Fundamentals of Computer Vision Project 2

Roșu Andrei

The current project is an application able to remove blank video frames, that don’t have any movement, from the real-time video. During the recording the bank frames will have a red X on them, this will notify the user that the frames are not being recorded. The ones that don’t have the X are recorded in a .avi file. For the implementation we will be using OpenCV and Numpy for the processing of the arrays in a timely manner.

The code is split in three sections:

1. The initialization and reading of the frames
2. The augmentation of the read frame
3. The thresholding and saving of the frame
4. Initialization and reading

cap = cv.VideoCapture(1)

width = int(cap.get(cv.CAP\_PROP\_FRAME\_WIDTH))

height = int(cap.get(cv.CAP\_PROP\_FRAME\_HEIGHT))

out = cv.VideoWriter('outpy.avi',cv.VideoWriter\_fourcc('M','J','P','G'), 20, (width,height))

if not cap.isOpened():

    print("Cannot open camera")

    exit()

i = 0

ret, frame = cap.read()

    # if frame is read correctly ret is True

if not ret:

    print("Can't receive frame (stream end?). Exiting ...")

    exit(1)

previous\_frame = cv.cvtColor(frame, cv.COLOR\_BGR2GRAY)

previous\_frame = cv.GaussianBlur(previous\_frame,(21,21),0)

i = 0

while True:

    ret, frame = cap.read()

    #skip one frame

    if (i==0):

        i = 1

        continue

    else:

        i = 0

    if not ret:

        print("Can't receive frame (stream end?). Exiting ...")

        break

Firstly, we initialize the capture video camera and the output video writer. If the camera is not opened the program will stop. We initialize the first frame of the video, saving it at previous\_frame and apply the GasussianBlur on it. After that we start the frame reading loop. Due to the slow pace of the camera I will skip every second frame so there will be no video jittering due to identical frames, after testing this seems to be the fix.

1. Augmenting the frame

   #getting the frame

    color\_frame = frame.copy()

    frame = cv.cvtColor(frame, cv.COLOR\_BGR2GRAY)

    #frame alterations

    frame = cv.GaussianBlur(frame,(21,21),0) #removes noise

I will apply a color change on the frame so I will keep only the grayscale channel, this will make detection of motion more clear, removing random coloring changes in the camera.

The GaussianBlur filter is used to remove noise from the frames.

1. Thresholding and saving

 #thresholding

    frame\_diff = cv.absdiff(frame, previous\_frame)

    ret,threshold = cv.threshold(frame\_diff,25,255,cv.THRESH\_BINARY)

    threshold = cv.dilate(threshold, None, iterations=2)

    avg = np.average(threshold)

    if (avg <= 0.1):

        cv.line(color\_frame,(0,0),(width, height),(0,0,255),5)

        cv.line(color\_frame,(width,0),(0, height),(0,0,255),5)

    else:

        out.write(color\_frame)

    previous\_frame = frame.copy()

In the last section of the implementation I am taking the differences between the fames and thresholding them in a (25,255) interval, all the values below 25 being ignored. After that the result is being dilatated so it can influence the overall average of the image in a more impactful way, without the dilatation small objects moving won’t activate the motion detection.

If the average is not above 0.1 the frame does not have any movement and it won’t be saved. I chose 0.1 as the value and not 0 due to the possibility of some frames to not be 0 even though no movement is present, another layer of protection against noise.

In the end the frame is saved for comparison with the next frame that will come from the camera.