PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Programa de Pós-Graduação em Informática

Rafael Henrique Brasil

DETECÇÃO DE AVANÇO DE SEMÁFORO VERMELHO UTILIZANDO CÂMERA EMBARCADA EM VEICULOS AUTOMOTORES

Belo Horizonte
2015

Rafael Henrique Brasil

DETECÇÃO DE AVANÇO DE SEMÁFORO VERMELHO UTILIZANDO CÂMERA EMBARCADA EM VEICULOS AUTOMOTORES

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. Alexei Manso Correa Machado

Belo Horizonte

2015

FICHA CATALOGRÁFICA
Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

Brasil, Rafael Henrique
B823d Detecção de avanço de semáforo vermelho utilizando câmera embarcada em veículos automotores / Rafael Henrique Brasil. Belo Horizonte, 2016.
52 f. : il.

Orientador: Alexei Manso Correa Machado
Dissertação (Mestrado) – Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Engenharia Elétrica.

1. Trânsito - Sinais e sinalização. 2. Detectores. 3. Sistemas inteligentes de veículos rodoviários. 4. Processamento de sinais - Técnicas digitais. 5. Câmaras de vídeo. 6. Percepção de padrões. I. Machado, Alexei Manso Correa. II. Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDU: 681.3.093

Rafael Henrique Brasil

DETECÇÃO DE AVANÇO DE SEMÁFORO VERMELHO UTILIZANDO CÂMERA EMBARCADA EM VEICULOS AUTOMOTORES

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica como requisito parcial para qualificação ao Grau de Mestre em Engenharia Elétrica pela Pontifícia Universidade Católica de Minas Gerais.

Prof. Dr. Alexei Manso Correa Machado (Orientador) – PUC Minas

---

Prof.ª Dr.ª Flávia Magalhães Freitas Ferreira – PUC Minas

Prof. Dr. Erickson Rangel do Nascimento – UFMG

Belo Horizonte, 21 de maio de 2015.

Aos meus pais, Leni e Mário, por todo incentivo e inspiração e pela fé que em mim depositaram.

AGRADECIMENTOS

Agradeço a todos direta e indiretamente envolvidos neste trabalho. Primeiramente a Deus por todas as bênçãos concedidas até o momento presente. Agradeço ao meu orientador, o Professor Alexei, por todo o incentivo, pela oportunidade e pelos novos conhecimentos. Agradeço também aos colegas de laboratório pela companhia prolongada e pelo apoio. Agradeço à Capes pela pela bolsa e, consequentemente, pela oportunidade. Agradeço aos colegas de trabalho, que compreenderam minhas indisponibilidades. E finalmente, agradeço aos meus pais, avós, irmãos e demais familiares, que sempre me motivam e, principalmente, compreendem minhas faltas.

"Os problemas significativos que enfrentamos não podem ser resolvidos no mesmo nível de pensamento em que estávamos quando os criamos."

Albert Einstein

RESUMO

O avanço de semáforo vermelho é uma infração muito comum. Atualmente o avanço de semáforos vermelhos é feito com sensores fixados na via. Contudo, em cidades como Belo Horizonte, uma pequena porcentagem dos semáforos são equipados com tais sensores. Por esse motivo, este trabalho propõe que a detecção de avanço de semáforo vermelho seja feita a partir de um sistema composto por uma câmera e um computador embarcados no veículo. Um algoritmo também é proposto para processar os videos gravados pela câmera e um protótipo foi implementado. O sistema captura imagens coloridas, utiliza filtro de média, erosão e dilatação para reduzir ruídos e busca por semáforos com a luz vermelha acesa com a imagem no espaço HSV. Detectado um semáforo vermelho, ele é rastreado usando Camshift. O objetivo do protótipo é monitorar veículos de trabalho, já que não aparenta ser atrativo para veículos pessoais de passeio. Além disso, não há qualquer intervenção ativa na condução, agindo em caráter meramente educativo. Trabalhos relacionados já lidam com detecção e reconhecimento de semáforos com câmera embarcada. Também há trabalhos que detetam avanço de semáforo vermelho com visão computacional, mas nenhum trabalho encontrado une as duas técnicas, propondo a detecção de avanço de samáforo com a câmera embutida no veículo. Testes são realizados com vídeos previamente gravados nas ruas de Belo Horizonte e com um vídeo de *benchmark* utilizando o protótipo implementado. Os resultados são comparados com base no tempo de execução, na exatidão e na taxa de erros.Com o vídeo na sua taxa de quadros original, o tempo de processamento demorou mais que a duração do vídeo. Porém, o processamento do vídeo em 5fps levou aproximadamente 1 minuto para cada 6 minutos do vídeo gravado. Para o vídeo de Belo Horizonte, taxa de acertos ficou superior a 95,8% durante o dia e acima de 93,7% para o vídeo noturno. Já no vídeo de *benchmark*, a taxa de acerto limitou-se a 64%, ainda sendo um valor aceitável. A principal contribuição consiste na nova forma de detectar os avanços mesmo nos semáforos sem detectores, dado que não foram encontrados trabalhos que utilizam o mesmo princípio.

Palavras-chave: Avanço de semáforo, Avanço de sinal, semáforo vermelho.

ABSTRACT

The red traffic light advance is a very common traffic viollation. Nowadays, vehicles running red traffic lights is detected by sensors fixed on the streets. However, in cities like Belo Horizonte a very small percentage of all traffic lights are equipped with such sensors. For this reason, this work proposes a red light runner detection to be performed by a system that consists of a camera and a computer embedded in the vehicle. An algorithm is also proposed to process the recorded videos and a prototype was implemented. The system captures colored images, applies mean filter, erosion and dilation to reduce noises and searches for traffic lights with the red light on using the HSV color space. Once the light is detected, it's tracked using Camshift. The prototype's goal is to monitor work vehicles since it's not attractive to ride cars. Also there is no intervention in driving, acting only in a educational way. Related works already deal with detection of traffic lights with embedded camera. There are also works that detect traffic light runners using Computer Vision, but none of them use both tecniques together to propose a traffic light runner detection using embedded camera in the car. Tests are performed with videos recorded in the streets of Belo Horizonte and in a benchmark video using the implemented prototype. The results are compared based in execution time, accuracy and errors rate. With the video in it's original frame rate, the time took to process was longer than the video duration. However, the video processing in 5fps took about one minute for every 6 minutes processed from the recorded video. The hit rate for the video from Belo Horizonte is above 95.8% in daytime and above 93.7% over night. For the benchmark video the hit rate is limited to 64%, but it's still acceptable. The main contribution is the new way to detect advances even at traffic lights without detectors since there were no works found that use the same principle.

Keywords: Running red light, Red light runners, Red traffic light.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

Com o crescente aumento da quantidade de veículos produzidos a cada ano, a indústria automotiva tem investido em sistemas de transporte mais inteligentes com o objetivo de auxiliar o condutor a tomar decisões e a pensar em tempo hábil em situações de risco, ajudando a poupar vidas e recursos. Um sistema que dê ao veículo a capacidade de perceber e interpretar o que há à sua volta pode melhorar diretamente a segurança desses condutores.

No trânsito cotidiano, pode acontecer de um condutor desatento deixar alguma sinalização passar despercebido. Da mesma forma, alguns condutores intencionalmente desrespeitam as leis de trânsito. Veículos modernos já incluem diversos sistemas de segurança, mas mesmo uma colisão frontal entre dois veículos a 40 km/h cada pode ter consequências dramáticas.

Sensores avançados embutidos, tais como sonares, radares e câmeras ganharam mais importância com intuito de garantir mais segurança na condução de veículos motorizados através do suporte ao condutor e no desenvolvimento de veículos autônomos. Um sistema com transmissores nas placas, semáforos e veículos que fosse capaz de transmitir informações entre estes elementos seria altamente eficaz, mas isso exigiria um transmissor em cada sinalização de trânsito para mermitir essa comunicação. A mesma limitação se aplica quando se trata da comunicação entre os veículos.

Como as características do trânsito e das vias diferem de uma via para outra, além da dificuldade de se adaptar a sinalização em uma cidade inteira, diversos sensores trabalham com a visão computacional, seguindo a teoria que a informação que um humano pode captar visualmente também pode ser usada pelo sistema que o auxilia. Dai a razão pela qual este trabalho lida com Visao Computacional para interpretar a sinalização e registrar uma possível má conduta diante de um semáforo vermelho.

1.1 Motivação e Objetivo

Diversos radares e sensores eletrônicos estão instalados em vias e pontos estratégicos de Belo Horizonte, tomando como base estudos técnicos que detectam áreas com maior fluxo de pedestres, o tipo e a quantidade de acidentes, entre outros. A regra, em geral, é a mesma aplicada ao longo das principais vias brasileiras.

As infrações mais frequentes registradas no primeiro semestre de 2013 em Belo Horizonte foram avanço de semáforo vermelho e estacionamento em local indevido. O estacionamento indevido é considerado infração leve, enquanto avanço sinal vermelho é infração gravíssima. A quantidade de infrações por avanço de sinal vermelho ter sofrido uma grande queda de 80% no primeiro semestre de 2014 em relação ao mesmo período

do ano anterior, de 123.878 para 25.470 (PARANAIBA; HOLANDA, 2014). Contudo, essa redução nos registros não necessariamente significa que as infrações não autuadas caíram na mesma proporção. Motoristas acabam fazendo um mapeamento dos semáforos que têm registro de avanço e, consequentemente, sabem onde "podem" ou não desrespeitar o semáforo.

Atualmente, a cidade conta com cerca de 47 equipamentos de registro de avanço de semáforo vermelho. Os aparelhos apresentam ótima precisão, às vezes aliada ao auxílio humano, como em Contagem (ANDRADE, 2011). Todavia, o custo elevado da instalação de tais equipamentos em todas as interseções com semáforo da cidade torna o projeto inviável num primeiro momento, motivo pelo qual as interseções candidatas são selecionadas, com base em dados estatísticos, para receberem os equipamentos de registro de avanço.

Com um sistema de detecção de avanço de sinal vermelho embarcado no veículo, o condutor poderia ter melhor conduta e respeitar a sinalização independentemente da existência de oficiais militares ou de radares fixos na via simplesmente por saber que está sendo monitorado. Com isso, poderia haver uma redução nas estatísticas de avanço e, principalmente, de acidentes.

Uma mesma câmera pode ser adotada para diversos fins. Dessa forma, se já houver uma câmera no veículo para outros sistemas (câmeras de segurança, por exemplo), o mesmo vídeo pode ser usado no processamento para verificação do avanço de semáforo. Por outro lado, o acoplamento de uma nova câmera num veículo permite que o mesmo vídeo seja adotado para outros fins futuramente.

