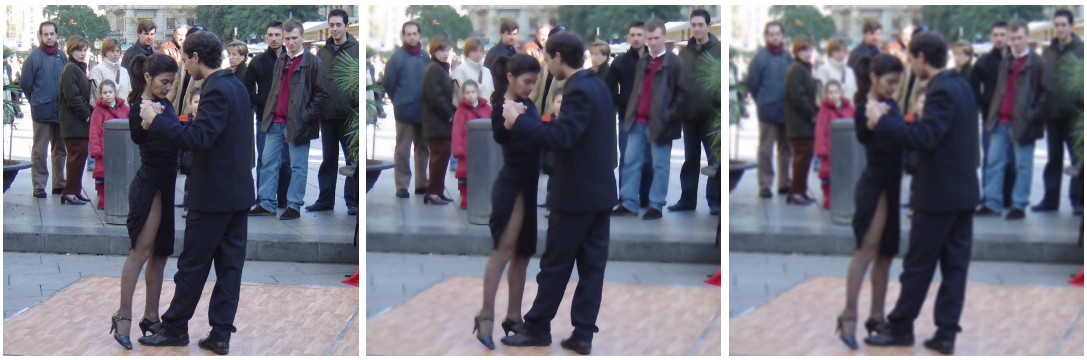
Em imagens coloridas, filtragens que preservam bordas podem ser aplicadas aos três canais de cor separadamente. Contudo, os padrões de intensidade nos três canais de

cor em geral são diferentes. Filtragem nos canais de cor separados geram padrões de cor diferentes nas bordas da imagem. O filtro bilateral resolve este problema combinando os três canais de cores e medindo a distância euclidiana no espaço combinado. É usado o espaço de cores CIE-Lab, pois somente as cores perceptualmente similares ao ser humano são suavizadas juntamente e somente as bordas perceptualmente diferentes são mantidas. A equação que define a filtragem bilateral em imagens coloridas é

$$h_v(\mathbf{x}) = \frac{1}{W_s} \sum_{\mathbf{x}_s \in S} G_{\sigma_s}(\|\mathbf{x} - \mathbf{x}_s\|) G_{\sigma_r}(\|f_v(\mathbf{x}) - f_v(\mathbf{x}_s)\|) f_v(\mathbf{x}_s) \tag{4.18}$$

onde $f_v(\mathbf{x})$ é um vetor 3-D correspondente às intensidades dos três canais de cor.

Na Figura 17, observa-se a utilização do filtro bilateral em uma imagem colorida usada no sistema de detecção de pedestres. A medida que se aumenta a dispersão do *kernel* de intensidades, as bordas vão sendo mais suavizadas, assemelhando-se a uma filtragem espacial com filtro gaussiano.



(a) (b) (c)

Figura 17 – Exemplos de filtragem usando o filtro bilateral, com tamanho da máscara $S = 11 \times 11$ pixels. (a) - Imagem original, (b) - $\sigma_s = 200$ e $\sigma_r = 0,2$, (c) - $\sigma_s = 10$ e $\sigma_r = 1$.

5 Metodologia usada na detecção de pedestres

Neste capítulo será apresentada uma visão geral da tarefa de detecção de pedestres, bem como o banco de dados utilizado.

5.1 Banco de dados

O banco de dados usado neste trabalho é o INRIA *Person Database*, produzido por Dalal e Triggs (2005) e disponível em (DALAL, 2014). As pessoas presentes nesse banco de dados estão usualmente em pé, mas podem aparecer em qualquer pose e orientação. Os pedestres aparecem em uma grande variedade de planos de fundo, criando assim um conjunto de imagens não controladas.

Para o treinamento do classificador, são separadas 1.208 imagens contendo apenas um pedestre. Como imagens de pessoas possuem muita variação, para se treinar um classificador efetivo, as imagens foram redimensionadas para o tamanho da janela de detecção (128×64 pixels) e centralizadas em torno do pedestre. Para melhorar o desempenho do classificador, as imagens são duplicadas, fazendo-se o espelhamento horizontal das mesmas, criando assim um conjunto total de 2.416 amostras de treinamento positivas. A Figura 18 ilustra alguns exemplos do conjunto de treinamento positivo. Nota-se nos exemplos a grande variação intra-classe, e também que algumas janelas possuem um aglomerado de pedestres. Fazer o treinamento com essas imagens cria um classificador mais robusto, capaz de detectar pedestres em meio a multidões e com diferentes poses.

Figura 18 – Exemplos de amostras de treinamento positivas. Nota-se a variação intra-classe das amostras.

Cada janela de treinamento positiva contém um pedestre com a altura aproximada de 96 pixels, para que sobrem 16 pixels de cada lado da imagem. Essa margem foi criada no intuito de eliminar qualquer efeito de fronteira no cálculo dos gradientes.

Para formar o conjunto de treinamento negativo, são fornecidas 1.218 imagens sem pedestres. De cada uma dessas imagens são extraídas 10 janelas aleatoriamente, formando um conjunto de 12.180 amostras negativas. A Figura 19 apresenta exemplos dessas amostras. A escolha das amostras de treinamento foi feita deste modo para concordar com outros trabalhos sobre detecção de pedestres (DALAL; TRIGGS, 2005; WALK et al., 2010; DOLLAR et al., 2012), que também usam o mesmo baco de dados.

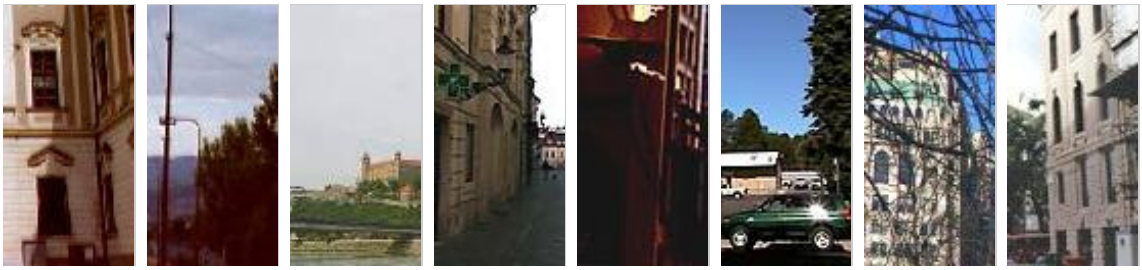


Figura 19 – Exemplos de amostras de treinamento negativas.

É importante mencionar que, uma vez escolhidas as amostras de treinamento negativas, essas são fixadas, e todos os treinamentos posteriores são feitos com o mesmo conjunto. Isso foi feito para que o conjunto de treinamento não interfira nos resultados da classificação.

Para testar o classificador treinado, foram fornecidas 288 imagens contendo pedestres, sendo que nenhum desses pedestres aparece no conjunto de treinamento. Todas as imagens possuem anotações, que consistem em uma caixa delimitadora centrada no tronco dos pedestres. As anotações originais do banco de dados não possuem uma razão de aspecto constante, pois foram feitas manualmente. Exemplos das imagens de teste com anotações podem ser vistos na Figura 20.

Figura 20 – Exemplos de imagens de teste com anotações em amarelo. Percebe-se na imagem que as anotações não possuem razão de aspecto constante.

5.2 Fases do processo de detecção de pedestres

A arquitetura geral de detecção de pedestres, baseada na arquitetura apresentada em (DALAL, 2006), é construída com base em um método para classificação de regiões individuais da imagem. Essa arquitetura é baseada em três fases: fase de treinamento, fase de varredura e fase de avaliação. A fase de treinamento cria um classificador binário que prevê decisões acerca da presença ou não presença de pedestre em regiões de tamanho fixo da imagem, chamadas janelas. A fase de varredura usa o classificador para fazer uma densa varredura em múltiplas escalas, fornecendo a presença preliminar de pedestres em cada local da imagem de teste. Essas detecções preliminares são então fundidas para a obtenção das detecções finais. A fase de avaliação compara as detecções finais com as anotações do banco de dados, e dependendo da sobreposição entre elas, incrementa os valores de verdadeiro positivo, falso positivo e falso negativo. O desempenho final do sistema de detecção de pedestres depende da acurácia e confiança do classificador binário e também de como as detecções preliminares foram fundidas na fase de varredura.

A abordagem da detecção de pedestres por janelas deslizantes possui várias vantagens. Ela permite que um classificador convencional seja usado para a classificação, retirando dele a responsabilidade de ser invariável a posição e a escala (entretanto invariância a outros tipos de transformação, a iluminação e a mudanças de pose devem ser assumidas). Nesta abordagem, o classificador trabalha em coordenadas relativas (posição das características em relação ao centro da janela de detecção), fazendo possível o uso de um conjunto de treinamento bem rígido em relação ao posicionamento dentro da janela. Porém, o classificador é usado para classificar um grande conjunto de janelas, fazendo com que o sistema seja custoso computacionalmente e sensível a taxa de falso positivo do classificador, pois o número de janelas negativas varridas nas imagens de teste é muito maior que o número de janelas positivas.

5.2.1 Fase de treinamento

A fase de treinamento tem como objetivo final criar um classificador binário capaz de dizer se existe um pedestre ou não em uma janela de tamanho fixo. O treinamento inicial é feito com 2.416 amostras positivas e 12.180 amostras negativas, totalizando 14.596 amostras.

Após o treinamento inicial, uma etapa de retreinamento é executada, que consiste em uma varredura nas 1.218 imagens de treinamento negativas a procura de falsos positivos (*hard examples*). Os *hard example*s são então adicionados ao conjunto de amostras negativas de treinamento e o classificador é retreinado com o conjunto de amostras aumentado (2.416 positivas e 12.180 + *hard examples* negativas). Para isso, varre-se a janela de detecção de 8 em 8 pixels, com uma razão de escala de 1,1 (detalhes sobre o processo de varredura

podem ser vistos na subseção 5.2.2).

Esta etapa de retreinamento é importante, pois o classificador inicial possui um valor alto de falsos positivos por janela. Como a quantidade de janelas negativas a serem varridas é muito maior que a quantidade de janelas positivas, este alto valor de falsos positivos se torna proibitivo, porque a quantidade de falsos positivos detectados em uma imagem se torna muito grande, afetando o desempenho final do sistema. O retreinamento com mais amostras negativas diminui este valor.

