Sistema de Multicamadas para Detecção de Placas Sinalizadoras em Tempo Real

Multi-Level System for Real-Time Detection of Traffic Sign

Renan de Freitas Leite[1]

Byron L. D. Bezerra[1]

Bruno José T. Fernandes[1]

1 Escola Politécnica de Pernambuco, Universidade de Pernambuco, Recife, Brasil.

E-m,ail do autor principal: Renan de Freitas Leite, rfl@ecomp.poli.b

Resumo

O desenvolvimento de sistemas de assistência ao condutor (ADAS, Advanced Driver Assistance Systems) originou uma demanda de técnicas de detecção de placas sinalizadoras em imagens digitais, que estão se tornando cada vez mais robustas. Porém, essas técnicas necessitam de muito recurso computacional para serem executadas em tempo real (30 quadros por segundo). Neste artigo, é apresentado um sistema de detecção de placas sinalizadoras capturadas por câmeras digitais. O modelo proposto consiste de 2 fases de detecção, com o objetivo de juntar técnicas de busca e extração de características, que utilizam o menor custo computacional possível. O modelo possui uma taxa de acurácia acima de 90% na base de dados GTSDB (German Traffic Sign Detection Benchmark) assim como os melhores modelos do estado da arte, porém possui um menor tempo de resposta. Por fim, o sistema foi testado em um ambiente real, por meio de uma câmera digital.

Palavras-Chave: *Detecção; Símbolos; Imagem; Câmera;*

Abstract

The development of driver assistance systems (or ADAS, Advanced Driver Assistance Systems) gave origin to a demand for traffic sign detection techniques on digital images, which are be-coming more robust. However, these techniques require a lot of computational resources to exe-cute in real time (30 frames per second). This article presents a system of traffic sign detection captured by digital cameras. The proposed model consists of two stages of detection, in order to join search techniques and feature extraction, with as low as possible computational cost. The model has an accuracy rate above 90% in GTSDB database (German Traffic Sign Detection Benchmark) as well as the best state-of-the-art models, but it has a shorter response time. Fi-nally, the system was tested in a real environment with a digital camera.

Keywords: *Detection; Sings; Image; Camera.*

http://dx.doi.org/10.25286/repa.v3i2.572

1 Introdução

A informação visual é comumente utilizada pelo ser humano para realizar suas tarefas. As tomadas de ações são algumas vezes inconscientes, por se tratar de ações muito simples, como por exemplo, o ato de se desviar de um objeto vindo lentamente em direção oposta.

Uma tarefa que precisa de informação visual é o reconhecimento de placas de sinalização. Para que o motorista possa desempenhar corretamente sua tarefa, as placas de sinalização devem ser vistas como instruções de proibição, de perigo e de obrigação, entre outras.

Um sistema de piloto automático de carro pode requerer o reconhecimento de placas de sinalização como uma de suas funcionalidades. Isso motivou o surgimento de diferentes abordagens que buscam detectar placas de sinalização **[1]**-**[11]**, além de bases de dados que abordam especificamente esse tema, tais como a LISA *Traffic Sign Database* (LTSD) [1], a *German Traffic Sign Detection Benchmark* (GTSD) [2] e a *German Traffic Sign Recognition Benchmark* (GTSR) [3]. Como consequência de todas essas contribuições, os sistemas de piloto automático estão se tornando cada vez mais confiáveis.

A implementação de um detector e reconhecedor de placas de sinalização não pode ser feita da maneira convencional. Isso porque existe uma grande quantidade de eventos possíveis para serem tratados, tais como variações na posição da placa e na iluminação do ambiente, oclusões parciais, além de uma infinidade de objetos de formato parecido.

Para resolver esse tipo de problema é necessário um algoritmo inteligente, ou seja, um sistema que se adapta aos diversos eventos que possam ocorrer. Alguns dos algoritmos inteligentes utilizados nas abordagens encontradas na literatura tem como base *deep learning* **[1]** e *machine learning* **[2]**-**[11]**. Essa capacidade de adaptação de um sistema às complexas entradas de um ambiente real é um dos focos da computação inteligente. Isso se deve às diversas combinações de eventos que podem ocorrer, gerando um problema, onde o sistema deverá estar apto a se aproximar da melhor solução.

A visão computacional faz parte da computação inteligente e é um conjunto de métodos e técnicas que permitem a interpretação de imagens. Essa interpretação pode ser definida como a transformação de uma imagem digital em uma estrutura de dados, que descreve a semântica ou significado de partes dessa imagem em um contexto qualquer.

O problema deste trabalho é o reconhecimento de placas de sinalização que possam transmitir informação útil para uma máquina ou um indivíduo, por meio de imagens digitais. Isso significa, por exemplo, que uma máquina poderá reconhecer outra quando esta entrar em seu campo de visão. Assim como permitir que a máquina possa tomar ações devido à sinalização de uma placa capturada na imagem, como por exemplo "seguir em frente". Isso favorece uma autonomia à máquina, não necessitando que um usuário controle algumas de suas ações.

