Computer Vision – Assignment 0 - Report

Shivani Chepuri
2018122004

Note – Please check all the input and output files in the provided google drive links.

https://drive.google.com/drive/folders/1n_TC0EnWJ4Q-6KqIeP8MugY3I86U_KL2?usp=sharing

Please find the folder named **cv** – that is being shared by the link. The contents and corresponding tasks are as follows:

Input self-video - "vv6.mp4", "vv6.webm"

Input Greenmat Butterfly video - "vv5.mp4"

Outputs:

Q1(a) - "images" folder

Q1(b) - "shivani_assg00.mp4"

Q2 - "images_webcam" , **"op_chroma.avi"**

Q3 – "final.mp4"

Question 1 – Video to Images:

Problem Description: Given a video, we are supposed to extract images from it. Given a set of images, make a video out of them, being able to vary fps if asked to.

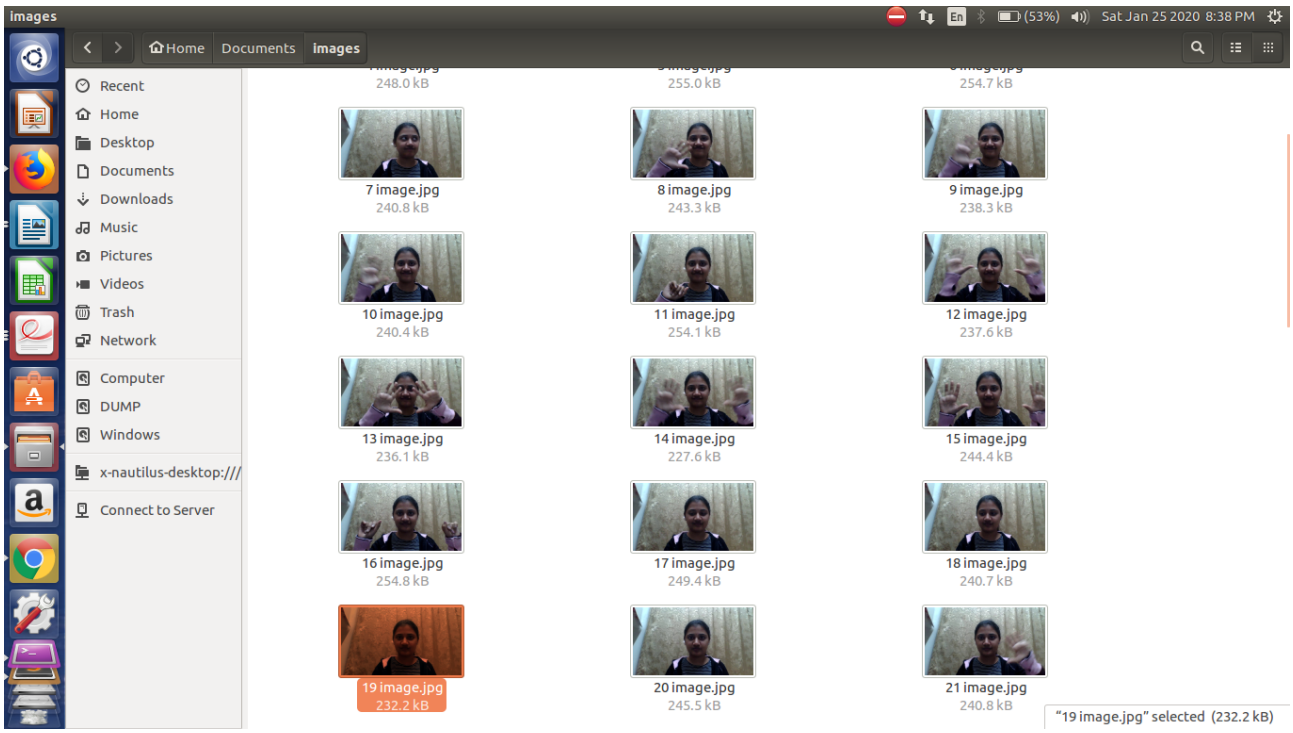
For this question, I have taken a video of myself using inbuit webcam, which I used through out the assignment. It is named "vv6.mp4" and "vv6.webm" (to expermiment between format conversions and number of frames, I have tried both the formats)

For output images please look in **"images" folder**

Solution and Challenges:
For any given video, depending upon how frequently we are sampling, we may get any number of frames. This is called frames per second or 'fps'. By manipulating fps, we can get the desired number of images from the video.

In the code that has been used to obtain the solution, I have taken the parameters as follows:
frame rate = 0.5
It implies that I will sample the image from the video every 0.5 seconds (or) the **fps = 2,** that is for every second the number of frames I extract is 2.

It was a bit challenging to save the file in the required directory and manipulating the fps, I had to tune the framerate variable, a bit.



Algorithm/Code Explanation:
CAP_PROP_POS_MSEC gives how many milliseconds we are at, in the video.