Walk et al. (2010) observaram que existe melhoria ao se fazer mais de uma rodada de retreinamento. Este fato não foi observado nesse trabalho, pois, para mais de um retreinamento, a quantidade de *hard examples* encontrada é ínfima (Tabela 1). O primeiro retreinamento já treina um classificador capaz de classificar quase que perfeitamente todas as janelas negativas varridas.

	Número de *hard examples*
Primeiro retreinamento	22672
Segundo retreinamento	5
Terceiro retreinamento	3

Tabela 1 – Quantidade de *hard examples* encontrados nos processos de retreinamento

É conveniente notar que o classificador, no trabalho aqui apresentado, foi treinado usando o mesmo conjunto de treinamento de (DALAL; TRIGGS, 2005; DOLLAR et al., 2012), logo, os resultados podem ser comparados sem levar em conta diferenças no treinamento.

5.2.2 Fase de varredura

A fase de varredura tem como objetivo gerar a detecção final do pedestre através de caixas delimitadoras (*bounding boxs*). Para isso, varre-se uma janela na imagem, de tamanho fixo igual a 128×64 pixels, em diferentes posições e escalas. O vetor de características é extraído de cada janela varrida e é feita uma classificação, usando o classificador treinado na fase anterior. O resultado dessa classificação fornece as detecções preliminares do sistema. O valor do deslocamento espacial escolhido para a janela foi de 8 pixels, tanto na horizontal como na vertical, pois corresponde ao tamanho da célula básica (8×8 pixels) dos descritores HOG e CSS. Fazendo assim, pode-se calcular o descritor na imagem inteira e depois seleciona-los na posição da janela, fazendo-se desnecessário o cálculo dos descritores a cada janela varrida.

Para a varredura na escala são criadas várias imagens em escalas diferentes, com a razão entre escalas igual a 1,05. O número exato de escalas depende do tamanho original da imagem, pois são criadas escalas até o limite onde só caiba uma janela de varredura na

imagem, seja na horizontal ou na vertical. A escala final é dada pela equação

$$S_e = min\left(\frac{W_i}{64}, \frac{H_i}{128}\right), \tag{5.1}$$

onde W_i e H_i são a largura e altura da imagem, respectivamente. Sendo $S_s = 1$ a escala inicial, o número de escalas por imagem é

$$S_n = \left\lfloor \left(\frac{\log(\frac{S_e}{S_s})}{\log(S_r)} + 1 \right) \right\rfloor \tag{5.2}$$

onde $S_r = 1,05$ é a razão entre escalas adjacentes. Idealmente, quanto menor a razão da escala, mais minuciosa é a varredura, porém o número de janelas varridas aumenta. Para o passo da janela e razão de escala escolhidos neste trabalho, a quantidade de janelas varridas nas 288 imagens de teste é igual a 11.350.629.

Após a varredura, um passo importante a ser feito é a fusão das múltiplas detecções sobrepostas. Ela consiste em fundir todas as detecções em torno de um pedestre, gerando assim uma única caixa delimitadora por pedestre. As técnicas utilizadas para fundir as detecções preliminares são o *mean shift* e o agrupamento hierárquico. O resultado da fusão são as detecções finais do sistema. A Figura 21 apresenta um fluxograma do processo geral de detecção de pedestres e do processo de varredura.

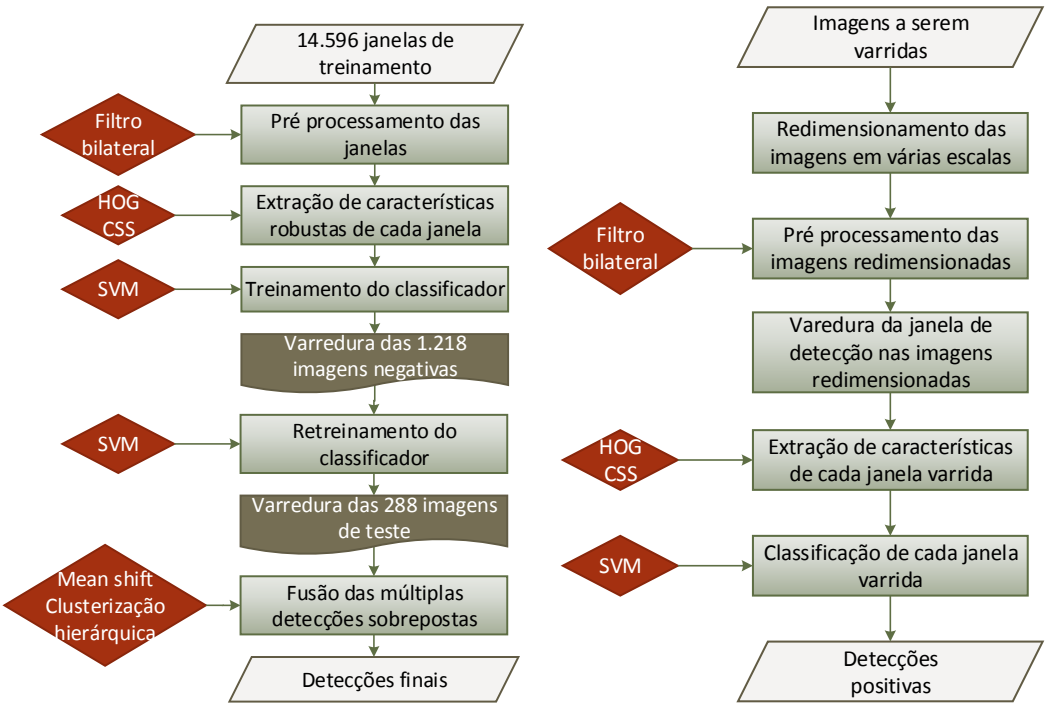


Figura 21 – Fluxograma do sistema de detecção de pedestres. (a) Fluxograma do processo geral, (b) fluxograma da etapa de varredura, usada na busca de *hard examples* nas 1.218 imagens negativas e na busca dos pedestres nas 288 imagens de teste.

5.2.3 Fase de avaliação

A fase de avaliação tem como objetivo comparar as detecções finais com as anotações do banco de dados. Um *bounding box* detectado (BB_{dt}) e um *bounding box* anotado (BB_{an}) formam um potencial casamento se eles se sobrepõem suficientemente. É usado neste trabalho a medida de Pascal (EVERINGHAM et al., 2010), que diz que a área de sobreposição entre os *bounding box* deve ser maior que 50% para o casamento, segundo a expressão

$$\frac{area(BB_{dt} \cap BB_{an})}{area(BB_{dt} \cup BB_{an})} > 50\%. \tag{5.3}$$

Cada BB_{dt} e BB_{an} podem ser casados somente uma vez. Quando um BB_{dt} se sobrepõe sobre mais de um BB_{an}, o casamento é feito sobre a dupla com maior sobreposição. O casamento de uma dupla conta como um verdadeiro positivo. BB_{dt}'s não casados contam como falsos positivos, enquanto BB_{an}'s não casados contam como falsos negativos.

6 Resultados

Neste capítulo serão apresentados resultados referentes aos diversos parâmetros e modificações testados neste trabalho.

Para comparar resultados, usa-se curvas de taxa de erro (*miss rate*) contra falsos positivos por imagem (FPPI). A taxa de erro é dada por

$$\frac{falso\ negativo}{verdadeiro\ positivo + falso\ negativo}, \tag{6.1}$$

representando quantos pedestres foram perdidos em relação ao total de pedestres nas imagens. A medida de falsos positivos por imagem é uma média da quantidade de falsos positivos em relação a todas as imagens. A escolha deste tipo de curva foi feita pois a mesma é preferida em relação à curvas de *precision-recall* para certas tarefas, como a detecção de pedestres, pois nessas aplicações existe um certo limite na quantidade aceitável de falsos positivos por imagem, independentes da densidade de pedestres. Os gráficos são criados varrendo-se o limiar de detecção do SVM na fase de varredura. Para cada limiar varrido, deve-se realizar a fusão de múltiplas detecções e prosseguir com a avaliação dos resultados.

Para sumarizar o desempenho do detector, é usada neste trabalho a média logarítmica da taxa de erro (*log average miss rate*), representada pela média aritmética de nove taxas de erro. Essas nove taxas de erro são referentes a nove valores de falso positivo por imagem, igualmente espaçados entre 10^{-2} a 10^{0} no espaço logarítmico. Como as curvas são aproximadamente lineares nesses pontos, a média logarítmica da taxa de erro é similar a taxa de erro a 10^{-1} FPPI, mas em geral mostra uma medida mais estável e informativa do desempenho do sistema. Todas as curvas de taxa de erro contra falsos positivos por imagem apresentam a média logarítmica da taxa de erro na legenda.

Os resultados são obtidos do sistema usando somente o descritor HOG, sem o descritor CSS e também sem o pré-processamento com o filtro bilateral, a não ser quando dito o contrário. Escolheu-se fazer assim para economia de tempo, pois o CSS e o filtro bilateral aumentam muito o tempo de execução do algoritmo. Os parâmetros usados para os descritores são os parâmetros descritos nos artigos de origem dos mesmos, a não ser quando dito o contrário. O valor padrão do parâmetro c do SVM é $0,01$, por ter demonstrado bons resultados em (DALAL; TRIGGS, 2005).

6.1 Razão de aspecto

Variações significantes na largura das anotações e das detecções podem trazer um efeito indesejável no resultado do sistema. A altura das anotações é uma reflexão da escala, enquanto a largura depende da pose do pedestre. O banco de dados usado fornece anotações feitas manualmente, logo as larguras são variadas, não possuindo uma razão de aspecto constante. As detecções finais, por sua vez, possuem razão de escala constante e igual a 0,5, a mesma razão de escala da janela de detecção usada (128×64 pixels). Essa não uniformidade da razão de escala ocasiona uma diminuição na sobreposição entre anotações e detecções, diminuindo assim a medida de Pascal e fazendo com que menos detecções sejam casadas.

