UM PROTÓTIPO DE SISTEMA DE VISÃO COMPUTACIONAL PARA RASTREAR A MOVIMENTAÇÃO DE VEÍCULOS

Ângelo L. B. Rodem*
Karl Phillip Buhr**

Resumo

A maior parte dos sistemas de monitoramento de trânsito empregados atualmente utilizam operadores humanos para monitorar ocorrências. Esse trabalho apresenta uma abordagem para detectar e rastrear automaticamente a movimentação de veículos em imagens digitais de tráfego rodoviário para permitir que os operadores identifiquem facilmente a quantidade de veículos e os horários de mais tráfego na rodovia, independente de a placa do veículo estar presente. Elaborou-se um protótipo de *software* com técnicas de processamento de imagens e visão computacional, que incluem operações morfológicas, limiarização, subtração de fundo com *MOG* (*Mixture of Gaussian*), entre outras, para trabalhar com vídeos de monitoramento de trânsito. Os vídeos utilizados durante os experimentos permitem ao sistema efetuar a contagem de veículos corretamente e de forma automática na maioria dos testes; entretanto, nesses resultados preliminares, identificaram-se situações nas quais o algoritmo perde a precisão momentaneamente e esses casos interessantes também são discutidos.
Palavras-chave: Detecção de objetos. Rastreamento de veículos. Visão computacional.

1 INTRODUÇÃO

Os automóveis possuem um papel fundamental na estruturação de qualquer sociedade moderna, pois permitem o transporte eficiente de pessoas e mercadorias. Somente nos últimos 11 anos, foram registrados 31.861.240 novos veículos (ASSOCIAÇÃO NACIONAL DOS FABRICANTES DE VEÍCULOS AUTOMOTORES, 2015). Ao se comparar esses dados com os 11 anos anteriores, é possível perceber um aumento significativo de 15.142.141 novos veículos, indicando que durante esse período o número de registros quase duplicou. Esse aumento traz vários novos desafios, inclusive o de manter a segurança nas rodovias.

Sistemas de monitoramento de trânsito estão sendo utilizados no Estado de Santa Catarina pela Polícia Rodoviária Federal (PRF-SC), pelo Departamento Estadual de Trânsito (Detran/SC) e pela Polícia Militar (PMSC) e permitem flagrar infrações de veículos que transitam acima do limite de velocidade ou no acostamento e visualizar motoristas que dirigem utilizando o telefone celular ou sem cinto de segurança, além de auxiliar durante investigações de outros tipos de ocorrências.

Conforme o Sindicato dos Centros de Formação de Condutores de Santa Catarina (2014), somente na BR-101 existem mais de 120 câmeras para monitorar a rodovia, cujo acompanhamento é realizado por uma central especializada da Policia Rodoviária Federal (PRF). Apesar de todo esse equipamento, a maior parte das rodovias catarinenses não são federais, e, portanto, não contam com esses sofisticados sistemas de monitoramento.

O presente trabalho apresenta uma abordagem de baixo custo ao utilizar *OpenCV*, um *framework* de visão computacional de código aberto, para efetuar automaticamente a detecção e o rastreamento de veículos em imagens de monitoramento de trânsito. A metodologia empregada permite rastrear veículos grandes e pequenos, independente de a placa do veículo estar visível na imagem, diferentemente dos sistemas mencionados anteriormente, que dependem fortemente da presença da placa do veículo para realizar a mesma tarefa.

* Graduando em Engenharia de Computação na Universidade do Oeste de Santa Catarina; angelorodem@gmail.com
** Mestre em Computação Aplicada; Professor do Curso de Engenharia de Computação na Universidade do Oeste de Santa Catarina; karl.buhr@unoesc.edu.br

2 **METODOLOGIA**

A visão computacional é a área da ciência que estuda métodos de aquisição, processamento, análise e entendimento de imagens e, no geral, informações do mundo real, com o intuito de produzir dados numéricos ou simbólicos para realizar alguma função ou tomar decisões (KLETTE, 2014).

