Universitatea Politehnica Timişoara
Piața Victoriei nr. 2
RO 300006 - Timişoara
Tel: +40 256 403000
Fax: +40 256 403021
rector@rectorat.upt.ro
www.upt.ro

Sistem de detectare facială

Universitatea *Politehnica* Timisoara

Student:
Filip Albin Rus
Coordonator:
Șl. Dr. Ing. Cristian Moldovan

email:

2017

Cuprins

- Despre sistem
- Componentele sistemului
- Funcționare
- Etape de prelucrare
- Simularea
- Învățarea unui nou model

Despre sistem

Universitatea Politehnica Timișoara

Sistem automat de detectare a feței și a ochiilor folosit folosind conceptul de Computer Vision.

Componentele sistemului

- Hardware
  - RaspberryPi
  - Cameră foto
  - Suport cameră
  - PC(Calculator personal)
- Software
  - Program de prelucrare și analizare a imaginilor

Componentele sistemului

Program de prelucrare și analizare

- Limbaj de programare: Python
- Bibliotecă de funcții: OpenCV,PiCamera

Funcționare

Etape de prelucrare

- Creare imagine OpenCV
- Încărcare clasificatoare Haar
- Transformare imagine în tonuri de gri
- Căutare elemente specifice
- Încadrarea elementelor găsite
- Salvarea imaginii finale

Simulare

eface.py - /home/pi/eface.py (2.7.9)

File Edit Format Run Options Windows Help

```
import io
import picamera
import cv2
import numpy

#Create a memory stream so photos doesn't need to be saved in a file
stream = io.BytesIO()

#Get the picture (low resolution, so it should be quite fast)
#Here you can also specify other parameters (e.g.:rotate the image)
with picamera.PiCamera() as camera:
    camera.resolution = (320,240)
    camera.capture(stream, format='jpeg')

#Convert the picture into a numpy array
buff = numpy.fromstring(stream.getvalue(), dtype=numpy.uint8)

#Now creates an OpenCV image
image = cv2.imdecode(buff, 1)

#Load a cascade file for detecting faces
face_cascade = cv2.CascadeClassifier('/usr/share/opencv/haarcascades/haarcascade
eye_cascade = cv2.CascadeClassifier('/usr/share/opencv/haarcascades/haarcascade_

#Convert to grayscale
gray = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)

#Look for faces in the image using the loaded cascade file
faces = face_cascade.detectMultiScale(gray, 1.1, 5)

print "Found "+str(len(faces))+" face(s)"

#Draw a rectangle around every found face
for (x,y,w,h) in faces:
    cv2.rectangle(image,(x,y),(x+w,y+h),(255,255,0),2)
    roi_gray = gray[y:y+h,x:x+w]
    roi_color = image[y:y+h,x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),1)


#Save the result image
cv2.imwrite('Imagine.jpg',image)
```

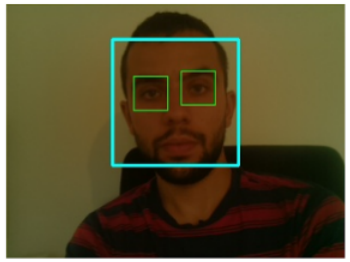
Ln: 12 Col: 32

pi@raspberrypi: ~

File Edit Tabs Help

```
pi@raspberrypi:~ $ python eface.py
Found 1 face(s)
pi@raspberrypi:~ $
```

Imagine.jpg (320x240) 100%



Învățarea unui nou model

- Probe pozitive
- Probe negative

Universitatea Politehnica Timişoara
Piața Victoriei nr. 2
RO 300006 - Timişoara
Tel: +40 256 403000
Fax: +40 256 403021
rector@rectorat.upt.ro
www.upt.ro

Sistem de detectare facială

Universitatea *Politehnica* Timisoara

Student:
Filip Albin Rus
Coordonator:
Șl. Dr. Ing. Cristian Moldovan

email:

2017