Therefore, v.set(cv2.CAP_PROP_POS_MSEC,c*1000) (frames function) gives a pointer to where we are in the video. As the loop goes on(while loop in the main function – runs until the video is over), we proceed to next point in the video which is at a distance of c*1000 (or 1 second)

In the next step we read the image or the frame captured at the point where we have just placed out pointer.

i.e, check, image = v.read()

After capturing the image and making sure **frame availability != 0**, using, cv2.imwrite(), we write the image to a file in the project directory.

```
#image or frame capture from video
#video to images

import cv2, glob, os
import numpy as np

p1 = '/home/shivani/Documents/vv6.mp4'
v = cv2.VideoCapture(p1)

path = '/home/shivani/Documents/images'
count = 1
c = 0
frameRate = 0.5

def frames(c):
    v.set(cv2.CAP_PROP_POS_MSEC,c*1000)
```

```
    check,image = v.read()
    if check == True  or check == 1:
        cv2.imwrite(os.path.join(path, str(count)+ " " +"image.jpg"), image)
    return check

s = frames(c)
while s:
    count += 1
    c = c + frameRate
    c = round(c, 2)
    s = frames(c)

print("please check 'images' folder for output")
v.release()
```

Images to Video:

I have used the same images or frames captured above to regenerate the original video (to set a mark for fps and all)

First, I tried to put all the images together in the video and later worked on arranging the images in the order of their capture.

Solutions and Challenges:

The part of videoWriting had to be separetly understood.
That is the lines,
x = cv2.VideoWriter_fourcc(*'DIVX')
out_var = cv2.VideoWriter('shivani_assg00.mp4',x, fps, (width,height))

which have so many parameters.

Width,height can be set to anything as per the necessary size of the video – kept same as the frame size.

Output file name is **“shivani_assg00.mp4”**

Algorithm:

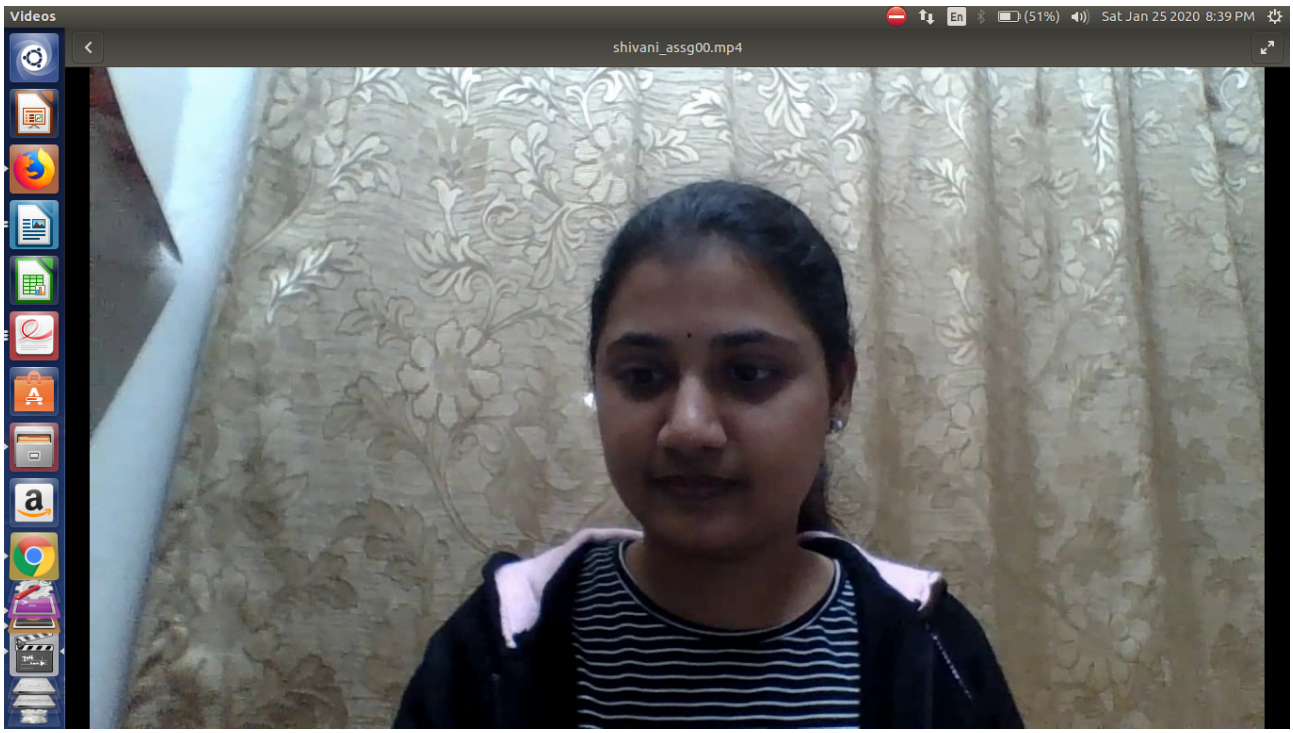
All the images names are arranged in an array according to the order of their capture.

Using this array, one by one, the images are read and their shapes are saved.

As the images are read, they are also put in an array, so that we ca use this image_array later on, put some fps and create a video.