Dollar et al. (2012) propõe a padronização da razão de escala das anotações, para remover qualquer efeito não desejado no resultado. As anotações podem ser obtidas em (DOLLAR, 2014), e possuem uma razão de aspecto igual a 0,41. Essas anotações foram modificadas apenas em suas larguras, mantendo-se a mesma altura das anotações originais. Um exemplo da diferença entre as anotações originais e modificadas pode ser visto na Figura 22.

(a) (b)

Figura 22 – Diferença entre anotações (a) originais e (b) modificadas.

6.2 Corte das detecções finais

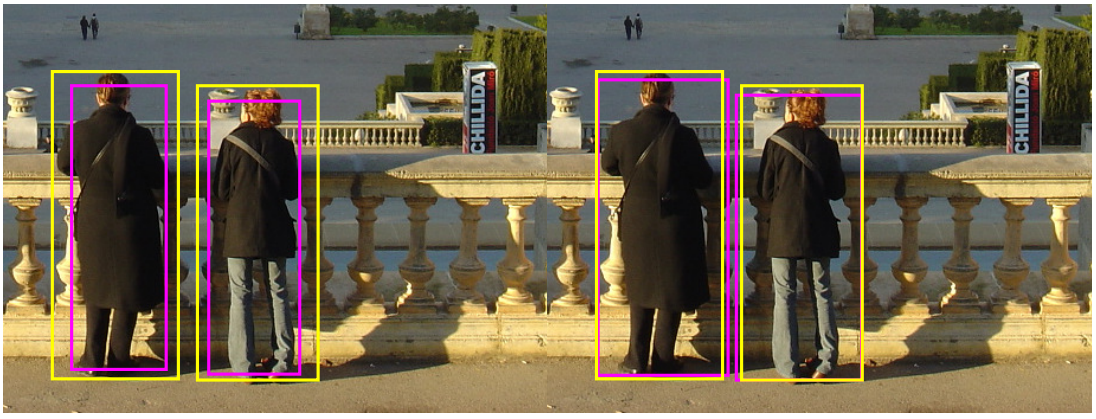
Como citado na seção 5.1, as janelas de treinamento possuem 16 pixels de margem em torno dos pedestres, logo, as detecções finais do sistema também possuem essa margem. As anotações, tanto as originais como as modificadas, não possuem tal margem, como pode ser visto na Figura 22. Logo, para que a área de sobreposição seja maior, deve-se cortar 16 pixels de cada lado das detecções (reduzir a imagem em 16 pixels de cada lado, jogando fora estes pixels) antes do cálculo da medida de Pascal. Desse modo, as detecções finais terão um tamanho de 96×32 pixels, possuindo uma razão de escala igual 0,333. Entretanto,

	16 Ver	14 Ver	12 Ver	10 Ver	8 Ver	6 Ver	4 Ver	2 Ver	0 Ver
16 Hor	0,4805	0,4738	0,4704	0,4676	0,4593	0,4560	0,4600	0,4637	0,4629
14 Hor	0,4597	0,4424	0,4389	0,4401	0,4398	0,4376	0,4387	0,4383	0,4385
12 Hor	0,4357	0,4355	0,4351	0,4346	0,4369	0,4366	0,4366	0,4393	0,4410
10 Hor	0,4348	0,4337	0,4359	0,4359	0,4387	0,4396	0,4408	0,4508	0,4615
8 Hor	0,4356	0,4358	0,4389	0,4405	0,4472	0,4599	0,4723	0,4854	0,5082
6 Hor	0,4396	0,4407	0,4491	0,4612	0,4735	0,4915	0,5295	0,5705	0,6096
4 Hor	0,4473	0,4604	0,4736	0,4957	0,5371	0,5797	0,6217	0,6814	0,7420
2 Hor	0,4720	0,4887	0,5278	0,5700	0,6201	0,6883	0,7510	0,8072	0,8556
0 Hor	0,5223	0,5583	0,6076	0,6807	0,7429	0,8068	0,8539	0,9004	0,9349

Tabela 2 – Média logarítmica da taxa de erro para vários valores de corte das detecções finais.

essa razão de escala difere bastante da razão de escala das anotações modificadas, que é igual a 0,41.

Foi proposto neste trabalho um corte diferente nas detecções finais, para que elas tenham um valor médio da medida de Pascal maior, quando comparadas com as anotações modificadas. Através de teste, foi observado que cortar as detecções finais (após fusão das múltiplas detecções) em 10 pixels nas laterais e 14 pixels em cima e em baixo produz a menor média logarítmica da taxa de erro, 5% menor do que para o corte de 16 pixels, como pode ser observado na Tabela 2. Com isso, as detecções finais terão um tamanho de 100×44 pixels, fazendo com que elas tenham uma razão de aspecto igual a 0,44. Mesmo não sendo a razão de aspecto ideal (0,41), esse corte é o que gera o melhor resultado. Um exemplo da diferença no corte das detecções finais pode ser visto na Figura 23.



(a) (b)

Figura 23 – Diferença entre detecções (a) com corte de 16 pixels e (b) com corte de 10 pixels na horizontal e 14 na vertical. As detecções estão em magenta e as anotações modificadas estão em amarelo. Nota-se o maior valor da medida de Pascal na imagem (b).

A Figura 24 sumariza os resultados discutidos nas seções 6.1 e 6.2, exibindo as

curvas de taxa de erro contra FPPI das possíveis combinações entre anotações e detecções.

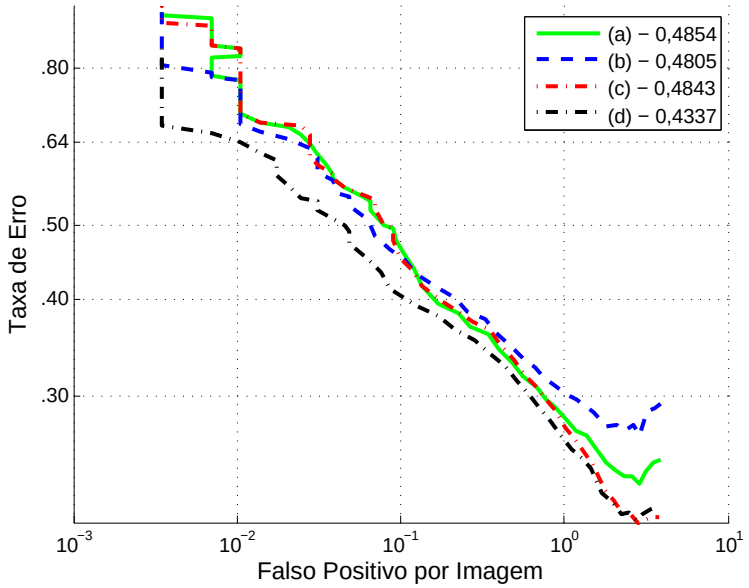


Figura 24 – Comparação de resultados, (a) anotações originais e detecções com corte de 16 pixels, (b) anotações modificadas e detecções com corte de 16 pixels, (c) anotações originais e detecções com corte de 10 e 14 pixels, (d) anotações modificadas e detecções com corte de 10 e 14 pixels.

6.3 Modificações no HOG

As modificações feitas no descritor HOG consistem na mudança da normalização local do histogramas e na proposta do cálculo do gradiente tratando cada pixel como um vetor, como descrito na seção 2.1. Os resultados mostrados daqui em diante ja levam em consideração as observações feitas nas seções 6.1 e 6.2. Ou seja, serão usadas as anotações modificadas e o corte modificado das detecções finais. A Figura 25 apresenta os resultados do sistema para as quatro normalizações testadas. Pode-se ver que as normalizações L1-sqrt, L2-hys e L2 apresentam resultados semelhantes, com uma leve vantagem da normalização L2. Já a normalização L1 apresentou um desempenho muito abaixo das outras normalizações.

A comparação dos resultados usando diferentes métodos para o cálculo dos gradientes pode ser vista na Figura 26. Observa-se que a proposta feita para o cálculo dos gradientes utilizando o método de Di Zenzo (ZENZO, 1986) apresentou resultados bem inferiores. Essa piora ocorreu porque o método de Di Zenzo só fornece orientações do gradiente pertencentes ao intervalo $(-45, 45)$ graus, logo a metade dos histogramas não acumula nenhum valor.

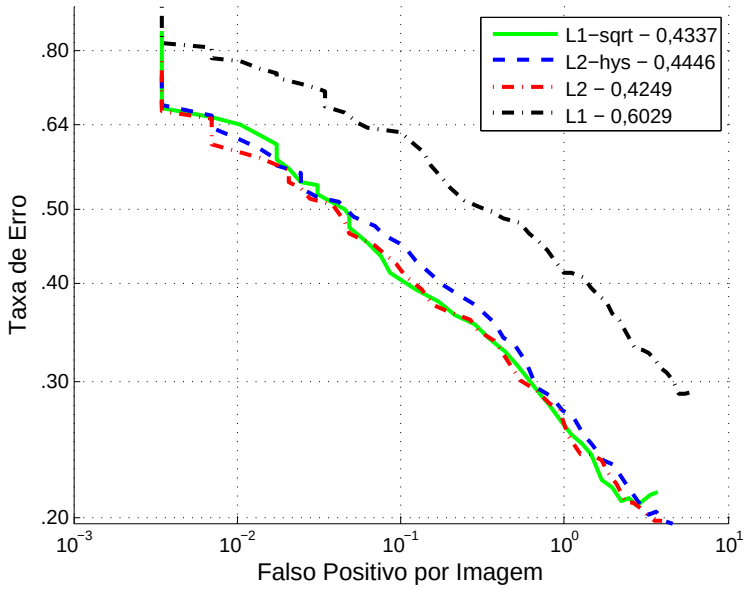


Figura 25 – Comparação de resultados entre diferentes normalizações locais do descritor HOG.

