

# Trabalho 6 - Background subtraction

Ruan Pires - 14103318

31 de outubro de 2018

## 1 Introdução

Este trabalho teve como objetivo desenvolver algoritmos, utilizando a linguagem de programação C++ e a biblioteca OpenCV, para realizar detecção de movimento, ou, equivalentemente a remoção de *background* em vídeo. O trabalho foi dividido em duas etapas: Na primeira, é demonstrado um método simplificado que utiliza como base para o *background* o primeiro *frame* do vídeo. Esse método demonstrou-se fácil de implementar, pois envolve apenas a diferença de dois *frames*, porém muito sensível a variações na posição da câmera, além de não levar em consideração objetos em movimento no primeiro frame ou objetos que estacionam na cena após algum tempo.

Na segunda etapa, no intuito de corrigir as falhas apresentadas pelo primeiro algoritmo, foi desenvolvido um algoritmo de subtração de *background* adaptativo. O algoritmo proposto soma o *frame* atual ao último *background* estimado de maneira ponderada, realizando assim uma média temporal e permitindo que o *background* estimado mude durante o vídeo. Para melhor avaliar a performance do algoritmo proposto na segunda etapa em uma aplicação real, este foi utilizado para contagem de carros transitando em uma estrada.

## 2 Primeira etapa - Background simples

O conceito de movimento, nessa primeira parte do trabalho, foi resumido à seguinte idéia: É a diferença entre um frame e o frame-base estático. Para representar este frame-base estático, foi escolhido o primeiro frame do vídeo a ser utilizado. Esta solução tem como ponto positivo a fácil implementação e, dependendo da aplicação, a alta sensibilidade ao movimento em cena. Por exemplo, se aplicada em uma linha de produção, em que o fundo da imagem é conhecido e estático, e existir a garantia de que a câmera não sofrerá perturbações, pode ser uma boa solução. Porém, casos como esse são raros, uma vez que em aplicações reais, perturbações e ruídos são inevitáveis.

Para testar o algoritmo, um vídeo de câmera, supostamente, estática em uma autoestrada foi utilizado. A imagem do *background* (primeiro frame do vídeo) pode ser visualizada abaixo:



Figura 1: Background simples

Após alguns frames a câmera sofre algumas perturbações, quase imperceptíveis para alguém que está assistindo ao vídeo, porém perceptíveis e percebidas como movimento para o algoritmo utilizado. O resultado da máscara de movimento pode ser visualizado abaixo:

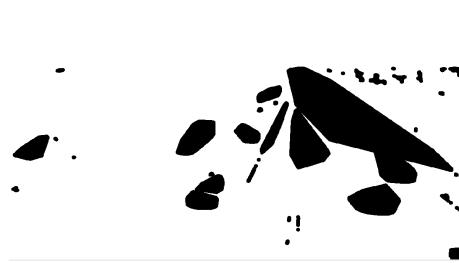


Figura 2: Máscara de movimento



Figura 3: Identificação de movimento

Claramente, analisando as imagens de máscara e recorte de movimento acima, é possível concluir que o algoritmo não é a melhor opção para detectar movimento em ambientes sujeitos a variações e ruídos, como uma câmera em uma autoestrada. Mesmo que a câmera fosse perfeitamente estática, se aplicado para esse caso o algoritmo falharia assim que a luminosidade da cena mudasse muito, pois o frame-base é estático.

### 3 Segunda etapa - background adaptativo

Visando corrigir os problemas apresentados pelo algoritmo de remoção de background baseado em um frame-base estático, foi proposta uma abordagem que utiliza um background que se adapta conforme a passagem dos frames da cena. Para realizar essa ponderação e atualizar o background estimado foi utilizada a função "accumulateWeighted" da biblioteca OpenCV. A função em questão toma três parâmetros: O frame atual, o background estimado anterior, e o peso que o frame atual terá para estimar o próximo background. O parâmetro do peso do frame atual dirá quanto este deve influenciar no background, e deverá ser ajustado para cada aplicação. Para o vídeo escolhido a seguinte abordagem foi seguida em relação à estimativa desse peso:

1. O valor inicia alto, em torno de 0.3. (Ou seja, 30% do frame atual deve ser adicionado ao background).
2. A cada iteração esse peso diminui em 0.005. (Conforme os frames passam, o peso individual diminui)
3. Estabelecer um limite para o peso. Quando o peso chega em 0.018, fica congelado nesse valor.

A abordagem escolhida assegura duas coisas importantes: A diferença entre o frame e o background "converge" rapidamente após alguns frames. Como o frame de background começa vazio, a intensidade da cor deve subir rapidamente, para não detectarmos diferenças falsas durante muitos frames. E além disso, garante que apenas objetos que fiquem muito tempo na cena sejam entendidos como background, eliminando ruídos e pequenas perturbações da câmera.