Now, before writing them into a video object, we first have to setup the object which is done in the variable, **out_var.**

Here, cv2.waitkey() mainly controls the fps. Fps is also controlled inside the object we are wrting the images to (to make the video)




```
#video creation from images
#images to video

from os import listdir
from os.path import isfile, join
import cv2
import numpy as np
import glob
import os

#Note - all the images are coming in order of their name(or generation) in the regenerated video
#Note - the playback speed or frame rate of the video is being controlled

path = '/home/shivani/Documents/images'
img_array = []
fps = 20
file_names = listdir(path)
a = sorted(file_names, key = lambda item: (int(item.partition(' ')[0])
                        if item[0].isdigit() else float('inf'), item))
print(a)

for _file in a:
    image = cv2.imread(path + '/' + _file)
    height, width, layers = image.shape
    img_array.append(image)

x = cv2.VideoWriter_fourcc(*'DIVX')
out_var = cv2.VideoWriter('shivani_assg00.mp4',x, fps, (width,height))

l = len(img_array)
for i in range(l):
```

```
    out_var.write(img_array[i])
    cv2.waitKey(1000)
out_var.release()

print("\n")
print("please check the file with name shivani_assg0.mp4 for output, change the 'fps' variable and waitkey parameter)
```

Question 2

Problem Description: Capture frames from webcam and store the images in a folder.

I have done a live streaming of webcam. A part of the code keeps capturing the images from the webcam. Another code takes a video of the streaming and saves it to a file in the project directory.

Solutions and Challenges:

Managing fps and frame rates for both capturing and making a video needed some time to tune.

We can also see the frames that are being captured being displayed in a window on the screen, and the captured video is saved in a .mp4 file.

Algorithm:

cv2.videocapture(0) – 0 is intrinsically for the webcam. This line activates the webcam to start capturing. This does not write or read anything but just initiates the object.

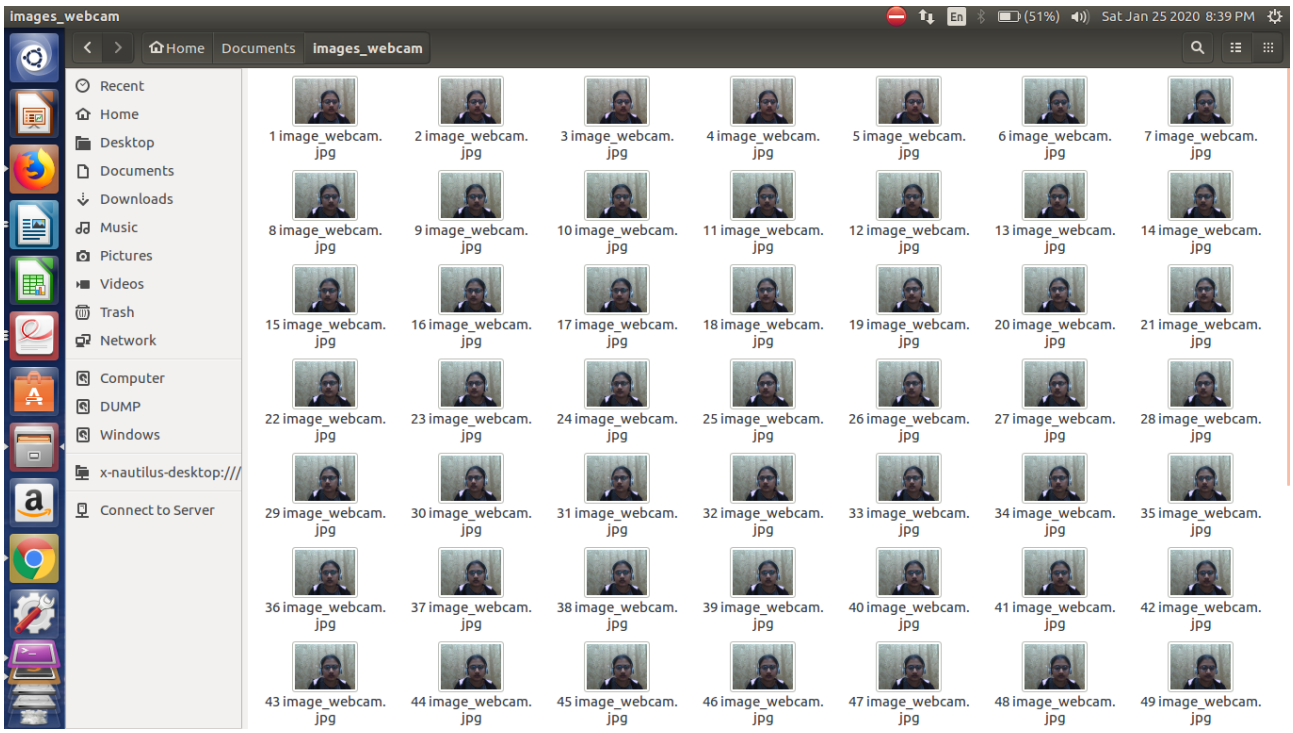
Therefore, we need to read from it and then write to a file, to capture a frame.
Then, we need to display the captured frame using cv2.imshow()

The capturing stops when we press “w”. It can also be set for some time, say 20 seconds, using a counting variable, i.e, “millis”; both are setup in the code.