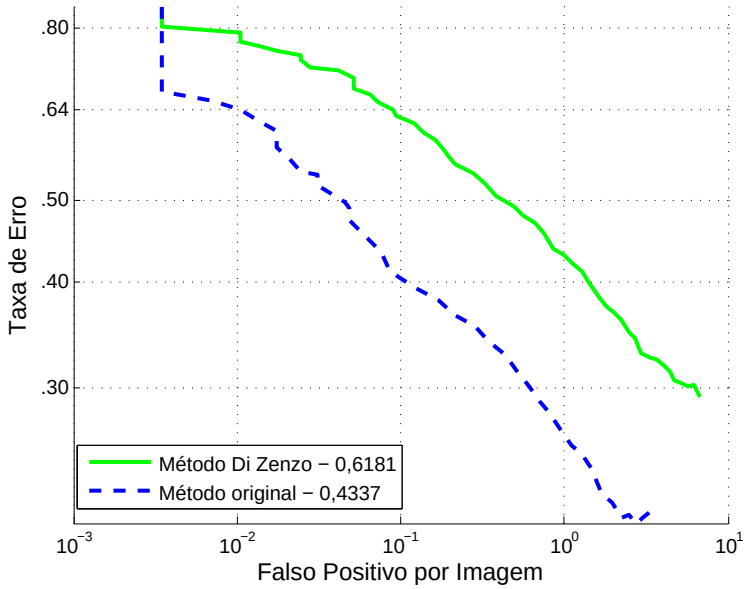


Figura 26 – Comparação de resultados entre diferentes métodos para o cálculo do gradiente no descritor HOG.

6.4 Modificações no CSS

Para realizar os teste descritos aqui, o descritor CSS foi concatenado ao descritor HOG, formando assim um descritor final de dimensão $3780(HOG) + 8128(CSS) = 11908$. Na Figura 27 pode-se ver os resultados dos testes usando diferentes tipos de normalização por janela no descritor, como mencionado na seção 2.2. Assim como no descritor HOG, a normalização L2 apresentou melhores resultados.

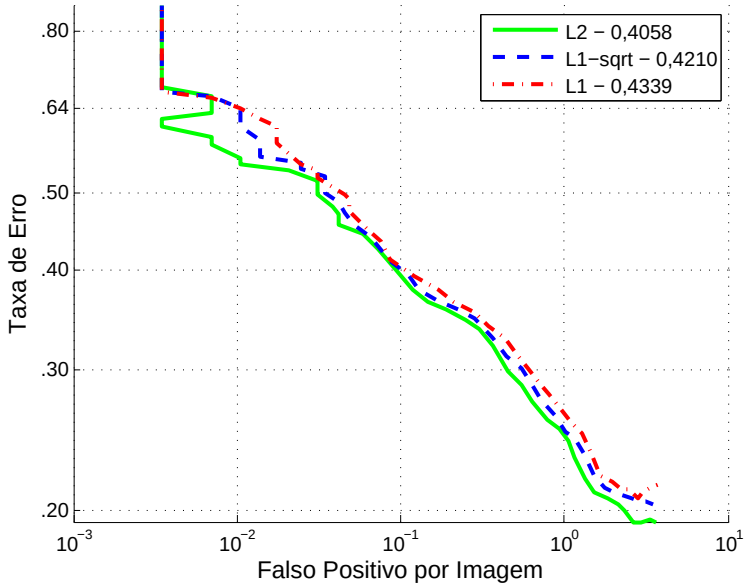


Figura 27 – Comparação de resultados entre diferentes normalizações por janela do descritor CSS.

As modificações propostas no descritor CSS consistem na normalização local dos histogramas de cor e na mudança da métrica de distância usada para o cálculo da distância entre histogramas. Estas propostas foram apresentadas na seção 2.2. A Figura 28 apresenta os resultados da normalização local dos histogramas, sendo que a normalização usada foi a L2. Observa-se que a normalização local, aliada a normalização por janela, apresenta uma pequena melhora no desempenho do sistema. Usar somente a normalização local resulta em uma queda de desempenho do sistema.

Na Figura 29 pode-se ver a comparação das métricas de distância usadas. Percebe-se que a distância correlação apresenta um melhor desempenho, com uma média logarítmica da taxa de erro igual a $0,3907$.

6.5 Implementações do SVM

Três implementações diferentes do SVM foram testadas: SVMlight (JOACHIMS, 1999), libsvm (CHANG; LIN, 2011) e a implementação padrão do Matlab. Como todo o

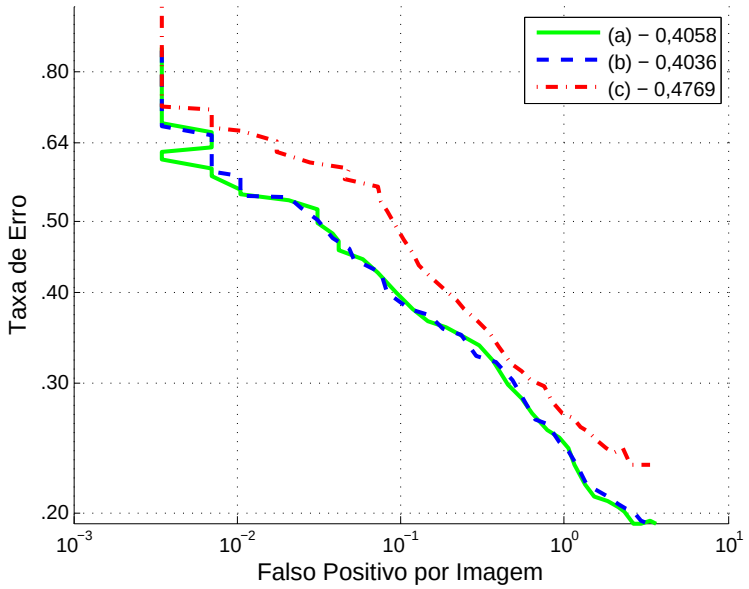


Figura 28 – Comparação de resultados usando a normalização local no descritor CSS.(a) - Somente normalização por janela, (b) - normalização local e por janela, (c) - somente normalização local.

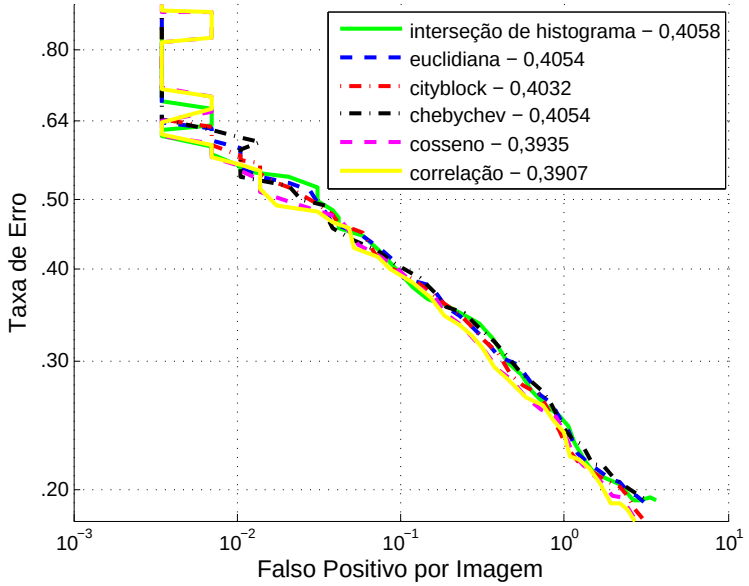


Figura 29 – Comparação de resultados para diferentes métricas de distância.

código foi implementado no Matlab, foi necessário o uso de interfaces para o SVMlight e o libsvm, disponibilizadas nos sites dos autores. As três implementações levaram um tempo semelhante para o treinamento do classificador, mas a implementação do Matlab foi muito superior em rapidez na classificação, como pode ser visto na Tabela 3. O tempo de classificação é um fator muito importante para sistemas baseados em janelas deslizantes, pois a quantidade de janelas classificadas é muito elevada.

	Classificação de 14596 vetores
Matlab	1,18 segundos
SVMlight	129,27 segundos
libsvm	148,87 segundos

Tabela 3 – Tempos de classificação para diferentes implementações do SVM

Outro parâmetro de importância no algoritmo do SVM é dado pelo valor de c, que indica o peso dado aos pontos que se encontram dentro da margem do SVM para a criação do vetor de pesos w. Alguns valores desse parâmetro foram testados, como pode ser visto na Figura 30. O melhor resultado corresponde ao valor de $c = 0,1$. O valor padrão de

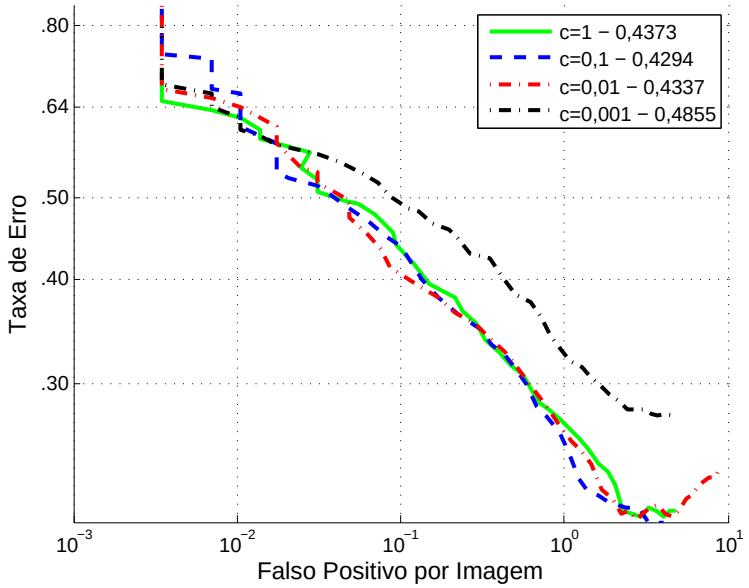


Figura 30 – Comparação de resultados para diferentes valores do parâmetro c do SVM.

$c = 0,01$ foi usado nos testes anteriores pois foi o que deu o melhor resultado em (DALAL; TRIGGS, 2005).

6.6 Combinação de classificadores

Neste trabalho foi testado o uso de classificadores separados para cada descritor, ao invés de um classificador para a concatenação dos descritores. Segundo Ho, Hull e Srihari (1994), usar diferentes classificadores separadamente para cada descritor pode ajudar a realçar os pontos fortes de cada um.

Nos testes feitos, são usados dois SVM's lineares, um para o descritor HOG e o outro para o descritor CSS. O processo de treinamento e retreinamento é feito separadamente, mas de igual modo para os dois, conforme a seção 5.2.1. Ao se fazer assim, é introduzido um novo parâmetro c para o SVM do descritor CSS.