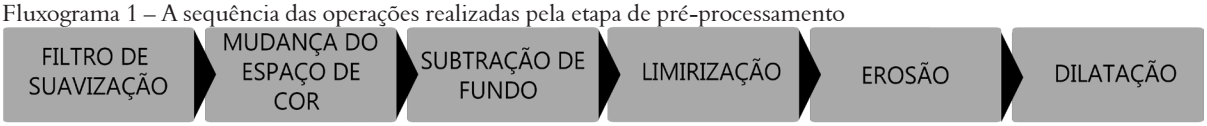
O algoritmo proposto é dividido em duas etapas, a primeira que envolve o pré-processamento da imagem, e a segunda que possibilita o rastreamento dos veículos. Esse sistema de detecção é projetado para trabalhar com vídeos gravados por câmeras de monitoramento de trânsito. Dessa forma, a solução apresentada é considerada iterativa e deve ser executada em cada *frame* (quadro) da imagem do vídeo para detectar a movimentação do objeto.

2.1 PRÉ-PROCESSAMENTO

O pré-processamento das imagens objetiva melhorar a aparência visual destas e enfatizar determinadas características para que a etapa subsequente de detecção de veículos possa executar sua tarefa corretamente, minimizando as chances de falsos positivos serem encontrados.

Nesse caso, o pré-processamento consiste na aplicação de filtros, conversão do espaço de cor, definição da área de interesse, subtração de fundo, operações morfológicas, entre outras operações, conforme indicado no Fluxograma 1.

Fluxograma 1 – A sequência das operações realizadas pela etapa de pré-processamento



Fonte: os autores.

2.1.1 **Filtro de suavização**

Um filtro de suavização é utilizado para borramento e redução de ruídos (GONZALES; WOODS, 2010, p. 100). Ele permite remover pequenos artefatos na imagem e também conectar pequenas descontinuidades para facilitar a extração de objetos por operações posteriores. O *OpenCV* permite especificar a quantidade de suavização do filtro baseado no tamanho de um *kernel* (núcleo), implicando o tamanho do borramento.

2.1.2 **Mudança de espaço de cor**

O propósito de um espaço de cores é facilitar a especificação das cores em alguma forma padronizada, amplamente aceita (GONZALES; WOODS, 2010, p. 264). O espaço de cor padrão do *OpenCV* é o *RGB*, em que cada cor aparece em seus componentes espectrais primários de vermelho, verde e azul, muito utilizado para monitores coloridos.

A conversão do espaço de cor para *HSV* (*Hue*, *Saturation*, *Value*) permite pensar em um ponto de cor como uma combinação de valores de matiz, saturação e intensidade de brilho. Separar a imagem nesses três canais permite ao algoritmo utilizar apenas a informação do canal de brilho para melhorar a precisão da detecção, conforme ilustrado pela Fotografia 1, e também diminuir drasticamente a quantidade de dados que as operações subsequentes devem processar, melhorando o desempenho geral da solução.

Fotografia 1 – Em A observa-se a imagem original em *RGB* e em B observa-se o resultado da conversão para *HSV* na qual a imagem resultante contém apenas a informação do brilho

Fonte: os autores.

2.1.4 **Subtração de fundo**

A subtração de fundo é o método básico para separar um objeto que está em movimento de seu *background* (fundo), resultando em uma imagem que mostra apenas o objeto em movimento (NIXON; AGUADO, 2012, p. 437). Dessa forma, o resultado desse processo de segmentação é um conjunto de *blobs* (manchas brancas).

A técnica de subtração de fundo utilizada é conhecida como *MOG* (*Mixture of Gaussian*), também disponível pela biblioteca do *OpenCV* (OPEN SOURCE COMPUDER VISION LIBRARY, 2015). Essa técnica implementa um método de aprendizagem automática, permitindo a remoção do fundo sem que o usuário tenha que especificar manualmente o *background*.

