Promotion 2016-2019 | Imérir

Documentation Générale

Projet Vision

Fournier Yannick
Breton Valentin
de Claverie Guillaume

26/01/2017
Version 1.0

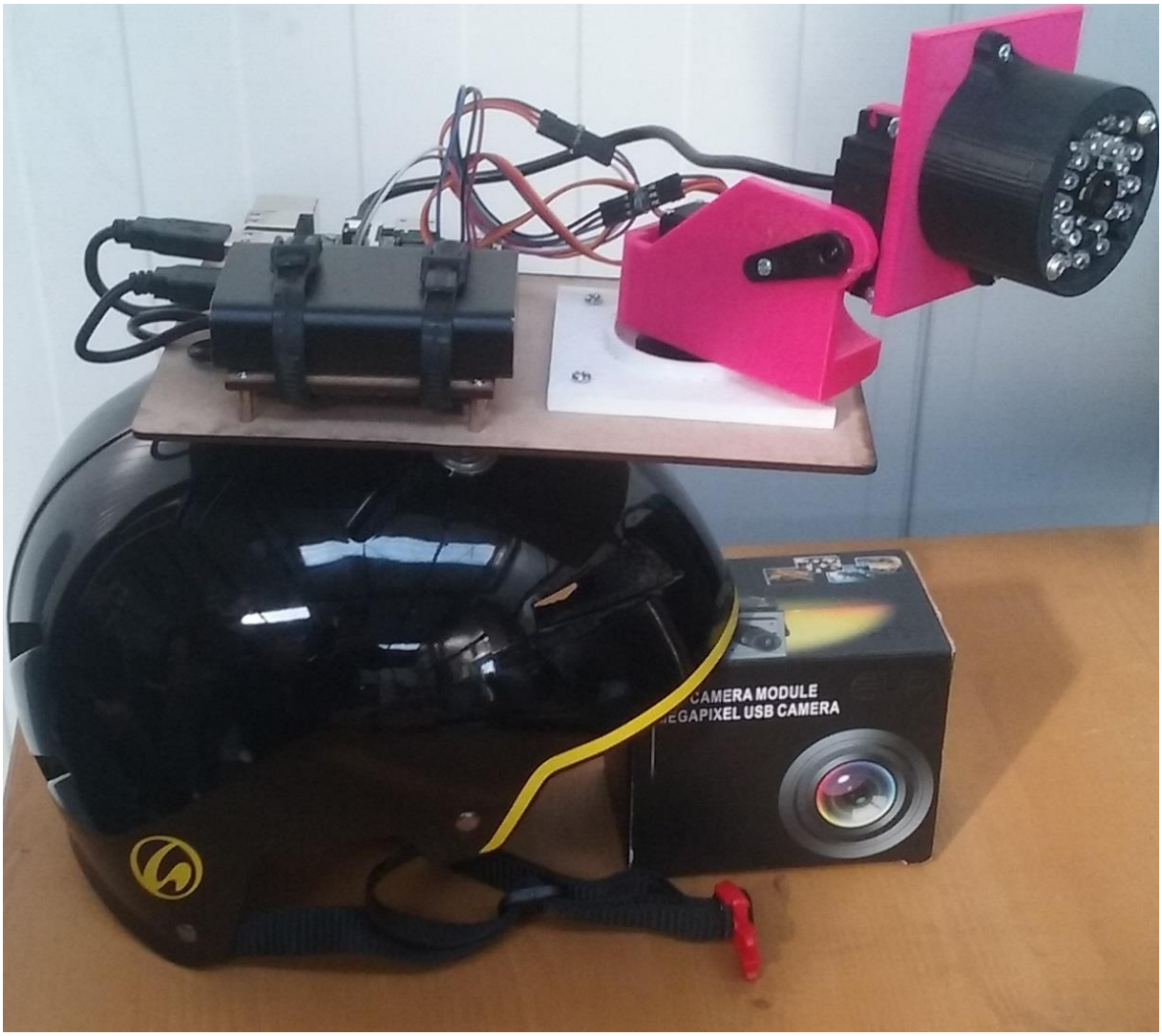
Historique des versions

Version	date	Modifications	Pages modifiées (1)
1.0	26/01/2017	Création du document	toutes

