mgr inż. Grzegorz Gołaszewski
pok. 216, D-10
golaszewski@agh.edu.pl

dr hab. inż. Szymon Łukasik
pok. 213, D-10
slukasik@agh.edu.pl

Analiza sekwencji wideo

Ćwiczenie laboratoryjne nr 6
Analiza i przetwarzanie obrazów

Wprowadzenie

Celem ćwiczenia jest zapoznanie uczestników kursu z przetwarzaniem sygnałów wideo w OpenCV – zarówno w aspekcie rejestracji/wyświetlenia sekwencji wideo jak i pracy z poszczególnymi jej klatkami.

Praca z sekwencjami wideo – przydatne polecenia

OpenCV reprezentuje sekwencję wideo jako obiekt `cv2.VideoCapture` . Powołujemy go do życia bądź to podając plik z którego sekwencja ma być wczytana:

```
obiekt_video = cv2.VideoCapture('test.avi')
```

bądź wskazując urządzenie typu VideoCapture:

```
obiekt_video = cv2.VideoCapture(0)
```

Nie ma w samym OpenCV łatwej metody identyfikacji urządzeń – zasadniczo są one indeksowane kolejnymi liczbami całkowitymi (od -1 lub od 0). Niektórzy w tym celu posiłkują się skryptami korzystającymi chociażby z systemowych API obsługujących multimedia: https://github.com/pvys/CV-camera-finder

Po zainicjowaniu obiektu przechwytywania pojedynczą ramkę zapisujemy tak:

```
ret, frame = obiekt_video.read()
```

Poza ramką otrzymujemy zmienną logiczną – która informuje czy przechwytywanie ramki zakończyło się sukcesem.

Dla urządzeń możemy odczytać parametry poleceniem `cap.get(parametr)` np (wycinek z dokumentacji).

- CAP_PROP_POS_MSEC Current position of the video file in milliseconds or video capture timestamp.
- CAP_PROP_POS_FRAMES 0-based index of the frame to be decoded/captured next.
- CAP_PROP_POS_AVI_RATIO Relative position of the video file: 0 - start of the film, 1 - end of the film.
- CAP_PROP_FRAME_WIDTH Width of the frames in the video stream.

mgr inż. Grzegorz Gołaszewski
pok. 216, D-10
golaszewski@agh.edu.pl

dr hab. inż. Szymon Łukasik
pok. 213, D-10
slukasik@agh.edu.pl

- CAP_PROP_FRAME_HEIGHT Height of the frames in the video stream.
- CAP_PROP_FPS Frame rate.
- CAP_PROP_FOURCC 4-character code of codec.
- CAP_PROP_FRAME_COUNT Number of frames in the video file.
- CAP_PROP_FORMAT Format of the Mat objects returned by `retrieve()`.
- CAP_PROP_MODE Backend-specific value indicating the current capture mode.
- CAP_PROP_BRIGHTNESS Brightness of the image (only for cameras).
- CAP_PROP_CONTRAST Contrast of the image (only for cameras).
- CAP_PROP_SATURATION Saturation of the image (only for cameras).
- CAP_PROP_HUE Hue of the image (only for cameras).
- CAP_PROP_GAIN Gain of the image (only for cameras).
- CAP_PROP_EXPOSURE Exposure (only for cameras).
- CAP_PROP_CONVERT_RGB Boolean flags indicating whether images should be converted to RGB.
- CAP_PROP_WHITE_BALANCE_U The U value of the whitebalance setting (note: only supported by DC1394 v 2.x backend currently)
- CAP_PROP_WHITE_BALANCE_V The V value of the whitebalance setting (note: only supported by DC1394 v 2.x backend currently)
- CAP_PROP_ISO_SPEED The ISO speed of the camera (note: only supported by DC1394 v 2.x backend currently)
- CAP_PROP_BUFFERSIZE Amount of frames stored in internal buffer memory (note: only supported by DC1394 v 2.x backend currently)

Jeśli któraś opcja nie jest wspierana przez urządzenie otrzymujemy 0 w wyniku pobierania jej wartości.

Dla niektórych z nich (np. rozdzielczości) jest możliwe analogicznie ustawienie danej opcji:

```
ret = obiekt_video = cap.set(3,640)
ret = obiekt_video.set(4,480)
```

Odtwarzanie sekwencji wideo: `cv2.imshow('frame', frame)` pokazuje jedną ramkę – iterujemy po niej.

Zapis – najpierw należy stworzyć obiekt codeca:

```
fourcc = cv.VideoWriter_fourcc(*'XVID')
```

potem określić nazwę, liczbę klatek na sekundę i rozdzielczość

```
wyjscie = cv.VideoWriter('plik.avi', fourcc, 20.0, (320, 200))
```

mgr inż. Grzegorz Gołaszewski
pok. 216, D-10
golaszewski@agh.edu.pl

dr hab. inż. Szymon Łukasik
pok. 213, D-10
slukasik@agh.edu.pl

Zapisujemy ramka po ramce (również iteracyjnie)
`wyjscie.write(frame)`

Strumień nie obsługuje ramek w odcieniach szarości, więc trzeba przeprowadzić odpowiednią konwersję.

Należy pamiętać o zwolnieniu strumienia
`wyjscie.release()`

W praktycznych problemach w których przetwarzanie ramek może zajmować sporo czasu warto popracować np. nad wielowątkowością opracowanego rozwiązania. Przykłady:
https://nrsyed.com/2018/07/05/multithreading-with-opencv-python-to-improve-video-processing-performance/

https://github.com/gilbertfrancois/video-capture-async

Realizacja ćwiczenia

Ćwiczenie stanowi kontynuację zadania z poprzednich laboratoriów. Proszę wczytać przykładowy krótki film przedstawiający obraz z kamery samochodu poruszającego się po drodze i zrealizować na nim metodę wykrywania linii poprzez transformację Hough. Wynik należy zwizualizować na sekwencji wideo.

Wczytać drugi film, przedstawiający nagranie samochodów na autostradzie. Przenieść do odcieni szarości. Stworzyć sekwencję różnic pomiędzy kolejnymi klatkami. Utworzone różnice wykorzystać jako maski na oryginalnych klatkach. (Jeżeli klatki są za duże, to można je przeskalować stosując `cv2.resize(...)`)