Algoritmo de Detecção de Sombras em Movimento

Princípios de Visão Computacional - UnB - 2º/2013
Professor : Flávio Vidal
Aluno : Juarez Aires Sampaio Filho - 11/0032829

I. Objetivos

Desenvolver um algoritmo de detecção de sombras em movimento utilizando as ferramentas do OpenCV.

II. Introdução

EM VISÃO COMPUTACIONAL muitas vezes estamos interessados em detectar objetos que se movem em uma cena. Na maioria nos casos o objeto está acompanhado por sua sombra e os algoritmo de extração de movimento irão detectá-la como outro objeto. De forma geral não estamos interessados em processar a sombra, somente o objeto em si. É preciso então detectar o que é sombra e o que é objeto para podermos assim saber o que deve ser ignorado.

O algoritmo de detecção de sombra que desenvolvemos é baseado em duas ideias chaves:

- a sombra que acompanha um objeto em movimento também está em movimento
- a sombra costuma ser mais escura que o objeto

Tendo em vista o primeiro princípio, nosso primeiro passo será procurar aquilo que está em movimento na cena. Isso será atingido ao fazermos uma subtração de fundo. Na subtração de fundo comparamos pixel a pixel um frame de um vídeo com seu anterior e marcamos aqueles pixel que sofreram uma mudança significativa. Dessa forma achamos na imagem regiões que estão se movendo. Aqui existem duas possibilidades: de um frame para o outro o pixel pode ter escurecido ou se tornado mais claro. O segundo princípio nos diz que as sombras em movimento escurecem mais a cena do que o objeto que a projeta. Essas duas ideias simples serão a base do nosso algoritmo.

Outro aspecto recorrente no algoritmo sugerido é a aplicação de filtros. A imperfeições da câmera, o formato de compactação de vídeo utilizado e as inumerosas fontes de erro na captura de imagens produzem ruídos que irão atrapalhar o processamento. Um objeto completamente cinza pode ter na sua imagem capturada, por exemplo, variações de tonalidade. Essas variações não são interessantes para nossa aplicação e as eliminamos com a aplicação de filtros que visam borrar a imagem, tornando-a mais homogênea. Apesar da perca de informação com essa abordagem, simplificamos bastante o problema e melhoramos significativamente os resultados.

Um último aspecto importante a ser introduzido é a dificuldade em se produzir um algoritmo genérico em visão computacional. Cada vídeo a ser analisado possui características que facilitam ou dificultam a aplicação de certos algoritmos. Fatores como iluminação e contraste entre os objetos em cena influenciam fortemente a aplicabilidade de muitos algoritmos. O algoritmos que discutimos pretende propor uma série de operações para detecção de sombras, os parâmetros dessas operações não podem, no entanto, serem previsto, sendo necessário calibração do processo para cada aplicação.

III. Materiais

O código elaborado foi feito em C++ utilizando a biblioteca OpenCV versão 2.4.6.

IV. Procedimentos

O algoritmo será descrito e sua funcionalidade exemplificado por uma aplicação em um vídeo de carros em certa rodovia. Todo o processamento é feito em tons de cinza e temos portanto apenas um canal para nos preocuparmos.

A. Subtração de Fundo

Seja C o frame atual e P o frame anterior. Seja $c_{i,j}$ e $p_{i,j}$ os pixels de C e P respectivamente na posição (i,j), defina a matriz X' como;

$$x'_{i,j} = c_{i,j} - p_{i,j}, \forall (i,j) \text{ válidos} \tag{1}$$

a operação implementa uma subtração de fundo direta. Em muitos vídeos, no entanto, mesmo em uma cena estática esse processo detecta algum movimento devido a pequenas variações de intensidades de um mesmo frame para outro, causadas principalmente pelo formato de compactação de vídeo. Para eliminar essas imperfeições, seja δ o limiar de variação, fazemos:

$$\begin{aligned} &\text{se } x'_{i,j} \in (-\delta, +\delta) \\ &\qquad x'_{i,j} = 255 \quad \forall (i,j) \text{ válidos} \end{aligned} \tag{2}$$

O valor 255 utilizado pinta de branco a área que não é de interesse.