A Fotografia 2 demonstra o resultado da subtração de fundo. Essa operação pode ser representada pela Equação simples (1):

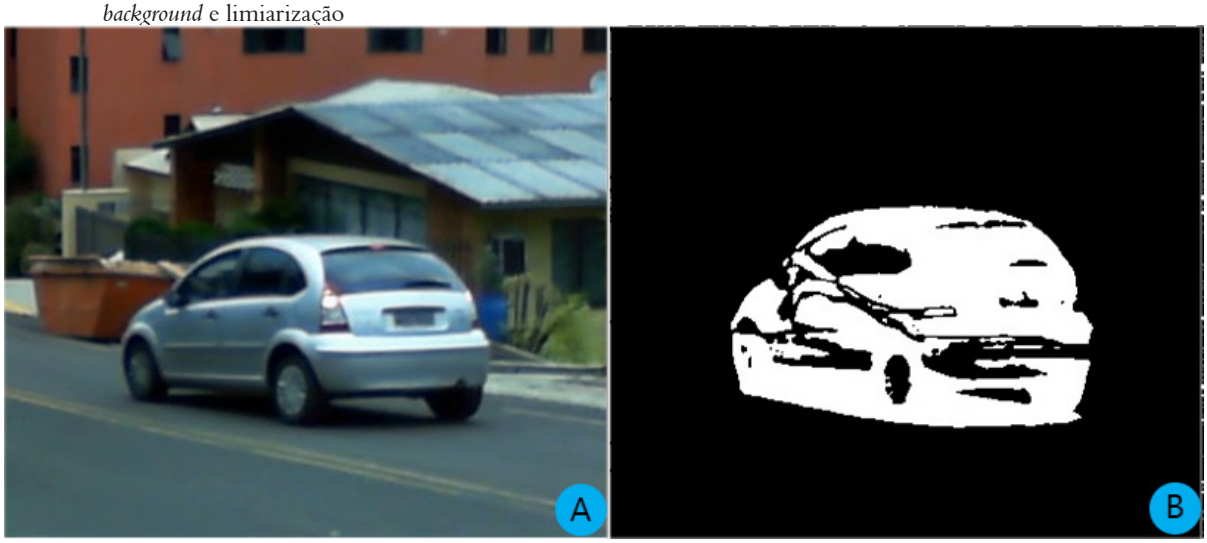
$$Seg = BG.Frm \quad (1)$$

Na qual *Seg* é o resultado de *BG* (*background*) pela operação lógica AND (.) (OU) em *Frm* (*frame* atual). O resultado contém apenas as diferenças entre as imagens, ou seja, são removidas as partes iguais entre o plano de fundo e o *frame*.

2.1.5 **Limiarização**

A operação de subtração de fundo resulta em uma imagem segmentada em polígonos brancos com várias tonalidades, o que dificulta a detecção, pois caso o tom seja muito próximo, mas não zero, a operação de rastreio pode acabar gerando um falso positivo. Então, é gerada uma imagem limiarizada em que os tons serão arredondados conforme a proximidade, ou seja, o resultado será branco e preto sem outras tonalidades. A Fotografia 2 mostra os resultados dos três últimos passos da operação de tratamento.

Fotografia 2 – Imagem A é a imagem original e a imagem B é o resultado do processo de corte de área de interesse, remoção de *background* e limiarização



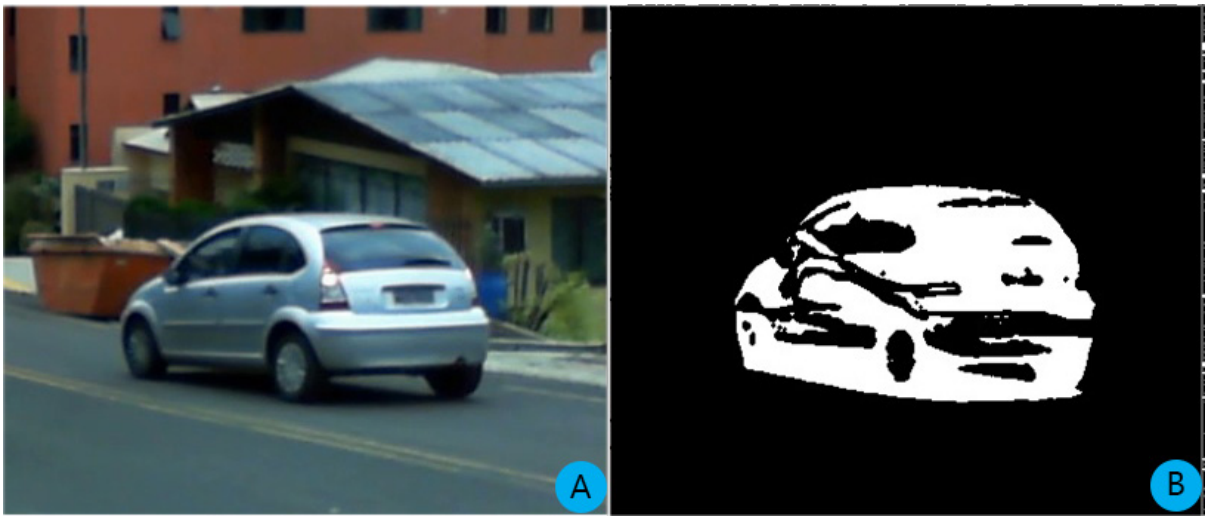
Fonte: os autores.

