INTERNET ARCHIVE Wayback Machine
http://www.femb.com.mx/people-counter/people-counter-2-opening-a-video-stream/ Go DEC FEB MAR
58 captures 03
23 Jun 2017 - 10 Sep 2019 2017 2018 2019
About this capture

People Counter 2 – Opening a video stream

In the second part of the tutorial we'll cover how to open a video stream in OpenCV.

A video stream may be a video recording in a file or video from a webcam. I'll show you how to work with both.

First, let's work with a video file. Download this sample video. We'll be using it throughout the tutorials.

Create a new Python script in the same location as the video and add the following code:

```
import numpy as np
import cv2

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file

while(cap.isOpened()):
    ret, frame = cap.read() #read a frame
    try:
        cv2.imshow('Frame',frame)
    except:
        #if there are no more frames to show...
        print('EOF')
        break

    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #release video file
cv2.destroyAllWindows() #close all openCV windows
```

As you can see, we begin by importing **numpy** and **openCV** to our script.

Then we open the video file with the **VideoCapture** object, giving it the location of the video file as a parameter.

The we read frames from the video and show them, one by one, until we reach the end of it. At that point, we exit the while loop and close the video file and the video window.

Using a webcam is very similar:

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

while(cap.isOpened()):
    ret, frame = cap.read()
    try:
        cv2.imshow('Frame',frame)
    except:
        print('EOF')
        break

    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

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

The only difference is how we create the **VideoCapture** object. This time we pass it **0** as parameter. This indicates that we want to use the webcam with ID 0. Is you hav emultiple webcams on your computer, such as a USB webcam and another one embedded onyour screen, you'll need to pass **0** or **1**, depending on which one you want to use.

A VideoCapture object has several properties that you can access and sometimes change:

- **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.
- **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** Currently not supported
- **CAP_PROP_RECTIFICATION** Rectification flag for stereo cameras (note: only supported by DC1394 v 2.x backend currently)

You can use the **get** method to show them and **set** to give them a new value.

Here´s an example on showing all of the properties' values and changing the width and height of the video stream from the webcam:

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

#show all video properties
for i in range(19):
    print i, cap.get(i)

cap.set(3,160) #set width
cap.set(4,120) #set height

while(cap.isOpened()):
    ret, frame = cap.read()
    try:
        cv2.imshow('Frame',frame)
    except:
        print('EOF')
        break

    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

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

This is all for this tutorial. Next time we'll work on the video stream, drawing a GUI on it.

Please leave any comments or questions you have. ☺

PS. If you have trouble opening the AVI file in openCV, try updating your video codecs. Downloading DIVX codec helped me when I had this issue.

Fede / June 11, 2016 / People counter / people counter

5 thoughts on "People Counter 2 – Opening a video stream"

talarari
June 18, 2016 at 9:01 am

Hey, Love the tutorial, any estimate on when you will release the rest?
we have a hackathon at work in a couple of weeks and i would like to do something similar, would be great if you could even just share the code on github or something and i'll try to figure it out (i know writing a blog post is time consuming ☺).

thanks again, awesome work!
Log in to Reply

Fede
September 6, 2016 at 6:08 pm

Next chapter is up, I don´t know if it´s still useful to you. Sorry for the delay, got caught up with work.
Log in to Reply

antonjames
January 12, 2017 at 10:21 pm

Hi there,

Unfortunately I can't open my avi video, cap.isOpened() simply returns false, what exactly do you mean by updating my video codecs? How exactly do you do that?
Log in to Reply

aali

INTERNET ARCHIVE Wayback Machine
http://www.femb.com.mx/people-counter/people-counter-2-opening-a-video-stream/ Go
58 captures
23 Jun 2017 - 10 Sep 2019
DEC FEB MAR
2017 03 2018 2019
About this capture

Log in to Reply

Fede

April 17, 2017 at 9:28 pm

Yes, both are possible. Not sure if real time, though.
The code works with a series of images (frames), that's what a video is, many images, one after another. If you can get and IP camera's frames one by one, then the code will work. Same goes for RPI camera

Log in to Reply