2 Trabalhos Relacionados

O estado da arte possui diferentes abordagens **[1]**-**[11]** visando resultados de detecção cada vez melhores. Uma característica em comum dessas abordagens é a utilização de propriedades de cor (canais) e de gradientes (geometria). Pelo menos duas etapas são utilizadas nessas abordagens: extração da região de interesse (segmentação ou filtragem) e reconhecimento (classificação das regiões).

A abordagem de Liang *et al.* **[10]** utiliza as cores e os gradientes das imagens para extrair as regiões de interesse, como pode ser observado na Figura 1. É realizada uma busca massiva por formas regulares em toda a imagem. Essas regiões são classificadas pela geometria, por um modelo HOG/SVM treinado.

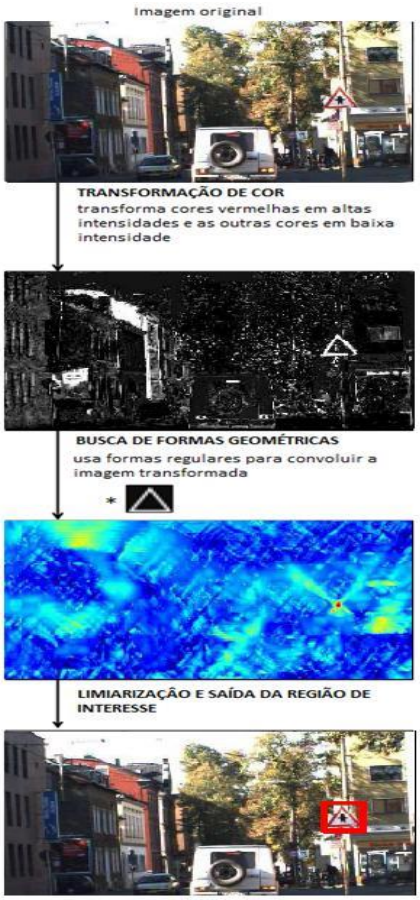

Figura 1: Busca por formas geométricas para extração das regiões de interesse da abordagem de Liang *et al.* (adaptado de [10])

Já a abordagem de Mathias *et al.* **[11]** realiza a extração das regiões de interesse utilizando o modelo de canais integrados (Figura 2). Todos os canais servem de entrada para um modelo HOG/SVM treinado, e a busca é feita por meio de uma janela deslizante em várias escalas.

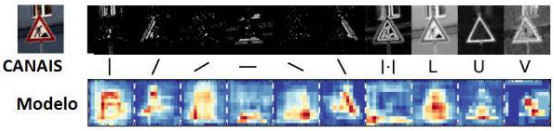

Figura 2: Características extraídas em 10 canais para a classe danger.

Da esquerda para direita: 6 canais de orientação, magnitude do gradiente, 3 canais correspondentes ao espaço de cor LUV. (adaptado de **[11])**

Visando uma elevada taxa de acerto, porém com um alto custo computacional, a abordagem de Houben *et al.* **[9]** utiliza o HOG para extração das regiões de interesse e um classificador LDA/SVM (*Linear Discriminant Analysis*) para o reconhecimento da placa. O método de extração de regiões de interesse dessa abordagem é capaz de encontrar uma forma de placa muito pequena na imagem (10 por 10 pixels, por exemplo), além de separar todas essas regiões com o uso do método *non-maximal suppression.*

3 Modelo Proposto

O modelo proposto é apresentado detalhadamente nesta seção. Como podemos observar na Figura 3, a ideia principal é reduzir o espaço de busca, por meio de uma segmentação feita na imagem pré-processada. O pré-processamento aumenta o contraste da imagem, por meio de uma equalização local. Essa abordagem visa minimizar o problema de iluminação da imagem, que pode resultar na perda da placa durante a fase de segmentação. Nesta seção também serão demonstrados os conceitos existentes no estado da arte referentes a cada uma das etapas do modelo proposto.

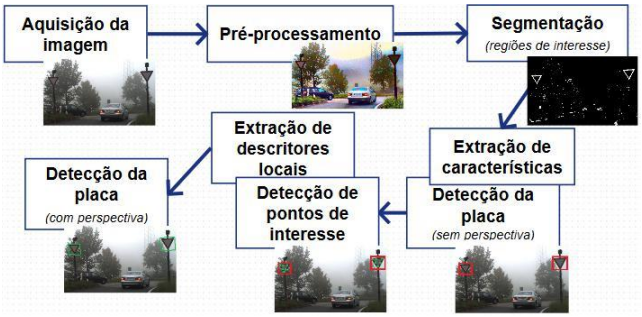

Figura 3 : Modelo proposto.

http://dx.doi.org/10.25286/repa.v3i2.572

3.1 Pré-processamento

As placas de sinalização possuem a característica de terem cores chamativas. Nos casos em que a placa de sinalização está em uma região muito escura ou muito clara da imagem, a diferença de coloração é reduzida por causa da diminuição do contraste.