2.1.6 **Erosão**

A Erosão é um processo simples de "corroer" imagens, permitindo remoção de ruído ou afinamento/redução de *Blobs* (Manchas brancas).

É aplicado a uma erosão nos *frames* com intuito de remover barulho presentes nestes, mas nessa imagem foi desnecessário, pois o *blur* foi suficiente para remover o ruído da imagem, notando-se pouca diferença, como é mostrado na Fotografia 3. (OPEN SOURCE COMPUTER VISION LIBRARY, 2015b; NIXON; AGUADO, 2012, p. 124).

Fotografia 3 – Imagem A é original e a imagem B é limiarizada após processo de erosão

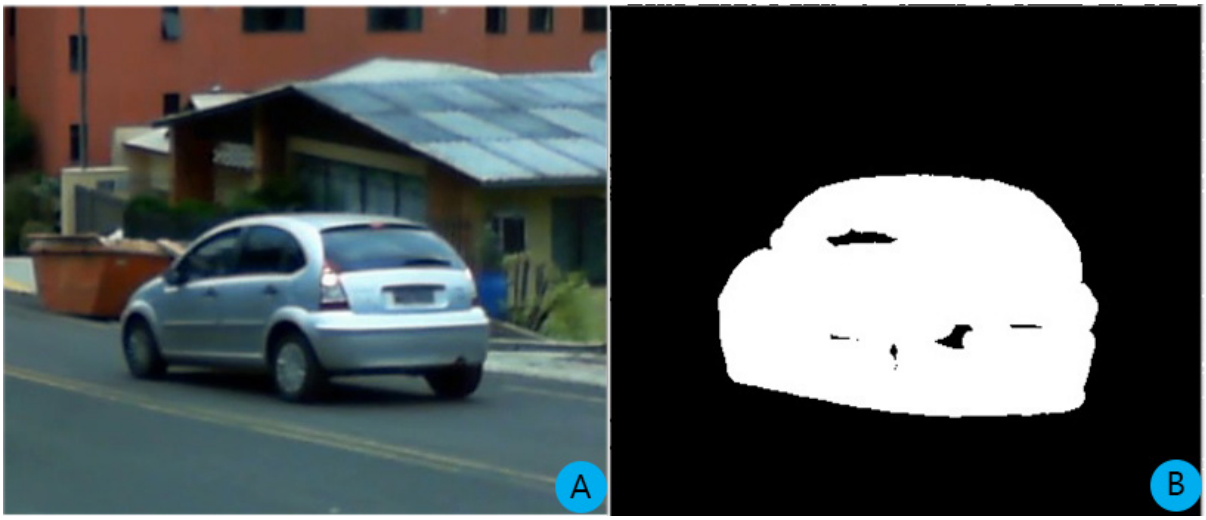


Fonte: os autores.

2.1.7 **Dilatação**

A dilatação faz o processo contrário da erosão, fazendo com que os *blobs* possam aumentar de tamanho. Essa operação é usada nesse algoritmo para gerar uma ligação com as outras partes desconectadas, diminuindo os buracos e solidificando o *blob*. A Fotografia 4 mostra o resultado na imagem teste (OPEN SOURCE COMPUTER VISION LIBRARY, 2015b; NIXON; AGUADO, 2012, p. 126).