No processo de varredura, os descritores HOG e CSS são extraídos de cada janela e classificados usando o SVM correspondente de cada um, gerando dois valores de *score*. Tais valores são combinados para gerar as detecções preliminares do sistema. Três métodos de combinação são testados aqui (YANG et al., 2013), conforme Tabela 4.

Método	Equação	Descrição
Média aritmética	$S = \frac{1}{M}\sum_{i=1}^{M} S_i$	Calcula a média aritmética dos *scores* S de cada classificador
Scores mínimos	$S = \min\{S_1, \ldots, S_M\}$	Seleciona o menor *score* dentre os classificadores
Scores máximos	$S = \max\{S_1, \ldots, S_M\}$	Seleciona o maior *score* dentre os classificadores

Tabela 4 – Métodos para combinação de *scores*

A Figura 31 mostra a comparação entre esses três métodos de combinação e o sistema com um único classificador para os dois descritores. O valor do parâmetro c usado no SVM do descritor CSS é igual 1. Pode-se ver pela Figura 31 que a combinação dos classificadores usando a média dos *scores* obtêm um melhor resultado.

Também foi testada a influência do parâmetro c do classificador do CSS, como pode ser visto na Figura 32, onde somente as curvas referentes a combinação dos *scores* usando a média foram plotados. Pode ser observado que a variação de c gera uma pequena variação da média logarítmica da taxa de erro, e que o valor de $c = 1$ gera o melhor resultado.

6.7 Uso do agrupamento hierárquico

O uso do agrupamento hierárquico foi proposto neste trabalho para a identificação dos *clusters* formados pelo *mean shift*, e principalmente para a diminuição do tempo de fusão das múltiplas detecções sobrepostas. Como dito na seção 4.2, para que as detecções

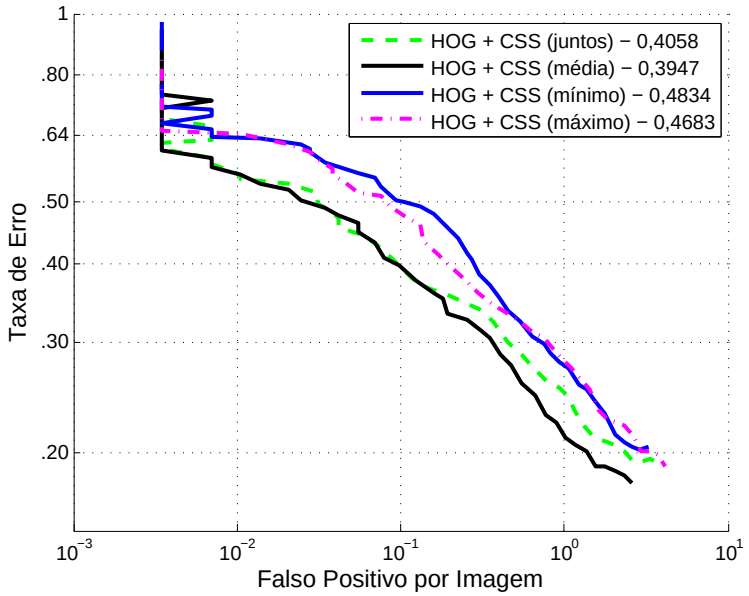


Figura 31 – Comparação de resultados para diferentes métodos de combinação dos *scores*.

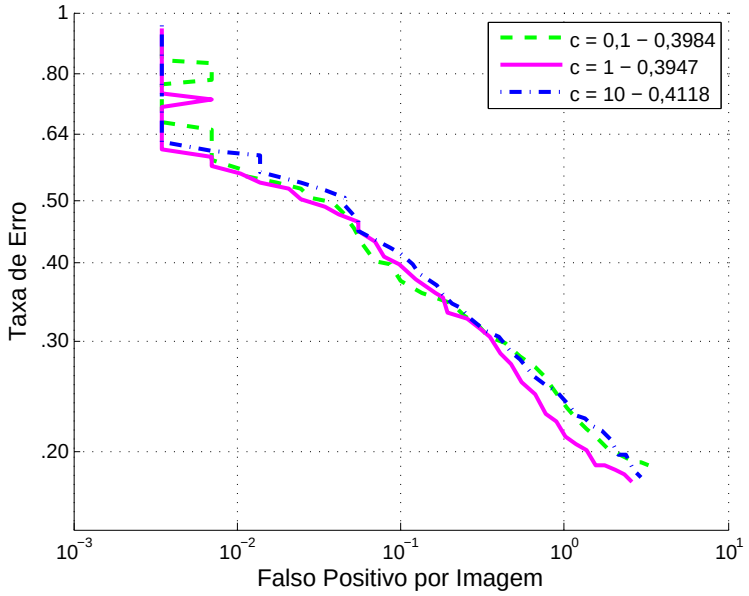


Figura 32 – Comparação de resultados para diferentes valores do parâmetro c usado no SVM do descritor CSS.

convirjam exatamente para o máximo local, são precisas muitas iterações do *mean shift* para cada detecção preliminar. Diminuindo-se o número de iterações e aplicando a técnica de agrupamento hierárquico, diminui-se o tempo de fusão.

Duas situações foram testadas para a fusão de 49 detecções preliminares. A primeira situação consiste no uso do *mean shift* com um limiar igual a $0,001$, e depois o arredondamento das dimensões da detecção e escolha dos valores únicos, correspondentes aos máximos locais. A segunda situação consiste no uso do *mean shift* com um limiar de detecção igual a $0,1$ e o uso do agrupamento hierárquico. a Tabela 5 apresenta os tempos de execução para cada situação. Pode-se observar que o uso do agrupamento

	Situação 1	Situação 2
Tempo de execução	0,526 segundos	0,113 segundos

Tabela 5 – Tempo de execução para diferentes processos de fusão das detecções preliminares.

hierárquico diminui em aproximadamente 5 vezes o tempo de execução da fusão das detecções preliminares.

6.8 Aplicação do filtro bilateral

O algoritmo do filtro bilateral usado neste trabalho possui três parâmetros, que são: tamanho da janela de filtragem espacial, desvio padrão do *kernel* espacial (σ_d) e desvio padrão do *kernel* de intensidades (σ_r). O tamanho de janela usado foi fixado em 11×11 pixels, e foram testados diversos valores de (σ_d) e (σ_r). A Tabela 6 mostra esses resultados na forma da média logarítmica da taxa de erro. Percebe-se pela tabela que valores de

	$\sigma_r = 0,1$	$\sigma_r = 0,05$	$\sigma_r = 0,025$
$\sigma_d = 5$	0,4545	0,4375	0,4462
$\sigma_d = 10$	0,4515	0,4298	0,4298
$\sigma_d = 20$	0,4531	0,4296	0,4416

Tabela 6 – Média logarítmica da taxa de erro para vários valores dos parâmetros do filtro bilateral.

$\sigma_d = 20$ e $\sigma_r = 0,05$ produzem a menor média logarítmica da taxa de erro.

A aplicação do filtro bilateral foi feita de duas formas diferentes. Na primeira forma, aplica-se o filtro apenas nas imagens originais e todas as imagens em escalas menores são redimensionadas a partir da imagem original filtrada. Na segunda forma, todas as imagens em escalas menores são redimensionadas a partir da imagem original, e todas são filtradas separadamente. A Figura 33 apresenta a comparação dos resultados aplicando o filtro bilateral nessas duas formas diferentes, com os parâmetros $\sigma_d = 5$ e $\sigma_r = 0,05$. A Figura

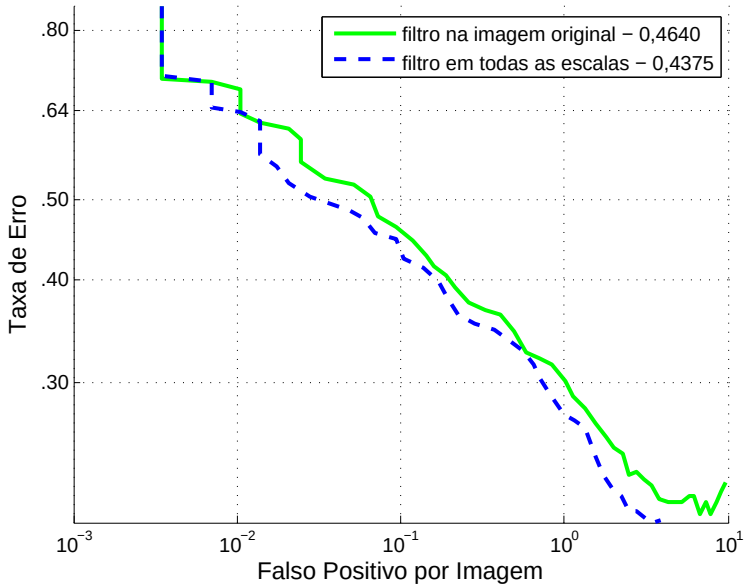


Figura 33 – Comparação de resultados para aplicação do filtro bilateral de duas formas diferentes.

33 mostra que o melhor resultado é alcançado fazendo-se a filtragem em todas as escalas. Esse resultados era esperado, pois no caso de se filtrar somente a imagem original, qualquer artefato criado nesta filtragem será transferido para as outras escalas. Filtrando-se cada escala separadamente, elimina-se este problema, mas aumenta-se o custo computacional do algoritmo.

Todos os testes acima foram feitos utilizando-se apenas o descritor HOG. Pode ser visto pela Tabela 6 que a aplicação do filtro bilateral utilizando apenas o descritor HOG no sistema trouxe uma pequena diminuição na média logarítmica da taxa de erro. A aplicação do filtro bilateral na combinação dos descritores HOG e CSS acarreta uma piora no desempenho do sistema, como pode ser visto na Figura 34.

6.9 Resultados Finais

Nesta seção serão apresentados os resultados finais, usando sempre os parâmetros de cada técnica que obtiveram os melhores resultados. Os resultados anteriores foram obtidos utilizando todos os parâmetros padrão (parâmetros descritos nos artigos de origem de cada técnica), modificando somente os parâmetros de interesse. Nos resultados expostos nesta seção, todas as técnicas terão seus parâmetros modificados ao mesmo tempo, logo, os resultados irão diferir dos apresentados nas seções anteriores.