(1) si modification globale, indiquer « toutes »

Sommaire

Projet Vision



Par Fournier Yannick, Breton Valentin, de Claverie Guillaume (1ere année Promotion Bourgine 2016-2019)

I. Présentation du projet

Le projet Vision consiste à la mise en place d'un support caméra, servomoteurs sur casque ou un robot afin de récupérer l'image de la caméra sur un smartphone. Le retour des données du gyroscope du téléphone permettront le contrôle des servomoteurs et donc une plus grande immersion avec les Google Cardboard.

II. Remerciements

Nous remercions, monsieur Navarro Joris, directeur du FabLab de Perpignan, monsieur Denet Laurent directeur de l'IMERIR de Perpignan, monsieur LeMenaheze Claude et les enseignants de l'IMERIR sans qui ce projet n'aurai ni pu avoir lieu ni pu aboutir. Nous vous remercions de l'aide et des ressources indispensables qui nous as été accordées durant tout la période du projet.

III. Ressources utilisées

- Raspberry Pi 3
- Carte SD 8Go
- Ecran d'ordinateur muni d'un port HDMI ou le rétroprojecteur de la salle
- Camera USB
- Imprimante 3D
- Casque
- Google CardBoard
- Servomoteurs
- Batteries Externes (une pour le smartphone et l'autre pour l'alimentation de la Raspberry)

Autres : ordinateurs, smartphones, clavier et souris pour la raspberry, visserie…

IV. Partie Programmation

1. Installation du programme de base pour faire fonctionner la raspberry pi 3

- Télécharger Raspbian sur le site officiel de Raspberry
- Formater si besoin le carte SD avec SDFormatter
- Mettre les dossiers Raspbian téléchargés sur la carte SD
- Démarrer la carte Raspberry, l'installation doit se lancer d'elle-même

A savoir : L'utilisateur par défaut est : « pi »**,** le mot de passe est : « raspberry »

2. (Facultatif) : Le SSH

Si vous ne voulez pas utiliser directement la raspberry ce qui oblige de trouver un écran, un clavier et une souris, le SSH est là. Le SSH permet de se connecter à la raspberry par le wifi et donc contrôler la carte sur son pc.

1ère Méthode :

Tutoriel : http://the-raspberry.com/ssh-raspberry-pi

Néanmoins si ce lien n'est plus valide voici un résumé :

- Sur la raspberry pi ouvrir un terminal
- Saisir la commande "sudo raspi-config" puis valider
- Sélectionner l'option "ssh"
- Sélectionner le choix "Enable"
- Confirmer le message par "Ok"
- Saisir la commande "sudo ifconfig" et relever l'adresse IP de la raspberry

2ème Méthode :

- Saisir la commande "sudo /etc/init.d/ssh start"

Néanmoins cette méthode a pour défaut de devoir être effectuée à chaque démarrage de la raspberry.

Pour les deux méthodes :

Sous Windows télécharger PUTTY: http://www.putty.org/

- Lancer Putty
- Dans la case « Host Name (or IP address)» saisir l'IP de la raspberry
- Rentrer vos identifiant (rappel : utilisateur = pi et mot de passe = raspberry)

Vous êtes maintenant connectés à votre raspberry. A présent il ne vous suffit plus que d'alimenter la carte, vous pouvez donc ranger le clavier, la souris et l'affichage.

