ISEL
INSTITUTO SUPERIOR
DE ENGENHARIA DE LISBOA

Licenciatura Engenharia Informática e Multimédia - 3º Ano

Processamento de Imagem e Visão

Trabalho prático 2

Elaborado por:
Rui Rita nº 33753
Pedro Costa nº 43254

Docente:
Eng.º André Lourenço

Ano letivo 2019/2020

Índice

1. Objetivos

Este trabalho prático tem como objetivo o processamento de um video no qual é representado um ambiente de sala de aula e respetivos alunos, em que, através de um algoritmo desenvolvido em *python*, se seja capaz de detetar, seguir, e contar o número de alunos presentes.

Recorrendo a bibliotecas como *OpenCV* e *Numpy*, deverá ser tratado o video recorrendo às técnicas estudadas na disciplina durante o semestre, de forma pré-processar, extrair, e classificar componentes existentes num video.

2. Estimação de imagem de fundo

A estimação da imagem de fundo é uma técnica utilizada para se conseguir remover componentes/objectos/pessoas de um determinado conjunto de imagens que representem a mesma cena, em que se recorre ao cálculo da mediana para se estimar quais as componentes que se matêm estáticas entre as várias imagens.

Para este cálculo em video é necessário primeiro definimos uma janela de *frames* (ou número de imagens) que pretendemos considerar para esta estimação (neste caso usamos 100 *frames* de video). Uma vez obtidas as imagens e ordenadas temporalmente é considerado (para cada *pixel*), qual o valor que se encontra na posição intermédia dessas 100 *frames* ordenadas (na posição 50), ou seja, o valor que ocorre mais vezes nessa janela de images.

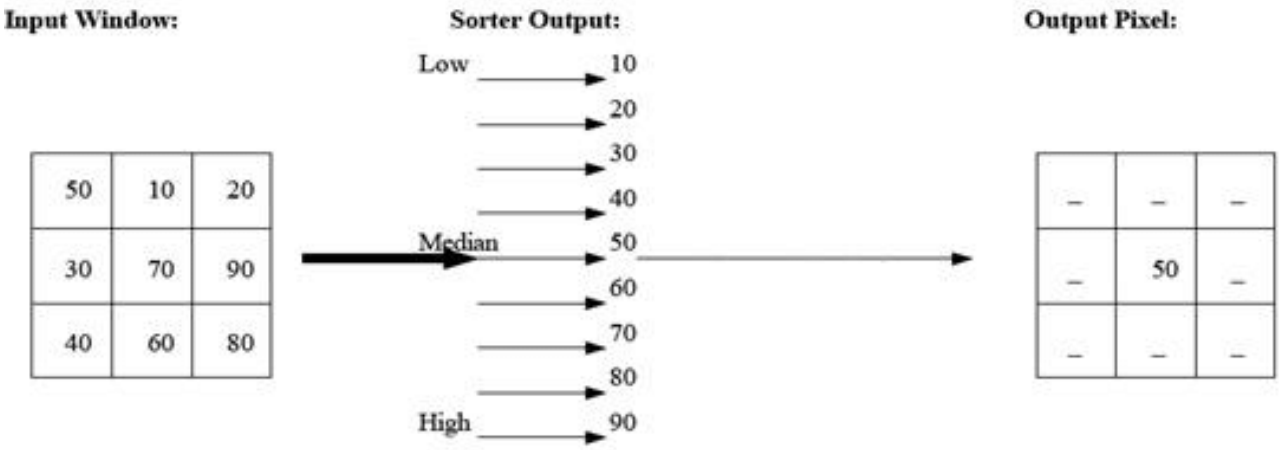

Fig.1 - Cálculo de mediana

Esta fórmula é então aplicada a todo o video numa fase de pré-processamento de modo a se estimar qual a mediana com menor variação ao longo do tempo, e, consequentemente, qual a imagem mais constante (ou imagem de fundo)

Embora com alguns resultados interessantes, apresenta vários problemas para o posterior tratamento da imagem, nomeadamente nas diferenças para cada *frame*, o ruído que se mantem na imagem (mudanças de luminosidade, sombras, etc), ou anulação (subtração) de *pixels* da *frame* úteis para a deteção de movimento.

