Contagem de veículos em uma via a partir de vídeo

Alex Teixeira Benfica
UFMG
Visão Computacional
Prof.: Mário Fernando Montenegro

***Abstract*—Este trabalho propõe uma abordagem simples e computacionalmente eficiente para contagem de veículos em vias de 1 pista e única direção. Os veículos são segmentados levando-se em consideração a detecção de movimentos por diferença do background e do frame anterior. A detecção e remoção de áreas de sombra é usada com o intuito de melhorar a precisão da segmentação. A partir do resultado da segmentação do veículo uma heurística simples com utilização de máquina de estados é usada para tratar o contador. Resultados experimentais mostram que esta proposta pode ser usada com sucesso para localização e contagem de veículos.**

***Keywords*-contagem de veículos, detecção de veículos, rastreamento de veículos**

I. Introdução

Para automatizar os sistemas de controle de tráfego nos dias atuais é necessário o emprego de substancial poder computacional aliado a imagens obtidas por câmeras. A utilização de sensores mais simples tem se tornado inviável a cada dia, seja pela quantidade limitada de informação que fornecem, ou mesmo pelas dificuldades inerentes à sua instalação e manutenção. Ao utilizar câmeras, por outro lado, a quantidade de informação a ser processada aumenta e isto exige softwares eficientes para tornar o processamento útil ao ser executado em tempo real.

A abordagem utilizada neste trabalho pode ser executada em tempo real e consiste de 3 processos em sequência: segmentação do veículo, detecção do comprimento do mesmo e, por fim, contagem de ocorrências.

A segmentação é feita em três passos consecutivos que geram imagens intermediárias a partir da diferença entre cada frame e o background e também com o frame anterior. Em cada frame do vídeo a sombra é detectada e removida. Com este processo é possível segmentar o veículo de maneira suficiente para o passo seguinte.

A detecção do comprimento de cada veículo é feita pela medida direta da distância entre o início e o fim do mesmo em cada frame. A precisão deste comprimento é melhor quanto mais realista for a remoção da sombra, tratada em III-E.

A partir do comprimento detectado, entra em cena a contagem de ocorrências de veículos, que rastreia o veículo em cada frame evitando a múltipla contagem de um mesmo veículo. Esta contagem é feita a partir de uma máquina de estados.

Esta abordagem se mostrou computacionalmente eficiente e pode ser adaptada para funcionamento em real-time utilizando linguagens de programação compiladas.

II. Trabalhos relacionados

O rastreamento de veículos tem sido tratado com diferentes abordagens ao longo dos anos, e esta abordagem se mostra influenciada pela crescente qualidade de vídeos das câmeras de vigilância mais recentes.

A detecção do background foi criada a partir de ideias de [1] onde são estudadas várias técnicas para subtração do background.

A ideia de utilizar transformações morfológicas e subtração da imagem de fundo surgiu em [2] onde a subtração da imagem de fundo e a segmentação da imagem baseada em transformações morfológicas como erosão e dilatação são feitas em cada frame do vídeo. O algoritmo proposto em [2] segmenta a imagem preservando bordas importantes que aumentam a confiabilidade da segmentação em relação às mudanças que ocorrem no ambiente da cena.

Os problemas devido à inerente complexidade do trânsito em algumas vias relatados em [3] levaram à inicial restrição de contar veículos apenas em 1 via e 1 sentido. Apesar de o trânsito não ser necessariamente confuso como em [3], simples casos de oclusão poderiam levar o presente algoritmo a falhar e isto justifica a restrição.

A motivação para utilização de vídeos de baixa resolução veio de [4] que também apresenta um algoritmo que é escalável com a resolução do vídeo de entrada e é processado em tempo real. [4] exige calibração das câmeras utilizadas, o que não é desejável no contexto deste trabalho.

Outros trabalhos como [5] sugerem abordagens para diminuir a influência da iluminação noturna na detecção dos veículos e ideias contidas neste trabalho serão de grande valia ao adaptar este algoritmo para funcionamento à noite.