A equalização de histograma adaptativa ou AHE (*Adaptative Histogram Equalization*) é um método utilizado nos casos em que, durante a aquisição da imagem, há uma grande variação de iluminação no ambiente. Diferentemente da equalização de histograma global, que faz a equalização do histograma de toda a imagem, a AHE computa vários histogramas correspondentes a diferentes regiões da imagem.

O CLAHE **[12]** (*Contrast Limited Adaptative Histogram Equalization*) é um tipo de AHE, que utiliza a diferença de contraste ao longo da imagem para efetuar as equalizações de histograma. O CLAHE foi desenvolvido para prevenir a amplificação de ruído que ocorre por meio da equalização de histograma global. Dessa forma, logo após a captura da imagem de entrada, esse pré-processamento é aplicado com o intuito de realçar características importantes da imagem relacionadas à coloração, como ilustramos na Figura 4.

Figura 4: Imagem original (esq.) e imagem pré-processada pelo CLAHE (dir.).

Com o objetivo de melhorar a velocidade de processamento, algumas abordagens do estado da arte **[1] [8]** realizam um redimensionamento na imagem. Porém, a abordagem utilizada neste trabalho envolve uma redução no espaço de busca e a utilização de técnicas de menor custo computacional de tal modo, que não será necessário redimensionar a imagem.

Portanto, a etapa de pré- processamento recebe a imagem colorida como entrada e retorna a imagem pré-processada pelo método CLAHE. A imagem pré-processada tem o mesmo tamanho da imagem de entrada e também é colorida.

3.2 Segmentação

A detecção de bordas em uma imagem é bastante útil na área de processamento de imagens, por simplificar muitas tarefas, tais como, a segmentação de regiões e o reconhecimento de objetos. Já foi desenvolvida uma grande variedade de algoritmos que buscam resolver este problema. O algoritmo de Canny foi criado para ser um detector de bordas ideal, de acordo com três critérios: baixa taxa de erro, boa localização e resposta mínima.

Os pontos de contorno detectados pelo Canny são como áreas de pixels nas quais ocorrem uma mudança brusca de nível de intensidade. O algoritmo de Canny utiliza várias máscaras **[12]**. Primeiramente é aplicado um filtro gaussiano, para o realce das bordas. Após isso é aplicado o filtro de Sobel na imagem. O *non-maximum suppression* serve para remover pixels que não são considerados partes de um contorno, e dessa forma somente linhas finas permanecerão. O último passo é aplicar a histerese, que é um tipo específico de limiarização **[12]**.

Figura 5: Resultado do detector de borda Canny.

Além da detecção de bordas Canny, a limiarização por cor também é aplicada. A imagem pré-processada tem seus componentes BGR separados em três matrizes, onde cada matriz representa um desses componentes em cada pixel. Dessa forma, quatro funções de limizarização são definidas para quatros tipos de cor de placa: azul, verde, vermelho e amarelo.

$$Ib(x,y)=|2*B(x,y)-R(x,y)-G(x,y)|>Lb \quad (1)$$
$$Ig(x,y)=|2*G(x,y)-R(x,y)-B(x,y)|>Lg \quad (2)$$
$$Ir(x,y)=|2*R(x,y)-B(x,y)-G(x,y)|>Lr \quad (3)$$
$$Iy(x,y)=|R(x,y)+G(x,y)-2*B(x,y)|>Ly \quad (4)$$

onde B(x,y), G(x,y) e R(x,y) são, respectivamente, as matrizes dos componentes B, G e R da imagem préprocessada; Lb, Lg, Lr e Ly são os limiares para cada função de limiarização, que possuem o valor normalizado entre 0 e 255; e Ib(x,y), Ig(x,y), Ir(x,y) e Iy(x,y) são, respectivamente, as imagens limiarizadas pelas cores azul, verde, vermelho e amarelo.

Com o objetivo de fazer uma separação entre as bordas das placas e o background de cor semelhante, as imagens segmentadas e a imagem resultante do detector de bordas Canny serão as entradas da seguinte função lógica:

$$Sb(x,y)=Ib(x,y)\cap\neg Ic(x,y) \quad (5)$$
$$Sg(x,y)=Ig(x,y)\cap\neg Ic(x,y) \quad (6)$$
$$Sr(x,y)=Ir(x,y)\cap\neg Ic(x,y) \quad (7)$$
$$Sy(x,y)=Iy(x,y)\cap\neg Ic(x,y) \quad (8)$$

onde Ic(x,y) é a imagem resultante do detector de bordas Canny; e Sb(x,y), Sg(x,y), Sr(x,y) e Sy(x,y) são, respectivamente, as imagens segmentadas pelas cores azul, verde, vermelho e amarelo.

Dessa forma, a etapa de segmentação recebe a imagem pré-processada como entrada e retorna as imagens segmentadas pelas funções de segmentação.

