Open CV – 3

1.Color detection

```
// import modules
cap = cv2.VideoCapture(1)
while(1):
        Capture a frame
        ret, frame = cap.read()
        #convert bgr to hsv
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
        # define range of desired color in HSV
        lower_orange = np.array([0,144,215])
        upper_orange = np.array([34,255,255])
        #threshold the hsv image to get desired color
        mask = cv2.inRange(hsv, lower_orange, upper_orange)
        # Bitwise-AND mask and original image
        res = cv2.bitwise_and(frame,frame, mask= mask)
        # morphological operations
        kernel = np.ones((5,5),np.uint8)
        erosion = cv2.erode(mask,kernel,iterations = 1)
        dilation = cv2.dilate(mask,kernel,iterations = 1)
        opening = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)
        closing = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
        cv2.imshow('Erosion',erosion)
        cv2.imshow(‘dilation',dilation)
        cv2.imshow('Original',frame)
        cv2.imshow('Mask',mask)
        cv2.imshow('Opening',opening)
        cv2.imshow('Closing',closing)

        k = cv2.waitKey(5) & 0xFF
        if k == 27:
                break

cv2.destroyAllWindows()
cap.release()
```

2.Simple Code for tracking a coloured object :
(Here we are tracking an Orange TT ball)

```
import cv2
import numpy as np
import cv2.cv as cv
import time
import serial
cap = cv2.VideoCapture(0)
while(1):
         #capture a frame
        ret, frame = cap.read()
        #convert bgr to hsv
        hsv = cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
        # define range of desired color in HSV
        lower_orange = np.array([0,144,215])
        upper_orange = np.array([34,255,255])
        #threshold the hsv image to get desired color
        mask = cv2.inRange(hsv, lower_orange, upper_orange)
        # Bitwise-AND mask and original image
        res = cv2.bitwise_and(frame,frame, mask= mask)
        # morphological operations
        kernel = np.ones((5,5),np.uint8)
        erosion = cv2.erode(mask,kernel,iterations = 1)
        dilation = cv2.dilate(mask,kernel,iterations = 1)
        opening = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)
        closing = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
        # Detect circles using HoughCircles
        circles =  cv2.HoughCircles(closing,cv.CV_HOUGH_GRADIENT,2,120,
        param1=100,param2=55,minRadius=10,maxRadius=0)
        #Draw Circles
        if circles is not None:
                for i in circles[0,:]:
                        # If the ball is far, draw it in green
                        cv2.circle(frame,(int(round(i[0])),int(round(i[1]))),int(round(i[2])),
                        (255,0,0),4)
                        cv2.circle(frame,(int(round(i[0])),int(round(i[1]))),1,(0,255,0),4)

        cv2.imshow('tracking',frame)
        cv2.imshow('Original',frame)
        cv2.imshow('Mask',mask)
        cv2.imshow('Result',res)

        k = cv2.waitKey(5) & 0xFF
        if k == 27:
                break

cv2.destroyAllWindows()
cap.release()
```

3. Code for finding HSV values using trackbars :

```
import cv2
import numpy as np

#function to apply HSV value
def getthresholdedimg(hsv):
        threshImg =cv2.inRange(hsv,np.array((cv2.getTrackbarPos('Hue_Low','Trackbars'),cv2.getTrackbarPos('Saturation_Low','Trackbars'),cv2.getTrackbarPos('Value_Low','Trackbars'))),np.array((cv2.getTrackbarPos('Hue_High','Trackbars'),cv2.getTrackbarPos('Saturation_High','Trackbars'),cv2.getTrackbarPos('Value_High','Trackbars'))))
        return threshImg

#function to get present trackbar value
def getTrackValue(value):
        return value

c = cv2.VideoCapture(0)
width,height = c.get(3),c.get(4)
print "frame width and height : ", width, height

#Create trackbars
cv2.namedWindow('Output')
cv2.namedWindow('Trackbars', cv2.WINDOW_NORMAL)
cv2.createTrackbar('Hue_Low','Trackbars',0,255, getTrackValue)
cv2.createTrackbar('Saturation_Low','Trackbars',0,255, getTrackValue)
```

```
cv2.createTrackbar('Value_Low','Trackbars',0,255, getTrackValue)
cv2.createTrackbar('Hue_High','Trackbars',0,255, getTrackValue)
cv2.createTrackbar('Saturation_High','Trackbars',0,255, getTrackValue)
cv2.createTrackbar('Value_High','Trackbars',0,255, getTrackValue)
cv2.createTrackbar('Caliberate','Trackbars',0,1, getTrackValue)
while(1):
    _,f = c.read()
    f = cv2.flip(f,1)
    # Convert BGR to HSV
    hsv = cv2.cvtColor(f,cv2.COLOR_BGR2HSV)
    thrImg = getthresholdedimg(hsv)


    #Morphological operations
    erode = cv2.erode(thrImg,None,iterations = 3)
    dilate = cv2.dilate(erode,None,iterations = 10)


    #finding contours
    contours,hierarchy =
    cv2.findContours(dilate,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)
    for cnt in contours:
        x,y,w,h = cv2.boundingRect(cnt)
        cx,cy = x+w/2, y+h/2
        cv2.rectangle(f,(x,y),(x+w,y+h),[0,0,255],2)

    if(cv2.getTrackbarPos('Caliberate','Trackbars') == 1):
        cv2.imshow('Output',thrImg)
    else:
        cv2.imshow('Output',f)


    if cv2.waitKey(10) & 0xFF == ord('q'):
        break

cv2.destroyAllWindows()
cap.release()
```

Python – Arduino Communication

4a. Python Program

```
import numpy as np
import cv2 as cv
import serial
import time

#Initializing the device the for communication
ser = serial.Serial('/dev/ttyUSB0', 9600)

while(1):

        #for example if you want to send tha data a, b , c
        a=100
        b=200
        c=300
        output = "X{0:d}Y{1:d}Z{2:d}".format(a,b,c)
        ser.write(output)
```

4b. Arduino program

```
int x,y,z;

void setup()
{
Serial.begin(9600);
}

void loop()
{
        if (Serial.available() > 0)
   {
        if (Serial.read() == 'X')
      {
        x = Serial.parseInt();
         if (Serial.read() == 'Y')
         {
                y = Serial.parseInt();
                if (Serial.read() == 'Z')
                {
                     z = Serial.parseInt();
                }
            }
         }

// Put your Code Here

          while (Serial.available() > 0)
     {
          Serial.read();
      }

     }

}
```

The above programs are just examples. You may need to send information of different data types.

Extended Task :
Write an open cv program to detect wether an object of a given color is to the right or left of a reference point and correspondingly blink One LED for Right and two LEDs for Left. Pass information from python to arduino to achieve this.