3. Deteção de *pixels* ativos

O método mais simples para estimarmos quais os *pixels* ativos em cada *frame*, seria através da subtração do fundo previamente cálculado, em que esta imagem de fundo é então aplicada durante o video *frame-a-frame*, a qual é subtraida por forma a identificarmos quais as componenete dessa *frame* que não fazem parte do fundo.

Uma vez que a subtração é feita com a imagem binarizada, um único *threshold* para trás alguns problemas quando é aplicada a cada *frame*, como por exemplo consideração de mudanças em sombras/luminosidade como zonas ativas, ou cortes de movimento devido a subtração da imagem binária de fundo. Com a aplicação de *blur* à imgem antes da binarização conseguimos eliminar algum do ruído, no entanto não o suficiente para que o *threshold* consiga caracterizar corretamente o movimento (ou a não existência de movement = sombra).

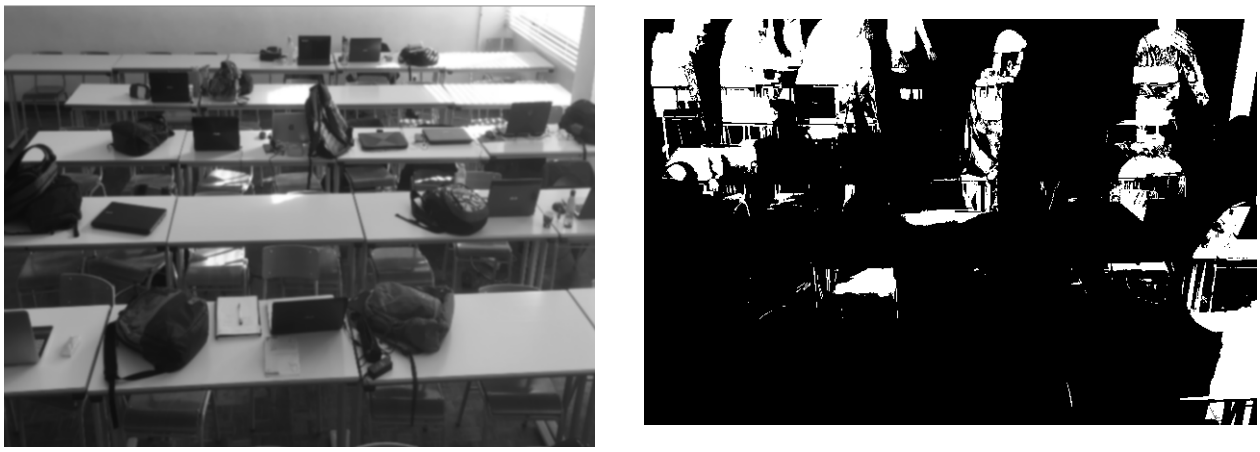

Fig.2 - Subtração do fundo *frame-a-frame*

Estes problemas ficavam mais patentes no momento de inter-ligar zonas de movimento que são a mesma pessoa, no entanto nesta máscara de subtração se encontram divididos, e serão dessa forma, classificados como 2 zonas independentes.

Devido a estes problemas identificados, optámos pela utilização do método *createBackgroundSubtractorMOG2*, da biblioteca *OpenCV,* para a estimação do movimento dentro de determinado conjunto de imagens. O que este método faz é verificar quais as variações de *pixels* nas últimas N imagens (N sendo definido na criação

do *subtractor*) em que é retornada a máscara de movimento em cada *frame* que é aplicado.

Através da criação do *subractor* (cv2.createBackgroundSubtractorMOG2(history=30, varThreshold=16, detectShadows=False)) conseguimos obter a diferença de movimentos mas comparando cada *frame* com as últimas 30.

Fig.3 - Deteção de movimento utilizando o *subtractor*

Como se pode ver na imagem, ainda existe algum ruído em relação às sombras e diferenças de iluminação, no entanto, após aplicação dos operadores morfológicos, conseguimos tratar estes dados corretamente, o que não acontece de forma tão correta na subtração direta da imagem de fundo.

4. Operadores morfológicos

A aplicação dos operadores morfológicos toma um papel muito importante no processamento de imagem, uma vez que permite a exclusão/inclusão, união/separação, aumento/diminuição, etc., dos componente presente numa imagem, o que torna o seu processamento mais simples.