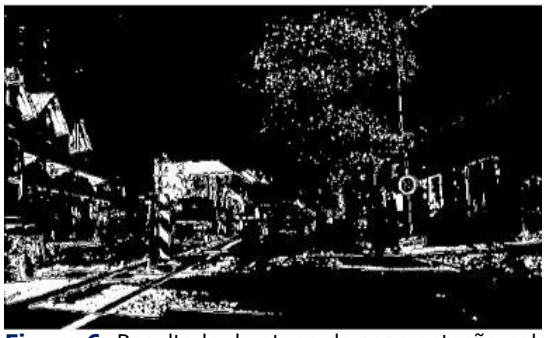

Figura 6: Resultado da etapa de segmentação pela cor vermelha

As imagens segmentadas possuem, portanto, o mesmo tamanho da imagem de entrada e estão normalizadas em preto e branco (zero ou um) e representam as regiões de interesse da imagem.

3.3 Extração de características

O algoritmo HOG (Histogram of Oriented Gradients) é usado com o objetivo de detectar objetos em uma imagem. A imagem é separada em vários blocos, e em cada bloco é calculada a distribuição local dos gradientes.

A saída do algoritmo HOG pode ser utilizada em um classificador para se obter o resultado de uma detecção. O classificador deverá ser treinado com exemplos positivos e negativos do tipo de objeto que se quer encontrar na imagem. O modelo deste trabalho utilizará um classificador binário amplamente utilizado nos probemas de visão computacional.

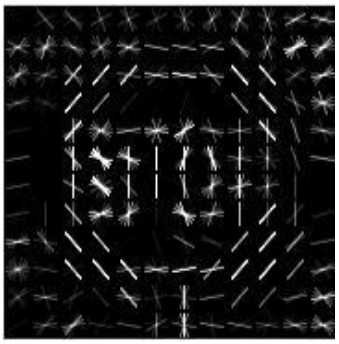

Figura 7: Resultado do HOG em uma placa de sinalização stop.

http://dx.doi.org/10.25286/repa.v3i2.572

Com a separação das formas encontradas na imagem na etapa de segmentação, a extração de características de cada região de interesse (e das combinações das mesmas) pode ser feita pelo HOG. Portanto, os descritores HOG de cada região de interesse da imagem pré-processada serão as saídas da etapa de extração de características.

3.4 Detecção da placa (sem perspectiva)

A teoria do modelo de classificação SVM (*Support Vector Machine*) é baseada na ideia de minimização do risco estrutural ou SRM (*Strucutral Risk Minimization*) **[2]**. Isso significa que a SVM é um classificador que busca minimizar o limite de risco esperado durante o seu treinamento.

No caso deste trabalho, a SVM serve para separar exemplos de placas de sinalização e exemplos de background **[2]**-**[9]**. O espaço de características é obtido através do HOG, e assim a quantidade de exemplos tem que ser o suficiente para minimizar os falsos positivos.

Com os descritores de cada região de interesse da imagem, a avaliação de cada região pode ser feita por um classificador HOG/SVM treinado. Se o resultado for da classe placa, então o modelo irá guardar a região da imagem pré-processada referente à placa, para ser realizada a detecção da perspectiva, ou seja, a posição da placa em relação ao eixo de captura. Portanto, as saídas dessa etapa de detecção da placa serão as imagens das placas retiradas da imagem pré-processada.

Figura 8: Resultado da etapa de detecção com o classificador HOG/SVM.

3.5 Detecção dos pontos de interesse

O SURF (*Speed Up Robust Features*) [13] é um extrator de características invariante à escala. Os pontos de interesse do objeto não mudam significativamente suas características ao serem aproximados ou afastados.

Para mensurar os pontos de interesse, o SURF busca regiões na imagem que se diferenciam em propriedades, tais como brilho ou cor, comparado a regiões próximas [13]. Em outras palavras, o SURF encontra diferentes regiões que possuem propriedades constantes ou aproximadamente constantes. Todos os pontos pertencentes a uma mesma região são parecidos entre si.

O detector SURF será aplicado em cada imagem de placa da etapa de detecção anterior. Porém, já que uma imagem digital é uma amostragem, um objeto em baixa resolução terá menos regiões ou pontos de interesse quando comparado ao mesmo em alta resolução. Desse modo, não poderá ser feita a detecção de pontos de interesse importantes da placa nos casos em que a mesma se encontra muito distante. As saídas da etapa de detecção dos pontos de interesse serão os pontos de interesse de cada placa detectada na etapa anterior.

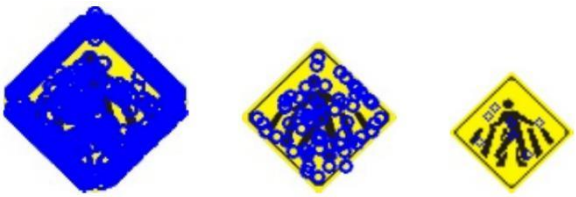

Figura 9: Diferença na detecção de pontos de interesses detectados pelo SURF, de acordo com a diminuição da resolução da imagem vista da esquerda para a direita.

