UFPB-CI-DSC
Disciplina: Visão Computacional
Prof. Augusto de Holanda B. M. Tavares
25 de Junho de 2024

15ª Atividade

Fluxo óptico

O código abaixo detecta pontos-chave através de Shi-Tomasi e realiza o seu rastreio temporal utilizando o fluxo óptico pelo método de Lucas-Kanade:

```
import numpy as np
import cv2 as cv

cap = cv.VideoCapture('corgi_race.mp4')

# params for ShiTomasi corner detection
feature_params = dict( maxCorners = 100,
                       qualityLevel = 0.3,
                       minDistance = 7,
                       blockSize = 7 )

# Parameters for lucas kanade optical flow
lk_params = dict( winSize = (15, 15),
                  maxLevel = 2,
                  criteria = (cv.TERM_CRITERIA_EPS | cv.TERM_CRITERIA_COUNT, 10,
                      0.03))

# Create some random colors
color = np.random.randint(0, 255, (100, 3))

# Take first frame and find corners in it
ret, old_frame = cap.read()
old_gray = cv.cvtColor(old_frame, cv.COLOR_BGR2GRAY)
p0 = cv.goodFeaturesToTrack(old_gray, mask = None, **feature_params)

# Create a mask image for drawing purposes
mask = np.zeros_like(old_frame)

while(1):
    ret, frame = cap.read()
    if not ret:
        print('No frames grabbed!')
        break
```

```
    frame_gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)

    # calculate optical flow
    p1, st, err = cv.calcOpticalFlowPyrLK(old_gray, frame_gray, p0, None,
        **lk_params)

    # Select good points
    if p1 is not None:
        good_new = p1[st==1]
        good_old = p0[st==1]

    # draw the tracks
    for i, (new, old) in enumerate(zip(good_new, good_old)):
        a, b = new.ravel()
        c, d = old.ravel()
        mask = cv.line(mask, (int(a), int(b)), (int(c), int(d)),
            color[i].tolist(), 2)
        frame = cv.circle(frame, (int(a), int(b)), 5, color[i].tolist(), -1)
    img = cv.add(frame, mask)

    cv.imshow('frame', img)
    k = cv.waitKey(30) & 0xff
    if k == 27:
        break

    # Now update the previous frame and previous points
    old_gray = frame_gray.copy()
    p0 = good_new.reshape(-1, 1, 2)

cv.destroyAllWindows()
```

Utilizando o código acima como base, realize as seguintes tarefas:

1. Modifique o código do para utilizar pontos-chave obtidos através do algoritmo ORB. Limite o número de pontos-chave detectados a 100.

2. Modifique o código para que sejam destacados com círculos coloridos apenas os pontos cujo erro de estimativa fique abaixo de um limiar de sua escolha.

Continuando, o código abaixo utiliza o algoritmo de Gunnar Farnerback para calcular o fluxo óptico denso em uma imagem, que é o caso em que o fluxo é calculado para todos os pontos da imagem:

```
cap = cv.VideoCapture(cv.samples.findFile('heavy_object.mp4'))

ret, frame1 = cap.read()

pyr_scale = 0.5
```

```
levels = 3
winsize = 15
iterations = 3
poly_n = 5
poly_sigma = 1.2
flags = 0

prvs_frame = cv.cvtColor(frame1, cv.COLOR_BGR2GRAY)
hsv = np.zeros_like(frame1)
hsv[..., 1] = 255

# Taking a matrix of size 5 as the kernel
kernel = np.ones((9, 9), np.uint8)

while(1):
    ret, frame2 = cap.read()
    if not ret:
        print('No frames grabbed!')
        break

    next = cv.cvtColor(frame2, cv.COLOR_BGR2GRAY)
    flow = cv.calcOpticalFlowFarneback(prvs, next_frame, None, pyr_scale, levels,
        winsize, iterations, poly_n, poly_sigma, 0)
    mag, ang = cv.cartToPolar(flow[..., 0], flow[..., 1])
    hsv[..., 0] = ang*180/np.pi/2
    hsv[..., 2] = cv.normalize(mag, None, 0, 255, cv.NORM_MINMAX)
    bgr = cv.cvtColor(hsv, cv.COLOR_HSV2BGR)
    cv.imshow('frame2', bgr)

    if cv.waitKey(1) == ord('q'):
        break

    prvs = next

cv.destroyAllWindows()
```

Utilizando o código acima como base, realize as seguintes tarefas:

3. Produza um *frame* a partir do *frame* original em que apenas os *pixels* em movimento são exibidos em cores, sendo os demais definidos em preto. Trabalhe o seu código para minimizar a incidência do ruído.

4. Repita o item anterior para o caso do vídeo da corrida de cachorros. O desempenho observado é melhor ou pior? Escreva um pequeno texto justificando a sua resposta.