B. Segmentando a Sombra

Veja que a matriz X' do processo anterior pode possuir valores negativos e positivos, pois um pixel pode ter escurecido ou se tornado mais claro de uma iteração para outra. Queremos tornar essas duas fases bem distintas. Para isso, seja ω o valor de separação e λ um valor de amplificação ou redução de diferenças.:

$$\begin{aligned} &\forall (i,j) \text{ válidos tal que } x'_{i,j} \neq 255 \\ &\text{se } (x'_{i,j} < 0)\text{: } x'_{i,j} = -1 * x'_{i,j} * \lambda \\ &\text{se } (x'_{i,j} > 0)\text{: } x'_{i,j} = x'_{i,j} * \lambda + \omega \end{aligned} \tag{3}$$

O valor λ se maior que a unidade aumenta a diferença entre os tons e se menor diminui. A ideia dessas duas constantes é separar as fases em cinzas escuros e claros bem distintos. A escolha delas envolve a calibração do método sobre sucessivos teste. Ao final dessa etapa teremos 3 fases distintas em X':

- onde não houve movimento significativo temos branco(255)
- onde os pixel escureceram temos cinza claro(acima de 100 a 255)
- onde os pixel esbranqueceram temos cinza escuro ou preto (de 0 a 100)

A figura 1a ilustra a saída dessa subtração de fundo com separação de fases.

Já sabemos agora que o está escurecendo e o que está clareando, buscamos a sombra no segundo grupo da lista a cima. É esperado que a sombra seja sensivelmente mais escura que o objeto, o que leva, na convenção a cima, que ela esteja mais clara que o objeto no frame X'. Definimos então os valores onde consideramos sombra. Essa escolha pode ser feita pela análise de histograma ou baseada em teste. A definição de uma 'look up table' junto com a função LUT na biblioteca imgproc do openCV segmenta então a matriz X' produzindo a matriz S que contém a nossa sombra. Um filtro de erosão e dilatação é aplicado sobre S para diminuir o ruído e tornar as zonas com sombras contínuas.Subtraímos então da matriz X' a sombra S para termos X: a matriz dos pixels que se movem e não são sombra. A imagem 1b ilustra a saída.

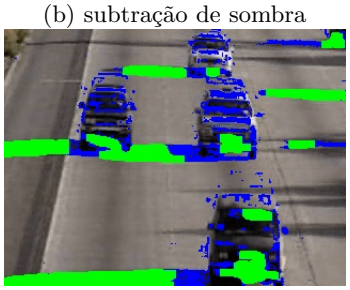
O que restou em X é a zona que foi descoberta por sombra de um frame para outro e o próprio objeto. Na nossa convenção, a zona descoberta por sombra é escura e o objeto é claro. Realizamos o mesmo processo para separar as duas classes e definimos o que é o objeto. Filtramos novamente e pintamos no frame colorido de entrada as zonas marcadas como sombra de verde e como objeto de azul.A figura 1c ilustra a saída.

V. Resultados

Os resultados são apresentados a seguir. Na figura 1a temos o resultado da subtração de background nos modos descritos. Note que nessa etapa vemos 4 grupos: a sombra, o carro e as áreas que foram descobertas pelo carro e pela sombra desde o frame anterior. Na figura 1b subtraímos a sombra e reduzimos os grupos a analizar. A figura 1c mostra o resultado final. Nela marcamos as sombras de verde e o objetos de interesse em azul. Notamos que a detecção de objetos não foi eficiente, mas que as sombras estão bem marcadas.

(a) subtração de fundo com separação de fases

(b) subtração de sombra



(c) resultado final

Figura 1: etapas do processamento

VI. Discussão e Conclusão

O método aqui discutido é de simples e rápida implementação e os resultados de detecção de sombra são evidentes como mostra a figura 1c. Como a figura também mostra, nosso algoritmo não foi capaz de selecionar os carros adequadamente, mas não era esse seu o foco. Em aplicações a deteccção de sombra poderia ser utilizada com outros artifícios de detecção próprios para objetos para melhorar o resultado. Algumas falhas do algoritmo são evidentes:

- no processo de filtragem com erosão sombras pequenas serão perdidas. Dessa forma nosso algoritmo funciona melhor com sombras grandes. No vídeo em análise, a deteccção melhora consideravelmente a medida que os carros se aproximam da tela e sua sombra aumenta.
- objetos escuros são detectados como sombra. Como nosso critério é baseado simplesmente em cor, partes escuras do objeto, como o para-brisa dos carros, serão detectadas como sombras. Parte desse ruído pode ser eliminadas com filtragem, mas não conseguiu-se remover por completo essas imperfeições.

Para generalizar o uso do algoritmo seria necessário automatizar a escolha dos parâmetros. No trabalho desenvolvido a escolha foi feita manualmente através de numerosos teste. Uma sugestão para trabalhos futuros

nessa linha de generalização do algoritmo é a análise de histograma para definir regiões de interesse.

O trabalho desenvolvido ilustra a facilidade de desenvolvimento de aplicações com o OpenCV. A biblioteca permite fácil manipulação de imagens e vídeos e oferece suporte para que, com uma rápida subtração de plano de fundo a aplicação de alguns filtros, desenvolva-se um aplicativo capaz de decidir áreas de um vídeo relacionadas com sombra.

Referências

[1] Forsyth, D.A. , *Computer Vision: a Modern Approach*, 1ªed.