III. Metodologia

A. Obtenção do vídeo

1) Características do vídeo: Vídeo com resolução de 320x240 e 24 bits de cor e 30 quadros por segundo. O algoritmo funciona também com outras resoluções, sendo

necessário ajustar os parâmetros citados em III-G. O tempo de processamento aumenta linearmente com o número de linhas do vídeo. O número de quadros é um fator importante pois sua diminuição implica no aumento das diferenças entre frames tratadas em III-D e pode ser necessário mudar partes significativas do algoritmo para lidar com taxas de frames muito inferiores à 30.

2) Posicionamento da câmera: A câmera captura os veículos lateralmente. O vídeo utilizado foi obtido a partir de uma câmera posicionada a uma altura de 8 metros e uma distância de 6 metros da lateral da via, medida no nível do chão. A câmera foi posicionada de forma que a base do vídeo ficasse paralela à lateral mais próxima da via. O posicionamento também permite que qualquer veículo que passe pela via seja integralmente capturado em relação à sua altura, não ficando portanto partes fora da imagem. Como o objetivo é apenas a contagem dos veículos, existem casos em que veículos grandes não tem seu comprimento totalmente capturado em nenhum frame do vídeo.

B. Definição preliminar do algoritmo

O algoritmo utilizado precisa segmentar o veículo em cada frame do vídeo para realizar a contagem, o que é feito seguindo cada um dos passos abaixo.

- Conversão para cinza.
- Segmentação por subtração do background, descrito em III-C2.
- Segmentação por subtração do frame anterior, descrito em III-D.
- Remoção de sombra, descrito III-E.
- Operações morfológicas, em III-F
- Detecção de início e fim de cada veículo, em III-G

Figure 1. Exemplo de um frame antes do processamento: IMG_{FR}

Por fim, a variação em cada frame destas coordenadas de início e fim do veículo é analisada indicando quando há um novo veículo na cena ou se a imagem é do mesmo veículo detectado no frame anterior.

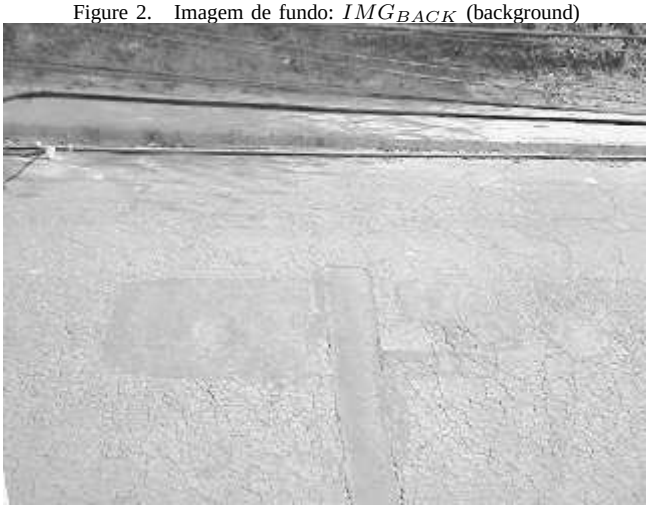
C. Segmentação por subtração do background

1) Criação da imagem de background: A segmentação por subtração do background exige uma imagem que represente o background da cena sem veículos ou outros elementos móveis. A imagem de background é adquirida a partir dos 50 primeiros frames do vídeo que são processados da seguinte forma:

- Converte-se o frame para cinza, gerando a imagem IMG_{FR} .
- Converte-se o frame anterior para cinza, gerando a imagem IMG_{FRA} .
- Calcula-se a diferença absoluta de cada pixel entre IMG_{FR} e IMG_{FRA} , gerando a imagem *frAbsDif*.
- Conta-se o número de pixels que são zero em *frAbsDif*, gerando o contador *pixelsDiferentesAbs*.
- É feita uma limiarização de *frAbsDif* usando limiar $=$ 45, gerando a imagem *frThrDif*.
- Conta-se o número de pixels com valor zero em *frAbsDif*, gerando o contador *pixelsDiferentesThr*.
- Se *pixelsDiferentesThr* $=$ 0, a imagem é adicionada à lista de candidatos à background: *candidatos*.

De todas as imagens na lista *candidatos*, o background é aquela que tem o menor valor no contador *pixelsDiferentesAbs*.

Figure 2. Imagem de fundo: IMG_{BACK} (background)



A imagem resultante deste passo é chamada de IMG_{BACK} no restante deste artigo.

