Changing color spaces

We can use the `cv2.cvtColor(image,flag)` function to convert the color space of an image. The flag defines the existing color space and the color space the image has to be converted to.

There are a lot of color conversion flags available in opencv. To get the list of all color spaces just type cv2.COLOR_ and press the crtl+space key to show the list.

We are going to look at 2 color conversion formats.

1. `cv2.COLOR_BGR2GRAY`
2. `cv2.COLOR_BGR2HSV`

- Hue range in opencv is from 0-179 (most of the online color pickers have a range of 0-360)
- Saturation range is 0-255
- Value range is 0-255

We use BGR2GRAY to create masks for images
The HSV color sapce is used intead of the BGR color sapce as thersholding for a particular color is easier in HSV color sapce

---

Before we start tracking the ball we must able to isolate the ball from the entire image. we will using the `inRange()` function to acheive this.

We will take a pic of the ball from the video and try to isolate the ball from the background.

In [3]:

```
from cv2 import cv2
import numpy as np

img = cv2.imread('images/blue_ball_pic.JPG')

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

def nothing(x):
    pass

cv2.namedWindow('image')

cv2.createTrackbar('H-Low','image',0,179,nothing)
cv2.createTrackbar('H-High','image',0,179,nothing)

cv2.createTrackbar('S-Low','image',0,255,nothing)
cv2.createTrackbar('S-High','image',0,255,nothing)

cv2.createTrackbar('V-Low','image',0,255,nothing)
```

```
cv2.createTrackbar('V-High','image',0,255,nothing)

while True:
    h_l = cv2.getTrackbarPos('H-Low','image')
    h_h = cv2.getTrackbarPos('H-High','image')

    s_l = cv2.getTrackbarPos('S-Low','image')
    s_h = cv2.getTrackbarPos('S-High','image')

    v_l = cv2.getTrackbarPos('V-Low','image')
    v_h = cv2.getTrackbarPos('V-High','image')

    low = np.array([h_l,s_l,v_l],dtype = 'uint8')
    high = np.array([h_h,s_h,v_h],dtype = 'uint8')


    mask = cv2.inRange(hsv, low,high)

    blue = cv2.bitwise_and(img,img,mask = mask)

    cv2.imshow('image',blue)
    cv2.imshow('mask',mask)


    key  = cv2.waitKey(50)

    if key == ord('q'):
        break

cv2.destroyAllWindows()
```

The high and low HSV threshold that we get from the tackbars can now be used to track the ball from the video.

low thresholds : `[32,86,109]`
high thresholds : `[127,255,199]`

Now since we have the high and the low threshold for the blue ball we can use this to actually track the ball in the video.

In [1]:

```
import cv2
import numpy as np

cap = cv2.VideoCapture('videos/blue_ball.mp4')

while(1):

    # Take each frame
    ret, frame = cap.read()

    if not ret:
        break
    # Convert BGR to HSV
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # define range of blue color in HSV
    lower_blue = np.array([32,86,109])
    upper_blue = np.array([127,255,199])
```

```
    # Threshold the HSV image to get only blue colors
    mask = cv2.inRange(hsv, lower_blue, upper_blue)

    # Bitwise-AND mask and original image
    res = cv2.bitwise_and(frame,frame, mask= mask)

    cv2.imshow('frame',frame)
    cv2.imshow('mask',mask)
    cv2.imshow('res',res)
    k = cv2.waitKey(5) & 0xFF
    if k == 27:
        break


cv2.destroyAllWindows()
```

We a bit of green with the above thresholds. The reason is we did'nt fine tuen the thresholds.

low thresholds : `[92,86,109]`
high thresholds : `[127,255,199]`

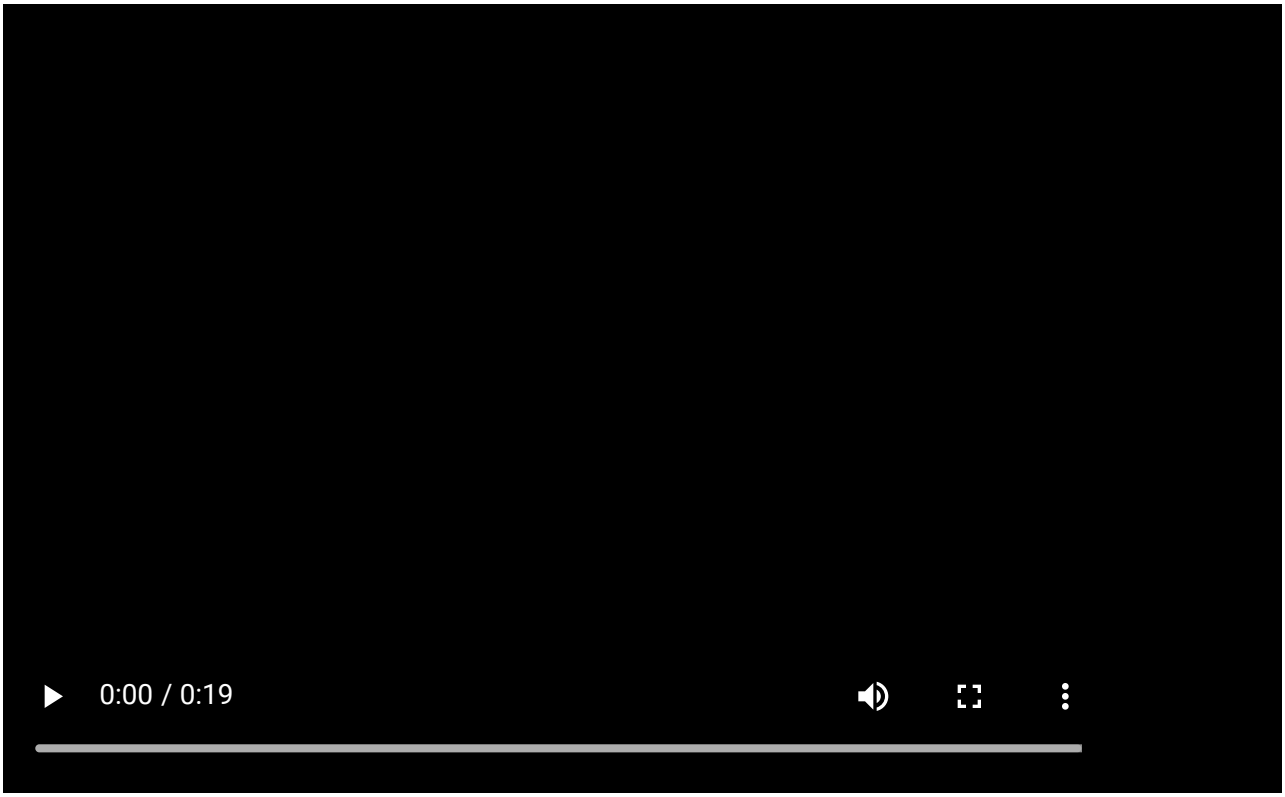
We can try again with the above thershold to get a better results