É feita uma comparação dos resultados deste sistema com o resultado apresentado em (DOLLAR et al., 2012), usando somente o descritor HOG. O mesmo banco de dados é usado, tanto para treinamento como para teste. A Figura 35 apresenta os resultados

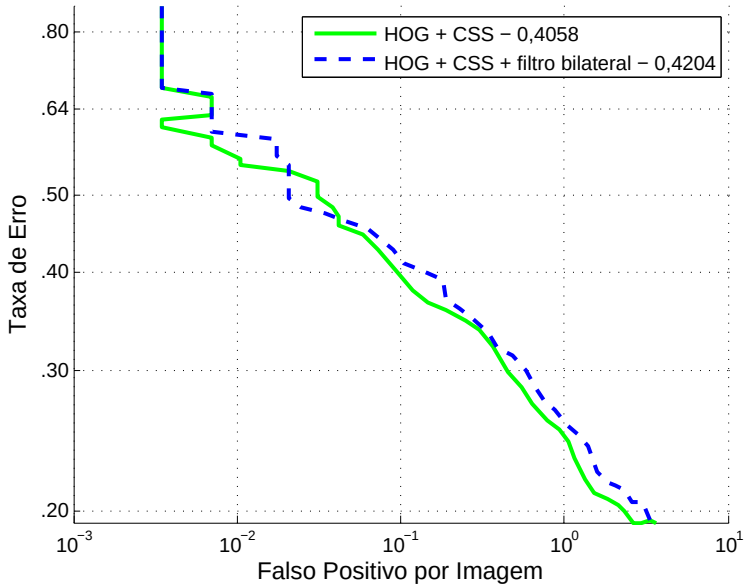


Figura 34 – Comparação de resultados para aplicação do filtro bilateral na combinação dos descritores HOG e CSS.

finais para a tarefa de detecção de pedestres. Percebe-se pelos resultados o acréscimo

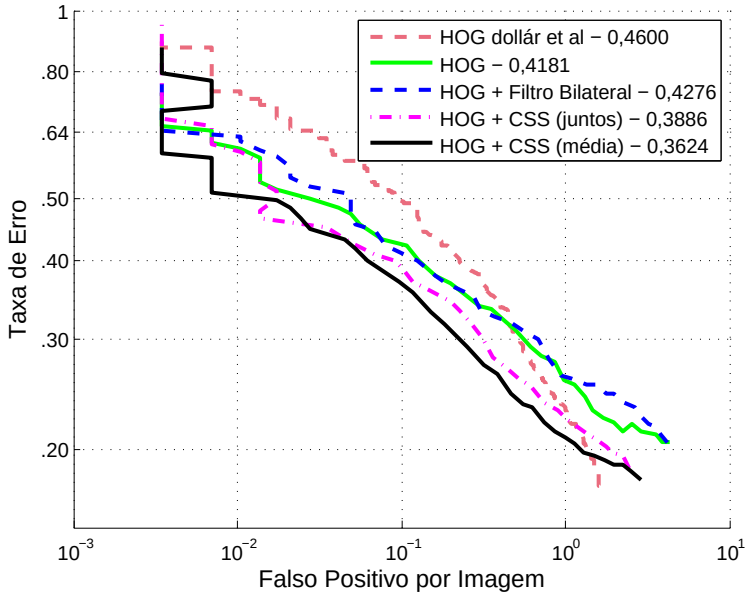


Figura 35 – Comparação final dos resultados.

de desempenho do sistema proposto neste trabalho em relação ao resultado exposto em (DOLLAR et al., 2012), usando apenas o descritor HOG. Curiosamente, a adição do filtro bilateral, apesar de apresentar uma pequena melhora quando aplicado ao sistema usando o descritor HOG original, como mostrado na seção 6.8, gera uma queda de desempenho quando usado com o descritor HOG com a normalização L2 e o parâmetro c do SVM igual

a 0,1. Pode-se ver ainda a melhora gerada pela adição do descritor CSS, diminuindo a média logarítmica da taxa de erro em 3%, quando se usa somente um classificador para os dois descritores. Com a combinação de classificadores, usando a média dos *scores*, a média logarítmica da taxa de erro diminui em mais 2,5%, chegando a 36,24%, o melhor resultado obtido neste trabalho, aproximadamente 10% abaixo do sistema similar utilizando HOG apresentado em (DOLLAR et al., 2012).

6.10 Teste no banco de dados ETH

Para testar a capacidade de generalização do sistema proposto, o mesmo foi testado no banco de dados ETH *Pedestrian Dataset* (ESS; LEIBE; GOOL, 2007). As imagens deste banco de dados foram coletadas na forma de vídeo, através de um par de câmeras fixas em um carro, com uma resolução de 640×480 pixels e uma taxa de quadros de 13 a 14 quadros por segundo. Este banco de dados possui uma densidade de pedestres maior que o banco de dados INRIA, assim como pedestres em maiores escalas.

As imagens usadas aqui estão disponíveis em (ESS, 2014). Três sequências são usadas, sendo elas: 999 imagens da sequência "BAHNHOF", 451 imagens da sequência "JELMOLI" e 354 imagens da sequência "SUNNY DAY", totalizando 1804 imagens. Essas mesmas sequências também foram usadas por Dollar et al. (2012) no seu *benchmark* de detecção de pedestres. As anotações usadas foram modificadas por Dollar et al. (2012) para se manter uma razão de aspecto constante, e estão disponíveis em (DOLLAR, 2014).

Para este teste, a fase de treinamento foi feita no banco de dados INRIA, e somente as fases de varredura e avaliação foram feitas no banco de dados ETH. Escolheu-se fazer assim para se ter uma melhor comparação de resultados, pois, em seu trabalho, Dollar et al. (2012) fez testes no banco de dados ETH utilizando classificadores previamente treinados nos bancos de dados originais de cada trabalho. Como a publicação original do HOG (DALAL; TRIGGS, 2005) foi treinada no banco de dados INRIA, o classificador usado por Dollar et al. (2012) também foi treinado no banco de dados INRIA, ja que tal classificador foi obtido diretamente dos seus autores originais.

O banco de dados ETH possui características de vídeo, com quadros consecutivos sem muita variação. Devido a este fato, as imagens foram varridas de 10 em 10, para evitar repetições desnecessárias. As anotações com pedestres menores que 50 pixels são excluídas, assim como em (DOLLAR et al., 2012).

Na Figura 36 são apresentados os resultados obtidos pelo sistema descrito nesta dissertação, varrendo-se as imagens do banco de dados ETH. Dois testes foram feitos: o primeiro teste foi feito com o sistema utilizando apenas o descritor HOG, e o segundo teste foi feito com o sistema utilizando os descritores HOG e CSS, treinados separadamente. Estes dois testes correspondem à segunda e à quinta curvas da Figura 35. Os parâmetros

utilizados foram os parâmetros que apresentaram melhor resultado nos testes utilizando o banco de dados INRIA.

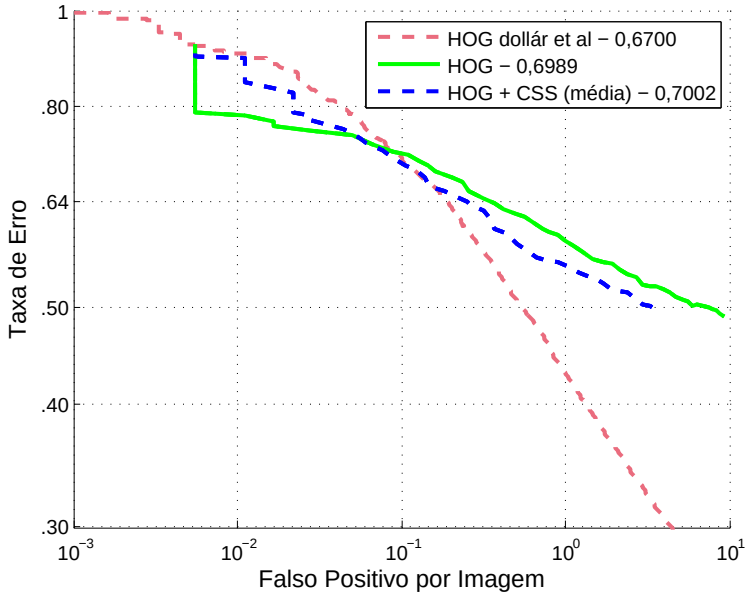


Figura 36 – Comparação dos resultados no banco de dados ETH.

Pode-se observar pela Figura 36 que o sistema proposto nesta dissertação não superou o desempenho do sistema testado em (DOLLAR et al., 2012), quando usado para detectar pedestres no banco de dados ETH. A adição de técnicas que mostraram ter melhor desempenho no banco de dados INRIA não melhoraram o resultado no banco de dados ETH.

Alguns motivos podem ser levantados para justificar estes resultados. Primeiro, o banco de dados ETH possui muitas pessoas em escalas pequenas, e foi mostrado em (DOLLAR et al., 2012) que o descritor HOG obtêm melhor resultado para pessoas em escalas médias e grandes. Também foi observado que a cor das vestimentas dos pedestres é muito parecida com a cor das edificações de fundo, trazendo dificuldades para o descritor CSS encontrar um padrão de auto similaridade entre as cores dos pedestres. Talvez o maior motivo do fraco desempenho neste banco de dados está no fato de que os parâmetros usados foram otimizados para o banco de dados INRIA. Provavelmente o desempenho do sistema seria superior se os parâmetros do sistema fossem otimizados para a detecção no banco de dados ETH. Para otimizar o sistema, seria necessário testar novamente todos os parâmetros, como foi feito para o teste no banco de dados INRIA. Tal tarefa foi deixada de lado devido a restrições de tempo.

7 Conclusões e Trabalhos Futuros

Neste trabalho foi proposta a união de diversas técnicas, advindas dos campos de processamento de imagens e reconhecimento de padrões, para a criação de um sistema de detecção de pedestres baseado em janelas deslizantes, além da definição de uma metodologia para detectar o pedestre em ambientes não controlados.