3.6 Extração de descritores locais

Cada ponto de interesse terá seu descritor SURF calculado por meio da distribuição de intensidade dos pixels vizinhos. A quantidade de informação do descritor implica na acurácia da combinação de pontos e na complexidade computacional. Um descritor grande pode discriminar muito bem um determinado ponto, porém pode ser menos robusto quando há distorções na imagem.

Portanto, as saídas da etapa de descritores locais são os descritores de cada ponto de interesse nas imagens das placas. Se a placa detectada não possuir uma quantidade de pontos de interesse suficiente, o modelo proposto terá como saída somente as detecções obtidas na etapa de detecção descrita na seção 3.4

3.7 Detecção da placa (com perspectiva)

O FLANN (*Fast Library for Approximate Nearest Neighbors*) é uma biblioteca para rápida execução de busca por vizinhos próximos em espaços de alta dimensionalidade [14]. Isso significa que cada descritor de uma imagem poderá ser combinado com um descritor muito parecido de uma outra imagem.

Primeiramente, é criado um índice de pesquisa do conjunto de dados inicial de pontos, índice utilizado mais tarde para rápidas pesquisas por vizinhos mais próximos no conjunto de dados. Se a estrutura contém um campo que especifica o tipo de índice para criar, a função irá criar um índice de *kd-tree*. Se o índice e índice de *kd-tree* não são especificados diretamente, a função irá primeiro tentar detectar automaticamente o melhor índice e índice de *kd-tree* a ser usado para a busca do vizinho mais próximo no conjunto de dados fornecido.

Utilizando os descritores do SURF, se encontra um problema de falsos positivos, por causa de ambiguidades encontradas em pontos de interesse parecidos (porém não iguais). Para contornar este problema utiliza-se o algoritmo RANSAC. O RANSAC (*RANdom SAmple Consensus*) é um método iterativo que busca por *outliers* (valores atípicos) em uma base de dados [15]. No caso, a base de dados será constituída pelos pares de descritores encontrados pelo FLANN, e os *outliers* serão os falsos positivos.

Com o método RANSAC é possível encontrar a homografia da imagem da placa, que é o resultado da busca pelo plano de projeção da placa (perspectiva). O modelo do RANSAC busca o elemento de regularidade entre os pontos de interesse encontrados pelo FLANN, ou seja, os pontos devem se encontrar em um mesmo plano de projeção.

As saídas dessa etapa de detecção são os vértices do quadrilátero que representa a perspectiva de cada placa de sinalização detectada. Com isso, pode ser feito um ajuste de perspectiva na placa que melhora o desempenho de classificadores multi-classes, que definem o significado (conteúdo) de cada placa de sinalização.

http://dx.doi.org/10.25286/repa.v3i2.572

Entrada: Imagem colorida

Saída: *bounding boxes* e planos de perspectiva da placa (representados pelos 4 vértices de cada um dos planos)

1	Aplicar CLAHE na imagem de entrada nos canais L*a*b* para aumentar o ganho de cor.
2	Realiza a segmentação na imagem pré-processada pelo CLAHE nas componentes vermelha, verde, azul e amarelo.
3	Remove as bordas das imagens segmentadas, utilizando a negativa do detector de bordas Canny.
4	Aplica-se o modelo HOG/SVM treinado nas regiões segmentadas.
5	Retorna os *bounding boxes* das regiões que obtiveram saída positiva.
6	Calcula os pontos de interesse e os descritores SURF das imagens em tons de cinza e binarizadas dos *bounding boxes*.
7	Combina os pontos de interesse das regiões dos *bounding boxes* com as suas correspondências nas imagens de treinamento.
8	Calcula a homografia das placas detectadas com o método RANSAC.
9	Retorna os planos de perspectiva das placas detectadas.

Figura 10: Resumo do modelo proposto.

4 Experimentos

Um protótipo foi implementado em C ++ para testar e validar o modelo. O framework OpenCV[4] foi usado para evitar retrabalho. Este protótipo tem como entrada as imagens a serem processadas e classificadas, exibindo os sinais detectados na imagem.

A LTSD é uma base de dados de imagens em tons de cinza tiradas de ruas dos Estados Unidos que contém muitas imagens negativas, ou seja, imagens que não possuem placas sinalizadoras. Já a GTSD e a GTSR contêm imagens coloridas, de modo que a segmentação de cor pode ser aplicada. As 900 imagens da base de dados GTSD são escolhidas para testar o modelo. Na GTSD, o principal objetivo é detectar 3 tipos de sinais de trânsito: *Prohibitory*, *Mandatory* e *Danger*.

Para realizar o treinamento dos modelos de classificação, as bases de dados LTSD e GTSR foram utilizadas. Neste modelo, 11634 imagens negativas (fotos das ruas sem sinalização) da LTSD e todas as 51885 imagens positivas de GTSR são usadas para treinar os classificadores. A razão para isso é porque essas bases de dados possuem uma variedade maior de imagens, já que a competição GTSD oferece somente 900 imagens para treinamento e validação. Há 3 classes, de modo que 3 listas de descritores SURF são criados, e 3 modelos HOG / SVM são treinados.