Tendo em conta a máscara que o *subtractor* retorna após a deteção de movimento (Fig. 3), aplicamos 2 operadores morfológicos distintos. Primeiro é aplicada uma erosão, de modo a serem eliminidas as zonas de ruído mais pequenas, ou quaisquer deteções de movimento minimas que ocorram em torno das pessoas. Para esta transformção foi utilizado um *kernel* de 5x5.

Fig.4 - Máscara de movimento antes e após aplicação da erosão

É visível a diminuição do ruído após a aplicação do operador de erosão, no entanto, a máscara transformada ainda se encontra com as zonas de movimento muito segmentadas, pelo que a sua classificação e identificação será comprometida desta forma. Uma maneira de contornar este problema é através do operador morfológico de dilatação, que, ao contrário da erosão, expande as zonas em torno de zonas identificadas como movimento, dando assim uma sensação de união entre os vários segmentos e também de algum restauro do tamanho original que foi afetado pela erosão.

Para esta transformação foi aplicado um *kernel* de 55x55. A razão para uma máscara do operador tão grande prende-se com o facto de tentarmos obter a conexão entre os vários segmentos da mesma pessoa identificados no movimento, e de restaurar algum do tamanho inicial.

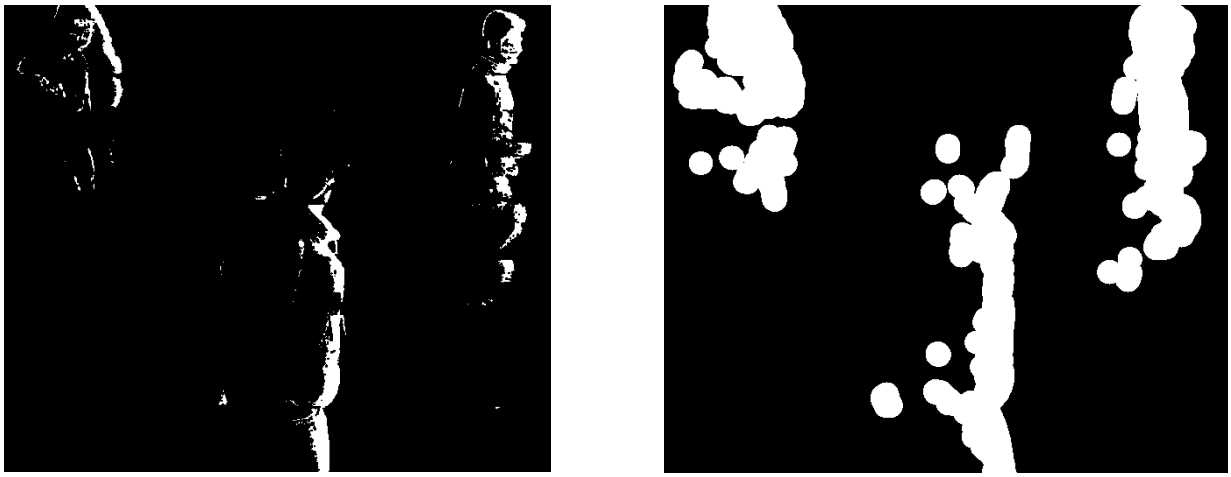

Fig.5 - Aplicação do operador morfológico de dilatação após erosão

5. Deteção de regiões ativas

Após o tratamento da imagem de detação de movimento e aplicação dos operador morfológicos, o passo seguinte passa por detectar as zonas de segmentos conexos, e para tar recorremos ao método *findContours* do *OpenCV*, que retorna as posições e dimensões de cada segmento da imagem. No entanto, e como é visível após aplicada a dilatação, ainda existem zonas que queremos excluír da clasificação como zona ativa. Para tal, através da área de cada zona, excluimos de considerar as zonas de tamanho muito baixo, eliminando dessa forma o restante ruído presente na máscara transformada.

Fig.6 - Criação de *boundingboxes* nas zonas de deteção de movimento

O útlimo tratamento feito ao video é tendo em conta a classficiação da imagem, e para tal é necessário retirar ponto de interesse em cada um dos segmentos em movimento. O modo como foi desenvolvido baseia-se na criação de uma máscara individual para cada um dos segmentos de movimento.

Fig.7 - Aplicação de máscaras a zonas de movimento

Dentro de cada máscara individualmente, retiramos os pontos que melhor caracterizam essa zona, através do método *goodFeaturesToTrack* do *OpenCV*.