In the end, we need to close the webcam and stop displaying the images.

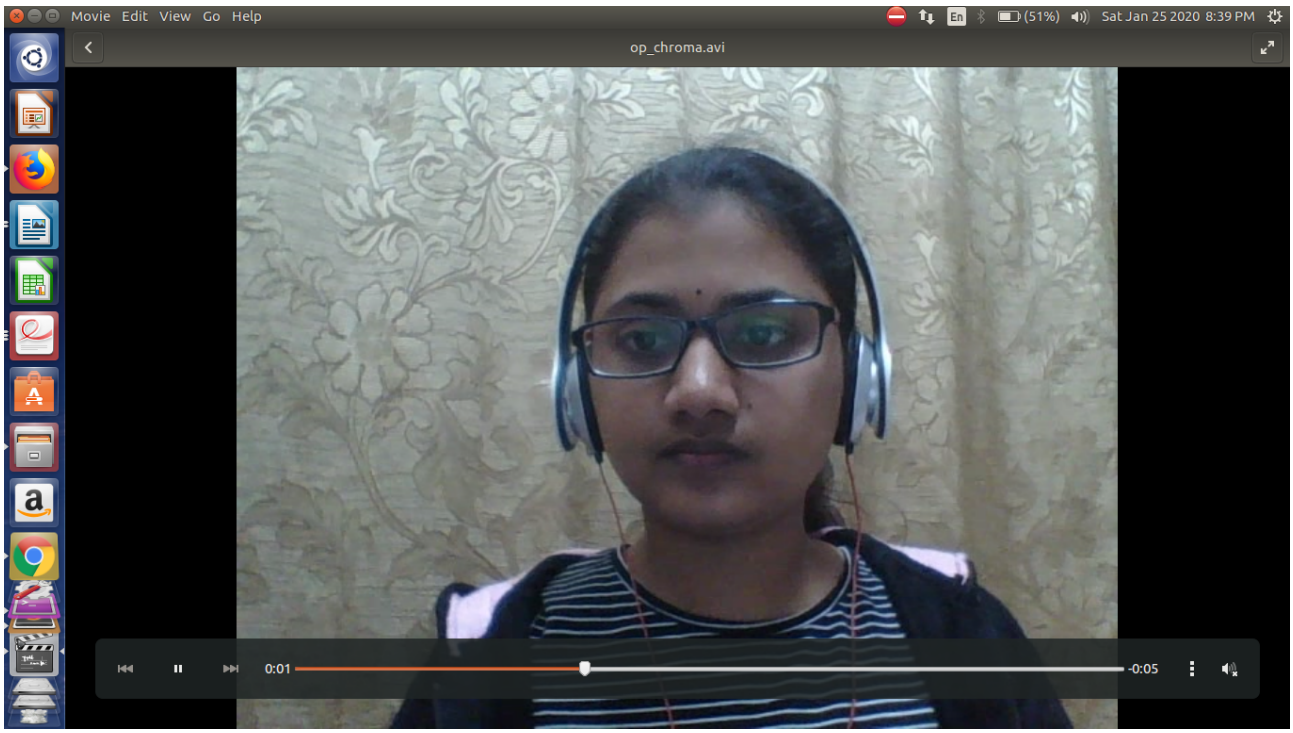
The following image shows webcam images output.
Look in **“images_webcam”** folder

https://drive.google.com/drive/folders/1n_TC0EnWJ4Q-6KqIeP8MugY3I86U_KL2?usp=sharing




```
p2 = '/home/shivani/Documents/images_webcam'
mills = 0

vidcap_webcam = cv2.VideoCapture(0)
while True:
    mills += 1
    check,frame = vidcap_webcam.read()
    #print(check)
    if check == True:
        cv2.imwrite(os.path.join(p2, str(mills)+ " " +"image_webcam.jpg"), frame)

    cv2.imshow("now_streaming", frame)
    k = cv2.waitKey(100)

    if k == ord('w'):
        break
    elif k == 50:
        break

        print("no. of milliseconds of streaming or number of frames captured", mills)
vidcap_webcam.release()
cv2.destroyAllWindows()
```

Video Capturing from webcam:
webcam video output is in “op_chroma.avi”



Algorithm:

The same concept as it is with images but we need to setup a videowriter and write to its object. We also need to define the output video name and write the whole thing to it.

```
import cv2
import numpy as np

#capture a video of myself
cap = cv2.VideoCapture(0)
output_file = 'op_chroma.avi'
c = 0
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
x = cv2.VideoWriter_fourcc(*'DIVX')
fps = 15
op_out = cv2.VideoWriter(output_file, x, fps, (width,height))
while (cap.isOpened()):
    c +=1
    check,frame = cap.read()

    if check:
        op_out.write(frame)
        cv2.imshow("now streaming",frame)
    else:
        break

    k = cv2.waitKey(100)
    if k == ord('w'):
        break
    #elif k == 25:
```

 # break

cap.release()
op_out.release()
cv2.destroyAllWindows()

Question 3:

Problem Description: Chroma Keying

Chroma key compositing, or chroma keying, is a visual effects/post-production technique for compositing (layering) two images or video streams together based on colour hues (chroma range). The technique has been used in many fields to remove a background from the subject of a photo or video – particularly the newscasting, motion picture, and video game industries.