O CLAHE utilizado na etapa de pré-processamento possui a parametrização padrão, com a janela mínima 8 por 8 e o corte de histograma de valor 8. Para o detector Canny, foram escolhidos os valores 100 e 200 para os limiares de valor mínimo e máximo respectivamente. Nas funções de limiarização da etapa de segmentação, foi assumido o valor 15 para todos os limiares. Esses valores foram assumidos após experimentos preliminares.

A implementação padrão do extractor de características HOG foi adotada. Os parâmetros foram definidos da seguinte forma: o tamanho da janela foi de 50 por 50; tamanho do bloco foi de 20 por 20; o tamanho da célula era de 10 por 10; *blockstride* foi de 10 por 10; orientação foi quantificada a 9 bins. Assim, a dimensão do vetor de característica é 576. Para diferentes escalas de sinais de trânsito detectadas pela segmentação de cor, a imagem é, em primeiro lugar redimensionada para caber o tamanho da janela HOG (50-por-50) e, em seguida, o descritor é calculado.

A implementação SVMLight[5] foi utilizada, pois o algoritmo utilizado tem requisitos de escalonamento de memória e pode lidar com problemas que contenham milhares de vetores de suporte. A SVM utilizada possui a parametrização padrão do SVMLight e configuração de *kernel* linear de aprendizado estatístico (redução do risco estrutural e empírico) C com valor 0,01. Todos esses valores foram assumidos após experimentos preliminares, tendo em vista o desempenho e a convergência do modelo de classificação durante a fase de treinamento. [11] deslizante). Os N' exemplos negativos que forem detectados pelo classificador (falsos positivos) são adicionados no conjunto de exemplos negativos (N+N') para um novo treinamento. Depois de K iterações, o detector está treinado. Essa abordagem é chamada de *hard negative mining* [16] e cada SVM é treinado para cada tipo de placa. Dessa forma, inicialmente o algoritmo recebe as imagens positivas de um tipo de placa da GTSR, determinadas para serem de treinamento, e as imagens negativas da LTSD. Após as interações a quantidade de imagens negativas serão suficientes para se obter uma boa taxa de classificação.

A detecção é considerada correta ou verdadeiro positivo quando as coordenadas correspondentes ao *bounding box* (região retangular que indica onde foi detectado o objeto) cobrirem pelo menos 60% da área do *ground truth*, que é a região da placa determinada pela base de dados, de acordo com o coeficiente de similaridade de Jaccard

$$J(S,G) = \frac{|S \cap G|}{|S \cup G|}$$

onde S e G são as regiões dos *bounding boxes* do modelo de detecção e do ground truth, respectivamente.

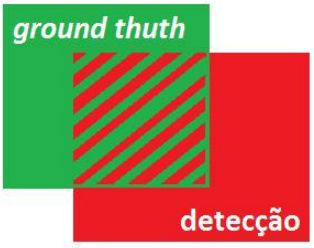

Figura 11: Coeficiente de similaridade de Jaccard.

A avaliação da detecção da placa na competição GTSDB é realizada baseada na curva de *precision-recall*, onde estes são calculados conforme a seguir:

$$recall = \frac{n^{o}\ de\ placas\ corretamente\ detecta}{n\acute{u}mero\ de\ placas} \quad (10)$$

$$precision = \frac{n^{o}\ de\ placas\ corretamente\ det}{n\acute{u}mero\ de\ placas\ detecta} \quad (11)$$

http://dx.doi.org/10.25286/repa.v3i2.572

O resultado é avaliado pela função padrão disponibilizada com a GTSD. Esse benchmark compara os métodos por meio da curva *precision-recall* (PR) e da área sob a curva (AUC). O modelo proposto obteve para o AUC de 98,27% para a classe *Prohibitory*, de 93,22% para a classe *Danger* e de 90,14% para a classe *Mandatory.*

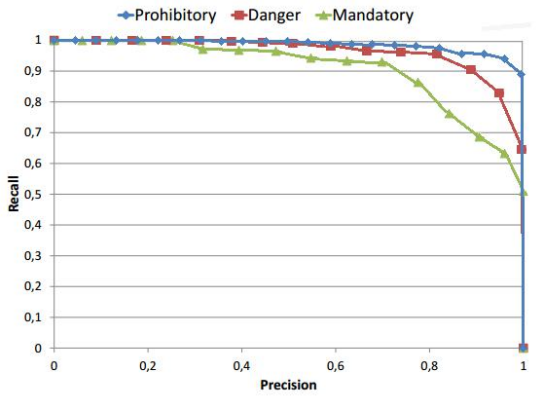

Figura 12: Precision-recall do modelo proposto no conjunto de teste do GTSD para cada superclasse

Os modelos do estado da arte que utilizam LDA (Linear Discriminant Analysis) [9][10], intChn (Integral channel features) [11] e HOG/SVM com extração de características de cor [10] atingiram um valor muito próximo de 100% da AUC.