2) Diferença entre o frame e o background: É calculada a diferença absoluta entre IMG_{FR} e IMG_{BACK} e feito uma limiarização com limiar 40. O resultado é imagem IMG_{DIFBG} .

D. Segmentação por subtração do frame anterior

É calculada a diferença absoluta entre IMG_{FR} e IMG_{FRA} e feito uma limiarização com limiar 40. O resultado é imagem IMG_{DIFFRA} .

Figure 3. Diferença em relação ao background: IMG_{DIFBG}

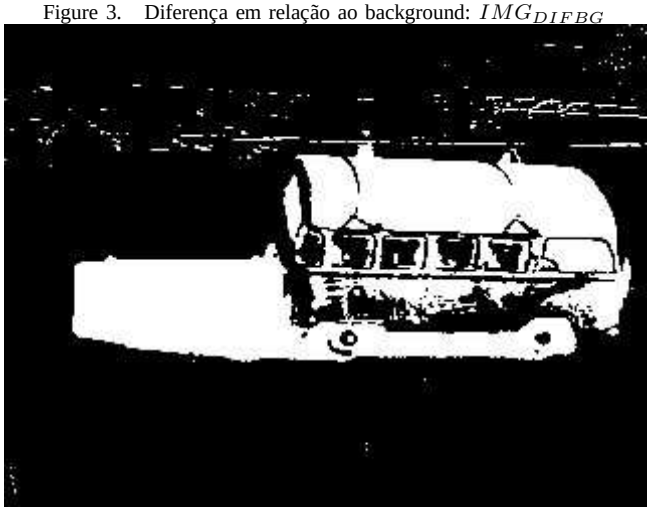
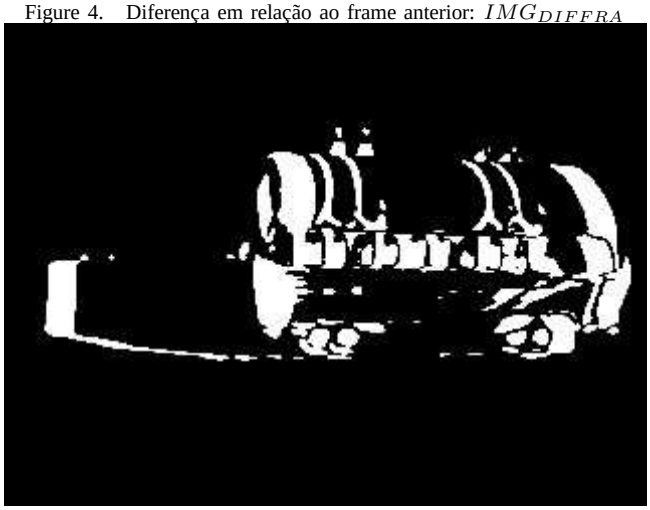


Figure 4. Diferença em relação ao frame anterior: IMG_{DIFFRA}



E. Remoção de sombras

As sombras que os veículos fazem sobre a pista causam distorções significativas no momento de segmentar o veículo e precisa ser removida. Em cada frame é aplicado o método descrito em [6] para remoção ou minimização dos efeitos das sombras na imagem. Este método é muito sensível à iluminação da cena e os 3 limiares utilizados por ele precisam ser escolhidos manualmente para cada iluminação. Entretanto é possível estimar estes parâmetros a partir de medidas no brilho geral de cada frame da imagem.

A saída do algoritmo em [6] é uma máscara chamada IMG_{SOMBRA} que descreve as regiões de sombra na imagem. Esta sombra geralmente apresenta *buracos* e regiões pouco definidas. Sobre esta máscara e as imagens IMG_{DIFFRA} e IMG_{DIFBG} são aplicadas algumas operações morfológicas, descritas abaixo.

