People Counter 3 – Drawing in the video window.

This part of the tutorial is going to be very simple. We'll just be drawing a simple interface in the video window, to display some information.

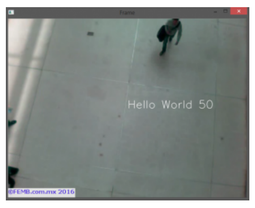
Let's start by using this code from the last chapter of this tutorial with some minor modifications:

```
import numpy as np
import cv2

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

w = cap.get(3) #get width
h = cap.get(4) #get height

mx = int(w/2)
my = int(h/2)

count = 0

while(cap.isOpened()):
    ret, frame = cap.read() #read a frame
    try:
        count = count + 1
        text = "Hello World " + str(count)
        cv2.putText(frame, text ,(mx,my),cv2.FONT_HERSHEY_SIMPLEX
                    ,1,(255,255,255),1,cv2.LINE_AA)
        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
```

First we use the cap.get() methods to calculate the middle coordinates in our video (width/2, height/2).

Then, before we call imshow() we use cv2.putText(). As the name suggests, this method writes text on the video frame. Usage is: `cv.PutText(img, text, org, font, color)`, where org is the origin (bottom-left corner) of the text to write.

If you run the code, you'll see this:



We can also draw lines, circles, etc. into the video frame, OpenCV has many methods to draw geometric shapes.

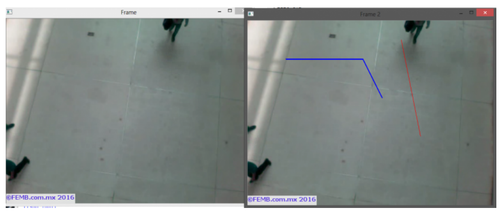
Let's draw some lines:

```
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)
        frame2 = frame
    except:
        #if there are no more frames to show...
        print('EOF')
        break

    line1 = np.array([[100,100],[300,100],[350,200]], np.int32).reshape((-1,1,2))
    line2 = np.array([[400,50],[450,300]], np.int32).reshape((-1,1,2))

    frame2 = cv2.polylines(frame2,[line1],False,(255,0,0),thickness=2)
    frame2 = cv2.polylines(frame2,[line2],False,(0,0,255),thickness=1)

    cv2.imshow('Frame 2',frame2)

    #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
```

This time, we're working outside the **try** block and using two video windows, one to display the raw video and another one to show the modified one with lines.

In order for **polylines** to work, it needs to receive a numpy array with the coordinate pairs (x and y) for each point in the line, in our case, beginning and end. If you want to specify the points like I did, you also need to call **reshape(-1,1,2)** for it to work with polylines().

If you run this code you´ll see:



Here´s OpenCV's documentation for drawing functions.

This is the end of this chapter. In the next one we'll use background substraction to actually start with the counter.

Fede / June 19, 2016 / People counter