Tabela 1: AOC's dos modelos propostos para cada classe de detecção com sua respectiva taxa de execução.

Método	P	D	M
HOG+LDA+SVM	100,00%	99,91%	100,00 %
HOG+SVM	100,00 %	98,85%	92,00%
intChn	100,00%	100,00%	96,98%
Color HOG+SVM	98,27%	93,22%	90,14%

Entretanto, o modelo proposto possui um tempo de execução muito inferior, devido ao baixo custo computacional das técnicas abordadas. A segmentação reduz a quantidade de imagens processadas pelo modelo de classificação HOG/SVM, evitando o processo de janela deslizante. Os principais modelos do estado da arte possuem fases de segmentação e de detecção de elevado custo computacional, aplicando vários filtros massivos em uma única imagem.

Método	Máxima taxa de quadros por segundo
HOG+LDA+SVM	abaixo de 1
HOG+SVM	1
intChn	2
Color HOG+SVM	acima de 70

Tabela 2: Modelos propostos com sua respectiva taxa de execução.

Após o teste, vimos que o protótipo não funciona bem em algumas situações nas quais a placa está inclinada, movimentada ou muito iluminada. Este comportamento acontece porque o modelo HOG/SVM, é robusto a rotação até o ponto em que os histogramas locais de cada região da imagem não coincidem. O modelo HOG/SVM também depende dos gradientes da placa, que podem ser degradados com a movimentação. Além disso, a segmentação de cor falha em alta luminosidade (saturação dos microssensores destinados a cada pixel). A taxa de erro aumenta nessas situações, porém ainda existe a detecção (verdadeiros positivos) nos casos em que a disposição dos gradientes e a coloração são suficientes para o modelo de classificação reconhecer a placa.

Figura 13: Exemplos de falhas na detecção (falsos negativos).

O *average overlap* representa a média do percentual da área da placa detectada em todas as imagens de teste, ou seja, os valores dos coeficientes de similaridade de Jaccard. Essa média inclui somente os verdadeiros positivos detectados pelos modelos de detecção. Isso

significa que quanto maior esse valor, a placa será enquadrada cada vez melhor. O valor 100% de average overlap indica que todas as regiões detectadas coincidem exatamente com o ground truth. O modelo proposto obteve 96,15%, 94,69% e 93,26% de average overlap na detecção do conjunto de teste para as classes *Prohibitory*, *Danger* e *Mandatory*, respectivamente.

O sistema desenvolvido obteve uma taxa de execução de 71-76 imagens por segundo no computador com processador Intel i7-4790 CPU 3.6GHz e 16GB de RAM DDR3, superando todas as 3 principais abordagens do estado da arte. A segmentação por cor com HOG-SVM e a detecção SURF-FLANN são fundamentais para se ter um rápido tempo de resposta. No estado da arte não foram encontrados trabalhos que tenham apresentado esta composição, tendo em vista que o modelo proposto busca identificar também a perspectiva da placa sinalizadora.

Foram realizados testes qualitativos do protótipo com uma webcam, e com isso, uma melhor avaliação da etapa de detecção de perspectiva foi possível. A Figura 14 mostra o resultado de uma das detecções realizadas com um exemplo de placa. O teste com a webcam foi realizado com poucas imagens nessa situação, somente para verificar o funcionamento do modelo no mundo real nessa situação. A webcam possui uma resolução de 800 por 600 *pixels* e uma taxa de captura de 30 quadros por segundo. Para uma melhor avaliação, se faz necessário experimentos com mais imagens e com uma maior variedade de situações.

Figura 14: Detecção de perspectiva durante o teste qualitativo realizado com uma webcam.

Também foram realizados testes com imagens do Google Street View[6] de Recife para verificar a aplicabilidade do modelo de detecção proposto. Mantendo a propriedade das placas do tipo *Prohibitory* das bases de dados GTSR e GTSD, os exemplos de detecção são mostrados na Figura 15. A avaliação foi realizada com 5 imagens do Street View. Para melhor avaliar o funcionamento do modelo no padrão de sinalização brasileiro, mais imagens são necessárias, sendo este um possível trabalho futuro.

Figura 15: Detecção de placas sinalizadoras durante o teste qualitativo com imagens do Google Street View.

5 Conclusão

A partir do protótipo desenvolvido, foi possível validar o modelo proposto. Observamos um funcionamento satisfatório ao

http://dx.doi.org/10.25286/repa.v3i2.572

detectar placas de sinalização em ambientes variados e em tempo real. Embora os experimentos tenham sido realizados com placas de sinalização alemãs, entende-se que o modelo funcionaria com outros tipos de placas, uma vez que os modelos de classificação são treinados com base nas características invariáveis das mesmas.

O modelo proposto falha na detecção em alguns casos que há borramento entre objetos próximos à placa de sinalização. Assim, necessita-se de uma melhoria na abordagem de detecção por cor, com a intenção de diminuir esse problema e aumentar os casos de detecção.