O objetivo deste trabalho é propor uma técnica para a detecção de avanço de semáforo vermelho. A detecção passa a ser, portanto, um monitoramento do veículo, não mais da via ou de uma interseção específica. O alvo desse trabalho são principalmente veículos de uso em trabalho, tais como caminhões, táxis, ônibus e carros de empresas. Afinal, para um veículo particular, de uso exclusivamente pessoal, a solução proposta é pouco atrativa.

Este trabalho aborda desde a captura das imagens até a detecção (ou não) do avanço, não se preocupando se algum alerta será emitido ao usuário. Também não há qualquer integração entre o sistema experimental e o veículo, nem para obtenção de informações, tampouco para interferir na direção. Há preocupação com o tempo de processamento, já que este trabalho visa o processamento em tempo real. Contudo, não há processamento em tempo real nos testes realizados e descritos nos próximos capítulos, uma vez que a máquina usada para processar os vídeos não está embarcada. Os vídeos foram capturados para processamento posterior.

1.2 Descrição do problema

Reconhecimento de sinalização de trânsito é uma aplicação da Visão Computacional que tem como objetivo a detecção automática e reconhecimento de sinalização de trânsito de imagens obtidas por uma câmera acoplada a um veículo em movimento. A utilização de imagens e da visão computacional pode ser um grande desafio devido a fatores externos do ambiente, tais como alterações na iluminação, sombras, rotações, oclusões, objetos similares a sinalização no ambiente, entre outros.

Para melhor entendimento do problema, alguns conceitos usados em todo este trabalho são:

a) Semáforo: é todo o semáforo suspenso na via, incluindo o anteparo preto e as três luzes (vermelha, amarela e verde);

b) Luz: é qualquer fonte de luminosidade, incluindo a lâmpada do semáforo acesa em dado momento;

c) Luz acesa: é a luz atualmente acesa num semáforo. Sabe-se que somente uma das luzes do semáforo fica acesa por vez;

d) Semáforo vermelho: é um semáforo onde a luz superior, ou seja, a vermelha, está acesa. Essa luz acesa é a parte mais importante do semáforo para os testes propostos neste trabalho.

Sabendo que, atualmente, registros de avanço de sinal são feitos a partir de equipamentos fixos nas vias, este trabalho propõe como problema a detecção de avanço de semáforo vermelho com base na análise de imagens capturadas com uma câmera embarcada no próprio veículo. A Figura 1 mostra os passos que compõem o processo de detecção de avanço de sinal.

a) Aquisição: a aquisição das imagens é feita com uma câmera de vídeo localizada no interior do veículo capaz de captar imagens coloridas. A câmera deve estar sempre direcionada para a frente. O vídeo capturado é armazenado para o processamento do sistema. A Figura 2 exibe um exemplo de um quadro capturado de uma câmera embarcada no veículo;

b) Pré-processamento: a fase de pré-processamento é encarregada de eliminar eventuais ruídos oriundos do processo de aquisição, tais como chuviscos e pequenas regiões com cores similares às luzes do semáforo.

c) Segmentação: determina quais regiões da imagem são relevantes para se prosseguir com o processamento. A segmentação é feita com base no tom (junção de matiz, saturação

Figura 1 – Passos do processamento das imagens capturadas

Fonte: Elaborado pelo autor

Figura 2 – Exemplo de imagem capturada do interior do veículo. A câmera deve estar voltada para a frente, de forma que tenha plena visão da sinalização.

Fonte: Charette (2013)

e brilho) das cores das luzes do semáforo, já que os semáforos têm tons aproximados de verde, amarelo e vermelho. A Figura 3 mostra a segmentação de uma imagem capturada a partir do tom de verde esperado num semáforo. Observe que, apesar de a árvore tambem ser verde, a segmentação é capaz de distingui-la do semáforo de acordo com a matiz, saturação e valor (*Hue, Saturation, Value* - HSV) dos *pixels* ao redor de cada objeto;

d) Detecção e Reconhecimento: consiste em detectar se uma região de interesse realmente é um semáforo. Havendo mais de um, esta fase já se encarrega de eleger o semáforo mais próximo do veículo, por exemplo, com base na altura do sinal dentro da imagem. Na Figura 4 pode-se ver um exemplo de semáforo detectado a partir da imagem segmentada na Figura 3;

e) Rastreamento: uma vez detectado um semáforo vermelho, ele é rastreado na imagem até que desapareça da região de rastreamento, ou seja, nas proximidades da região detectada no quadro anterior, se houver. O semáforo pode ter alternado para a cor verde ou ter sido ignorado pelo condutor. Nesse último caso, o avanço é contabilizado.

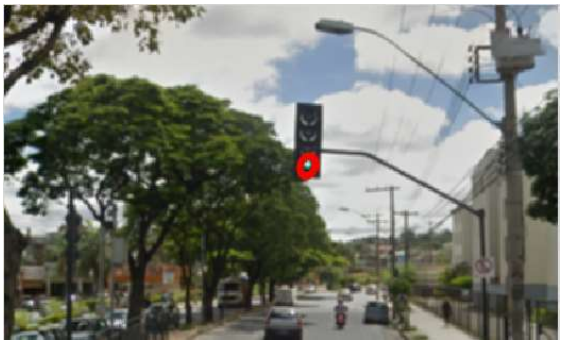
1.3 Organização do texto

No próximo capítulo, algumas técnicas de detecção e reconhecimento de sinalização de trânsito são apresentadas. O capítulo seguinte detalha os métodos e técnicas adotados, seguido de experimentos e conclusão. A análise do resultado deve considerar a quantidade de semáforos pelos quais o veículo passou, a quantidade de avanços detectados e a quantidade de avanços omitidos.

Figura 3 – À esquerda, imagem original. À direita, região de interesse estimada com base na cor caso a cor verde fosse a de interesse.

Fonte: Imagem capturada pelo autor

Figura 4 – Exemplo de semáforo detectado. A lâmpada destacada será monitorada e rastreada no passo posterior.



Fonte: Testes da pesquisa

2 TRABALHOS RELACIONADOS

Há diversos trabalhos relacionados com detecção e reconhecimento de sinalização de trânsito em geral utilizando processamento de imagens. Duas formas muito comuns para se iniciar a detecção e reconhecimento de sinais de trânsito através de imagens são a segmentação com base na cor e a segmentação com base em bordas. As técnicas que utilizam cores tendem a ter melhor controle sobre a iluminação. Um exemplo utilizando o espaço de cores HSV é descrito por Fleyeh (2006).

Os trabalhos de Paulo e Correia (2007), Soetedjo e Yamada (2005), Souki, Boussaid e Abid (2008) e Xu (2009) propõem métodos de detecção e reconhecimento de placas de trânsito. No trabalho de Paulo e Correia (2007) a análise da sinalização é dividida é três estágios: (1) detecção, responsável por definir regiões de interesse nas imagens; (2) classificação, onde a sinalização é classificada como obrigação ou proibição; e (3) reconhecimento, que identifica exatamente a sinalização em questão. A detecção e a classificação são feitas com base na cor (segmentação de cores) e na forma (círculos, triângulos, octógonos e quadrados).

Uma boa estratégia adotada por Paulo e Correia (2007) e Xu (2009) para identificação da cor da sinalização é converter a imagem de um espaço RGB para um espaço HSV, assim como neste trabalho, para desassociar as informações de cor e intensidade. Isso permite trabalhar na imagem minimizando o impacto da luminosidade ambiente. Paulo e Correia (2007) propõem valores ideais de H e S (espaço HSV) para identificar cores; uma transformada *Fast Radial Symmetry* (FRS - Simetria radial rápida) é utilizada para identificar círculos, e a detecção de cantos de Harris para identificar triângulos e quadriláteros. Os resultados chegam a ser satisfatórios, com detecções de 93,1% e 97,7% dos sinais em baixa e média luminosidades, respectivamente, mas com apenas 29.4% com iluminação em excesso. A falha mais recorrente (mais de 46% das vezes) é reconhecer octógonos (placas de PARE) como círculos. No trabalho de Xu (2009), sinais agrupados são separados e analisados de forma individual. Os limites externos de placas parcialmente oclusas são extraídos utilizando evolução de curvas discretas. Uma otimização é aplicada nos limites externos dos objetos candidatos antes de fazer a comparação com os padrões. A técnica se mostra eficiente, uma vez que é necessário apenas um padrão para cada forma. Além disso, o reconhecimento através de formas individuais torna o reconhecimento invariante à translação, rotação e escala, também lidando melhor com imagens de placas parcialmente oclusas. Ainda assim, a oclusão ainda é um problema, representando taxas de erro superiores a 16% em imagens capturadas durante o dia e mais de 25% em imagens noturnas.

Soetedjo e Yamada (2005) comparam abordagens utilizando-se correspondência com padrão (comparação de candidatos com as placas) e Algoritmos Genéticos (GA - *Genetic Algorithm*), onde a primeira é sensível a rotações e escala, exigindo um grande

número de padrões, e a segunda exige muito esforço computacional. Dessa forma, uma nova técnica é proposta para se detectarem placas compostas por círculos vermelhos, apesar de a técnica ser expansível para outras placas. A técnica utilizada para encontrar elipses (uma placa redonda se apresenta como uma elipse para um observador em posição oblíqua em relação à placa) foi chamada de fragmentação geométrica. Tal técnica consiste em extrair pontos das bordas dos objetos da imagem utilizando GA e trabalha em cada metade da elipse separadamente. Dessa forma, aprimora-se o tempo de processamento e o tratamento de imagens parcialmente oclusas (considerando que mais da metade da placa estará visível). Como resultados, uma média de mais de 86% das placas é detectada corretamente em um tempo médio de 4 segundos por imagem, um tempo considerado alto em relação à proposta de Souki, Boussaid e Abid (2008), em que o tempo se mantém numa média de 3 segundos/imagem. A fase de detecção descrita por Souki, Boussaid e Abid (2008) consiste em procurar regiões de interesse a partir de características das placas, como cor e forma. A forma circular é buscada pela transformada de Hough por ser um método menos sensível a dados imperfeitos e ruídos, chegando a reconhecer alguns objetos semiocultos. Em seguida, na fase de reconhecimento, cada área de interesse candidata passa por um filtro de mediana, para reduzir ruído, e por binarização. Os zeros são removidos das placas por serem considerados informação redundante na maioria das possíveis placas de velocidade e, por fim, a placa é testada contra um conjunto de padrões para decidir se é ou não sinalização de trânsito e qual é. Os resultados revelam que a transformada de Hough representa a maior parte do tempo de processamento tanto no computador de teste quanto no hardware proposto.