Figure 5. Exemplo de sombra detectada em um frame: IMG_{SOMBRA}

F. Operações morfológicas

As seguintes operações são realizadas em sequência sobre IMG_{SOMBRA} :

- Duas operações de dilate e o resultado, binário, é invertido.
- Do resultado da operação anterior, é feito um AND com IMG_{DIFBG} gerando $IMG_{semSombra}$

Agora, sobre $IMG_{semSombra}$ são feitas as seguintes operações em sequência: $IMG_{semSombra}$ é copiado em IMG_B

- Filtro de mediana 3x3
- Erode 2x (função cvErode do OpenCV)
- Dilate 3x (função cvDilate do OpenCV)
- AND com IMG_B que gera IMG_C

Agora, são feitos dois Dilates sobre IMG_{DIFFRA} e do resultado, um AND com IMG_C , gerando finalmente IMG_{Carro}

Todas estas operações morfológicas tem o objetivo de reduzir ruídos. Os principais ruídos da imagem são causados por sombras fracas que não são sempre bem detectadas por [6] ou por ruídos inerentes à gravação que aparecem na diferença entre quadros. Outro tipo de ruído que se mostra comum é o ruído causado pela mudança na iluminação do fundo com o passar do tempo. Como o fundo vai se tornando diferente em cada quadro, a imagem IMG_{BACK} deixa de ser uma boa representação do fundo.

G. Detecção de início e fim de cada veículo

A partir da imagem IMG_{Carro} os pixels iniciais e finais de cada veículo são detectados. Em cada linha da imagem os pontos que correspondem ao primeiro e último pixel do veículo são marcados. Estes pixels são o primeiro e o último pixel branco de cada linha de IMG_{Carro} , respectivamente.

Em cada linha da imagem a coordenada x destes pixels é variável e por vezes encontrada em regiões distantes do real

Figure 6. Carro segmentado: IMG_{Carro}

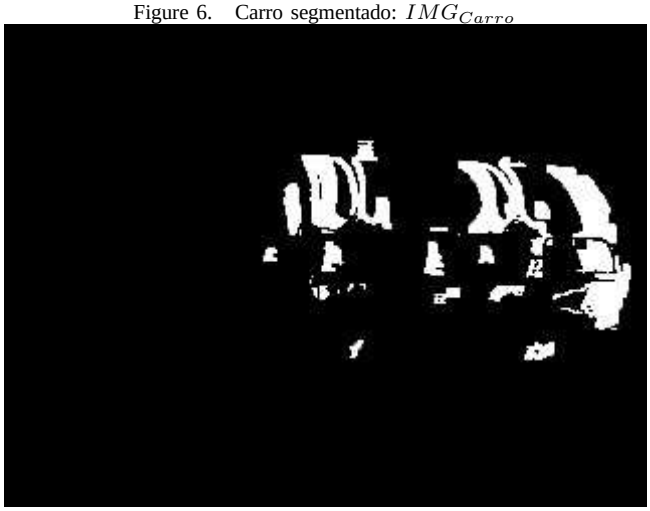


Figure 7. Início e fim do veículo: linhas $minLinhaInicial$ e $maxLinhaFinal$

início do veículo, seja sobre o veículo ou fora dele em outras partes da cena. Tal comportamento é ruim pois diminui a precisão da medida. Para diminuir a influência deste ruído, são considerados apenas alguns destes pixels detectados:

- Nos pixels que determinam o início do veículo, a coordenada x_{inicio} que determina o início do veículo é dada pela média das $n/2$ menores coordenadas x dos pixels iniciais.
- Nos pixels que determinam o final do veículo, a coordenada x_{final} que determina o final do veículo é dada pela média das $n/2$ maiores coordenadas x dos pixels finais.

A partir de x_{inicio} e x_{final} o valor $distLinhas$ é calculado como sendo a diferença absoluta entre x_{final} e x_{inicio} .

A heurística para detecção de veículos é realizada então a partir dos 3 valores x_{final} , x_{inicio} e $distLinhas$ e dos seguintes limiares:

- $distMinExisteVeic$: distância mínima entre as linhas para considerar a existência de algum veículo na cena. Diminuir este limiar permite contar veículos menores, mas pode causar erros ao contar também pedestres.
- $minLinhaInicial$: posição mínima da linha inicial no quadro.
- $maxLinhaFinal$: posição máxima da linha final no quadro.

Os limiares $minLinhaInicial$ e $maxLinhaFinal$ são relacionados à resolução do vídeo e distância da câmera até a pista. Podem inclusive ser calculadas dinamicamente em função destas informações.

O algoritmo de contagem é uma máquina de estados com apenas 2 estados, $temVeic$ e $naoTemVeic$, iniciando no estado $naoTemVeic$. A mudança para o estado $temVeic$ ocorre quando os seguintes requisitos são atendidos.