Figura 4: Background após 10 frames



Figura 5: Background apóis vários frames ( $> 20$ )

A medida em que o background é estimado, a diferença entre o frame atual e o último background é computada. Para calcular essa diferença a função "absdiff" foi utilizada. Como muitas vezes as cores dos carros se parecem bastante com a cor do asfalto, e existem sombras e reflexos presentes, algumas operações morfológicas foram feitas para suavizar o frame das diferenças. A matriz da diferença entre frame e background e sua equivalente suavizada podem ser visualizadas abaixo:

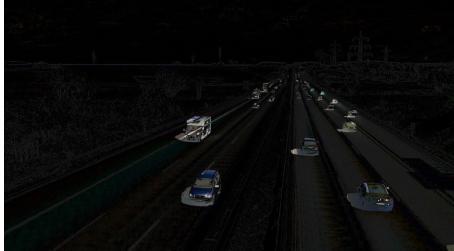


Figura 6: Diferença frame-background



Figura 7: Diferença suavizadas

Após computar a diferença entre os frames, foi necessário aplicar um threshold para binarizar a imagem e mais facilmente estimar onde estão os contornos dos carros. A máscara binarizada e o frame de recorte equivalente podem ser visualizados abaixo:



Figura 8: Máscara binarizada

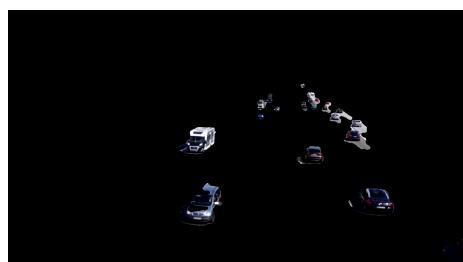


Figura 9: Recorte equivalente

Observando a máscara binarizada e o recorte equivalente obtidos é possível perceber que os contornos não estão englobando todo o objeto de interesse na maioria dos casos, possivelmente devido à similaridade entre cores das sombras e do asfalto. Para corrigir esse problema, para cada contorno obtido a partir dessa imagem, é calculado o *convex hull*, a partir da função "convexHull" da biblioteca opencv. Esse função garante que os contornos não serão côncavos, e portanto não apresentarão tanta granularidade.

A máscara binarizada e o recorte equivalente após a aplicação da função "convexHull" podem ser visualizadas abaixo:

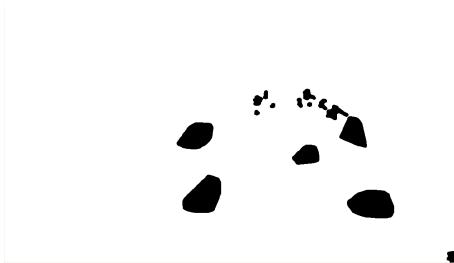


Figura 10: Máscara binarizada

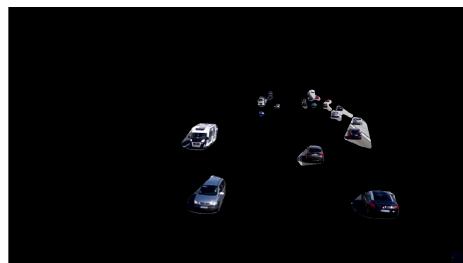


Figura 11: Recorte equivalente

Após obter a máscara binarizada e recorte equivalente convexas, são calculados para cada contorno, de área maior que um determinado threshold, um retângulo que o engloba. O ponto central desse retângulo será utilizado para de fato "contar" os carros que passam. Foi desenhado um retângulo ao longo da pista, que servirá como sensor virtual para a detecção de carros. Quando algum ponto de centro de massa de um determinado retângulo de contorno pertencer também à área do sensor virtual, o contador de carros é incrementado.

Um frame do contador funcionando pode ser visualizado abaixo:

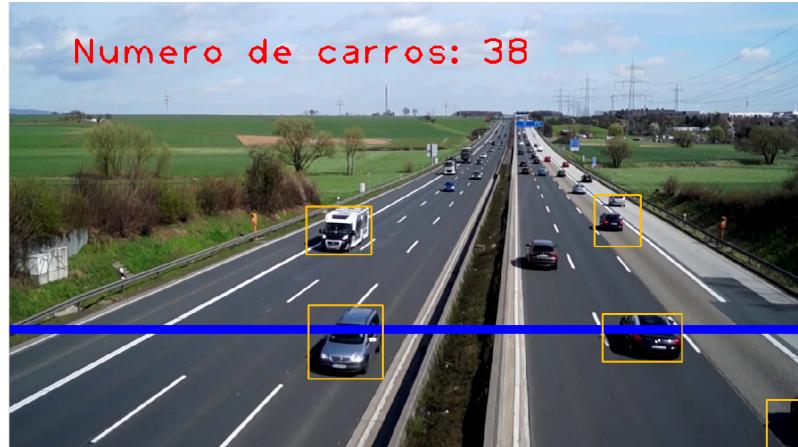


Figura 12: Contador finalizado

## 4 Conclusão

Foi possível observar, após finalizar este trabalho, que mesmo aplicando técnicas simples de visão computacional foi possível obter um resultado relativamente bom para essa aplicação de contagem de carros a partir de *background subtraction*. Um dos problemas encontrados foi o fato de um contorno quando muito próximo a outro, juntar-se e formar um contorno só, impossibilitando a precisão nas operações. Uma possível solução seria aplicar *labels* em cada contorno para que mesmo quando fundidos, fossem identificados como dois contornos distintos. Algumas técnicas como *optical flow*, ou redes neurais poderiam auxiliar na identificação dos carros, e evitar que um carro não seja contado ou contado mais de uma vez.

O vídeo do teste final pode ser visualizado em: <https://youtu.be/ZbsFnERAM48>