Report on video: "Detecting colors (Hsv Color Space) - Opencv with Python"

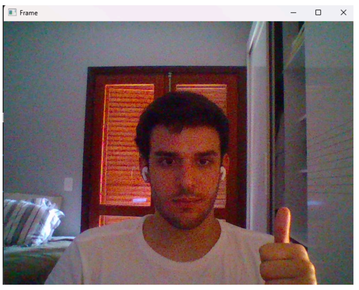
*_https://www.youtube.com/watch?v=Q0lTPYlK-4A_*

Below, a color detecting algorithm using OpenCv will be tested

In [1]:
```
import cv2 # Importing the Opencv library
import numpy as np
```

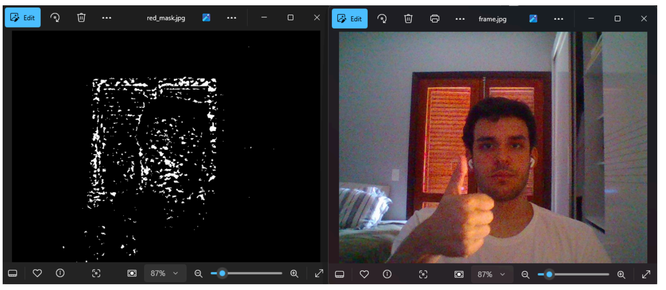
In [2]:
```
cap = cv2.VideoCapture(0)
# This is essentially just instantiating an object that accesses the computer camera.
```

In [3]:
```
# Loading the frames and reading them.
while True:
    _, frame = cap.read()

    cv2.imshow("Frame", frame) # Will load the Webcam image

    key = cv2.waitKey(1)
    if key == 27: # 27 is the ID for the Escape key on the keyboard, to stop showing the webcam image.
        break
```



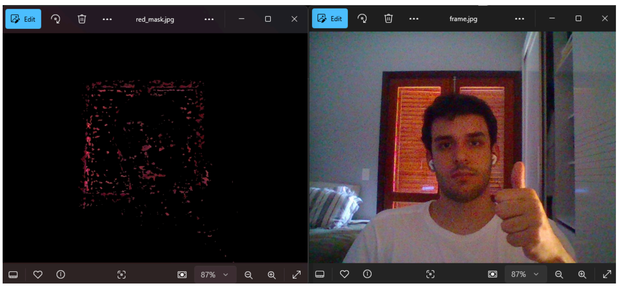
In [11]:
```
from time import sleep
# Loading the frames and reading them.
for i in range(1):
    sleep(1)
    ret, frame = cap.read()
    if not ret:
        break

    # Here, the hsv_frame will now hold an instance of the webcam, but
    # using HSV instead of the default BGR cholor scheme from opencv.
    # Once it is converted, it's possible to create a range for the color where
    # we define the lowest part of a specific color so the lowest/highest values of
    # Hue, Saturation and Value (HSV).

    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Red color masking:
    low_red = np.array([161,155,84])
    high_red = np.array([179,255,255])
    red_mask = cv2.inRange(hsv_frame, low_red, high_red)
    # Arranging the HSV image to Red

    cv2.imwrite("frame.jpg", frame)     # Saving the normal camera image
    cv2.imwrite("red_mask.jpg", red_mask) # Saving the altered camera image
```

With this, whichever kind of red that belongs to the `low_red -- high_red` interval will be highlighted as white (the red window border gets highlighted in this image)



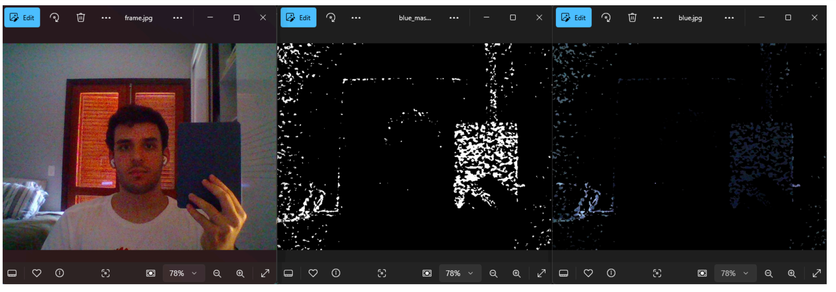
Instead of showing as white, now using the bitwise operator "and" to use it as red:

In [15]:
```
from time import sleep
# Loading the frames and reading them.
for i in range(1):
    sleep(1)
    ret, frame = cap.read()
    if not ret:
        break

    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Red color masking:
    low_red = np.array([161,155,84])
    high_red = np.array([179,255,255])
    red_mask = cv2.inRange(hsv_frame, low_red, high_red)

    red = cv2.bitwise_and(frame, frame, mask=red_mask)

    cv2.imwrite("frame.jpg", frame)     # Saving the normal camera image
    cv2.imwrite("red_mask.jpg", red) # Saving the altered camera image
```



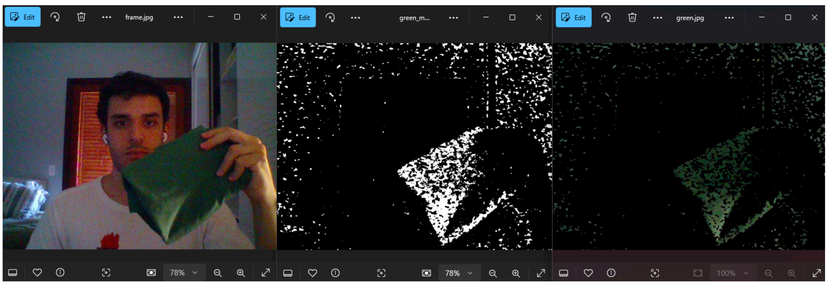
Now, doing the same thing but with blue color:

In [17]:
```
from time import sleep
# Loading the frames and reading them.
for i in range(1):
    sleep(3)
    ret, frame = cap.read()
    if not ret:
        break

    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Blue color masking:
    low_blue = np.array([94,80,2])
    high_blue = np.array([126,255,255])
    blue_mask = cv2.inRange(hsv_frame, low_blue, high_blue)

    blue = cv2.bitwise_and(frame, frame, mask=blue_mask)

    cv2.imwrite("frame.jpg", frame)     # Saving the normal camera image
    cv2.imwrite("blue_mask.jpg", blue_mask)     # Saving the blue masked camera image
    cv2.imwrite("blue.jpg", blue) # Saving the blue camera image
```



Lastly, doing the same process with the green color:

In [20]:
```
from time import sleep
# Loading the frames and reading them.
for i in range(1):
    sleep(3)
    ret, frame = cap.read()
    if not ret:
        break

    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # green color masking:
    low_green = np.array([35,50,50])
    high_green = np.array([85,255,255])
    green_mask = cv2.inRange(hsv_frame, low_green, high_green)

    green = cv2.bitwise_and(frame, frame, mask=green_mask)

    cv2.imwrite("frame.jpg", frame)           # Saving the normal camera image
    cv2.imwrite("green_mask.jpg", green_mask) # Saving the green masked camera image
    cv2.imwrite("green.jpg", green)           # Saving the green camera image
```



In short, to filter colors in an image, its needed to determine an interval of color values that are going to be highlighted in the image.