O trabalho de Fleyeh (2006) lida com imagens em variadas condições de luz (alto brilho, sobras parciais, sombra total, noite, etc.). O foco não é o reconhecimento, mas a segmentação das cores somente. As imagens são previamente processadas, realçadas e segmentadas de acordo com as propriedades das placas, tais como cor e forma. Na fase de reconhecimento, cada área candidata é testada de forma a se decidir se é uma placa ou não. O comportamento da imagem é comparado diante de quatro espaços de cores: RGB, NRGB, intensidade e HSV, onde o espectro HSV se mostra mais invariante à luminosidade. O trabalho então é realizado na imagem no espaço HSV normalizado. Os resultados chegam a revelar acerto na segmentação em mais de 95% dos casos investigados.

Os trabalhos de Fairfield e Urmson (), Shen et al. (2009), Levinson et al. (2011) tratam da detecção e reconhecimento de semáforos de trânsito. Fairfield e Urmson () extraem, de uma imagem de 5 megapixels, uma região de 2040 x 1080, resolução suficiente para permitir detecção a 150m do veículo. Esta distância é considerada razoável para se parar o veículo a uma velocidade de 90 km/h. Considerando que semáforos costumam ser localizados em cruzamentos, Fairfield e Urmson (), Levinson et al. (2011) e Almagambetov, Velipasalar e Baitassova (2015) utilizam consultas geoespaciais, através da API do

Google Maps, para descartar imagens distantes de cruzamentos. Todavia, esta regra não se aplica no Brasil, onde, em vários lugares, é comum se encontrarem semáforos em uma simples travessia de pedestres. Além disso, o sistema se tornaria altamente dependente da conexão com um *Global Positioning System* (GPS - Sistema de Posicionamento Global). Imagens são rotuladas a partir de fontes de luz vermelhas, amarelas ou verdes. A distância do semáforo é estimada com base em cálculos envolvendo o tamanho real dos objetos. Os resultados apresentam alta taxa de falsos negativos (cerca de 38% de todos os positivos reais). Shen et al. (2009) propõem uma modelagem de matiz e saturação (espaço HSI) com base em distribuições Gaussianas 2D, onde os parâmetros são obtidos a partir de candidatos rotulados manualmente. Adicionalmente, é proposto um método para eliminar falsos candidatos com base em informações de forma numa fase de pós-processamento da imagem. O resultado final é uma combinação de informações vindas do pós-processamento e de uma base de histórico. Os resultados dessa abordagem, no entanto, chegam a apresentar uma precisão superior a 99% para alguns vídeos.

O trabalho de Almagambetov, Velipasalar e Baitassova (2015) visa suprir a limitação de condutores com deficiências visuais que causem distorções nas cores, como daltonismo, por exemplo. As imagens são convertidas para o espaço HSV como neste trabalho. Uma cópia das imagens são mantidas em RGB e cada canal (R, G e B) é separado para cálculo do gradiente para detecção de bordas. A detecção envolve a via, a haste do semáforo e o próprio semáforo utilizando a Transformada de Hough. Um sinal sonoro é emitido ao condutor dependendo da cor do semáforo, sendo um sinal diferente para cada uma das três cores. O procedimento leva um tempo médio de 275 milissegundos por quadro (menos de 4 quadros por segundo, em média) e, no geral, superou 96% de acerto na detecção.

Diferentemente de Fairfield e Urmson () e Shen et al. (2009), onde a câmera está localizada no veículo, um método de detecção de avanço de sinal vermelho é proposto por Yung e Lai (2001) e Luo, Huang e Qin (2008) com a câmera localizada em um ponto fixo da via de onde sejam visíveis o semáforo, a faixa de retenção e os carros de passagem. O método de Yung e Lai (2001) identifica o semáforo mais próximo (o maior na imagem) e o utiliza no processamento. A detecção é feita com base na cor: branca para faixa de retenção e vermelha, amarela e verde para o semáforo. Os resultados dos testes alcançaram 100% de acerto nos testes. Luo, Huang e Qin (2008) realizam os testes em uma interseção com direita livre, ou seja, onde o semáforo não é válido para convergência à direita (somente para os veículos que convergem à esquerda ou seguem em frente). Os resultados apresentam precisão um pouco inferior ao trabalho de Fairfield e Urmson (), em torno dos 90%.

O rastreamento do semáforo é tratado como foco principal nos trabalhos de Gong et al. (2010) e Yelal et al. (2006). Gong et al. (2010) utilizam um método já existente cha-

mado Camshift (BRADSKI, 1998), enquanto Yelal et al. (2006) propõe um novo método. Fleyeh (2006) e Gong et al. (2010) utilizam espaço de cores HSV para buscar possíveis semáforos, aplicam erosão e dilatação para reduzir ruídos e fazem o reconhecimento com algoritmo baseado em aprendizado de máquina. Por fim, o semáforo reconhecido é rastreado na imagem usando Camshift, que é comparado com a simulação de Monte Carlo por Cadeias de Markov (TU; LI, 2000) e é tido como o mais rápido entre ambos. Por sua vez, o método proposto por Yelal et al. (2006) utiliza detecção das bordas da rodovia para estimar as possíveis posições de um semáforo, considerando a haste que o segura. O semáforo é detectado e rastreado independentemente da cor da luz acesa no momento.

O Quadro 1 contem um resumo de todos os trabalhos relacionados citados neste capítulo. As principais técnicas são mencionadas pelo nome (quando houver). O objetivo de cada trabalho é marcado com um X na respectiva coluna.

Quadro 1 – Resumo dos trabalhos relacionados

Autores	**Detecção**			**Fixo na via ou embarcado**	**Principais técnicas**
	Sinalização	Semáforo	Avanços		
Paulo e Correia (2007)	X			Embarcado	FRS, Harris, Evolução de curvas discretas
Soetedjo e Yamada (2005)	X			Embarcado	Correspondência com padrão, GA, Fragmentação geométrica
Souki, Boussaid e Abid (2008)	X			Embarcado	Hough, filtro mediana
Xu (2009)	X			Embarcado	Evolução de curvas discretas,comparação com padrões
Fleyeh (2006)	X			Embarcado	NRGB
Fairfield e Urmson ()		X		Embarcado	GPS, Transformação linear dircta, kd tree
Shen et al. (2009)		X		Embarcado	Distribuições Gaussianas
Levinson et al. (2011)		X		Embarcado	GPS
Almagambetov, Velipasalar e Baitassova (2015)		X		Embarcado	GPS, Hough
Yung e Lai (2001)			X	Fixo	Detecção de bordas, detector de loop virtual
Luo, Huang e Qin (2008)			X	Fixo	Hough
Gong et al. (2010)		X		Embarcado	Camshift, erosão, dilatação, Monte Carlo
Yelal et al. (2006)		X		Embarcado	CSTRV, La*b*
Este trabalho			X	Embarcado	Filtro média, Hough, Camshift

Fonte: Levantamento bibliográfico

3 METODOLOGIA

Sistemas de detecção de avanço se semáforo vermelho usualmente utilizam câmeras fixadas em pontos estratégicos da via, tornando-se parte do sistema de controle do trânsito naquele local. O problema proposto nesta parte do princípio de que a câmera seja embarcada como parte do veículo, transformando o sistema em um modo de monitoramento do veículo e do condutor.

Um sistema de detecção de semáforos deve ser robusto o suficiente para não se enganar com luzes vermelhas originadas em fontes diversas, como lanternas de veículos, outdoors, luzes na calçada, etc. Além disso, o desaparecimento de um semáforo, independente de qual luz encontra-se acesa, não indica necessariamente que o veículo passou por ele, pois pode ser apenas uma oclusão momentânea. É indispensável a este trabalho a capacidade de rastrear um semáforo já detectado para casos de oclusão, evitando falsos positivos, e para otimizar a varredura na imagem, mantendo o processamento somente na região próxima à detecção do quadro anterior.

3.1 Fases do processo

A técnica aplicada para a resolução do problema é dividida nas seguintes fases:

a) Aquisição: fase responsável por capturar as imagens que serão processadas quadro a quadro;

b) Pré-processamento: elimina eventuais ruídos oriundos do processo de aquisição;

c) Segmentação: a fase de segmentação é responsável por eleger candidatos a partir de áreas de interesse na imagem capturada;

d) Detecção e reconhecimento: verifica os candidatos e elege um semáforo na imagem;

e) Rastreamento: monitora a posição do semáforo e a cor acesa, detectando um possível avanço e as mudanças de cores em um mesmo semáforo.

As fases de Aquisição, Pré-processamento e Segmentação utilizam técnicas que já costumam ser utilizadas por vários outros trabalhos relacionados. Já a Detecção e Reconhecimento e o Rastreamento utilizam de técnicas que, apesar de serem genéricas, se fizeram necessárias para lidar com o problema deste trabalho.

3.1.1 Aquisição

O veículo monitorado deve possuir uma câmera embarcada capaz de armazenar os vídeos para processamento posterior. As imagens são capturadas para processamento

futuro, ou seja, essa fase não ocorre paralelamente com as demais. Os vídeos devem ser gravados com cores e com resolução mínima de 640x480. A câmera deve, ainda, ter um ângulo de visão de pelo menos 60º, para que seja capaz de capturar o semáforo mesmo se o veículo for o primeiro a parar sob ele.

3.1.2 Pré-processamento

O pré-processamento prepara a imagem para a próxima fase. Aplica na imagem, respectivamente, filtro de média, Erosão e Dilatação a fim de remover eventuais ruídos. Esse passo pode ser utilizado em diversos momentos do processamento, a fim de preparar a imagem para alguma técnica em que se faça necessário o um novo pré-processamento. Trabalhos relacionados utilizam filtro de mediana para reduzir ruídos. Contudo, utilizou-se o filtro de média já que o leve borrão gerado por este filtro suaviza bordas e deixa o vermelho dos semáforos mais uniforme.