3. La Caméra USB

Pour obtenir un flux vidéo et le diffuser sur un réseau local suivre les instructions ici : http://www.pobot.org/Vision-par-webcam-avec-une.html?lang=fr

Encore une fois si ce lien ne fonctionne plus suivre ces étapes :

sudo apt-get update
sudo apt-get upgrade
sudo apt-get install v4l-utils v4l-conf

- Installer le logiciel "luvcview" pour avoir un streaming (lancer avec la commande "luvcview" start)
- Installer le logiciel "uvccapture" pour avoir un streaming (lancer avec la commande "uvccapture" start)

Utiliser un serveur pour voir le streaming en réseau local :

1. pi@raspberrypi ~ $ **sudo apt-get install** subversion
2. pi@raspberrypi ~ $ **mkdir** webcams
3. pi@raspberrypi ~ $ **cd** webcams
4. pi@raspberrypi ~/webcams $ sudo svn co https://svn.code.sf.net/p/mjpg-streamer/code mjpg-streamer
5. pi@raspberrypi ~/webcams $ **sudo apt-get install** libjpeg8-dev
6. pi@raspberrypi ~/webcams $ **sudo apt-get install** imagemagick
7. pi@raspberrypi ~/webcams $ **cd** mjpg-streamer/mjpg-streamer
8. pi@raspberrypi ~/webcams/mjpg-streamer/mjpg-streamer $ **make**
9. pi@raspberrypi ~/./mjpg_streamer -i "./input_uvc.so -y -r 320x240 -f 27" -o "./output_http.so -w ./www"

A savoir : le « –f 27 » sur la dernière ligne de code correspond à vos fps, ici on aura donc 27 fps.

4. Création d'une application pour récupérer le flux vidéo

Le flux vidéo étant sur le réseau local cela a facilité sa récupération.

On utilise MIT app inventor qui est le logiciel qui permet la création de l'application : https://knowledge.parcours-performance.com/application-android-afficher-page-web-dun-raspberry-pi/

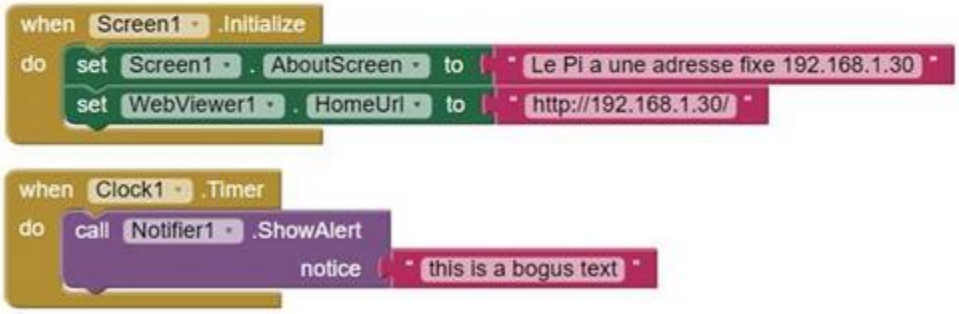
Le résumé de l'explication :

Solution avec webviewer

En mode « designer » :

- screen1 est en `sizing` « fixed », `ScreenOrientation` en « sensor » ;
- Dans les « non visible components », on trouve trois composants :
  - webviewer, qui visualise une page web ;
  - notifier1 réglé avec `BackgroundColor` et `TextColor` sur « none » ;
  - clock1 avec `TimerAlwaysFires` et `TimerEnabled` cochés, `TimerInterval` sur 5000 (ce sont des millisecondes en principe, donc ici 5 secondes)