Solution and Challenges:
In the current, task we are considering 2 videos, one with myself in the background and another video which has an object (butterfly flying) with a green or bluemat, in the foreground.

We want to mask the butterfly and make it the main object in the foreground.
My video will be in the background and all the area except for the shape of butterfly will be masked in this video.
When we add up these two videos (a series of frames) that is,
masked area in the foreground + masked area in the background,
we get the final video of me with a butterfly flying accorss.

We need to setup upper and lower limit to the pixel's values of RGB or BGR to do the mask, it was a bit challenging and has taken time.
Also, the shapes of the frames was different for the two videos which needed cropping. Finally settled :))

Small green dots along the edges are still there even after tuning the thresholds.

Algorithm:

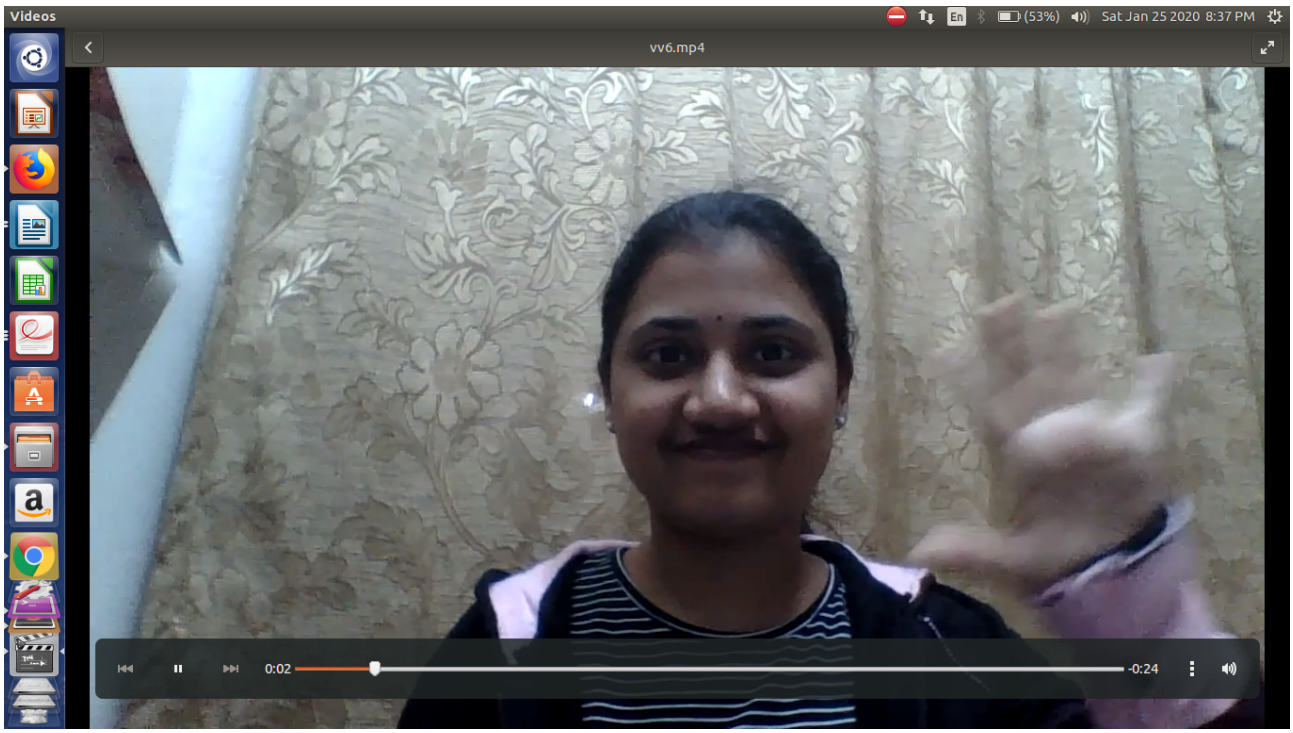
First we read the two videos using "videocapture".
We then define the thresholds for picture values which depends on the content of the videos, so that the masks are almost perfect.