3.1.3 Segmentação

O processo de segmentação converte a imagem para o espaço HSV como na maioria dos trabalhos relacionados, por ser este menos sensível à luminosidade ambiente, funcionando melhor em sombras e à noite, e busca por elementos vermelhos de forma aproximadamente arredondada. Assim, assume-se que um vídeo consiste de imagens coloridas It, onde t é o tempo, definidas como

$$I_t = \left\{ f_t(x,y) = \begin{bmatrix} f_{t,H}(x,y) \\ f_{t,S}(x,y) \\ f_{t,V}(x,y) \end{bmatrix} \right\} \tag{3.1}$$

onde (x, y) denota as coordenadas de um pixel e $f_{t,H}(x, y)$, $f_{t,S}(x, y)$ e $f_{t,V}(x, y)$ os valores de matiz, saturação e brilho, respectivamente. Como a iluminação de semáforos segue um padrão de cor (normalmente tons de vermelho, verde e amarelo aproximados), é possível concluir que os semáforos em geral utilizam uma faixa estreita do espectro. As regiões vermelhas $R_{R,K}$, amarelas $R_{Y,K}$ e verdes $R_{G,K}$ são definidas como

$$\begin{aligned}
R_{R,k} &= \left\{ \begin{array}{c} f_t(x,y) : |f_{t,H}(x,y)| < t_h \quad \text{e} \\ f_{t,S}(x,y) > t_s \end{array} \right\} \\
R_{Y,k} &= \left\{ \begin{array}{c} f_t(x,y) : \left|f_{t,H}(x,y) - \frac{\pi}{3}\right| < t_h \quad \text{e} \\ f_{t,S}(x,y) > t_s \end{array} \right\} \\
R_{G,k} &= \left\{ \begin{array}{c} f_t(x,y) : \left|f_{t,H}(x,y) - \frac{2\pi}{3}\right| < t_h \quad \text{e} \\ f_{t,S}(x,y) > t_s \end{array} \right\}
\end{aligned} \tag{3.2}$$

onde th e ts são valores de tolerância definidos durante os testes. Essa tolerância permite pequenas variações na cor devido a condições adversas que modifiquem a percepção da cor. As fontes de luz identificadas neste passo são consideradas como áreas de interesse. A Figura 5 mostra um exemplo de imagem onde a cor é extraída corretamente como informação relevante.

Ainda na Figura 5, é possível perceber que é comum surgirem pequenas distorções na cor que poderiam confundir o processo de segmentação (canto esquerdo da imagem da direita). Assim, um passo de erosão seguido de dilatação faz o trabalho de anular esses pequenos ruídos, de modo que regiões de interesse que não tiverem uma área mínima (dependendo do tamanho da imagem) serão facilmente eliminados. Para fins de avanço de semáforo, apenas a luz vermelha foi considerada como relevante.

3.1.4 Detecção e reconhecimento

Considerando a dimensão padrão da lâmpada como 20 cm de diâmetro, valor estabelecido pelo Conselho Nacional de Trânsito (CONTRAN), e que as três luzes estão sempre bem próximas, a partir do tamanho de uma das luzes na imagem It, é possível estimar o tamanho esperado do semáforo em pixels, com 2/3 do semáforo abaixo da lâmpada vermelha, 2/3 acima da verde e 1/3 acima com 1/3 abaixo da lâmpada amarela.

As regiões de interesse definidas passam por um realce de bordas Canny descrito em (GUPTA et al., 2011) na área onde possivelmente há um semáforo. Em seguida um passo de transformada de Hough busca por círculos nas áreas de interesse. O objetivo é verificar se a região segmentada é um círculo vermelho (luz do semáforo) ou não. Canny é muito utilizado por ser simples e eficaz no realce de bordas na maioria das aplicações se compararmos com Sobel e Shen&Castan, por exemplo (HASSAN et al., 2008). Já a transformada de Hough foi adotada por sua precisão, mesmo com objetos parcialmente oclusos, em relação ao tempo de processamento se comparada com outras técnicas citadas no capítulo anterior.

A Figura 6 exemplifica um possível resultado do realce de bordas numa região de interesse, onde uma luz vermelha está acesa e deve ser objetivo da busca da transformada de Hough.

O algoritmo só processa a região ao redor da área encontrada na Figura 5. Caso a região destacada como de interesse seja um círculo, a existência do semáforo é confirmada.

Como as luzes traseiras (lanternas traseiras e luzes de freios) dos veículos são usualmente vermelhas, além de ser possível que tenham forma arredondada, propõe-se que somente se procurem semáforos na parte superior. Outro motivo para tal é que os semáforos somente interessam no momento em que o veículo passa por ele, tornando desnecessário o processamento de semáforos distantes. A Figura 7 exemplifica a similaridade dos tons

Figura 5 – Extração da cor verde como região de interesse

Fonte: Imagem capturada pelo autor

Figura 6 – Exemplo de realce de bordas por Canny em uma região de interesse. A região de interesse compreende os arredores da possível lâmpada detectada na Segmentação.

Fonte: Imagem elaborada pelo autor

de vermelho entre os diversos semáforos e as luzes de freio de um veículo.

3.1.5 Rastreamento

A fase de rastreamento é responsável por definir se o semáforo saiu da área da imagem ou se houve oclusão ou mudança de cor (de vermelho para verde, por exemplo) dentro dos limites da imagem. Um semáforo detectado passa a ser monitorado a cada quadro subsequente. Ele deve, preferencialmente, ainda que se desloque também na horizontal, deslocar-se para posições superiores da imagem. A partir do deslocamento horizontal e vertical do semáforo, é possível se estimar a continuidade da sua trajetória e, assim, inferir se o semáforo sumiu da imagem ou se sofreu uma oclusão, seja ela parcial ou total.

O rastreamento é realizado usando Camshift, mesma técnica adotada em (BRADSKI, 1998), por ser robusta suficiente em imagens com fundo estático (o fundo da imagem varia pouco quando o carro se desloga em linha reta) (CHEN et al., 2014). O algoritmo Camshift é aplicado nas proximidades da última detecção a procura da posição da luz, independente da cor. Se encontrada nova posição, reposiciona a região de interesse com base na nova posição da luz. Caso contrário, é necessário aguardar alguns quadros pela volta do semáforo. Se isso não ocorrer, considera-se que a luz mudou para verde ainda dentro do campo de visão da câmera. Se a luz vermelha foi perdida na extremidade do campo de visão, caracteriza-se então o avanço do sinal.

3.2 Algoritmo

O algoritmo reune tecnicas já difundidas e utilizadas em muitos trabalhos de Visão Computacional. O processamento do vídeo de entrada segue a seguinte lógica:

Procedimento ProcessaVideo(v)

```
1  avancos ← 0;
2  enquanto HÁ QUADROS A SEREM PROCESSADOS faça
3      q ← próximo quadro de v
4      q ←FiltroMédia(q);
5      q ←Erosão(q);
6      q ←Dilatação(q);
7      verm ← RegioesVermelhas(q);
8      se ESTÁ RASTREANDO ALGUM SEMÁFORO então
9          Rastrear(sem);              ▷ busca a luz vermelha nas proximidades da
            encontrada no quadro anterior
10         se NÃO ENCONTROU SEMÁFORO PRÓXIMO DO ENCONTRADO NO
            QUADRO ANTERIOR então
11             BuscaSemaforo(q); ▷ Busca semáforos em toda a região superior da
                imagem
12         fim
13         se ENCONTROU SEMÁFORO então
14             sem ← nova região do semáforo;
15         senão
16             se SEMÁFORO FOI ENCONTRADO EM 2 QUADROS CONSECUTIVOS
                então
17                 avancos ← avancos + 1;        ▷ Registra o avanço e incrementa
                    contador
18             fim
19             PararRastreamento(sem)
20         fim
21     senão
22         BuscaSemaforo(q);      ▷ Busca semáforos em toda a região superior da
            imagem
23     fim
24 fim
```

Procedimento BuscaSemáforo(q)

```
1 bordas ←Canny(imagem);
2 sem ← região circular mais alta de bordas;      ▷ Considera apenas círculos e
    despreza as demais formas
3 se ENCONTROU REGIÃO CIRCULAR então
4 |   IniciarRastreamento(sem);    ▷ Marca a região para rastrear nos próximos
  |     quadros
5 fim
```

Para cada quadro, o Filtro de média, a Erosao e Dilatação se encarregam de reduzir pequenos ruídos. Caso não haja semáforos sendo rastreados, apenas é feita uma busca e, se algum semáforo for encontrado, o rastreamento é iniciado. Para cada quadro subsequente, se já houver algum semáforo sendo rastreado, o semáforo é buscado nos arredores da última região para otimizar o algoritmo. Se não for encontrado semáforo, a busca é refeita em toda a região superior da imagem, numa espécie de segunda tentativa. Se ainda sim não for encontrado semáforo o rastreamento é interrompido e o contador de avanços é incrementado se, e somente se, o semáforo desapareceu na borda superior da imagem e se algum semáforo tiver sido detectado e rastreado em 2 quadros consecutivos antes de desaparecer. Semáforos que desaparecem longe da borda superior da imagem são considerados casos de oclusão e, portanto, são desconsiderados até que voltem para a imagem.

As técnicas de Filtro de Média, Erosão, Dilatação e Canny, por serem altamente difundidas e consideradas simples, encontram-se nos Apêndices A, B e C. As demais são explicadas a seguir nas próximas seções.

3.3 Transformada de Hough

A Transformada de Hough (HOUGH, 1962) é uma técnica que pode ser usada para isolar características de uma forma particular em uma imagem. Como a técnica requer que as características desejadas sejam especificadas de forma paramétrica, a transformada de Hough clássica é mais adotada para detecção de curvas regulares, tais como linhas, círculos e elipses. Uma forma genérica da transformada de Hough pode ser aplicada onde uma descrição analítica simples das características é inviável, mas este trabalho lida apenas com a forma clássica por ser computacionalmente mais simples.

O caso mais simples é a transformada para detectar linhas retas. No espaço da imagem, a linha reta pode ser descrita como $y = mx + b$ onde m é a inclinação da reta e b é a interseção com o eixo das ordenadas. A ideia da transformada de Hough é considerar a linha como tendo características não de pontos discretos de uma imagem, mas de acordo com o modelo contínuo no plano. Com isso, de forma geral, a linha reta

pode ser representada como um ponto (b, m) no espaço dos parâmetros. Contudo, linhas verticais apresentam-se como um problema, pois permitiriam valores infinitos para m. Por esse motivo, Duda e Hart (1972) propuseram o uso de um par de parâmetros diferente, denotados como r e θ, para as linhas na transformada de Hough. Esses dois valores juntos definem uma coordenada polar.

O parâmetro r representa a distância algébrica entre a linha e a origem, enquanto θ é o ângulo do vetor ortogonal à linha e apontando para os quadrantes superiores do plano conforme a Figura 8. Usando esses parâmetros, a equação da linha pode ser escrita como

$$y = \left(-\frac{cos\theta}{sin\theta}\right) x + \left(\frac{r}{sin\theta}\right) \tag{3.3}$$

que pode ser reescrita como

$$r = xcos\theta + ysin\theta \tag{3.4}$$

Portanto, é possível associar com cada segmento de reta da imagem um par (r, θ) que é único se $\theta \in [0, \pi)$ e $r \in \mathbf{R}$. Essas representações correspondem a uma curva senoidal no plano (r, θ) do espaço de Hough. Se as curvas correspondentes a dois pontos se sobrepuserem, a localização no espaço de Hough onde eles se cruzam corresponde a uma linha na imagem original. Em outras palavras, um conjunto de pontos que formam uma linha reta na imagem original produzirá senóides no espaço (r, θ) que se cruzam nos parâmetros para tal linha. Um exemplo pode ser visto na Figura 9.