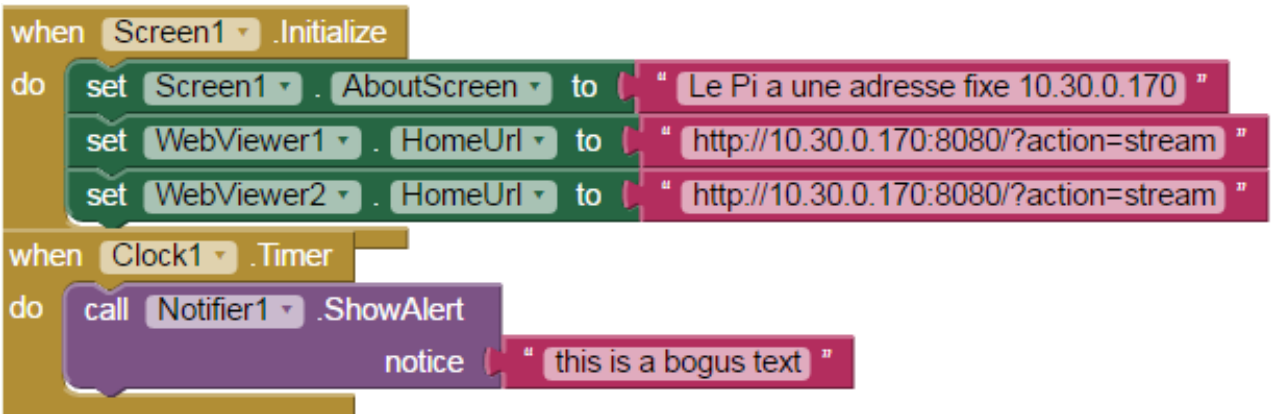
En mode « blocks » :



Voici le fichier en .aia utilisable sur MIT App Inventor (à renommer en .aia au lieu de .zip) : raspberry_pi_dashboard_webviewer

Dans notre solution, nous avons rajoutés une ligne de code (« set WebViewer2.HomeUrl to… ») afin d'obtenir deux flux vidéo sur le smartphone ce qui nous a permis après de nombreux essais de dimensionnement d'avoir une meilleure image et un meilleur rendu avec le Google Cardboard. C'est la méthode de l'image en Side By Side (SBS) utilisée en réalité virtuelle.

Ce qui nous donne donc :



A savoir : Une fois le code créée vous pouvez générer un QRcode qu'il vous suffira de scanner afin de télécharger votre application sur votre smartphone.

5. Contrôle des servomoteurs avec le gyroscope du smartphone

Tout d'abord deux applications :

La première, « Sensor Kinetics » : cette application vous permettra d'être sûr que votre téléphone dispose d'un gyroscope. Si vous avez bien un retour de donnée pour le gyroscope alors vous pouvez désinstaller l'application et continuer sinon regarder sur le voisin à partir de maintenant.

La deuxième, « Wireless IMU » est indispensable. Elle complète le code ci-dessous et permet de lire les données du gyroscope de votre téléphone et de les transmettre à la raspberry. Pour faire fonctionner l'application il suffit de rentrer l'adresse IP de la raspberry dans la case « Target IP Address » et de mettre en « on ».

A savoir : Cocher dans Wireless IMU la case « Run in background » et lancé l'application vidéo créée à l'aide de MIT app inventor au premier plan.