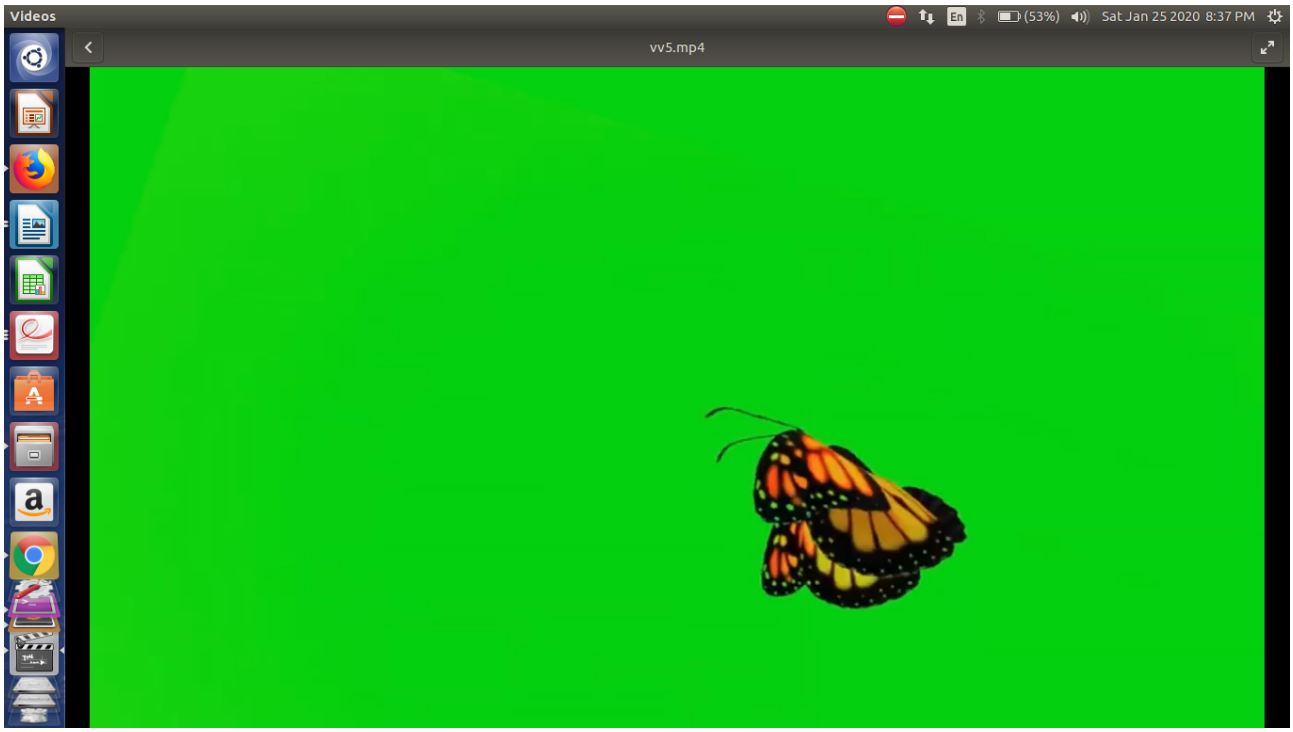
As explained in the solution, we create complimentary masks and add them to get the final video.
We also had to crop the foreground video to match shapes of the frames.

masked_frame[mask != 0] = [0, 0, 0] - if mask = 0, it is shown in black and if mask = 1 it is shown in white. So, here we are making the unmasked reqion [0,0,0], that is no pixel information – so that we can put something else there.

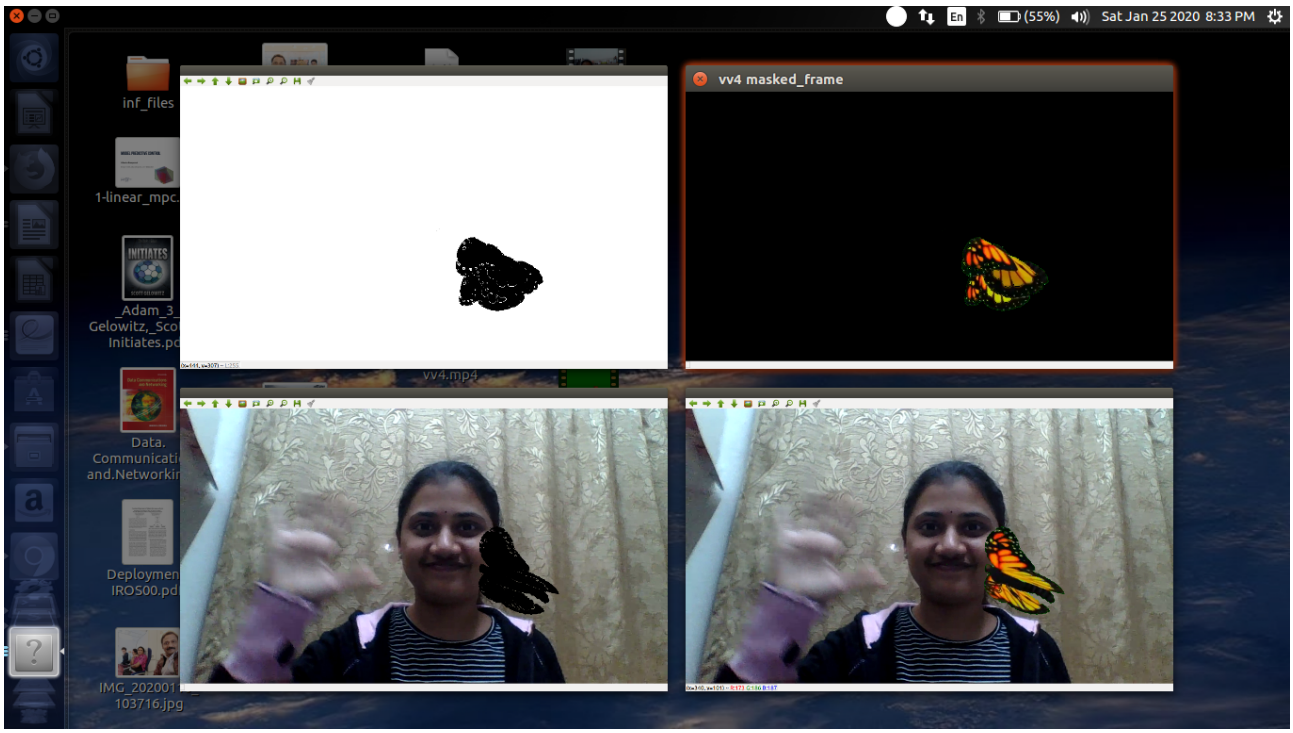
Similarly,
frame_back_crop[mask == 0] = [0,0,0] --> We are making the masked region zero i.e, [0,0,0] , no pixel information – so that the foreground's masked region can take its place.