A transformação pode ser generalizada ou especializada de acordo com a necessidade. Isso inclui a utilização para curvas além das linhas retas já mencionadas. Por exemplo, se precisamos de um método que detecte pontos em uma disposição que forme um círculo, podemos partir da representação paramétrica válida para todos os círculos e transformar cada ponto da figura.

Na detecção de círculos, a equação paramétrica geral é

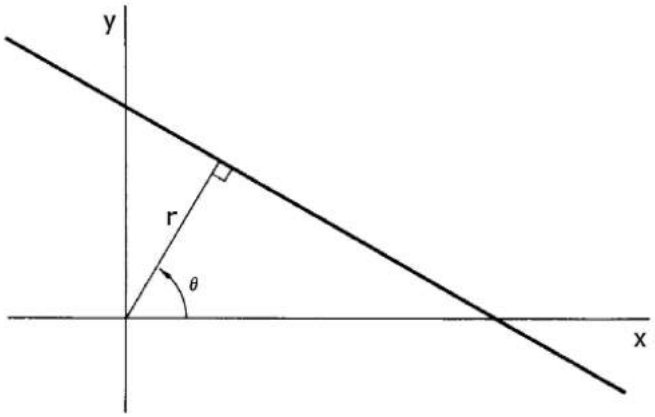
$$r^2 = (x - a)^2 + (y - b)^2 \tag{3.5}$$

onde a e b são as coordenadas do centro do círculo e r é o raio. Nesse caso, a complexidade do algoritmo começa a aumentar, já que há três coordenadas no espaço dos parâmetros, ou seja, cada ponto da figura original seriá transformado em um cone circular reto no espaço tridimencional (a, b, r). Se os cones correspondentos a vários pontos da figura original se interceptam num ponto, então estes pontos da figura original estão todos no círculo. A representação do círculo no espaço de Hough será um conjunto de senoidais que cruzam não em um, mas em infinitos pontos, formando uma linha. Cada ponto da

Figura 7 – Via com um semáforo próximo outros semáforos ao fundo

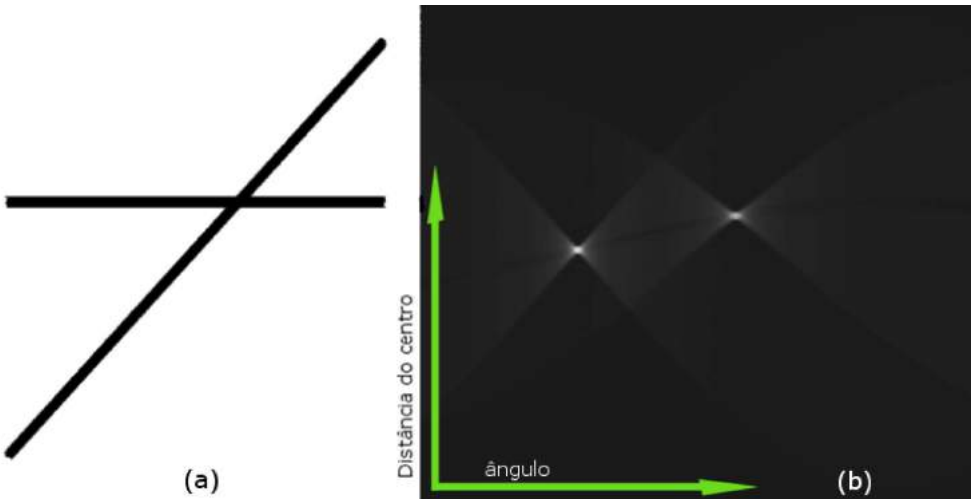
Fonte: Charette (2013)

Figura 8 – Representação gráfica de uma linha. A reta é representada no espaço de Hough pela sua normal que, por sua vez, é descrita em função de r e θ



Fonte: Duda e Hart (1972)

Figura 9 – (a) Retas na imagem original; (b) Representação das retas no espaço de Hough. Cada ponto das retas na imagem original (*a*) é representado por uma senoidal em função de r e θ no espaço de Hough (*b*). Senóides que se cruzam (pontos brancos mais intensos) indicam pontos colineares na imagem original.



Fonte: Elaborado pelo autor

linha tem os parâmetros correspondentes para cada ponto do círculo detectado. Se r for fixado, teremos um problema bidimensional que pode ser representado como na Figura 10, onde podem-se ver as senoidais correspondentes à transformada de Hough para círculos.

3.4 Mean-shift e Camshift

Mean-shift é uma técnica para localizar o máximo de uma função densidade com base em dados discretos dessa função. É útil para detectar as modas dessa densidade. É um método iterativo que parte de uma estimativa inicial x. Dada uma função *kernel* $K(x_i - i)$, essa função determina os pesos dos pontos próximos para reestimativa da média. Na distância para a estimativa corrente é comum utilizar um kernel Gaussiano

$$K(x_i - i) = e^{-c||x_i - x||^2} \tag{3.6}$$

A média ponderada da densidade da janela determinada por K é

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - i)x_i}{\sum_{x_i \in N(x)} K(x_i - i)} \tag{3.7}$$

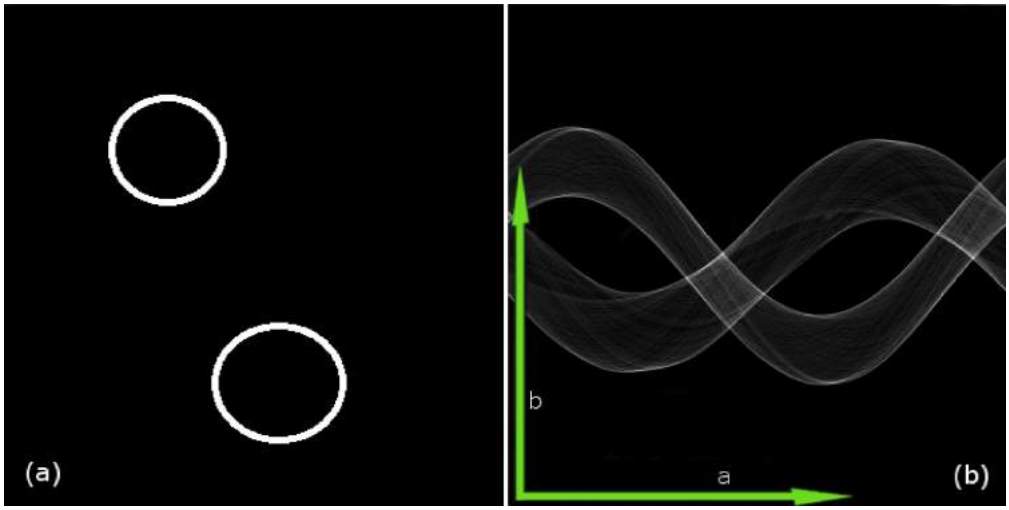
onde $N(x)$ é a vizinhança de x, um conjunto de pontos para os quais $K(x) \neq 0$. Então, o algoritmo agora tem $x \leftarrow m(x)$ e repete a estimativa até $m(x)$ convergir.

Em processamento de imagens, o algoritmo mean-shift é adotado para rastreamento

de objetos numa sequência de imagens. A técnica cria um mapa de confiança na nova imagem com base no histograma de cores do objeto na imagem anterior e busca o pico do mapa de confiança próximo à posição anterior do objeto de interesse. O mapa de confiança é uma função densidade de probabilidade da cor do pixel que atribui a cada pixel da nova imagem uma probabilidade de a cor do pixel ocorrer na imagem anterior.

O algoritmo Camshift utiliza o mesmo princípio do mean-shift. Contudo, a cada iteração, ele reajusta o tamanho e encontra uma rotação ótima para a janela de busca. O reajuste é feito com base no novo centro de massa e na nova área que possui a cor de interesse, sabendo que essa área é proporcional à distância entre o objeto e a câmera. A Figura 11 exemplifica a execução do Camshift para rastreamento de um veículo em uma sequência de imagens.

Figura 10 – (a) Círculos na imagem original; (b) Representação dos círculos no espaço de Hough em duas dimensões fixando r. Cada ponto dos círuclos na imagem original (a) é representado por uma circunferência em função de a e b no espaço de Hough (b). Circunferências que se cruzam (linhas brancas mais intensos) indicam pontos no mesmo círculo na imagem original.



Fonte: Imagem elaborada pelo autor

Figura 11 – Exemplo de rastreamento utilizando Camshift. À medida que o veículo se desloca, a janela o acompanha. Além disso, o algoritmo é capaz de redimensionar a janela de rastreamento à medida que o veículo se aproxima da câmera.

Fonte: Documentação OpenCV

4 EXPERIMENTOS E ANÁLISE DOS RESULTADOS

Para verificar o funcionamento da técnica proposta neste trabalho, foram utilizados dois vídeos distintos. Os vídeos foram armazenados e submetidos à implementação do algoritmo que é explicado no capítulo anterior.

4.1 Coleta das imagens

Um vídeo utilizado foi gravado com uma câmera veicular em Minas Gerais, nas ruas de Belo Horizonte e de Contagem. O vídeo foi gravado em 25 quadros por segundo em resolução de 640x480 pixels, 16 bits por pixel. O vídeo tem duração aproximada de 48 minutos, sendo 24 minutos durante o dia e 24 minutos durante a noite, totalizando 72.000 quadros. O vídeo foi testado com a taxa de quadros inicial e com uma redução na taxa de quadros para 5 e 2 quadros por segundo apenas a caráter experimental, e cada quadro armazenado como uma imagem em formato *Joint Photographic Experts Group* (JPEG). Durante a etapa diurna, o veículo passa por 24 semáforos, dos quais 8 estão vermelhos. Já na fase noturna, são 11 semáforos, dos quais 4 são vermelhos. Nesse vídeo, gravado para testes neste trabalho, a câmera foi devidamente posicionada a fim de capturar os semáforos na posição logo acima do veículo para facilitar a detecção de avanço. A Figura 12 contém dois quadros extraídos do vídeo gravado.