- $distLinhas > distMinExisteVeic$ e $x_{inicio} > minLinhaInicial$ e $x_{final} > maxLinhaFinal$

Somente no momento da transição para o estado $temVeic$, o contador de veículos é incrementado. Para voltar ao estado $naoTemVeic$, é necessário detectar que o veículo não está mais completamente presente no frame. A mudança para o estado $naoTemVeic$ ocorre quando:

- $distLinhas < distMinExisteVeic$ ou ($x_{inicio} < 80$ e $x_{final} < 80$) or ($x_{inicio} > 240$ e $x_{final} > 240$) or $distLinhas > 280$

Os valores constantes 80, 240 e 280 são proporcionais à largura do vídeo que, no caso do experimento tem 320 pixels.

IV. Experimentos e resultados

Foram realizadas medidas com 2 vídeos adquiridos em situações de luminosidade bem diferentes. Um com uma cena bem clara ($video_c$)e outro com uma cena mais escura ($video_e$). Em $video_c$ são 12 veículos e 1 pedestre. Em $video_e$ são 23 veículos, sem pedestres.

Para cada um dos vídeos os valores dos limiares para detecção de sombras precisou ser ajustado manualmente. Nestes ajustes a detecção da sombra foi satisfatória na maior parte dos quadros, exceto em situações de iluminação muito intensa e que a sombra se apresentou muito clara em termos de luminosidade. Nos casos em que a detecção de sombras foi deficiente, a sombra foi considerada parte dos objetos em movimento e contribuiu para aumentar as medidas de comprimento dos mesmos. Esta falha levou um pedestre a ser contado como veículo visto que a sombra o deixava com comprimento bem maior na imagem.

A contagem obteve índices de acertos diferente em cada um dos vídeos testados, sendo nv o número de veículos em cada vídeo.

Os seguintes critérios foram avaliados:

1) Contagem correta do veículo: de nv, quantos foram contados corretamente.
2) Contagem múltipla: de nv, quantos foram contados mais de uma vez.
3) Número de pedestres ou outros elementos contados incorretamente

A tabela a seguir mostra o resultado para cada um dos critérios nos dois vídeos avaliados.

Critério	$video_c$	$video_e$
1	11/12 (91%)	22/23 (95%)
2	01/12 (8.3%)	00/23 (0%)
3	01/01 (100%)	00/00 (0%)

A. Desempenho computacional

Todos os testes foram realizados na seguinte configuração de hardware e software, citando apenas os itens mais relevantes.

- Processador Intel Core i5 650
- 4GB RAM
- Virtualização através do VirtualBox executando Ubuntu com 2GB de RAM sobre sistema operacional host Windows 7.
- OpenCV 2.2 com binding para Python 2.6

A virtualização não entrega ao sistema convidado o mesmo poder computacional do processador físico, mas a queda de desempenho nesta configuração não foi tão significativa. O desempenho do disco não é também o gargalo deste algoritmo visto que o o tempo de leitura de cada frame do vídeo, já diluído o tempo inicial de carregamento do vídeo, representa menos de 0.2% do tempo de processamento de cada frame.

As funções mais pesadas do algoritmo foram feitas utilizando a linguagem Python, mas através da iteração sobre os pixels do frame buscados um a um por chamadas a funções da OpenCV. O processamento de cada frame utiliza em média 175ms. Como o vídeo tem 30 frames por segundo, este tempo precisa ser reduzido em pelo menos 6x para possibilitar o processamento em tempo real de cada novo frame em 33ms.

Todas as linhas do vídeo são tratadas em todos os casos e isto nem sempre é necessário. O mesmo vale para a resolução, pois há situações em que o processamento do frame pode ser feito em resolução menor.

Portanto, esta redução de tempo de processamento é sem dúvida possível, bastando realizar algumas otimizações e implementar o algoritmo em C ou outra linguagem compilada.

V. Conclusão

A contagem de veículos em vias de 1 faixa e apenas 1 direção apesar de ter aplicações mais restritas do que algoritmos mais genéricos e pode ser aplicada com sucesso em diversos casos.

Em um dos vídeos a máquina de estados contou 2 vezes um veículo que parou e recomeçou o movimento. Tal situação é também simples de resolver pois não envolve diretamente o ponto mais complexo que é a segmentação do veículo.

