Universidade de Brasília
Princípios de Visão Computacional
Projeto Final

Detecção de Objetos
e
Previsão de Colisões com Filtro de Kalman

---

4 de Dezembro de 2013

Professor: Flávio Vidal
Alunos:
Juarez A.S.F 11/0032829
Rodrigo Lima 11/xxxxxxx

I. Objetivos

Desenvolver um algoritmo para detecção de objetos e previsão de suas trajetórias e possíveis colisões utilizando para isso o conjunto de ferramentas disponíveis no OpenCV.

II. Introdução Teórica

Em aplicações de monitoramento de sistemas móveis estamos muitas vezes interessados na **previsão de colisão entre dois objetos**. Com essa previsão podemos tomar decisões para controlar a trajetória dos objetos de modo a evitar a colisão ou, ao menos, reduzir os danos causados. Com esse objetivo, **devemos ser capazes detectar os objetos**, sua trajetória e sermos capazes de **extrapolar a trajetória e prever as posições futuras** dos objetos sobre monitoramento.

As técnicas de visão computacional são comumente utilizadas na detecção de objetos. Vários procedimentos podem ser utilizadas para esse fim: como fizemos nos trabalhos anteriores, podemos detectar objetos pelo seu contorno(através de gradientes e a transformada Canny), sua forma(círculos e retas pela transformada Hough) e pelo movimento(subtração de fundo). Nesse trabalho, no entanto, usaremos uma técnica mais simples e que é eficiente se conhecermos de antemão a cor dos objetos: **detecção por cor**.

Se as cores dos objetos envolvidos forem conhecidas e aproximadamente constantes, e essas cores forem bem distintas das cores do fundo da imagem, então basta varremos a imagem procurando os pixeis pertencentes a uma certa faixa em torno das cores de interesse e temos detectados os objetos. Uma dificuldade que surge nesse algoritmo, é que cores parecidas podem ser geradas com combinações diferentes de canais RGB. Para contornar esse problema, **torna-se útil trabalhar com a imagem em componentes HSV**. HSV é sigla para:

- *hue*: indica a cor
- *saturation*: indica o quanto essa cor está misturado com branco
- *value*: indica o quanto a cor está misturada com preto

veja que a componente de cor(*hue*) é obtida diretamente e podemos dar maior importância a essa componente na hora da busca.

Utilizando técnicas de transformação morfológicas, como cálculo de momentos da imagem, ou então usando o kmeans para agrupar dados semelhantes, podemos obter o centro dos objetos sendo detectados e, monitorando esses centros, guardarmos a trajetória dos objetos a medida que o vídeo evolui. Possuindo dados sobre a posição em diferentes instantes de tempo podemos **determinar um modelo** envolvendo velocidade e aceleração para calcular uma curva de trajetória e assim prever posições futuras para o movimento dos objetos.

Esse processo de previsão de uma trajetória pode ser feito utilizando filtragem de Kalman. **Filtro de Kalman** é um processo que junta informações de um modelo para

o sistema sob análise, medições realizadas por diferentes sensores e métodos e as incertezas sob o modelo e as medições para **estimar o real valor da variável sendo medida** e ainda a **incerteza** resultante dessa estimação.

Podemos utilizar o filtro de kalman com um modelo de equação do movimento envolvendo posição, velocidade e aceleração junto com as medidas feitas pelo detector de objetos descrito anteriormente como entrada para o filtro para estimar a real posição do objeto. Mas para quê utilizarmos o filtro se a detecção por cor já é razoável? Basta implementar a detecção para ver o problema: devido a ruídos inerentes à captura de imagens, é muito provável que, mesmo que o movimento do objeto seja suava, **a detecção perceba uma vibração do centro do objeto**. Ao utilizarmos o filtro essas vibrações são eliminadas pelo modelo e pelo **conhecimento prévio de um erro gaussiano** nas medidas. Ou seja, **o filtro não acredita fielmente nas informações dos sensores, ao invés disso, utiliza o modelo para corrigir a detecção**. O quanto o filtro acredita nas medidas e no modelo é definido pelo erro associado a cada um. Esse erro é, portanto, parte da definição do filtro.

Além de estabilizar a detecção, o filtro de Kalman pode ser utilizado para *prever o futuro*. Isso é feito ao entrarmos no filtro os dados previstos sucessivamente. Isto é, para um conjunto de medidas realizada o filtro retorna a posição estimada para o objeto, se dissermos para o filtro que essa informação estimada é uma outra medida, ele irá processá-la normalmente e retornará a posição seguinte. Repetindo o processo, temos uma **previsão de posições futuras baseada nos dados medidos e em previsões anteriores do filtro**.

Ao fazermos isso teremos uma previsão razoável do futuro dos objetos. Falta então procurar por colisões. Para isso **supomos nossos objetos como sendo circulares**. O centro da partícula é o centro do objeto detectado e para obter o raio podemos usar transformações morfológicas e obter o menor círculo que engloba os pixeis relacionados a um objeto, ou utilizar o raio do objeto detectado pelo kmeans. Para melhorar a previsão, podemos somar ao raio a informação de incerteza na posição informada pelo filtro de Kalman. Tendo as partículas circulares que englobam os objetos detectados, **basta medir a distância entre os dois centros e comparar com a soma dos raios dos objetos**. Se a distância for menor que a soma dos raios, então os objetos estarão em colisão. Para prever o tempo até a colisão dividimos a distância entre a posição atual a posição prevista de colisão pela velocidade atual do objeto.

III. Materiais

O código elaborado foi feito em C++ as bibliotecas:

- core
- imgproc
- highgui
- background_segm
- tracking

do **OpenCV** versão 2.4.6.

IV. Descrição Experimental

V. Resultados

VI. Discussão

VII. Conclusão

Referências

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

[2] Vidal, F.B. e Alcalde, V.H.C. *Motion Segmentation in Sequential Images Based on the Differential Optical Flow*

[3] Documentação do OpenCV Disponível em: http://docs.opencv.org Acesso em 21 de Novembro de 2013.

[4] Dan Casas, *how to plot velocity vectors as arrows using sigle static image*. Disponível em: http://stackoverflow.com/questions/10161351/opencv-how-to-plot-velocit y-vectors-as-arrows-in-using-single-static-image Acesso em 21 de Novembro de 2013.

[5] Utkarsh Sinha, *K-Means clustering in OpenCV*. Disponível em: http://www.aishack.in/2010/08/k-means-clustering-in-opencv/ Acesso em 21 de Novembro de 2013.

[6] Mateusz Stankiewicz, *Background detection with OpenCV*. Disponível em : http://mateuszstankiewicz.eu/?p=189 Acesso em 21 de Novembro de 2013.