Outro vídeo para testes foi gentilmente cedido por Charette (2013). O vídeo, com taxa de atualização de 25fps (*frames per second* - quadros por segundo), em resolução 640x480 pixels, 8 bits por pixel, colorido, foi separado quadro a quadro em imagens de formato JPEG, forma como é distribuído. O vídeo possui um total de 11.179 quadros. A taxa de atualização foi reduzida para 5 e 2 quadros por segundo para uma comparação justa com o vídeo brasileiro. O veículo portador da câmera passa, ao longo do vídeo, por 8 semáforos vermelhos, mas não avança nenhum. As imagens foram obtidas durante um trajeto de cerca de 8 minutos em ruas de Paris, na França. Este é um vídeo genérico para *benchmark* em técnicas de processamento de vídeo, logo, a câmera não foi posicionada especificamente para visualizar semáforos suspensos acima dos veículos. A Figura 13 contém um quadro extraído do vídeo.

Para validar o funcionamento em caso de um avanço real, simulações de avanço foram feitas com a câmera sendo inclinada, verticalmente, de cima para baixo, para que o semáforo saia da visão da câmera na região superior da imagem. O comportamento simulado é similar ao que ocorre quando o veículo passa por baixo do semáforo. O veículo mantém-se parado durante o processo. Essas simulações de avanço são exemplificadas pela Figura 14 e permitem os testes de detecção e de falsos negativos. São dois vídeos, de 41 e 56 segundos, sendo um para o dia e outro para noite, respectivamente, com um total de 16 simulações de avanço durante o dia e 11 durante a noite. Como a simulação

Figura 12 – Exemplo de imagens do vídeo gravado em Belo Horizonte. (a) Imagem diurna. (b) Imagem noturna

Fonte: Imagens capturadas pelo autor

Figura 13 – Exemplo de imagem do vídeo obtido em Paris

Fonte: Charette (2013)

visa comparar a taxa de acertos e não o desempenho, os testes foram feitos somente com 5 e 2 fps.

Em ambos os vídeos, os semáforos contemplam tanto interseções quanto simples travessias de pedestres. Para cada um dos vídeos, são comparados os tempos de execução e a relação entre a quantidade de falsos positivos e falsos negativos para cada situação.

4.2 Implementação

O código foi implementado em C++ com uso da biblioteca OpenCV. Os testes foram realizados em um computador com processador Core i3 2.3 GHz dual core, com 6GB de RAM e executando Ubuntu 14.04. Alguns dos parâmetros, quando não especificado, foram definidos com base em testes durante a implementação em imagens de semáforos disponíveis no Google Street View para que semáforos fossem detectados corretamente nessas imagens. Os parâmetros utilizados, tanto para dia quanto para noite, foram os seguintes:

a) Pré-processamento: O filtro de média utiliza matriz de dimensões 3x3 e Dilatação e Erosão com elemento estrutural quadrado de dimensões 3x3 *pixels*;

b) Segmentação: No vídeo gravado para esta pesquisa, a busca por elementos vermelhos aceita somente os pontos no espaço HSV onde S e V são superiores a 75% e o H para vermelho deve ser próximo de 0 com tolerância de 20º para mais ou para menos. Para o vídeo de benchmark, H foi definido como 40º com tolerancia de 10º. Estes parâmetros foram diferentes pois os vídeos foram gravados com câmeras diferentes;

c) Detecção e reconhecimento: Canny utiliza limiar inferior 40 e limiar superior 100 com K sendo uma matriz 3x3. A transformada de Hough para círculos aceita uma distância mínima de 30 *pixels* entre os centros dos círculos para as imagens teste de 640x480 *pixels* para que não haja excesso de círculos detectados. O raio mínimo e máximo permitidos são 3 e 20, respectivamente, baseado no tamanho esperado do semáforo dentro da imagem de 640x480 *pixels*;

d) Rastreamento: o método Camshift executa no máximo 10 iterações na busca da convergência de $m(x)$, valor padrão utilizado no método de rastreamento do OpenCV.

4.3 Resultados

Durante os testes, verificou-se que os semáforos foram devidamente detectados para os vídeos diurnos. Contudo, alguns elementos com luz vermelha foram confundidos com luz vermelha de semáforo durante a noite.

O vídeo de Belo Horizonte foi separado em dois videos de 24 minutos cada. A parte diurna e a parte noturna foram processadas e avaliadas separadamente para facilitar a análise e a comparação. O processamento do vídeo gravado durante o dia com 25 quadros por segundo levou um tempo médio de 26,1 minutos, um pouco acima da duração do vídeo, com 10 falsos positivos. Com 5 quadros por segundo, levou um tempo médio de 247 segundos sem falsos positivos. Reduzindo a taxa de atualização para 2 quadros/segundo, o tempo caiu para 98 segundos ainda sem falsos positivos. A queda no número de falsos positivos ocorreu pois a redução da taxa de atualização (fps) reduziu a chance de um falso semáforo ser detectado em dois quadros consecutivos.

A Figura 15 contém um semáforo detectado e processado corretamente como não avanço.

Para o processamento das imagens noturnas, os tempos para 25 fps, 5 fps e 2 fps foram, respectivamente, 25,4 minutos, 238 segundos e 98 segundos, respectivamente. Os falsos positivos, chegaram a 9, 1 e 1 avanços, respectivamente, valores considerados excessivamente elevados considerando a existência de apenas 11 semáforos e nenhum avanço no percurso. O principal motivo da detecção de falsos positivos é a distorção nas cores causada pela qualidade da lente da câmera, que provocou alterações nas cores reais da iluminação do semáforo. Outro motivo é a iluminação de vapor de sódio de algumas vias, que possui cor amarelada e causa alterações nas cores do ambiente. A Figura 16 contém uma imagem noturna onde a distorção das cores é nítida.

É possível perceber que o amarelo da iluminação pública possui um tom de cor próximo ao tom de vermelho da luz traseira do ônibus. As cores seriam muito diferentes a olho nu, sem câmeras e processos de digitalização. Outra observação importante é que, com a redução da taxa de quadros, houve redução nos falsos positivos. Isso ocorreu porque, com uma taxa menor de atualização, variações maiores entre dois quadros reduziram a chance de a distorção afetar dois quadros consecutivos e, com isso, o próprio algoritmo eliminou o erro. A distorção nas cores e a ocorrência de muitos falsos positivos não afetou de forma significativa o tempo de processamento.

Para a simulação de avanço, nos testes para o vídeo diurno, o tempo de processamento com 25 fps levou 22 segundos com 17 registros de avanço (1 falso positivo). Com 5 fps o processamento levou 6 segundos e encontrou 15 avanços de semáforo dentre os 16 simulados, enquanto com 2 fps foi de 3 segundos e 9 detecções de avanço foram registradas. O problema desse tipo de simulação é que ao elevar o semáforo para que ele saia da imagem, diversos outros elementos da imagem são elevados simultaneamente, incluindo outros elementos vermelhos (como luzes traseiras de veículos). A Figura 17 exemplifica uma detecção correta. A Figura 18 exemplifica um caso em que o vermelho de outro elemento vermelho (um caminhão) se eleva na imagem, mas o algoritmo foi robusto o suficiente para ignorar o caminhão e detectar o avanço simulado. A Figura 19 mostra um

Figura 14 – Exemplo de simulação de avanço com movimento da câmera. Observe que toda a imagem se desloca verticalmente para cima, consequência de a câmera ser deslocada para baixo.

Fonte: Imagens capturadas pelo autor

Figura 15 – Exemplo de imagem diurna onde houve processamento correto. O semáforo mudou de vermelho para verde e o algoritmo não computou o avanço.

Fonte: Imagens capturadas pelo autor

Figura 16 – Exemplo de imagem noturna com distorção de cores por falha no processo de captura. Luzes que, a olho nu, são nitidamente diferentes, tornam-se parecidas devido à baixa qualidade da câmera.

Fonte: Imagem capturada pelo autor

exemplo em que o semáforo saiu do campo de visão da câmera antes de se aproximar o suficiente da borda superior da imagem. É principal a consequência da redução da taxa de quadros por segundo do vídeo.

Quanto à simulação durante a noite, o tempo de processamento foi de 28 segundos com 21 avanços (10 falsos positivos) para 25fps, 8 segundos e 11 avanços para o vídeo de 5 fps e 3 segundos com 10 avanços para o vídeo de 2 fps.

Observe que, de modo geral, a taxa de quadros 5 fps teve um resultado melhor em relação aos acertos. Como essa configuração é suficiente para execução em tempo real, ela será utilizada para comparação com o vídeo de benchmark. A Tabela 1 contém a quatidade de erros e acertos para o processamento diurno incluindo os avanços simulados. A Tabela 2 contém a quatidade de erros e acertos para o processamento noturno.

Tabela 1 – Resultados do processamento do vídeo de Belo Horizonte gravado em período diurno

	Avanço real	Não avanço	Total
Avanço detectado	15	0	15
Avanço não detectado	1	8	9
Total	16	8	

Fonte: Testes da pesquisa

Tabela 2 – Resultados do processamento do vídeo de Belo Horizonte gravado em período noturno

	Avanço real	Não avanço	Total
Avanço detectado	11	1	12
Avanço não detectado	0	4	4
Total	11	5	

Fonte: Testes da pesquisa

Para o vídeo de Paris, o processamento levou 137 segundos com 5 fps e 58 segundos a 2 fps. A taxa de erros revelou-se alta com 2 fps, com 9 avanços em apenas 8 semáforos, mas a quantidade de erros ainda foi melhor que os 15 avanços com 5 fps (sendo 8 falsos positivos). Isso deve-se principalmente ao posicionamento da câmera, que é praticamente horizontal. Com isso, várias luzes e elementos vermelhos, como as luzes de outros veículos, por exemplo, são confundidos com luzes de semáforos por estarem na parte superior da imagem, mesmo que não estejam realmente suspensos sobre a via. Na Figura 20 pode-se ver um caso de processamento errôneo com o veículo consideravelmente longe do semáforo devido ao mal posicionamento da câmera. Todavia, este foi considerado um acerto pois, num cenário em que a câmera estivesse devidamente posicionada, o avanço exemplificado seria considerado um sucesso, já que o semáforo vermelho saiu do campo de visão da câmera pela parte superior da imagem.

Tabela 3 – Resultados do processamento do vídeo de benchmark

	Avanço real	Não avanço	Total
Avanço detectado	7	8	15
Avanço não detectado	1	9	10
Total	8	17	

Fonte: Testes da pesquisa

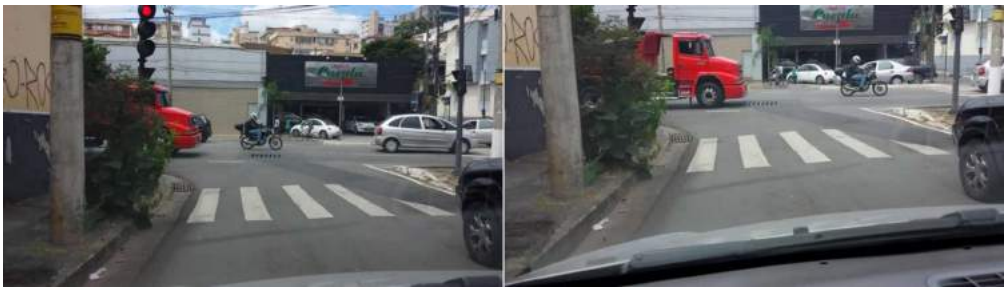
Apesar do número elevado de falsos positivos, houve casos em que a detecção ocorreu corretamente e não foi computado avanço. A Figura 21 mostra um caso em que houve processamento correto no vídeo francês.