As contribuições que mais se destacam neste trabalho, utilizando o banco de dados INRIA para teste, são:

- Mudanças no descritor CSS (mudança da métrica de distância entre histogramas e normalização local dos histogramas) diminuíram a média logarítmica da taxa de erro do sistema em aproximadamente 1,5%.
- O corte exato das detecções finais para uma maior sobreposição às anotações modificadas gerou uma diminuição da média logarítmica da taxa de erro do sistema em aproximadamente 5%.
- O uso do agrupamento hierárquico após o *mean shift* reduz o tempo da fusão de múltiplas detecções em aproximadamente 5 vezes, além de ser um método simples de identificar os *clusters* formados pelo *mean shift*.
- O uso da combinação de classificadores, combinando os *scores* dos descritores HOG e CSS, ocasionando uma diminuição de aproximadamente 2,5% na média logarítmica da taxa de erro em relação ao sistema usando somente um classificador sobre a concatenação dos dois descritores.

Algumas propostas, dadas abaixo, não alcançaram o resultado desejado.

- O uso do método exposto em (ZENZO, 1986) para o cálculo dos gradientes no descritor HOG aumentou a média logarítmica da taxa de erro do sistema em aproximadamente 18%.
- O uso da filtragem bilateral em todas as imagens redimensionadas, apesar de proporcionar uma pequena melhora com os parâmetros padrão do HOG e do SVM, produziu uma perda no desempenho se usado no sistema com os parâmetros otimizados. Deve-se levar em conta ainda o aumento do custo computacional gerado pela adição desta técnica.

7.1 Limitações do sistema

O sistema proposto aqui possui algumas limitações, que são intrínsecas ao seu modelo.

- O modelo de detecção baseado em janelas deslizantes requer que os objetos dentro da janela de detecção tenha um contorno recorrente e uma estrutura relativamente parecida. O treinamento de classificadores deste tipo requer um grande conjunto de janelas de treinamento para poder varrer a maioria dos casos possíveis da aparência de um pedestre. Ainda, o objeto a ser detectado deve possuir uma pequena variação inter-classe no que diz respeito a sua informação de contorno e auto similaridade de cor, para que o detector treinado possa classifica-lo corretamente. A classe de objeto 'pedestre' se encaixa bem nesta descrição, mas classes que possuem uma maior variação inter-classes, como algumas classes de animais, não são adequadas a este tipo de sistema.
- A dimensionalidade do vetor de características é muito elevada. Usando os descritores HOG e CSS concatenados, a dimensão do vetor é de 11.908. O grande tamanho do vetor de características trás problemas ao treinamento do classificador, limitando o número de amostras a serem usadas, pois todas as amostras precisam estar disponíveis na memória para o treinamento. Isso limita a quantidade de *hard examples* a serem usados no retreinamento, resultando em um classificador não tão robusto como poderia ser. Técnicas de redução de dimensionalidade, como PCA e LDA (DUDA; HART; STORK, 2001), podem ser usadas para resolver este problema.
- O custo computacional de sistemas baseado em janelas deslizantes é alto, pois as janelas devem ser varridas na imagem por completo, e em cada janela as características precisam ser extraídas e a janela classificada. Como o número de parâmetros a serem otimizados é muito grande, é extremamente inviável fazer o teste de todas as possíveis combinações de parâmetros.

7.2 Trabalhos futuros

Para pesquisas futuras, acredita-se que melhorias possam ser alcançadas com mudanças no método de cálculo do descritor CSS, pois o mesmo é pouco desenvolvido na literatura atual.

Pesquisas relacionadas ao setor de reconhecimento de padrões também podem aumentar o desempenho do sistema. Técnicas mais recentes, como o HIKSVM (*Histogram Intersection Kernel* SVM) (MAJI; BERG; MALIK, 2008) e a rede neural conhecida como *Extreme Learning Machine* (HUANG; ZHU; SIEW, 2006) são boas candidatas para realizar a classificação do pedestre no sistema proposto aqui.

O uso de bancos de dados baseados em vídeo, como o criado por Dollar et al. (2012), abre novas alternativas para a criação de um sistema mais robusto. O uso do conceito de movimento dos pedestres entre diferentes *frames* do vídeo pode ajudar a fazer a distinção entre pedestres e fundo. O fluxo ótico (DALAL; TRIGGS; SCHMID, 2006), combinado com descritores estáticos, pode ser usado para desenvolver sistemas mais precisos e robustos.

Referências

AHONEN, T.; HADID, A.; PIETIKAINEN, M. Face description with local binary patterns: Application to face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 28, n. 12, p. 2037–2041, 2006. Citado na página 22.

ANDRADE, L. de et al. Brazilian road traffic fatalities: A spatial and environmental analysis. *PLOS ONE*, Public Library of Science, v. 9, n. 1, p. e87244, 2014. Citado na página 15.

BELONGIE, S.; MALIK, J.; PUZICHA, J. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 24, n. 4, p. 509–522, 2002. Citado na página 21.

CHANG, C.-C.; LIN, C.-J. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, ACM, v. 2, n. 3, p. 27, 2011. Citado na página 65.

COMANICIU, D. An algorithm for data-driven bandwidth selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 25, n. 2, p. 281–288, 2003. Citado na página 44.

COMANICIU, D.; MEER, P. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 24, n. 5, p. 603–619, 2002. Citado 3 vezes nas páginas 24, 44 e 46.

DALAL, N. *Finding people in images and videos*. Tese (Doutorado) — Institut National Polytechnique de Grenoble-INPG, 2006. Citado 5 vezes nas páginas 8, 18, 30, 42 e 56.

DALAL, N. *INRIA Person Database*. 2014. Disponível em: <http://pascal.inrialpes.fr/data/human/>. Citado na página 54.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: IEEE. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. [S.l.], 2005. v. 1, p. 886–893. Citado 12 vezes nas páginas 13, 21, 22, 27, 29, 33, 54, 55, 57, 60, 67 e 73.

DALAL, N.; TRIGGS, B.; SCHMID, C. Human detection using oriented histograms of flow and appearance. In: *Computer Vision–ECCV 2006*. [S.l.]: Springer, 2006. p. 428–441. Citado na página 77.

DERPANIS, K. G. Mean shift clustering. *Lecture Notes. http://www. cse. yorku. ca/˜ kosta/CompVis_Notes/mean_shift. pdf*, 2005. Citado 2 vezes nas páginas 9 e 47.

DOLLAR, P. *Caltech Pedestrian Detection Benchmark*. 2014. Disponível em: <http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/>. Citado 3 vezes nas páginas 23, 61 e 73.

DOLLAR, P. et al. Pedestrian detection: An evaluation of the state of the art. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 34, n. 4, p. 743–761, 2012. Citado 13 vezes nas páginas 13, 19, 21, 23, 24, 55, 57, 61, 71, 72, 73, 74 e 77.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern classification*. [S.l.]: John Wiley & Sons, 2001. Citado 3 vezes nas páginas 8, 43 e 76.

ESS, A. *ETH Person Dataset*. 2014. Disponível em: <http://www.vision.ee.ethz.ch/~aess/dataset/>. Citado na página 73.

ESS, A.; LEIBE, B.; GOOL, L. V. Depth and appearance for mobile scene analysis. In: IEEE. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. [S.l.], 2007. p. 1–8. Citado na página 73.

EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. *International journal of computer vision*, Springer, v. 88, n. 2, p. 303–338, 2010. Citado na página 59.

FUKUNAGA, K.; HOSTETLER, L. The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, IEEE, v. 21, n. 1, p. 32–40, 1975. Citado 2 vezes nas páginas 24 e 44.

GERONIMO, D. et al. Survey of pedestrian detection for advanced driver assistance systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 32, n. 7, p. 1239–1258, 2010. Citado na página 15.

GONZALEZ, R. C.; WOODS, R. E.; EDDINS, S. L. *Digital image processing using MATLAB*. [S.l.]: Pearson Education India, 2009. Citado na página 29.

GU, C. et al. Recognition using regions. In: IEEE. *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. [S.l.], 2009. p. 1030–1037. Citado na página 19.

HO, T. K.; HULL, J. J.; SRIHARI, S. N. Decision combination in multiple classifier systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 16, n. 1, p. 66–75, 1994. Citado na página 68.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: theory and applications. *Neurocomputing*, Elsevier, v. 70, n. 1, p. 489–501, 2006. Citado na página 76.

JOACHIMS, T. Svmlight: Support vector machine. *SVM-Light Support Vector Machine http://svmlight. joachims. org/, University of Dortmund*, v. 19, n. 4, 1999. Citado na página 65.

JONES, W. D. Building safer cars. *Spectrum, IEEE*, IEEE, v. 39, n. 1, p. 82–85, 2002. Citado na página 15.

LEIBE, B.; SEEMANN, E.; SCHIELE, B. Pedestrian detection in crowded scenes. In: IEEE. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. [S.l.], 2005. v. 1, p. 878–885. Citado na página 19.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, Springer, v. 60, n. 2, p. 91–110, 2004. Citado 2 vezes nas páginas 27 e 28.

MAJI, S.; BERG, A. C.; MALIK, J. Classification using intersection kernel support vector machines is efficient. In: IEEE. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. [S.l.], 2008. p. 1–8. Citado na página 76.

MIT. *MIT Pedestrian Dataset*. 2014. Disponível em: <http://cbcl.mit.edu/software-datasets/PedestrianData.html>. Citado na página 22.

PAPAGEORGIOU, C.; POGGIO, T. A trainable system for object detection. *International Journal of Computer Vision*, Springer, v. 38, n. 1, p. 15–33, 2000. Citado 4 vezes nas páginas 13, 19, 20 e 21.

SEEMANN, E. et al. An evaluation of local shape-based features for pedestrian detection. In: CITESEER. *BMVC*. [S.l.], 2005. v. 5, p. 10. Citado na página 19.

THEODORIDIS, S.; KOUTROUMBAS, K. *Pattern Recognition*. Elsevier Science, 2008. ISBN 9780080949123. Disponível em: <http://books.google.com.br/books?id=QgD-3Tcj8DkC>. Citado 9 vezes nas páginas 8, 9, 24, 35, 36, 37, 39, 48 e 50.

TOMASI, C.; MANDUCHI, R. Bilateral filtering for gray and color images. In: IEEE. *Computer Vision, 1998. Sixth International Conference on*. [S.l.], 1998. p. 839–846. Citado 3 vezes nas páginas 9, 51 e 52.