Maintenant le code (en python) :

```
#Fait par Yannick Fournier
#1er année IMERIR
#Promotion Bourgine 2016-2019

import struct
import socket, traceback
import RPi.GPIO as GPIO
from math import *
import sys

#init for gyro
host = ''
port = 5555
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
s.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
s.bind((host, port))
#initialisation des servos
GPIO.setmode(GPIO.BCM)        #Choix du système de numérotation (BCM ou BOARD)
GPIO.setwarnings(False)
GPIO.setup(18, GPIO.OUT)    #On prépare le pin 18 de la Raspberry Pi 3 en sortie
GPIO.setup(13, GPIO.OUT)    #On prépare le pin 13 de la Raspberry Pi 3 en sortie
Frequence = 50
Servo1x = GPIO.PWM(18, Frequence)   #Création d'un objet Servo1xpour le PWM du port 18 à "Frequence" Hertz
Servo2y = GPIO.PWM(13, Frequence)   #Création d'un objet Servo2ypour le PWM du port 18 à "Frequence" Hertz
positionInitiale = 6.5              #Gros servo  = 6.5      Petit servo 9
Servo1x.start(positionInitiale)     #On initialise Servo1x à sa position centrale
Servo2y.start(positionInitiale)     #On initialise Servo2y à sa position centrale

#Déclaration des constantes et variables
rotationX = 0
rotationY = 0
tmpx = positionInitiale
tmpy = positionInitiale
in_min = -100
in_max = 100
out_min = 2     #Gros servo  = 2      Petit servo = 1
out_max = 12    #Gros servo  = 11      Petit servo = 12
rotationMax = 10



print("Start listening")
message = s.recvfrom(8192)  #Recupere les donnees du portable
message = s.recvfrom(8192)
#print(message)

while 1:
    try:
        message = s.recvfrom(8192)
        #print(message)

        coordonnees = message[0]
        messageSplitter = coordonnees.split(b',')
        #print(messageSplitter[6])
        #print(messageSplitter[7])
```

```
if (len(messageSplitter) > 8):
    rotationX = float(messageSplitter[6].strip())   #On recupere les coordonnees en X du gyroscope
    rotationY = float(messageSplitter[7].strip())   #On recupere les coordonnees en Y du gyroscope
    #print("rotationX : ", rotationX)
    #print("rotationY : ", rotationY)

    #Si on tourne trop vite le portable
    if (rotationX < -rotationMax or rotationX > rotationMax or rotationY < -rotationMax or rotationY > rotationMax):
        #print("continue : ", rotationX)
        #print("continue : ", rotationY)
        continue
"Mise à l'échelle des donnes du gyroscope entre out_min et out_max"
x = ((rotationX - in_min)*(out_max - out_min) / (in_max - in_min)) + out_min
y = ((rotationY - in_min)*(out_max - out_min) / (in_max - in_min)) + out_min
#print("rotationX : ", rotationX)
#print("rotationX : ", rotationY)

#print("x : ", x)
#print("y : ", y)

#print(((x - (out_max + out_min) / 2)) * 2)
#print(((y - (out_max + out_min) / 2)) * 2)
tmpx += ((x - (out_max + out_min) / 2)) * 2
tmpy += ((y - (out_max + out_min) / 2)) * 2
#print("tmpx : ", tmpx, "tmpy : ", tmpy)

"Conditions aux limites"
if (tmpx > out_max):
    tmpx = out_max
if (tmpx < out_min):
    tmpx = out_min
if (tmpy > out_max):
    tmpy = out_max
if (tmpy < out_min):
    tmpy = out_min
#print("tmpx : ", tmpx, "tmpy : ", tmpy)
        #Mise à jour de la position des servo moteurs
        Servo1x.ChangeDutyCycle(tmpx)
        Servo2y.ChangeDutyCycle(tmpy)


except (KeyboardInterrupt, SystemExit):
    raise
except:
    traceback.print_exc()
```

V. Partie Conception

On utilisera le logiciel SketchUp qui est compatible avec l'imprimante 3D et qui a l'avantage d'être gratuit. De plus SketchUp est plus simple d'installation et d'utilisation que SolidWorks.

- Télécharger SketchUp

- Lancer SketchUp

- Créer votre pièce

Nous sommes partie d'une pièce qui existait déjà pour la création du support (site Thingiverse : http://www.thingiverse.com/thing:887075).

La pièce a ensuite été entièrement repensée et modifiée pour correspondre aux dimensions voulues.

A savoir :

- Le format de base de SketchUp est ".skb "