Para 5 e 2 quadros por segundo, os tempos de processamento foram 137 e 58 segundos, respectivamente. Percebe-se que o tempo de processamento está aproximadamente na mesma proporção com a taxa de quadros por segundo. Seguindo a mesma linha, a quantidade de falsos positivos caiu de 58 para 34 por haver menos quadros consecutivos com a câmera mal posicionada para confundir o algoritmo no processo.

Figura 17 – Exemplo de imagem diurna com detecção correta de avanço simulado. A luz vemelha sai da imagem na parte superior e o avanço é computado.

Fonte: Imagens capturadas pelo autor

Figura 18 – Exemplo de imagem da simulação diurna com a elevação de um caminhão vermelho. Apesar de o caminhão ser vermelho, sua forma não arredondada evita que seja confundido com uma luz de semáforo.

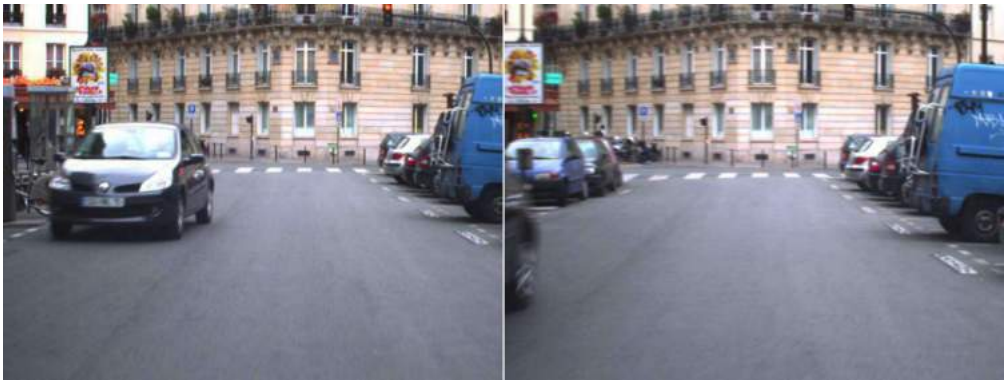


Fonte: Imagens capturadas pelo autor

Figura 19 – Exemplo de simulação em que o semáforo vermelho não foi devidamente detectado. O semáforo desaparece antes de chegar à borda superior da imagem e, como consequência, o algoritmo considera que não houve avanço.

Fonte: Imagens capturadas pelo autor

Figura 20 – Exemplo de imagem do video francês onde há fuga do semáforo do campo de visão devido ao mal posicionamento da câmera. Contudo, pela característica, este foi considerado um avanço real.



Fonte: Charette (2013)

Figura 21 – Exemplo de imagem do video francês com detecção correta do semáforo e do não avanço

Fonte: Charette (2013)

5 CONCLUSÃO E TRABALHOS FUTUROS

Em Belo Horizonte, menos de 1% dos semáforos tem equipamento para detecção de avanço de sinal vermelho. Com o pequeno número de semáforos monitorados, condutores com má índole fazem uma espécie de mapeamento mental dos semáforos com detectores de avanço e avançam os demais semáforos quando vermelhos. Com o grande número de infrações por avanço de semáforo todo ano, fez-se necessária uma nova forma de identificar esses avanços nos semáforos não dotados do equipamento adequado para registro. Por esse motivo, este trabalho propõe que o monitoramento seja feito com foco nos veículos de trabalho, não mais nas interseções da via. A proposta parte da utilização câmera embarcada e técnicas de processamento de imagem. As imagens são gravadas com câmera veicular e os vídeos processados posteriormente.

As imagens são convertidas para o espaço HSV para reduzir os efeitos da má luminosidade. Passam por um processo de filtro de média, erosão e dilatação, respectivamente, para reduzir eventuais ruídos. Em seguida, procura-se luz vermelha de algum possível semáforo e, se encontrado, este semáforo é rastreado até sair do campo de visão da câmera.

A precisão foi satisfatória para vídeos gravados durante o dia desde que a câmera esteja bem posicionada, além de o tempo ter sido extremamente favorável, levando cerca de 4 minutos para processar 24 minutos de vídeo com exatidão superior a 95,8% para o vídeo diurno e 93,7% para o vídeo noturno. A implementação requer ajustes e de uma melhor definição de parâmetros para tratar melhor a iluminação noturna e a distorção de cores. O tempo de execução é menor do que a duração dos vídeos para o processamento com redução na taxa de quadros/segundo, abrindo possibilidade para processamento em tempo real.

Para o vídeo de benchmark, houve muitos falsos positivos pelo mal posicionamento da câmera. Com isso, o uso de uma câmera genérica (usada para outros propósitos) tornou-se um problema, já que o semáforo sai do campo de visão com o veículo ainda muito distante do semáforo. Mesmo com muitos falsos positivos, a taxa de acertos foi de 64% dos semáforos processados corretamente.

Devido à impossibilidade de avançar semáforos legalmente e capturar vídeos de testes com estes avanços reais, testes foram realizados com simulação de avanço baseado na movimentação da câmera diante do semáforo vermelho. Os testes funcionaram melhor para o vídeo de 5 fps, onde somente um dos avanços simulados não foi detectado. A exatidão na simulação chegou a 96,2%, mesmo com o problema de outras luzes e itens vermelhos se colocarem no lugar dos semáforos.

Este trabalho traz como contribuição a nova metodologia de detecção de avanço de semáforo com a câmera embarcada, independente se o semáforo é ou não monitorado por outras formas de registro de avanço. Trabalhos relacionados encontrados lidam com

um ponto fixo da via, com diversas técnicas utilizadas. Apesar de menos preciso que um detector de avanço fixo na via, ainda sim a solução aparenta ser viável pela flexibilidade de registro de avanço em qualquer semáforo não monitorado.

5.1 Trabalhos futuros

Há vários pontos de melhoria na técnica proposta. Com base nos resultados, torna-se interessante o processamento em tempo real com um computador embarcado no veículo e ligado diretamente à câmera. Recomenda-se, nesse caso, validar o desempenho (tempo de execução) em um sistema de baixo consumo, já que este trabalho adotou um computador comum. Outra proposta são testes com vídeos em alta resolução gravados com uma câmera melhor, reduzindo a distorção de cores, e com maior taxa de quadros por segundo para comparar precisão e desempenho. Há também necessidade de lidar com semáforos com setas, onde o semáforo restringe a circulação somente para determinados veículos, pois esse tipo de abordagem não foi tratada no escopo deste trabalho. Uma forma de calibrar automaticamente a imagem para adaptação automática entre dia e noite seria de grande valia.

Apesar de o vídeo de benchmark utilizado nos testes deste trabalho (CHARETTE, 2013) não ser apropriado pelo posicionamento da câmera, testes com outros vídeos de benchmark, por abranger testes com mais sistemas similares, podem ser uma alternativa interessante. Testes com sistemas de registro de avanço similares podem ser feitos envolvendo, além de precisão e tempo, a mobilidade e o custo de implementação e implantação.

REFERÊNCIAS

ALMAGAMBETOV, A.; VELIPASALAR, S.; BAITASSOVA, A. **Mobile Standards-Based Traffic Light Detection in Assistive Devices for Individuals with Color-Vision Deficiency**. Intelligent Transportation Systems, IEEE Transactions on, PP, n. 99, p. 1–16, 2015. ISSN 1524-9050.

ANDRADE, G. **Multas por avanço de sinal geram polêmica**. [S.l.]: O Tempo Contagem, jul. 2011. Disponível em `http://www1.otempo.com.br/otempocontagem/noticias/?IdNoticia=6442`.

BRADSKI, G. R. **Computer vision face tracking for use in a perceptual user interface**. Intel Technology Journal Q2 '98, Citeseer, 1998.

CANNY, J. **A computational approach to edge detection**. Pattern Analysis and Machine Intelligence, IEEE Transactions on, IEEE, n. 6, p. 679–698, 1986.

CHARETTE, R. d. **Traffic Lights Recognition (TLR) public benchmarks**. out. 2013. Disponível em: <http://www.lara.prd.fr/benchmarks/trafficlightsrecognition>.

CHEN, C. et al. **Real-Time Robust Hand Tracking Based on Camshift and Motion Velocity**. In: Digital Home (ICDH), 2014 5th International Conference on. [S.l.: s.n.], 2014. p. 20–24.

DUDA, R. O.; HART, P. E. **Use of the Hough Transformation To Detect Lines and Curves in Pictures**. In: Communications of the ACM. [S.l.: s.n.], 1972.

FAIRFIELD, N.; URMSON, C. In: Robotics and Automation (ICRA), 2011 IEEE International Conference on. [S.l.: s.n.].

FLEYEH, H. **Shadow And Highlight Invariant Colour Segmentation Algorithm For Traffic Signs**. In: Cybernetics and Intelligent Systems, 2006 IEEE Conference on. [S.l.: s.n.], 2006. p. 1–7.

GONG, J. et al. **The recognition and tracking of traffic lights based on color segmentation and CAMSHIFT for intelligent vehicles**. In: Intelligent Vehicles Symposium (IV), 2010 IEEE. [S.l.: s.n.], 2010. p. 431–435. ISSN 1931-0587.

GUPTA, A. et al. **DGW-canny: An improvised version of Canny edge detector**. In: Intelligent Signal Processing and Communications Systems (ISPACS), 2011 International Symposium on. [S.l.: s.n.], 2011. p. 1–6.

HASSAN, M. A. A. et al. **Evaluation of Sobel, Canny, Shen & Castan using sample line histogram method**. In: 2008 International Symposium on Information Technology. [S.l.: s.n.], 2008. v. 3, p. 1–7. ISSN 2155-8973.

HEDBERG, H. et al. **A low complexity architecture for binary image erosion and dilation using structuring element decomposition**. In: CIRCUITS AND SYSTEMS, 2005. ISCAS 2005. IEEE INTERNATIONAL SYMPOSIUM ON. [S.l.: s.n.], 2005. p. 3431–3434 Vol. 4.

Paul V. C. Hough. **Method and means for recognizing complex patterns**. December 1962. 3069654. Disponível em: <http://www.freepatentsonline.com/3069654.html>.

