TP1
Getting started
Video sur Raspberry Pi

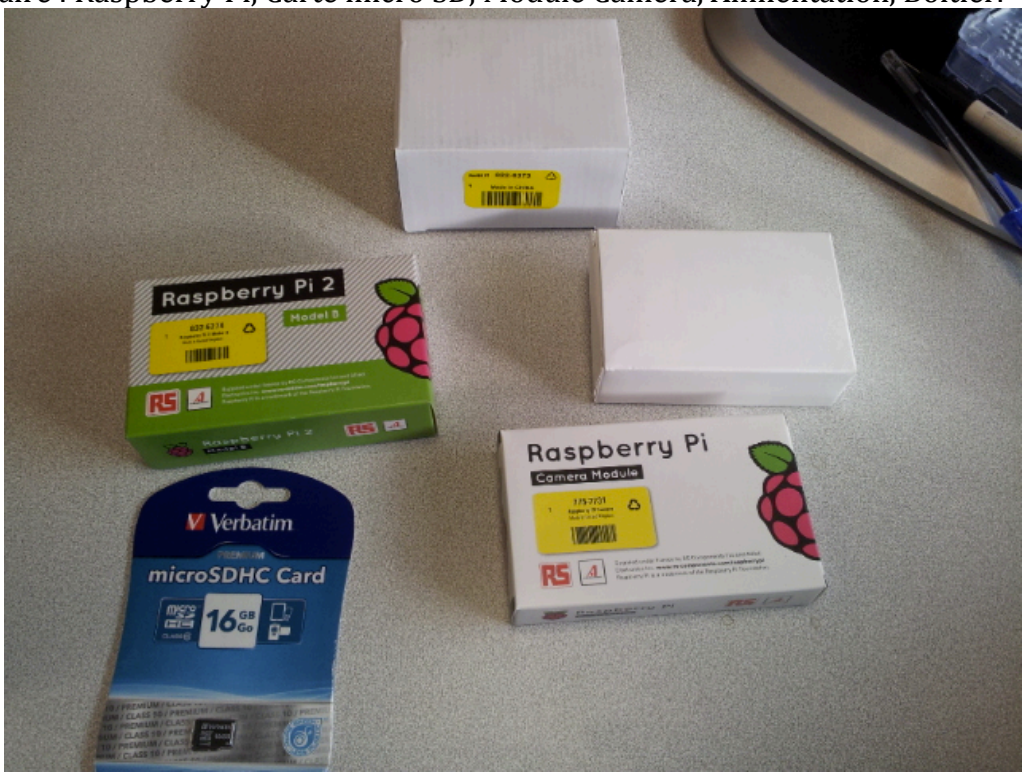
Romain Raveaux

Objectif

Assembler votre Raspberry Pi (RPI) et intégrer le capteur caméra
Environnent de développement
Manipulations de vidéo

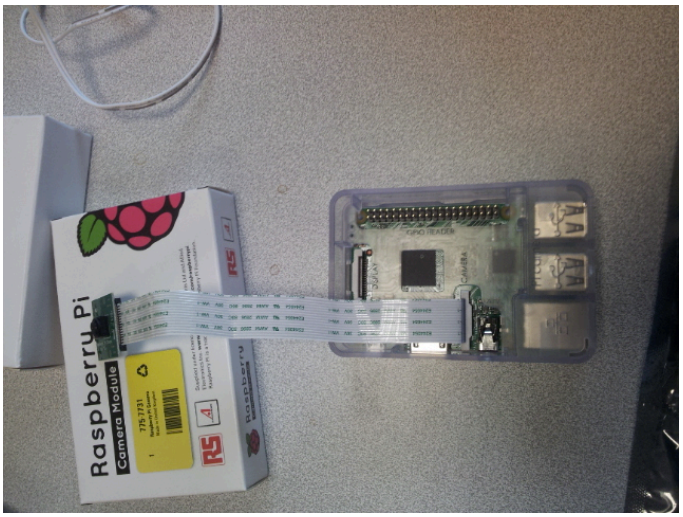
Matériel et Montage

Inventaire : Raspberry Pi, Carte micro SD, Module Caméra, Alimentation, Boitier.