---

Home Work

1) Tracking Multiple colored balls

Track the blue, red and the green colored balls in this video



Solution - problem 1

In [1]:

```
from cv2 import cv2
import numpy as np

img = cv2.imread('images/all_ball_pic.JPG')

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

def nothing(x):
    pass

cv2.namedWindow('track_bar')

cv2.createTrackbar('H-Low','track_bar',0,179,nothing)
cv2.createTrackbar('H-High','track_bar',0,179,nothing)

cv2.createTrackbar('S-Low','track_bar',0,255,nothing)
cv2.createTrackbar('S-High','track_bar',0,255,nothing)

cv2.createTrackbar('V-Low','track_bar',0,255,nothing)
cv2.createTrackbar('V-High','track_bar',0,255,nothing)

while True:
    h_l = cv2.getTrackbarPos('H-Low','track_bar')
    h_h = cv2.getTrackbarPos('H-High','track_bar')

    s_l = cv2.getTrackbarPos('S-Low','track_bar')
    s_h = cv2.getTrackbarPos('S-Hign','track_bar')

    v_l = cv2.getTrackbarPos('V-Low','track_bar')
    v_h = cv2.getTrackbarPos('V-High','track_bar')

    low = np.array([h_l,s_l,v_l],dtype = 'uint8')
    high = np.array([h_h,s_h,v_h],dtype = 'uint8')


    mask = cv2.inRange(hsv, low,high)

    blue = cv2.bitwise_and(img,img,mask = mask)

    cv2.imshow('image',blue)
    cv2.imshow('mask',mask)


    key  = cv2.waitKey(50)

    if key == ord('q'):
        break

cv2.destroyAllWindows()
```

Blue Ball

low thresholds : `[80,144,134]`

high thresholds : `[114,255,182]`

Red Ball - 1

low thresholds : `[0,86,88]`

high thresholds : `[6,113,255]`

Red Ball - 2

low thresholds : `[172,86,88]`

high thresholds : `[179,113,255]`

Green Ball

low thresholds : `[33,70,81]`

high thresholds : `[98,179,255]`

In [29]:

```
import cv2
import numpy as np

cap = cv2.VideoCapture('videos/all_ball.mp4')

while(1):

    # Take each frame
    ret, frame = cap.read()

    if not ret:
        break
    # Convert BGR to HSV
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # define range of blue color in HSV
#     lower_blue = np.array([80,144,134])
#     upper_blue = np.array([114,255,182])

#     lower_green = np.array([33,70,81])
#     upper_green = np.array([98,179,255])

    lower_red_1 = np.array([0,106,102])
    upper_red_1 = np.array([8,255,255])

    lower_red_2 = np.array([172,86,88])
    upper_red_2 = np.array([179,255,255])

    # Threshold the HSV image to get only blue colors
#     mask_b = cv2.inRange(hsv, lower_blue, upper_blue)
#     mask_g = cv2.inRange(hsv, lower_green, upper_green)
    mask_r1 = cv2.inRange(hsv, lower_red_1, upper_red_1)
    mask_r2 = cv2.inRange(hsv, lower_red_2, upper_red_2)


#     mask = cv2.add(mask_b,mask_g,mask_r1,mask_r2)
#     mask1 = cv2.bitwise_or(mask_b,mask_g)
    mask2 = cv2.bitwise_or(mask_r1,mask_r2)

#     mask = cv2.bitwise_or(mask1,mask2)

    # Bitwise-AND mask and original image
    res = cv2.bitwise_and(frame,frame, mask= mask2)

    cv2.imshow('frame',frame)
    cv2.imshow('mask',mask2)
    cv2.imshow('res',res)
    k = cv2.waitKey(20) & 0xFF
    if k == 27:
        break
```

```
cv2.destroyAllWindows()
```