Os trabalhos futuros incluem o refinamento da detecção por cor, utilizando mecanismo de detecção de características geométricas da placa para separação das diferentes formas. Esse tipo de detecção é utilizado em aplicações de realidade aumentada, para encontrar padrões geométricos que aparecem na imagem.

A distância da placa de sinalização é uma informação importante a ser investigada nos trabalhos futuros. Imagens de sensores RGB-D possuem a distância de cada *pixel* podendo ser utilizada na segmentação. Até mesmo com duas ou mais câmeras convencionais é possível extrair a distância dos objetos por meio da estereoscopia, que é uma técnica de cálculo de distância bastante utilizada nos problemas de detecção de objeto.

Com a finalidade de verificar a eficácia do modelo, se faz necessário compará-lo com outros trabalhos que abordam outras bases de dados. Essa comparação constitui um dos trabalhos futuros.

Referências

[1] LIM, Kwangyong et al. Real-time traffic sign recognition based on a general purpose GPU and deep-learning. **Journal PLoS one**, v. 12, n. 3, p. e0173317, 2017.

[2] LAFUENTE-ARROYO, S. et al. Traffic sign shape classification evaluation I: SVM using dis-tance to borders. In: IEEE Proceedings. Intelligent Vehicles Symposium, 2005. **Anais**... IEEE, 2005. p. 557-562.

[3] VIRUPAKSHAPPA, Kushal; HAN, Yan; ORUKLU, Erdal. Traffic sign recognition based on prevailing bag of visual words representation on feature descriptors. In: 2015 IEEE Interna-tional Conference on Electro/Information Tech-nology (EIT). **Anais**... IEEE, 2015. p. 489-493.

[4] WANG, Gangyi et al. A robust, coarse-to-fine traffic sign detection method. In: Neural Net-works (IJCNN), The 2013 International Joint Conference on. **Anais**... IEEE, 2013. p. 1-5.

[5] ZAKLOUTA, Fatin; STANCIULESCU, Bog-dan. Warning traffic sign recognition using a HOG-based Kd tree. In: Intelligent Vehicles Symposium, 4., 2011. **Anais**... IEEE, 2011. p. 1019-1024.

[6] HOUBEN, Sebastian. A single target voting scheme for traffic sign detection. In: Intelligent Vehicles Symposium, 6., 2011. **Anais**... IEEE, 2011. p. 124-129.

[7] ALTI, Samuele et al. A traffic sign detection pipeline based on interest region extraction. In: Neural Networks (IJCNN), The 2013 Interna-tional Joint Conference on. **Anais**... IEEE, 2013. p. 1-7.

[8] AGRAWAL, Subhash Chand; JALAL, Anand Singh; TRIPATHI, Rajesh Kumar. A Hybrid Method for Image Categorization Using Shape Descriptors and Histogram of Oriented Gradients. In:RAMAN, B. ET AL (Eds.) **Proceedings of International Conference on Computer Vision and Image Processing:** CVIP 2016, v.1. Singapore: Springer, 2017. p. 285-295.

[9] HOUBEN, Sebastian. A single target voting scheme for traffic sign detection. In: Intelligent Vehicles Symposium,4., 2011 **Anais**... IEEE, 2011. p. 124-129.

[10] LIANG, Ming et al. Traffic sign detection by ROI extraction and histogram features-based recognition. In: Neural Networks (IJCNN), The 2013 International Joint Conference on, 2013. **Anais**...IEEE, 2013. p. 1-8.

[11] MATHIAS, Markus et al. Traffic sign recognition—How far are we from the solution?. In: Neural Networks (IJCNN), The 2013 Interna-tional Joint Conference on. 2013. **Anais**... IEEE, 2013. p. 1-8.

[12] ZUIDERVELD, Karel. Contrast limited adaptive histogram equalization. In: HECKBERT, P. S. (Ed.) **Graphics gems IV**. San Diego: Academic Press Professional, 1994. p. 474-485.

[13] KARAMI, Ebrahim; PRASAD, Siva; SHE-HATA, Mohamed. **Image matching using SIFT, SURF, BRIEF and ORB**: Performance compari-son for distorted images. arXiv preprint arXiv:1710.02726, 2017.

[14] HAN, Yan; VIRUPAKSHAPPA, Kushal; ORUKLU, Erdal. Robust traffic sign recognition with feature extraction and k-NN classification methods. In: Electro/Information Technology (EIT), 2015 IEEE International Conference on, 2015. **Anais**... IEEE, 2015. p. 484-488.

[15] KIM, Seong-Uk; LEE, Joon-Woong. Traffic Sign Recognition, and Tracking Using RANSAC-Based Motion Estimation for Autono-mous Vehicles. **Journal of Institute of Control**, Robotics and Systems, v. 22, n. 2, p. 110-116, 2016.

[16] HENRIQUES, Joao F. et al. Beyond hard nega-tive mining: Efficient detector learning via block-circulant decomposition. In: proceedings of the IEEE International Conference on Computer Vision, 2013. **Anais**... IEEE, 2013. p. 2760-2767.

http://dx.doi.org/10.25286/repa.v3i2.572