TORRESAN, H. et al. Advanced surveillance systems: combining video and thermal imagery for pedestrian detection. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Defense and Security*. [S.l.], 2004. p. 506–515. Citado na página 16.

TUZEL, O.; PORIKLI, F.; MEER, P. Human detection via classification on riemannian manifolds. In: IEEE. *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. [S.l.], 2007. p. 1–8. Citado na página 22.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: IEEE. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. [S.l.], 2001. v. 1, p. I–511. Citado na página 22.

VIOLA, P.; JONES, M. J. Robust real-time face detection. *International journal of computer vision*, Springer, v. 57, n. 2, p. 137–154, 2004. Citado 4 vezes nas páginas 13, 20, 24 e 43.

VOLKSWAGEN. 2014. Disponível em: <http://www.vwerl.com/get_file/167>. Citado 2 vezes nas páginas 8 e 16.

WALK, S. et al. New features and insights for pedestrian detection. In: IEEE. *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. [S.l.], 2010. p. 1030–1037. Citado 7 vezes nas páginas 13, 23, 24, 31, 34, 55 e 57.

WANG, X.; HAN, T. X.; YAN, S. An hog-lbp human detector with partial occlusion handling. In: IEEE. *Computer Vision, 2009 IEEE 12th International Conference on*. [S.l.], 2009. p. 32–39. Citado 2 vezes nas páginas 13 e 22.

WOJEK, C.; SCHIELE, B. A performance evaluation of single and multi-feature people detection. In: *Pattern Recognition*. [S.l.]: Springer, 2008. p. 82–91. Citado 2 vezes nas páginas 13 e 21.

WOJEK STEFAN WALK, B. S. C. *TUD-Brussels Dataset*. 2014. Disponível em: <https://www.mpi-inf.mpg.de/departments/computer-vision-and-multimodal-computing/research/

people-detection-pose-estimation-and-tracking/multi-cue-onboard-pedestrian-detection/>. Citado na página 23.

WU, B.; NEVATIA, R. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In: IEEE. *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*. [S.l.], 2005. v. 1, p. 90–97. Citado na página 22.

WU, B.; NEVATIA, R. Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection. In: IEEE. *Computer vision and pattern recognition, 2008. cvpr 2008. IEEE conference on*. [S.l.], 2008. p. 1–8. Citado 2 vezes nas páginas 13 e 22.

YANG, K. et al. An extreme learning machine-based pedestrian detection method. In: IEEE. *Intelligent Vehicles Symposium (IV), 2013 IEEE*. [S.l.], 2013. p. 1404–1409. Citado na página 68.

YU, L.; YAO, W. Pedestrian detection fusion method based on mean shift. In: IEEE. *Machine Vision, 2009. ICMV'09. Second International Conference on*. [S.l.], 2009. p. 204–207. Citado 4 vezes nas páginas 24, 44, 47 e 48.

ZENZO, S. D. A note on the gradient of a multi-image. *Computer vision, graphics, and image processing*, Elsevier, v. 33, n. 1, p. 116–125, 1986. Citado 3 vezes nas páginas 29, 63 e 75.

APÊNDICE A – Cálculo de histogramas com interpolação

O cálculo dos histogramas para os descritores HOG e CSS envolve distribuir o peso do pixel para *bins* de espaço, orientação e cor. Uma tática simples de distribuição seria atribuir o peso total do pixel para o *bin* mais próximo, acarretando assim a existência do efeito de *aliasing*. Tal efeito pode produzir mudanças bruscas no cálculo do descritor. Um exemplo disto acontece no cálculo do descritor HOG de uma imagem que possua uma borda forte. Se essa borda estiver no limite de uma célula em uma imagem e, devido a alguma mudança nas condições do sistema, a mesma borda, em outra imagem, estiver em outra célula, a contribuição total desta borda terá sido calculada para células diferentes nas duas imagens. Para evitar este efeito, calcula-se os histogramas com interpolação linear para cada dimensão, fazendo com que a contribuição de cada pixel seja calculada para diferentes *bins*, devido a distância em que as coordenadas do pixel se encontram do centro desses mesmos *bins*.

A interpolação linear acontece da seguinte forma. Seja $\mathbf{h}$ um histograma com distância entre *bins* igual a b. $\mathbf{h}(x)$ corresponde ao valor do *bin* do histograma centrado em x. Assumindo que se deseja distribuir um peso w que se encontra no ponto x, sendo que o ponto x se encontra entre dois *bins* do histograma, x_1 e x_2, a interpolação linear distribui esse peso segundo as equações

$$\mathbf{h}(x_1) \leftarrow \mathbf{h}(x_1) + w \times \left(1 - \frac{x - x_1}{b_x}\right) \tag{A.1}$$

$$\mathbf{h}(x_2) \leftarrow \mathbf{h}(x_2) + w \times \left(\frac{x - x_1}{b_x}\right). \tag{A.2}$$

A extensão para o caso 3-D, usado no descritor HOG, é simples. Seja $\mathbf{p} = [x, y, z]$ a posição do ponto a ter o seu peso w interpolado. Considerando $\mathbf{p}_1$ e $\mathbf{p}_2$ como os dois vértices do cubo no histograma contendo p, onde cada componente de $\mathbf{p}_1 \leq \mathbf{p} < \mathbf{p}_2$. Seja o espaço entre *bins* dado por $\mathbf{b} = [b_x, b_y, b_z]$. A interpolação trilinear distribui o peso w

aos 8 *bins* ao redor de p conforme as equações

$$\mathbf{h}(x_1, y_1, z_1) \leftarrow \mathbf{h}(x_1, y_1, z_1) + w \times \left(1 - \frac{x - x_1}{b_x}\right) \times \left(1 - \frac{y - y_1}{b_y}\right) \times \left(1 - \frac{z - z_1}{b_z}\right) \quad \text{(A.3)}$$

$$\mathbf{h}(x_1, y_1, z_2) \leftarrow \mathbf{h}(x_1, y_1, z_2) + w \times \left(1 - \frac{x - x_1}{b_x}\right) \times \left(1 - \frac{y - y_1}{b_y}\right) \times \left(\frac{z - z_1}{b_z}\right) \quad \text{(A.4)}$$

$$\mathbf{h}(x_1, y_2, z_1) \leftarrow \mathbf{h}(x_1, y_2, z_1) + w \times \left(1 - \frac{x - x_1}{b_x}\right) \times \left(\frac{y - y_1}{b_y}\right) \times \left(1 - \frac{z - z_1}{b_z}\right) \quad \text{(A.5)}$$

$$\mathbf{h}(x_2, y_1, z_1) \leftarrow \mathbf{h}(x_2, y_1, z_1) + w \times \left(\frac{x - x_1}{b_x}\right) \times \left(1 - \frac{y - y_1}{b_y}\right) \times \left(1 - \frac{z - z_1}{b_z}\right) \quad \text{(A.6)}$$

$$\mathbf{h}(x_1, y_2, z_2) \leftarrow \mathbf{h}(x_1, y_2, z_2) + w \times \left(1 - \frac{x - x_1}{b_x}\right) \times \left(\frac{y - y_1}{b_y}\right) \times \left(\frac{z - z_1}{b_z}\right) \quad \text{(A.7)}$$

$$\mathbf{h}(x_2, y_1, z_2) \leftarrow \mathbf{h}(x_2, y_1, z_2) + w \times \left(\frac{x - x_1}{b_x}\right) \times \left(1 - \frac{y - y_1}{b_y}\right) \times \left(\frac{z - z_1}{b_z}\right) \quad \text{(A.8)}$$

$$\mathbf{h}(x_2, y_2, z_1) \leftarrow \mathbf{h}(x_2, y_2, z_1) + w \times \left(\frac{x - x_1}{b_x}\right) \times \left(\frac{y - y_1}{b_y}\right) \times \left(1 - \frac{z - z_1}{b_z}\right) \quad \text{(A.9)}$$

$$\mathbf{h}(x_2, y_2, z_2) \leftarrow \mathbf{h}(x_2, y_2, z_2) + w \times \left(\frac{x - x_1}{b_x}\right) \times \left(\frac{y - y_1}{b_y}\right) \times \left(\frac{z - z_1}{b_z}\right) \quad \text{(A.10)}$$

O caso de interpolação em 5 dimensões, como a feita no descritor CSS, pode ser facilmente derivada do caso de interpolaçao em 3 dimensões. Cada ponto terá seu peso distribuído em $2^5 = 32$ *bins* do histograma 5-D.

Antes de realizar o cálculo, é necessário que se defina os parâmetros $\mathbf{p}$, $\mathbf{b}$ e w. Para o cálculo dos histogramas no descritor HOG, as coordenadas x, y e z são, respectivamente, a posição horizontal do pixel na célula, a posição vertical do pixel na célula e a orientação do gradiente do pixel. O peso w é dado pela magnitude do gradiente do pixel. Os tamanhos dos espaços entre *bins*, b_x e b_y, são definidos pelo tamanho da célula, e no caso do descritor HOG, são iguais a 8. O valor de b_z é definido pela quantidade de *bins* nos histogramas de orientação. Conforme definido na seção 2.1, os histogramas de orientação possuem 9 *bins* entre 0 e 180, logo, o valor de b_z é $180 \div 8 = 22,5$.

Para o cálculo dos histogramas de cor no descritor CSS, as 5 coordenadas são: a posição horizontal do pixel na célula, a posição vertical do pixel na célula e os valores dos três *bins* de cor do modelo de cor usado. Aqui, o peso w é unitário, pois os histogramas são formados apenas com a informação de cor, sem serem pesados por outros valores. Assim como no descritor HOG, o tamanho do espaço entre *bins* das duas primeiras dimensões são iguais a 8, pois este é o tamanho da célula na qual os histogramas são calculados. O espaço entre *bins* das três últimas dimensões são definidos pela quantidade de *bins* nos histogramas de cor. Já que o valor dos *bins* é normalizado entre 0 e 1, e, conforme a seção 2.2, cada histograma possui 3 *bins*, o valor do espaço entre *bins* das três últimas dimensões é $1 \div 2 = 0,5$