Mettre le module Camera sur la Raspberry Pi. Voir la vidéo :
https://www.raspberrypi.org/help/camera-module-setup/

Mettre la RPI dans le boitier.



Logiciel

Nous allons utiliser le système d'exploitation Raspbian.
Lors des TP nous aurons aussi besoin de bibliothèques et de certains logiciels.
Demander à votre chargé de TP de mettre sur votre carte SD l'image du système d'exploitation.
Mettre la carte SD dans la RPI

Architecture réseau

Brancher la RPI à votre ordinateur via un câble Ethernet.
L'adresse IP de la RPI est la suivante : 192.168.1.253/24
Brancher la RPI sur le secteur
Connectez vous en SSH sur votre RPI :
ssh –X pi@192.168.1.253
Mot de passe : raspberry

Si votre RPI est sur un réseau avec plusieurs RPI, il faut alors changer l'adresse IP afin que cette dernière soit unique.

Environnement de développement

Une fois connecté, taper les commandes suivantes :
cd pythonwork
workon cv
./runidle.py

Bibliothèque

http://docs.opencv.org/2.4/

Lire le flux de la camera : PiCam

```
# import the necessary packages
from picamera.array import PiRGBArray
from picamera import PiCamera
import cv2
import time


# initialize the camera and grab a reference to the raw camera capture

camera = PiCamera()
camera.vflip = True
camera.hflip = True
#camera.brightness = 60
camera.resolution = (640, 480)
camera.framerate = 32
rawCapture = PiRGBArray(camera, size=(640, 480))
# allow the camera to warmup
time.sleep(0.1)


# capture frames from the camera
for frame in camera.capture_continuous(rawCapture, format="bgr",
use_video_port=True):

    # grab the raw NumPy array representing the image, then initialize the timestamp
    # and occupied/unoccupied text
    image = frame.array
    image.flags.writeable = True

    cv2.imshow("Frame", image)

    # clear the stream in preparation for the next frame
    rawCapture.truncate(0)

    # if the `q` key was pressed, break from the loop
    key = cv2.waitKey(1) & 0xFF
    if key == ord("q"):
        break
camera.close()
```

Lire une vidéo

```
import numpy as np
import cv2

cap = cv2.VideoCapture('vtest.avi')

while(cap.isOpened()):
   ret, frame = cap.read()

   gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

   cv2.imshow('frame',gray)
   if cv2.waitKey(1) & 0xFF == ord('q'):
      break

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

Lire/Ecrire une image

```
import cv2
im = cv2.imread('3x2.png')
cv2.imshow("Frame", im)
print(im)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
# pour le niveau de Bleu
print im[1][2][0]
# pour le niveau de Vert
print im[1][2][1]
# pour le niveau de Rouge
print im[1][2][2]
```

```
import cv2
im = cv2.imread('3x2.png')

im[1][1] = [0, 255, 255]

# sauvegarde du resultat
cv2.imwrite('resultat.png',im)
```

Ecrire une chaine de caractères dans une image

```
cv2.putText(image, "fps=%s" % (fps),
            (10, 75), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255))
```

Calculer et afficher le nombre de frame par seconde

```
import time

#permet de récupérer le temps à un instant t
t1=time.time()
```

Traitement utile

Redimensionner une image

```
gray = cv2.resize(gray, (0,0), fx=0.5, fy=0.5)
ou
gray = cv2.resize(gray, (64,64))
```

Créer une matrice

Avec des zéros

```
res=numpy.zeros((w, h))
```

Avec des uns

```
res=numpy.ones((w, h))
```

De la matrice (float) à l'image (entier)

```
im=numpy.uint8(matrice)
```