Opencv code

import numpy as np

import cv2

maiorArea = 0

import time

import math

import serial

cap = cv2.VideoCapture(0)

ser=serial.Serial("COM9")

if not(cap.isOpened()):

cap.open()

while(cap.isOpened()):

ret, frame = cap.read()

cv2.imshow("Webcam", frame)

bkg=frame.copy()

fundo = cv2.GaussianBlur(bkg,(3,3),0)

print("OK")

if cv2.waitKey(1) == 27:

cv2.destroyWindow("Webcam")

break

while True:

t1=time.time()

ret, imagem = cap.read()

mascara=imagem.copy()

cinza=imagem.copy()

#cv2.imshow("Webcam", imagem)

imagem = cv2.GaussianBlur(imagem,(3,3),0)

cv2.absdiff(imagem,fundo,mascara)

gray = cv2.cvtColor(mascara, cv2.COLOR\_BGR2GRAY)

ret,thresh1 = cv2.threshold(gray,0,255,cv2.THRESH\_BINARY+cv2.THRESH\_OTSU)

kernel = np.ones((3,3),np.uint8)

dilated = cv2.dilate(thresh1,kernel,iterations = 18)

cinza = cv2.erode(dilated,kernel,iterations = 10)

\_,contours,heir=cv2.findContours(cinza,cv2.RETR\_TREE,cv2.CHAIN\_APPROX\_NONE)

imagem=cv2.drawContours(imagem,contours,0,(0,255,0),2)

cv2.imshow("imagem",imagem)

cv2.imshow("Mascara", mascara)

cv2.imshow("Cinza", cinza)

cv2.imshow("Webcam", imagem)

cv2.imshow("Dilated", thresh1)

#cv2.imshow("Fundo", dilated)

M=cv2.moments(contours[0])

cx1= M['m10']/M['m00']

cy1= M['m01']/M['m00']

print cx1,cy1

t2=time.time()

ret, imagem = cap.read()

mascara=imagem.copy()

cinza=imagem.copy()

#cv2.imshow("Webcam", imagem)

imagem = cv2.GaussianBlur(imagem,(3,3),0)

cv2.absdiff(imagem,fundo,mascara)

gray = cv2.cvtColor(mascara, cv2.COLOR\_BGR2GRAY)

ret,thresh1 = cv2.threshold(gray,0,255,cv2.THRESH\_BINARY+cv2.THRESH\_OTSU)

kernel = np.ones((3,3),np.uint8)

dilated = cv2.dilate(thresh1,kernel,iterations = 18)

cinza = cv2.erode(dilated,kernel,iterations = 10)

\_,contours,heir=cv2.findContours(cinza,cv2.RETR\_TREE,cv2.CHAIN\_APPROX\_NONE)

imagem=cv2.drawContours(imagem,contours,0,(0,255,0),2)

cv2.imshow("imagem",imagem)

cv2.imshow("Mascara", mascara)

cv2.imshow("Cinza", cinza)

cv2.imshow("Webcam", imagem)

cv2.imshow("Dilated", thresh1)

#cv2.imshow("Fundo", dilated)

M=cv2.moments(contours[0])

cx2= M['m10']/M['m00']

cy2= M['m01']/M['m00']

print cx2,cy2

print t2-t1

vel=math.sqrt((cx2-cx1)\*(cx2-cx1)+(cy2-cy1)\*(cy2-cy1))

vel=vel/(t2-t1)

print vel

print "velocity is"

print vel\*0.0022292 ,"m/s"

# 480pixel at 1m distance gives a range of 107cm=> 1pi=0.22292cm

if vel <100:

ser.write("l")

elif vel>100 and vel<400:

ser.write("m")

else:

ser.write("h")

if cv2.waitKey(1) & 0xFF == ord('q'):

break

# Release everything if job is finished

cap.release()

cv2.destroyAllWindows()

Arduino code

#include<avr/io.h>

void usart\_int(void)

{

UCSR0B=(1<< RXEN0)|(1<< TXEN0);//TRANSMIT AND RECEIVE ENABLE

UCSR0C=(1<<UCSZ01)|(1<<UCSZ00);//ASYNCHRONOUS, 8 BIT TRANSFER

UBRR0L= 0x67 ; //BAUD RATE 9600

}

void usart\_send(int ch )

{

while(UCSR0A!=(UCSR0A|(1<<UDRE0)));//waiting for UDRE to become high

UDR0= ch;

if (ch=='l')

PORTB=0b00000001;

else if (ch=='m')

PORTB=0b00000011;

else if( ch=='h')

PORTB=0b00000111;

}

int main(void)

{

DDRB=0xFF;

PORTB=0x00;

usart\_int();

unsigned char ch;

while(1)

{

while(UCSR0A!= (UCSR0A|(1<<RXC0)));

ch=UDR0;

usart\_send(ch);

}

}

s