Fotografia 4 – A imagem A é original e a imagem B é a limiarizada após processo de dilatação

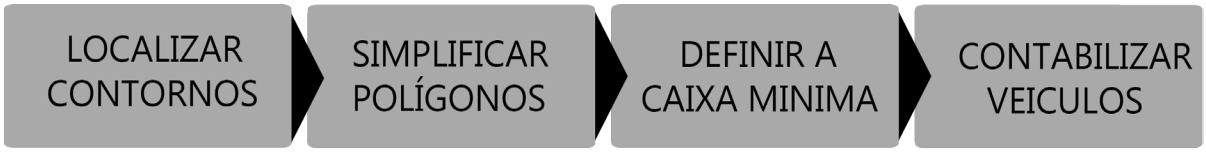


Fonte: os autores.

2.2 OPERAÇÃO DE RASTREIO DE VEÍCULOS

Essa operação tem o papel de localizar os carros na imagem recebida do processo de tratamento e desenhar um retângulo ao redor do veículo detectado como forma de indicar que certo automóvel foi localizado. Essa operação segue a ordem indicada no Fluxograma 2.

Fluxograma 2 – Operações para rastrear a movimentação



Fonte: os autores.

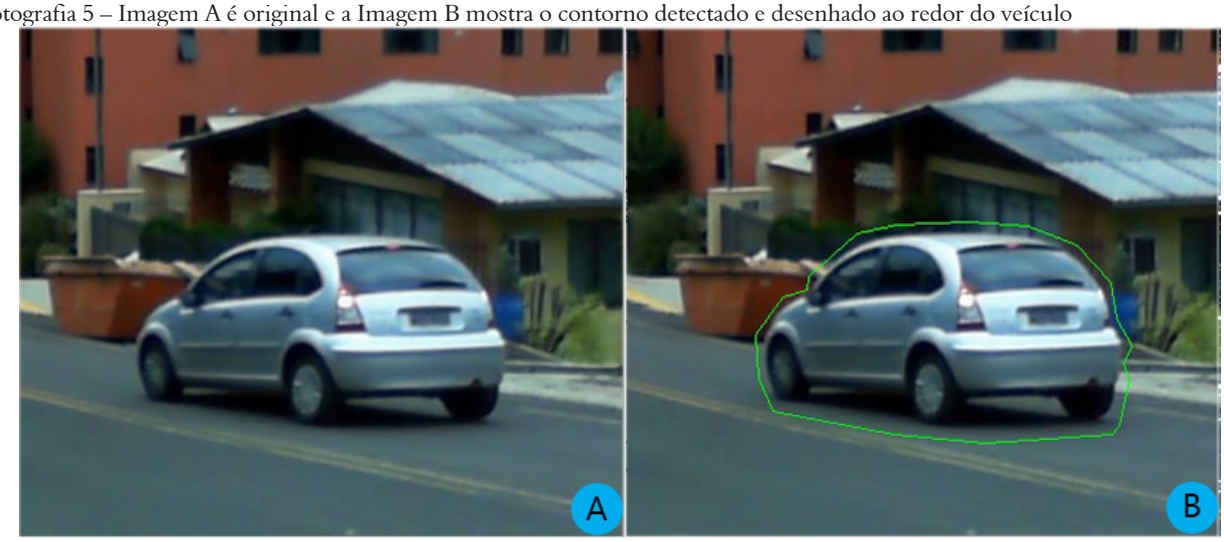
2.2.1 **Achar contornos**

Utilizando o algoritmo *FindCountours*, são achados contornos de todos os objetos na imagem limiarizada, essa operação retorna um vetor de vetores de pontos que indicam a estrutura de um ou vários polígonos que representam o contorno ao redor dos veículos na imagem (OPEN SOURCE COMPUTER VISION LIBRARY, 2015b).