Após serem retirados esses pontos de interesse, os mesmos são desenhamos em cada frame, de modo a serem seguidos entre as várias imagens

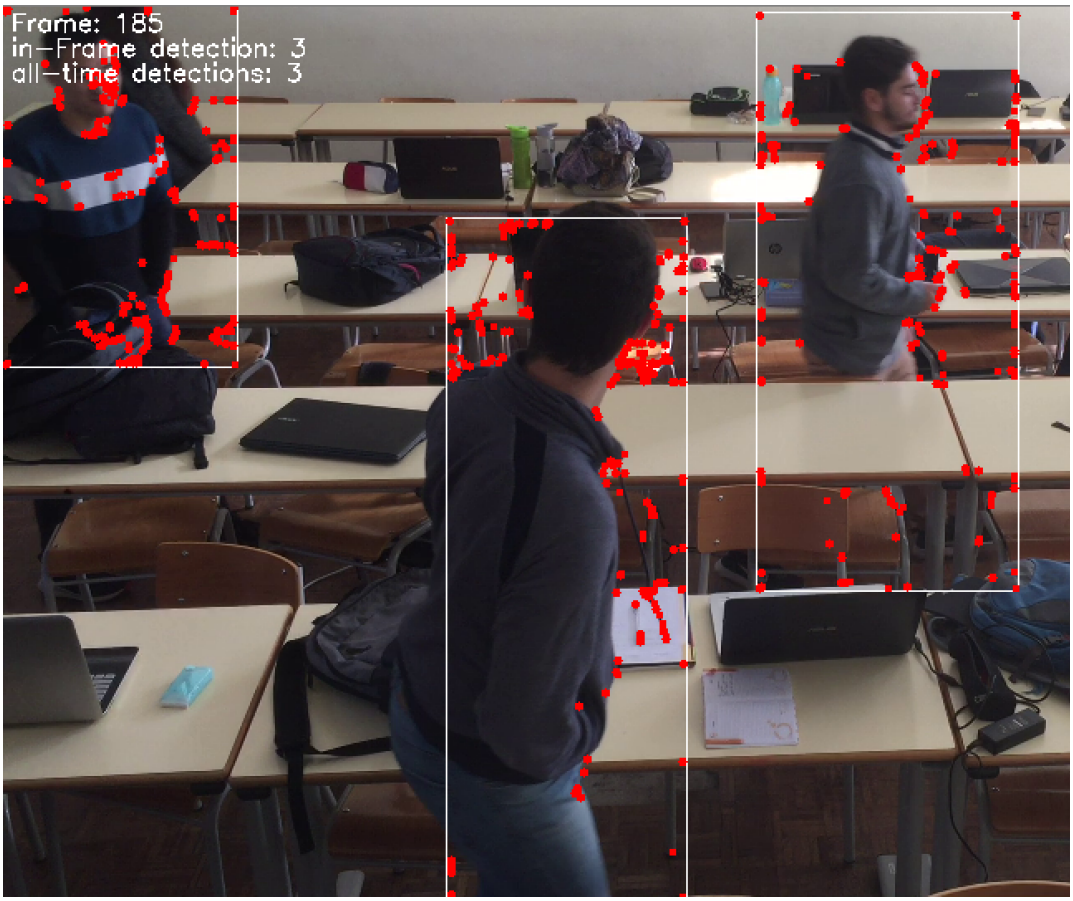

Fig.8 - Pontos com *features* de cada zona de movimento da *frame*

6. Conclusões

Finalizado este trabalho prático, a principal conclusão retirada é que um bom pré-processamento de imagem, seja através da subtração de fundo, deteção de movimento, aplicação de operadores morfológicos, ou todos em conjunto, faz toda a diferença no movimento de caracterizar conteúdos presentes na mesma.

A maior dificuldade prendeu-se com o *tracking* a ser efetuado a cada aluno, uma vez que os pontos que são retornados para classificação em cada ROI (*region of interest)* não são os ideais para se conseguir calcular corretamente o *optical flow* dos mesmos, uma vez que são marcados pontos em objetos do fundo e não da própria figura em movimento.

7. Bibliografia

1. Fórmula mediana: https://www.researchgate.net/figure/A-graphical-depiction-of-the-median-filter-operation_fig1_280925268