Os níveis de acerto do algoritmo podem ser muito melhorados posicionando-se a câmera em posição acima e mais afastada da via, tendendo à perpendicularidade. Os níveis de acerto de até 95% como no caso do $video_e$ são exemplos de resultados onde todos os parâmetros foram ajustados de forma manual e em $video_c$ os mesmos parâmetros ajustados em $video_e$ foram usados. Esta diferença entre estes valores sugere que se forem feitos ajustes sistemáticos e automáticos relacionados à iluminação de cada frame do vídeo, a segmentação do veículo, que depende da detecção da sombras poderá ser beneficiada.

As variações de iluminação e posição das sombras de objetos fixos (como folhas das árvores) tornaram obsoleta a imagem de fundo determinada no início do vídeo. Alguns minutos depois esta imagem não mais representava de forma fiel o background dos frames. Isto não chegou a causar problemas mas foi fácil perceber pelas linhas exibidas no vídeo de depuração que as variações das linhas se tornavam cada vez mais instáveis, o que poderia ter ocasionado erros.

Apesar de terem sido processados mais de 10 mil frames, os testes foram feitos com um número bem limitado de vídeos adquiridos no mesmo local, o que é pouco representativo para fins estatísticos.

VI. Trabalhos futuros

O fato do algoritmo não contar corretamente a existência de mais de um veículo no frame ou em direções diferentes pode se completamente resolvido trabalhando a heurística usada em III-G sem que nada seja modificado no processamento anterior de imagem dos frames.

A questão da detecção de sombras pode ter sua precisão aumentada se os parâmetros do algoritmo de sombras forem melhor determinados em tempo de execução, ao invés de fixos. O sistema está modular, logo outros algoritmos de detecção de sombra podem ser facilmente testados desde que retornem uma máscara.

A detecção da velocidade dos veículos pode ser implementada analisando-se a variação de $minLinhaInicial$ e $maxLinhaFinal$, bem como a detecção de categoria do veículo pode ser feita a partir do comprimento do mesmo, dado por $distLinhas$.

A estimativa do background pode ser refeita sempre que não for detectado movimento no vídeo ou usando qualquer das técnicas propostas em [1]. Isto sem dúvida alguma irá melhorar a precisão da segmentação dos veículos.

É necessário também definir uma base de dados de testes com maior variação de situações para aferir de forma mais sistemática os resultados do algoritmo.

Seria também muito interessante para fins comerciais que o algoritmo pudesse funcionar com câmeras de CFTV com baixa resolução e frame rate.

AGRADECIMENTOS

Este trabalho somente foi possível graças à excelente motivação oferecida ao observar e viver o profissionalismo e entusiasmo do professor Mário e dos monitores da disciplina de Visão Computacional. Apesar de não ter havido nenhuma contribuição direta e específica, destaco que as aulas do monitor Erickson Nascimento foram de excelente valia no quesito técnico ao mostrar e comparar as ferramentas como OpenCV e MATLAB, mas foram ainda mais importantes do ponto de vista motivacional ao deixar transparecer a enorme paixão pela área.

Registro também o reconhecimento da imprescindível colaboração dos colegas da disciplina ao ajudar nas várias conversas e discussões de métodos e mesmo avaliação de resultados preliminares que foram atingidos. Especiais agradecimentos ao Fabrício, Celso, Leonardo e Samuel que participaram com extremo interesse tanto pessoalmente quanto através de tópicos na lista de discussão da disciplina.

REFERENCES

[1] S.-C. S. Cheung and C. Kamath, “Robust techniques for background subtraction in urban traffic video.”

[2] P. P.M.Daigavane, “Real time vehicle detection and counting method for unsupervised traffic video on highways.”

[3] N. N. A. Le Quoc Anh, “Applying computer vision to traffic monitoring system in vietnam,” 2005.

[4] C. K. M. Maire, “Tracking vehicles in traffic surveillance video.”

[5] M. L. Kim-Sung Jie, “Computer vision based real-time traffic monitoring system.”

[6] J. L. Gordillo, “Navdyn:navigation of an autonomous vehicle in dynamic environments,” http://dali.mty.itesm.mx/ autonomos/Navdyn/node11.html.