2.2.2 **Simplificação de polígonos**

Essa etapa recebe o vetor encontrado anteriormente contendo os polígonos (contornos). O objetivo desse passo é reduzir a quantidade de vértices no polígono, usando um algoritmo que foi desenvolvido com base no algoritmo de Ramer–Douglas–Peucker, assim, permitindo que a operação de acoplamento consiga concluir sua tarefa de forma mais rápida sem perder a precisão. A Fotografia5 mostra com intuito demonstrativo o contorno detectado desenhado ao redor do carro. (KARTHAUS, 2012; WU; MARQUEZ, 2005; OPEN SOURCE COMPUTER VISION LIBRARY, 2015e).

Fotografia 5 – Imagem A é original e a Imagem B mostra o contorno detectado e desenhado ao redor do veículo



Fonte: os autores.

2.2.3 **Acoplamento em retângulos**

Essa etapa coleta os polígonos simplificados da etapa anterior e calcula o menor retângulo possível que conforte todos os vértices de um polígono encontrado (OPEN SOURCE COMPUTER VISION LIBRARY, 2015e).

2.2.4 **Desenhando nos *frames***

Desenhar nos *frames* é uma tarefa simples; na Fotografia 6 uma operação de desenhar contornos chamada *drawContours* foi usada para fazer a demonstração dos contornos encontrados, mas nessa etapa intencionou-se desenhar retângulos em cima dos carros encontrados. Então, com as dimensões coletadas da operação anterior de acoplamento, serão desenhados os retângulos (OPEN SOURCE COMPUTER VISION LIBRARY, 2015e, 2015a).

2.2.5 **Contagem de veículos**

Essa etapa de contagem de veículos cria um retângulo virtual que fica armazenado na memória. A detecção funciona de modo que a cada *frame* que um carro é localizado, uma verificação é feita para calcular se este tem o ponto central de seu retângulo dentro do retângulo virtual. Caso sim, assume-se que um carro está passando pela área contadora, ele incrementa o contador e espera o carro sair do retângulo para poder detectar um outro carro.

Na Fotografia 6, encontra-se o resultado completo do algoritmo contador, a linha vermelha serve como indicador da parte inferior do retângulo, caso um carro seja contado, essa linha piscará com a cor verde.

Fotografia 6 – A imagem A é original e a Imagem B é o resultado do algoritmo contador de carros

Fonte: os autores.

3 **RESULTADOS**

Os vídeos utilizados para teste podem ser encontrados em <https://mega.co.nz/#F!xhRQWSLR!RmVsnHBtHBPLz0UBGW4tkw> (mar. 2015).

Na Fotografia 7 (A), pode-se observar o momento em que o algoritmo apresentou uma falha na detecção, em virtude da supersegmentação. Isso acontece quando a etapa de subração de fundo resulta em muitos blobs e a etapa de dilatação não consegue conectar todos esses segmentos.

Um caso de sucesso pode ser observado na Fotografia 7 (B), que mostra os veículos sendo detectados corretamente. O fato de eles estarem mais próximos da câmera ajuda a evitar os artefatos produzidos pela supersegmentação, aumentando a precisão da detecção.

A Tabela 1 apresenta o tempo médio de processamento dos frames nos vídeos testados.

Fotografia 7 – Na imagem A, uma falha momentânea na detecção em razão da supersegmentação do objeto; na imagem B o mesmo objeto sendo detectado com sucesso alguns instantes depois, mais próximo à câmera.

Fonte: os autores.

Tabela 1 – Desempenho do algoritmo desconsiderando impressão na tela ou salvamento em disco

Nome do arquivo	Tam. vídeo (Mb)	Duração (Minutos)	Resolução (Pixel)	Nº total de frames	Tempo total de processamento (Segundos)
Arq1.mkv	2000	1:46	1280x720	3156	81,593
Arq2.mp4	211	2:55	1280x720	5239	108,614
Arq3.mp4	185	2:34	1280x720	4609	94,254

Fonte: os autores.

4 DISCUSSÃO

Os resultados mostram que é possível utilizar a solução proposta para rastrear a movimentação de veículos e de objetos com um alto grau de sucesso. Para diminuir o problema da supersegmentação, é recomendada a utilização de filtros lineares de suavização mais eficientes, como o filtro bilateral.