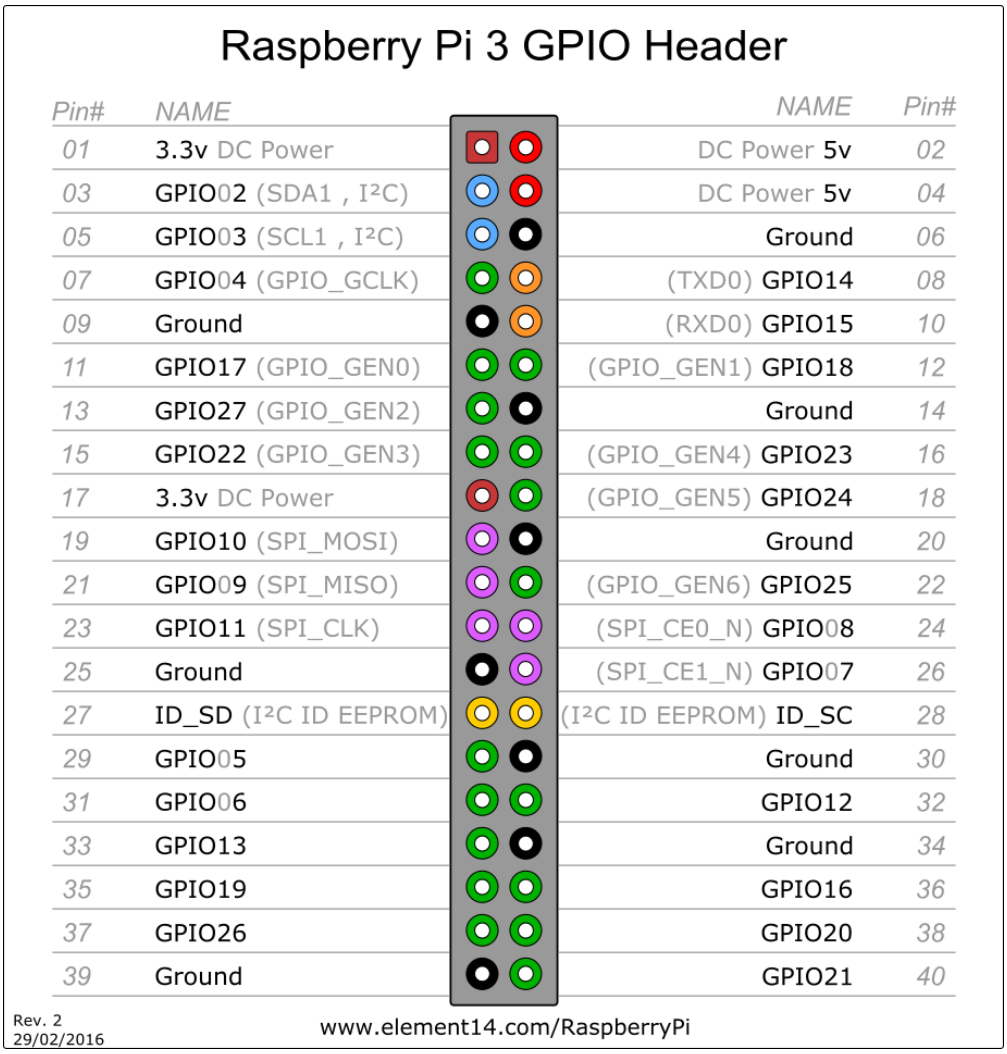
- Le format pour imprimer en 3D est " .stl "

Il faut donc pour passer du format ".skb" au format ".stl"

- Onglet Fenêtre dans Sketchup
- Extension Warehouse
- ScketchUp STL dans la barre recherche
- Télécharger ScketchUp STL

VI. Partie Connexion

Au niveau de la connectique :



Les servomoteurs sont alimentés en 5V :

- Pour le 1er servomoteur : Pin# 02, 06 et 12
- Pour le 2ème servomoteur : Pin# 04, 34 et 33

A savoir : La raspberry pi est alimentée via une batterie externe donc directement connectée sur son port d'alimentation est non sur les bornes.

VII. Annexes

Camera USB :

- Appareil photo USB 720p, 3.6mm lens,24 LED Light avec IR CUT, HD Webcam avec 1M usb Cable
- La technologie de compression H.264 sur caméra USB pour des images nettes et des couleurs précises
- Super petit aperçu et le profil mince pour l'embarqué
- Haut cadre à tarifs livrer 30 fps en 720p pour le temps réel vue vidéo à la maison ou dispositif à l'intérieur
- Rentable - l'utilisateur n'a pas besoin d'DVR. Seuls quelques caméras USB et un ordinateur peuvent faire un système de surveillance afin de maintenir la sécurité de votre maison

Servomoteurs :