LEVINSON, J. et al. **Traffic light mapping, localization, and state detection for autonomous vehicles**. In: ROBOTICS AND AUTOMATION (ICRA), 2011 IEEE INTERNATIONAL CONFERENCE ON. [S.l.: s.n.], 2011. p. 5784–5791. ISSN 1050-4729.

LUO, D.; HUANG, X.; QIN, L. **The Research of Red Light Runners Video Detection Based on Analysis of Tracks of Vehicles**. In: COMPUTER SCIENCE AND INFORMATION TECHNOLOGY, 2008. ICCSIT '08. INTERNATIONAL CONFERENCE ON. [S.l.: s.n.], 2008. p. 734–738.

PARANAIBA, G.; HOLANDA, T. d. **BH tem aumento de pegas e de veículos com velocidade acima de 50% do limite**. [S.l.]: Estado de Minas, ago. 2014. Disponível em `http://www.em.com.br/app/noticia/gerais/2014/08/21/interna_gerais,560828/`.

PAULO, C.; CORREIA, P. **Automatic Detection and Classification of Traffic Signs**. In: IMAGE ANALYSIS FOR MULTIMEDIA INTERACTIVE SERVICES, 2007. WIAMIS '07. EIGHTH INTERNATIONAL WORKSHOP ON. [S.l.: s.n.], 2007. p. 11–11.

SHEN, Y. et al. **A robust video based traffic light detection algorithm for intelligent vehicles**. In: INTELLIGENT VEHICLES SYMPOSIUM, 2009 IEEE. [S.l.: s.n.], 2009. p. 521–526. ISSN 1931-0587.

SOETEDJO, A.; YAMADA, K. **Fast and Robust Traffic Sign Detection**. In: SYSTEMS, MAN AND CYBERNETICS, 2005 IEEE INTERNATIONAL CONFERENCE ON. [S.l.: s.n.], 2005. v. 2, p. 1341–1346.

SOUKI, M.; BOUSSAID, L.; ABID, M. **An embedded system for real-time traffic sign recognizing**. In: DESIGN AND TEST WORKSHOP, 2008. IDT 2008. 3RD INTERNATIONAL. [S.l.: s.n.], 2008. p. 273–276.

TU, Z.; LI, R. **Automatic recognition of civil infrastructure objects in mobile mapping imagery using a markov random field model**. INTERNATIONAL ARCHIVES OF ISPRS, v. 33, n. 2, 2000.

XU, S. **Robust traffic sign shape recognition using geometric matching**. INTELLIGENT TRANSPORT SYSTEMS, IET, v. 3, n. 1, p. 10–18, 2009. ISSN 1751-956X.

YELAL, M. R. et al. **Color-based signal light tracking in real-time video**. In: IEEE. VIDEO AND SIGNAL BASED SURVEILLANCE, 2006. AVSS'06. IEEE INTERNATIONAL CONFERENCE ON. [S.l.], 2006. p. 67–67.

YUNG, N. H. C.; LAI, A. H. S. **An effective video analysis method for detecting red light runners**. VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON, v. 50, n. 4, p. 1074–1084, 2001. ISSN 0018-9545.

APÊNDICE A - FILTRO DE MÉDIA

O filtro de média, também conhecido como filtro de caixa normalizada, é um filtro linear simples que suaviza bordas e transições nas imagens.

Ao se aplicar o filtro de média, o valor de cada pixel $g(i, j)$ na imagem resultante é determinado como a soma ponderada dos pixels de entrada. Essa soma é definida por

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(j,l) \tag{1}$$

onde h é a matriz chamada *kernel* com os coeficientes do filtro

Para o filtro comum de média, a matriz kernel utilizada é

$$K = \frac{1}{K_{largura} * K_{altura}} \begin{bmatrix} 1 & 1 & 1 & ... & 1 \\ 1 & 1 & 1 & ... & 1 \\ . & . & . & ... & 1 \\ . & . & . & ... & 1 \\ 1 & 1 & 1 & ... & 1 \end{bmatrix} \tag{2}$$

onde $K_{largura}$ e K_{altura} são as dimensões da matriz K. A imagem resultante é a imagem inicial com os elementos de alta frequência suavizados como na Figura 1.

Figura 1 – (a) Imagem original; (b) Exemplo de resultado do filtro de média

Fonte: Documentação OpenCV

APÊNDICE B - EROSÃO E DILATAÇÃO

Na morfologia matemática, a função de erosão visa eliminar a contração de uma região através da contração de seus limites. Regiões menores do que a região estrutural podem ser eliminadas. Erros de reconhecimento podem ser evitados e pequenas sombras e ruídos podem deixar de ser identificados como objetos ou regiões de interesse. Por outro lado, dilatação e uma operação que amplia áreas ou objetos em uma imagem, também partindo de um elemento estrutural.

Erosão (também chamada de diferença morfológica) é a operação de subtração de vetores de elementos de um conjunto e Dilatação é a fusão de dois conjuntos. Suponha que X seja a imagem original e B o elemento estrutural, as operações de erosão e dilatação são definidas por

$$E(X) = X \ominus B = \{(x, y) | B_{xy} \subseteq X\} \tag{1}$$

$$D(X) = X \oplus B = \left\{(x, y) | \left[\hat{B}_{xy} \cap X\right] \neq \emptyset\right\} \tag{2}$$

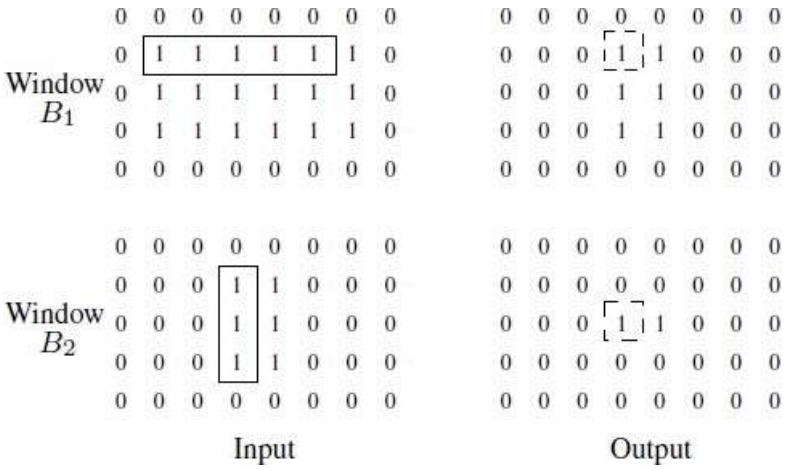
onde

1. X é a imagem original
2. $E(x)$ é o conjunto da imagem binária após a erosão
3. $D(x)$ é o conjunto da imagem binária após a dilatação
4. B é o elemento estrutural, totalmente contido em X, com cada valor no elemento sendo 0 ou 1, podendo ter qualquer forma gráfica e um ponto central
5. (x, y) sao coordenadas de pontos da imagem

Ambas são técnicas que utilizam janela deslizante para percorrer X. Essa janela desloca-se horizontal e verticalmente na imagem, de modo a percorrer todo o elemento estrutural de B. Na erosão, para cada posição da janela deslizante, o pixel central é mantido. Já na dilatação, em cada posição da janela deslizante cujo centro pertence a B, toda a janela é preenchida. Na Figura 1 pode-se ver um exemplo de erosão, com janela em linha horizontal de 5 pixels seguida por uma janela em linha vertical de 3 pixels.

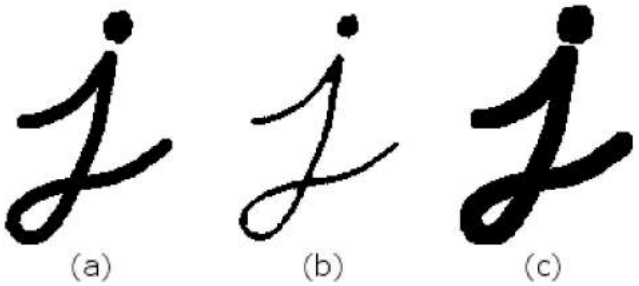
O resultado dos processos de erosão dilatação, respectivamente, são exibidos na Figura 2.

Figura 1 – Exemplo de janela deslizante no processo de erosão



Fonte: Hedberg et al. (2005)

Figura 2 – (a) Imagem original; (b) Exemplo de resultado de erosão; (c) Exemplo de resultado de dilatação



Fonte: Documentação OpenCV

APÊNDICE C - CANNY

O detector de bordas Canny foi proposto por Canny (1986). A detecção busca satisfazer os seguintes critérios:

1. Baixa taxa de erros: a detecção deve, de forma precisa, capturar o máximo possível das bordas na imagem;

2. Boa localização: os pontos da borda detectada devem, de forma precisa, localizar-se no centro da borda na imagem;

3. Resposta mínima: uma borda na imagem somente deve ser marcada uma vez e ruídos não pode criar falsas bordas.

A detecção se inicia com um filtro Gaussiano para suavizar a imagem e eliminar ruídos que poderiam causar falsos positivos, ou seja, detecção de falsas bordas. Uma matriz de $(2k+1)$ x $(2k+1)$ é usada na convolução. Uma possível matriz *kernel* (K) de dimensões 5x5, por exemplo, seria

$$K = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \tag{1}$$

O resultado do filtro Gaussiano é então usado para se encontrar o gradiente da imagem, ou seja, a transição das bordas. O procedimento é análogo ao operador Sobel, com duas máscaras de convolução

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{2}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{3}$$

A intensidade (ou força) do gradiente e a direção são dados respectivamente por

$$G = \sqrt{G_x^2 + G_y^2} \tag{4}$$

$$\Theta = \arctan \frac{G_y}{G_x} \tag{5}$$

sendo que a direção é arredondada para um dos seguintes ângulos: 0º, 45º, 90º ou 135º.

O próximo passo, com o objetivo de satisfazer o critério 3, busca o suprimir os gradientes exceto os máximos locais. Isto é feito comparando a força do gradiente de cada *pixel* com a força do gradiente dos *pixels* nos sentidos positivo e negativo do gradiente. Bordas menores que a dos demais *pixels* na máscara numa mesma direção serão suprimidas. Por fim, para o último passo, chamado Histerese, o detector Canny utiliza dois parâmetros: $limiar superior$ e $limiar inferior$. Gradientes que superam o limiar superior são considerados bordas; gradientes estiverem abaixo do limiar inferior são descartados; gradientes entre os dois limiares são aceitos se estiverem conectados com algum gradiente que supera o limiar superior. Na Figura 1 há um exemplo do resultado do detector de bordas Canny.

Figura 1 – Exemplo de detecção de bordas por Canny

Fonte: Documentação OpenCV