Capturing videos from the webcam ¶

We can use the `cv2.VideoCapture()` function to read video files from a camera or a video file.

To capture a video, you need to create a VideoCapture object. Its argument can be either the device index or the name of a video file. Device index is just the number to specify which camera. Normally one camera will be connected (as in my case). So I simply pass 0 (or -1). You can select the second camera by passing 1 and so on. After that, you can capture frame-by-frame. But at the end, don't forget to release the capture.

The below script opens the primary camera and the converts it into greyscale and shows the video.

In [1]:

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Display the resulting frame
    cv2.imshow('frame',gray)
    if cv2.waitKey(2000) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

It is recommended to use 0xFF to mask off the unwanted bits in 64bit machines.
The wait time inside the waitkey function defines the frame rate of the video capture
at the end we must always release the cap(ie the camera) else other apps wont be able to access it.

`cap.read()` return 2 value 1st is a bool depending on if the frame read was successful or not. The second value of the frame itself(numpy array containing the image data)

In the above example cap is a video capture object. Once a video capture object is cerated we can call method on it check its state.

Reading videos from file

In [2]:

```
import numpy as np
import cv2

cap = cv2.VideoCapture('videos\\test.avi')

while(cap.isOpened()):
    ret, frame = cap.read()

#     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

```
---------------------------------------------------------------------------
-
error                                     Traceback (most recent call las
t)
<ipython-input-2-3de7ad7a045f> in <module>
      9 #     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
     10
---> 11     cv2.imshow('frame',frame)
     12     if cv2.waitKey(1) & 0xFF == ord('q'):
     13         break

error: OpenCV(4.2.0) C:\projects\opencv-python\opencv\modules\highgui\src
\window.cpp:376: error: (-215:Assertion failed) size.width>0 && size.heigh
t>0 in function 'cv::imshow'
```

The above video plays in higher speed as the frame rate we specified is approx 1000fps which is not the same as the actual videos frame rate.
We can change this by changing the waitkey value

Also the window crashes as we are not detecting the end of video we can do this with the first value given by cap.read()

In [5]:

```
import numpy as np
import cv2

cap = cv2.VideoCapture('videos\\test.avi')

while(cap.isOpened()):
    ret, frame = cap.read()

    if ret == True:
        cv2.imshow('frame',frame)
        if cv2.waitKey(100) & 0xFF == ord('q'):
            break
    else:
        break

cap.release()
cv2.destroyAllWindows()
```

We give a waitkey of 100 which means we wait for 100ms after each frame which accounts for 10fps which is the actual fps of the video.

Writing video files

A video codec isn't the same as a video format or container. A container is a bundle of files. Inside it, you can find data that has been compressed by using a particular codec. For example, an AVI file can contain video compressed by XviD, or DivX, or MPEG-2 codecs. Usually, a container comprises a video and audio codecs, plus it can also contain other files like subtitles and chapters. Popular video formats or containers are AVI, MP4, WMV, MKV, MOV, FLV, etc.

In [9]:

```
import numpy as np
import cv2

cap = cv2.VideoCapture("videos/test.avi")

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')

# creating the video capture object
out = cv2.VideoWriter('videos\\test_gray.avi',fourcc, 10.0, (640,480))

while(cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)

        cv2.imshow('frame',frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

# Release everything if job is finished
cap.release()
out.release()
cv2.destroyAllWindows()
```

The video size should be near the actual video size value. Else the video output wont play.
so we can us the cap.get() methods to read the attributes of a video file.

The below Link shows all the readable attributes of a video file. We can just pass in the index numbers of these paramters
https://docs.opencv.org/2.4/modules/highgui/doc/reading_and_writing_images_and_video.html#videocapture-get
(https://docs.opencv.org/2.4/modules/highgui/doc/reading_and_writing_images_and_video.html#videocapture-get)

In [1]:

```
import numpy as np
import cv2

cap = cv2.VideoCapture("videos/test.avi")

w = cap.get(3)
h = cap.get(4)

print(w,'x',h)

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('videos\\test_gray.avi',fourcc, 10.0, (int(w),int(h)))

while(cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)

        cv2.imshow('frame',frame)
        if cv2.waitKey(100) & 0xFF == ord('q'):
            break
    else:
        break

# Release everything if job is finished
cap.release()
out.release()
cv2.destroyAllWindows()
```

```
768.0 x 576.0
```