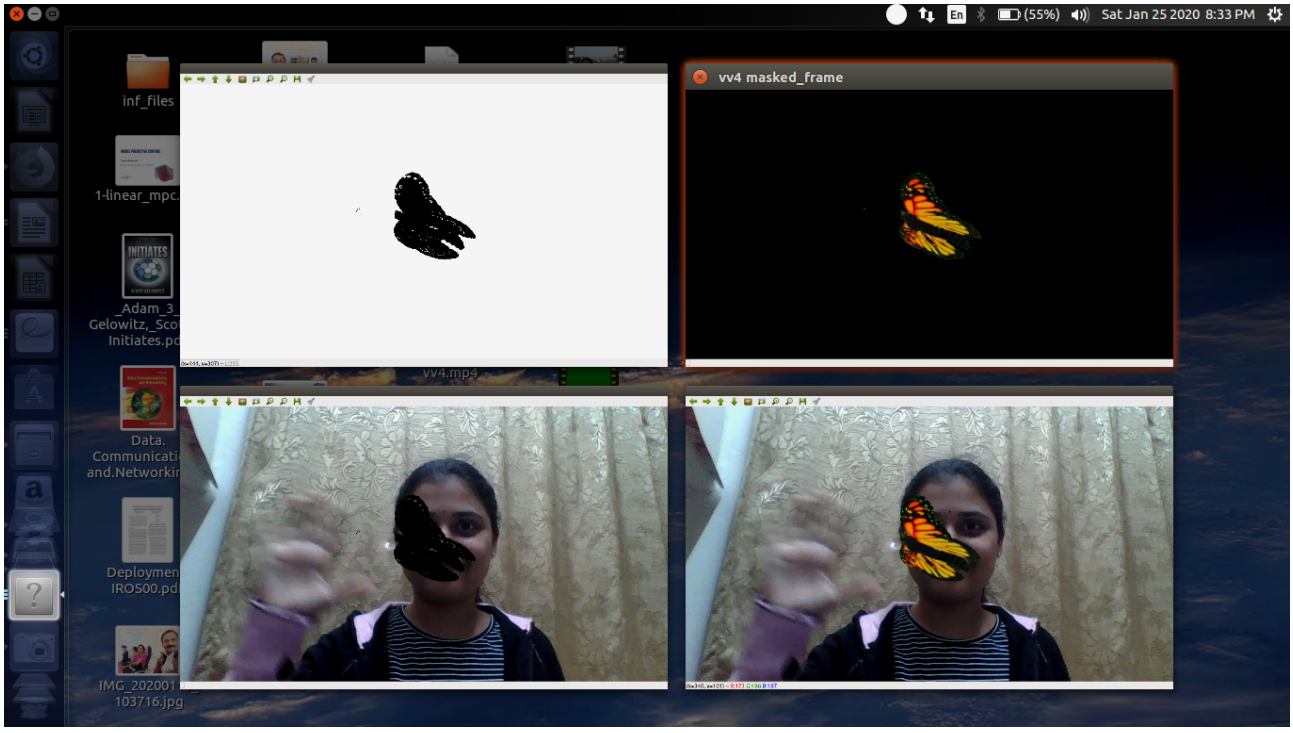
Original Videos (screenshots of a frame from the video)
Please find the final video as "final.mp4"

