

**树莓派安装opencv打开摄像头实现实时传输**

安装OpenCV库之前，我们需要更新一下软件源：

1、 sudo apt-get update

然后开始安装OpenCV库：

2、sudo apt-get install libcv-dev

**#include <opencv2/highgui/highgui.hpp>**

**#include <opencv2/core/core.hpp>**

**using namespace cv;**

**int main() { VideoCapture cap(0); //打开摄像头**

**// 如果要打开本地视频采用 VideoCapture cap("\*\*\*.avi"); if(!cap.isOpened()) return -1; //检测一下摄像头是否打开**

**Mat frame;**

**while(1){ cap>>frame; //读取当前帧**

**// 此处可添加图像处理算法，对图像进行处理，当然了，我们可以不做任何操作，只打开一下摄像头**

**imshow("Xiaorun OpenCV CAM", frame); //显示一下**

**if(waitKey(20) >=0) break; // 等待按键，跳出循环**

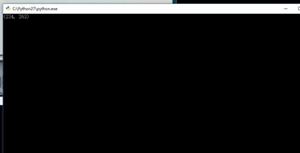
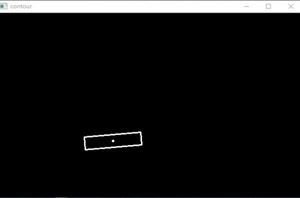
**}**

**return 0;**

**}**

**OpenCV获取零件位置**

**具体思路如下：**



* 获取图片
* 提取黑色的区域
* 将黑色区域进行二值化处理
* 获取中心区域轮廓及坐标
* 获取坐标中心点位置
* 绘制矩形及中心点 输出图片，黑色区域图，二值化图，轮廓及中心点图片。

# coding:UTF-8

import cv2

import numpy as np

class Findposition:

def \_\_init\_\_(self,path):

#获取图片

self.img=cv2.imread(path)

self.gray=cv2.cvtColor(self.img,cv2.COLOR\_BGR2GRAY)

self.hsv=cv2.cvtColor(self.img,cv2.COLOR\_BGR2HSV)

#提取黑色的区域

def Get\_black(self):

#get black area

low\_black=np.array([0,0,0])

high\_black=np.array([100,100,100])

mask=cv2.inRange(self.img,low\_black,high\_black)

black=cv2.bitwise\_and(self.hsv,self.hsv,mask=mask)

return black

#将黑色区域进行二值化处理

def Get\_contour(self):

#change to gray

black=self.Get\_black()

black\_gray=cv2.cvtColor(black,cv2.COLOR\_HSV2BGR)

black\_gray=cv2.cvtColor(black\_gray,cv2.COLOR\_BGR2GRAY)

#binaryzation

\_, thresh=cv2.threshold(black\_gray,10,255,cv2.THRESH\_BINARY+cv2.THRESH\_OTSU)

img\_morph=cv2.morphologyEx(thresh,cv2.MORPH\_OPEN,(3,3))

cv2.erode(img\_morph,(3,3),img\_morph,iterations=2)

cv2.dilate(img\_morph,(3,3),img\_morph,iterations=2)

return img\_morph

#获取中心区域轮廓及坐标

def Find\_contour(self,img):

img\_cp=img.copy()

cnts,\_=cv2.findContours(img\_cp,cv2.RETR\_TREE,cv2.CHAIN\_APPROX\_SIMPLE)

cnt\_second=sorted(cnts,key=cv2.contourArea,reverse=True)[1]

box =cv2.minAreaRect(cnt\_second)

return np.int0(cv2.cv.BoxPoints(box))

#绘制轮廓

def Draw\_contour(self,points):

mask=np.zeros(self.gray.shape,np.uint8)

cv2.drawContours(mask,[points],-1,255,2)

return mask

#获取中心位置

def Get\_center(self,points):

p1x,p1y=points[0,0],points[0,1]

p3x,p3y=points[2,0],points[2,1]

center\_x,center\_y=(p1x+p3x)/2,(p1y+p3y)/2

center=(center\_x,center\_y)

return center

#绘制中心点

def Draw\_center(self,center,mask):

cv2.circle( mask,center,1,(255,255,255),2)

return mask

#主函数

def main\_process(self):

morph=self.Get\_contour()

black=self.Get\_black()

points=self.Find\_contour(morph)

mask=self.Draw\_contour(points)

center=self.Get\_center(points)

draw\_center=self.Draw\_center(center,mask)

center\_x,center\_y=self.Get\_center(points)

print(center\_x,center\_y)

cv2.imshow('black',black)

cv2.imshow('morph',morph)

cv2.imshow('img',self.img)

cv2.imshow('contour',draw\_center)

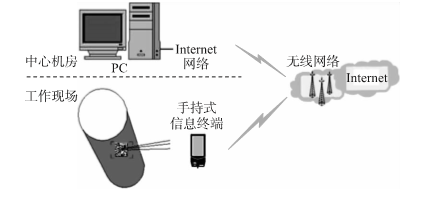
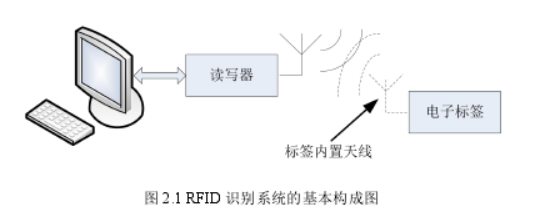
cv2.waitKey(0)

if \_\_name\_\_== '\_\_main\_\_' :

path='C:\Users\KaiyuanCao\Desktop\sample.jpg'

d = Findposition(path)

d.main\_process()



**线程实现rfid多标签读取**

private void btnStop\_Click(object sender, EventArgs e)

{

// MessageBox.Show(tread1.ThreadState.ToString());

tread1.Abort();

//MessageBox.Show(tread1.ThreadState.ToString());

}

/// <summary>

/// 循环读卡

/// </summary>

public void ReadCard()

{

while (true)

{

lock (m\_TheadLocker)

{

SingelTag();

}

//每2秒检查一次

Thread.Sleep(2000);

}

}

//读取单个标签

void SingelTag()

{

if (ValideHandel)

{

byte[] tag = new byte[12];

if (EPCSDKHelper.IdentifyUploadedSingleTag(\_handel, tag, null, null))

{

string code = TextEncoder.ByteArrayToHexString(tag);

listBox1.Items.Add("EPC:" + code);

}

}

}

private void btnclear\_Click(object sender, EventArgs e)

{

listBox1.Items.Clear();

}

}

}