No entanto, para utilizar essa solução em tempo real, é necessário investigar formas de aumentar o desempenho da etapa de tratamento de vídeo. Os resultados vistos na Tabela 1 mostram apenas o desempenho de processamento, sem considerar o salvamento das imagens ou a impressão na tela. Uma outra forma de acelerar a execução do algoritmo seria ignorar os frames pares, já que nenhuma modificação significativa acontece de um frame para outro.

Em trabalhos futuros recomenda-se a análise dos contornos como forma de classificar o veículo pelo tamanho do objeto detectado, já que motos, carros e caminhões podem ser identificados e classificados pelo seu tamanho.

Development of a computer vision system for vehicle tracking

Abstract

Most traffic monitoring systems currently deployed utilize human operators to monitor occurrences. This study presents an approach to automatically detect and track the movement of vehicles on digital images of road traffic to allow operators to easily identify the number of vehicles and the hours of more traffic on the highway, regardless of the license plate is present. We developed a software prototype with image processing and computer vision techniques that includes morphological operations, limiarization, background subtraction with MOG (Mixture of Gaussian), among others, to work with traffic monitoring videos. The videos used during the experiments allow the system to perform the vehicles counting properly and automatically in most tests, however,in these preliminary results were identified situations where the algorithm momentarily loses accuracy and these interesting cases are also discussed.
Keywords: Object detection. Vehicle tracking. Computer vision.

REFERÊNCIAS

ASSOCIAÇÃO NACIONAL DOS FABRICANTES DE VEÍCULOS AUTOMOTORES. **Anuário da indústria automobilística brasileira**. Disponível em: <http://www.anfavea.com.br/anuario.html>. Acesso em: 07 mar. 2015.

GONZALES, R. C.; WOODS, R. E. **Processamento digital de imagens**. 3. ed. São Paulo: Pearson Prentice Hall, 2010.

KARTHAUS, M. **JavaScript implementation of the Ramer Douglas Peucker Algorithm**. 2012. Disponível em: <http://karthaus.nl/rdp/>. Acesso em: 04 mar. 2015.

KLETTE, R. **Concise Computer Vision**. [S.l.: s.n.], 2014.

NIXON, M. S.; AGUADO, A. S. **Feature Extraction and Image Processing for Computer Vision**. 3. ed. Elsevier, 2012.

OPEN SOURCE COMPUTER VISION LIBRARY. **Drawing Functions**. Disponível em: <http://docs.opencv.org/modules/core/doc/drawing_functions.html>. Acesso em: 04 mar. 2015a.

ERODING AND DILATING. Disponível em: <http://docs.opencv.org/doc/tutorials/imgproc/ erosion_dilatation/erosion_dilatation.html>. Acesso em: 03 mar. 2015b.

HOW TO USE BACKGROUND SUBTRACTION METHODS. Disponível em: <http://docs.opencv.org/trunk/doc/tutorials/video/background_subtraction/background_subtraction.html>. Acesso em: 03 mar. 2015c.

IMAGE FILTERING. Disponível em: <http://docs.opencv.org/modules/imgproc/doc/filtering.html>. Acesso em: 03 mar. 2015d.

STRUCTURAL ANALYSIS AND SHAPE DESCRIPTORS. Disponível em: <http://docs.opencv.org/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html>. Acesso em: 04 mar. 2015e.

SINDICATO DOS CENTROS DE FORMAÇÃO DE CONDUTORES DE SANTA CATARINA. **Câmeras de monitoramento na BR-101 em Santa Catarina ajudam no trabalho de policiais rodoviários**. Blumenau, 2014. Disponível em: <http://www.sindemosc.com.br/index.php?option=com_content&task=view&id=6406&Itemid=92>. Acesso em: 18 mar. 2014.

WU, S. T.; MARQUEZ, M. R. G. **A non-self-intersection Douglas-Peucker Algorithm**. 2005. Disponível em: <http://www.dca.fee.unicamp.br/~ting/Publications/P2001-2005/wu-roci-2003-rfm.pdf>. Acesso em: 04 mar. 2015.