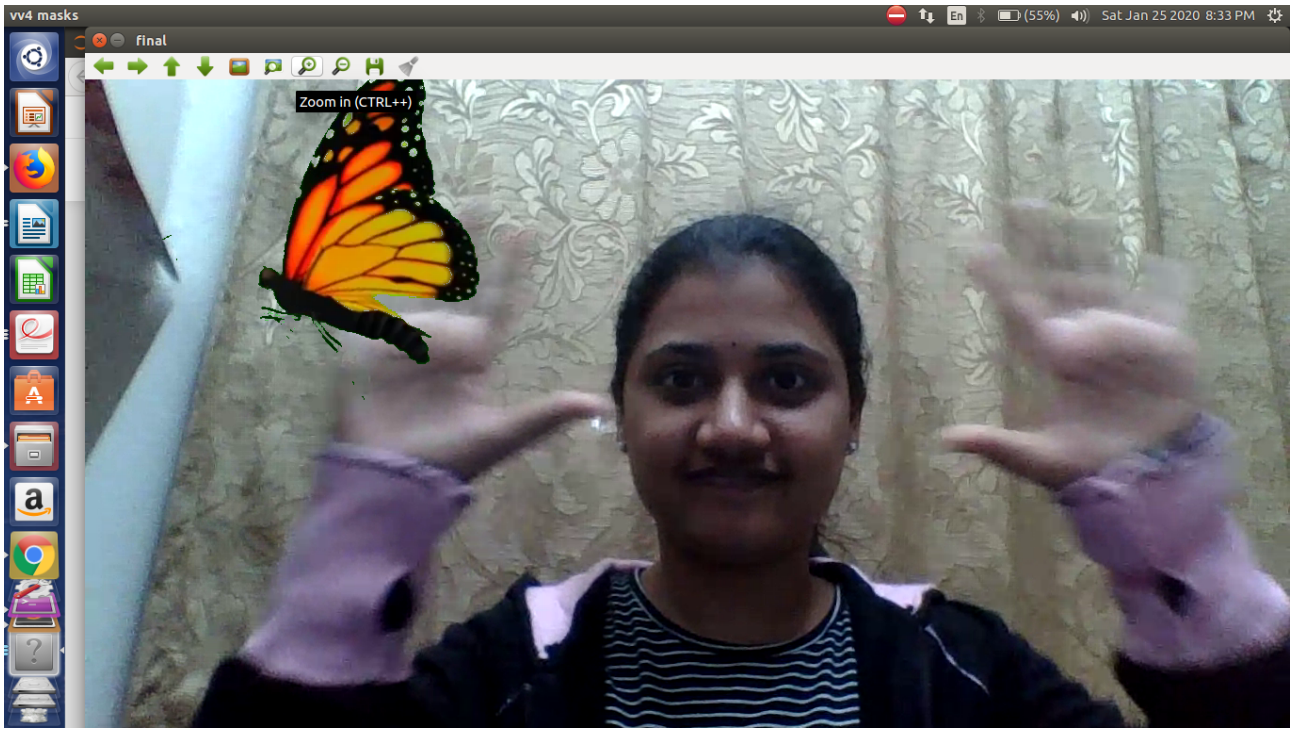




Please look at all the following images (screenshots of a frame from the video outputs)


Sat Jan 25 2020 8:33 PM
vv4 masked_frame
vv4.mp4
inf_files


Sat Jan 25 2020 8:33 PM
vv4 masked_frame
vv4.mp4
inf_files




```
#read a video with green mat and 2 or 3 objects
import cv2
import numpy as np
import matplotlib.pyplot as plt

#vv5 is the video with butterflies
#vv6 my video

vid = '/home/shivani/Documents/vv5.mp4'
#vid2 = '/home/shivani/Documents/op_chroma.avi'
vid2 = '/home/shivani/Documents/vv6.webm'

cap2 = cv2.VideoCapture(vid)
cap3 = cv2.VideoCapture(vid2)

lower_green = np.array([0, 80, 0])
upper_green = np.array([80, 255, 120])
img_array2 = []
fps2 = 15
while(cap2.isOpened() and cap3.isOpened()):
    check,frame = cap2.read()
    check_back,frame_back = cap3.read()
    if check and check_back:
        frame_copy = np.copy(frame)
        frame_back_copy = np.copy(frame_back)
        height, width = frame.shape[:2]
        #print(height,width)
        #frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        #frame_back = cv2.cvtColor(frame_back, cv2.COLOR_BGR2RGB)
```

```
        #cv2.imshow("vv4 frames", frame)
        #cv2.imshow("vv4 frames back", frame_back)

        mask = cv2.inRange(frame, lower_green, upper_green)
        cv2.imshow("vv4 masks", mask)
        masked_frame = np.copy(frame)
        masked_frame[mask != 0] = [0, 0, 0]
        cv2.imshow("vv4 masked_frame", masked_frame)

        frame_back_crop = frame_back[0:height, 0:width]
        frame_back_crop[mask == 0] = [0,0,0]
        cv2.imshow("vv4", frame_back_crop)

        final_frame = frame_back_crop + masked_frame
        cv2.imshow("final", final_frame)

        #I = cv2.imread(final_frame)
        #H, W, L = I.shape
        img_array2.append(final_frame)

    k = cv2.waitKey(10)
    if k == ord('w'):
        break

x2 = cv2.VideoWriter_fourcc(*'DIVX')
out_var2 = cv2.VideoWriter('final.mp4',x2, fps2, (width,height))

l2 = len(img_array2)
for i in range(l2):
    out_var2.write(img_array2[i])
    cv2.waitKey(1)

out_var2.release()
cap2.release()
cap3.release()
cv2.destroyAllWindows()
```

https://drive.google.com/drive/folders/1n_TC0EnWJ4Q-6KqIeP8MugY3I86U_KL2?usp=sharing

The ENd