Attraction mutante QIFF

(En cours / page créée le 31 janvier 2022)

Boîte interactive avec webcam et écran pour le QIFF

Galerie en ligne : https://lpl-qiff.imgbb.com/

Mise en boîte

en cours!

Matériel

Lenovo tiny M73

client léger Lenovo M73 (Pentium CPU G3320T @ 2.60 GHz (2 cores) RAM 4GB)

Quelle version d'openGL ?

Pour connaître la version d'openGL utilisable

```
su -
apt install mesa-utils
glxinfo | grep'version'
```

renvoie

```
server glx version string: 1.4
client glx version string: 1.4
GLX version: 1.4
    Max core profile version: 4.5
    Max compat profile version: 3.0
    Max GLES1 profile version: 1.1
    Max GLES[23] profile version: 3.1
OpenGL core profile version string: 4.5 (Core Profile) Mesa 18.3.6
OpenGL core profile shading language version string: 4.50
OpenGL version string: 3.0 Mesa 18.3.6
OpenGL shading language version string: 1.30
OpenGL ES profile version string: OpenGL ES 3.1 Mesa 18.3.6
OpenGL ES profile shading language version string: OpenGL ES GLSL ES 3.10
```

Autre méthode :

```
glxgears -info|grep GL_VERSION
```

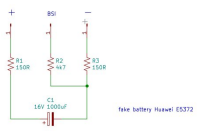
Renvoie

```
GL_VERSION    = 3.0 Mesa 18.3.6
```

Utilisation d'un modem 4G Huawei E5372

Le modem est là mais il n'a plus de batterie, en attendant d'en recevoir une nouvelle (disponible chez Reichelt), on tente de fabriquer une «fake battery» pour permettre de le faire fonctionner par le port USB.

Le schéma est refait d'après des vidéos de bricoleurs indiens captées sur les internet.





TEST EN COURS (Laisser le temps au condensateur de se charger!)
> Ca ne fonctionne pas, le condensateur ne se charge pas, peut-être est il défectueux ?
Test inabouti, on utilise plutôt la nouvelle batterie!

Adresse du hotspot (à confirmer) 192.168.8.1

Ecrans

1024×768

Webcam PS-Eye

Modèle indiqué (sous le pied) : PS-Eye SLEH-00448
Identifiant de la caméra avec `lsusb` :

```
Bus 002 Device 010: ID 1415:2000 Nam Tai E&E Products Ltd. or OmniVision Technologies, Inc. Sony Playstation Eye
```

Démontage

Démontage de la caméra d'après Hacking Sony PS3-Eye, en version courte :

- faire sauter les 4 caches de vis à l'arrière de la cam, dévisser
- enlever le capot plastique arrière en faisant levier avec un tournevis (il faut forcer un peu)
- dévisser les 2 vis relient la caméra au pied,
- dévisser les 5 vis qui relient la carte électronique au capot avant (3 du haut sont reliées au “cache micro”, 2 au capot avant)

On obtient




Impression d'un boîtier d'après un modèle récupéré sur thingiverse (Playstation Eye Go-Pro Mount Case)

Modifications du boitier

Sur la pièce top : des pattes de fixation sont ajoutées et des trous sont percés pour les micros
Sur la pièce bottom : les pattes de fixation sont supprimées, le passage de cable agrandi



Champ de vision et réglages

La PS-Eye a deux angles de champ de vision : 56° (point rouge à gauche) pour les applications de type chat et 75° (point bleu à droite) pour les applications qui nécessitent un grand angle (vue du corps complet, etc.). Détails sur le champ de vision de la ps-eye



(source)

Pour attraction mutante régler les objectifs à gauche (56°)

Notes d'installation en vrac

Pour un affichage basculé sur le côté: `xrandr -o right`

Installation d'Arduino

- installer (télécharger, décompresser)
- ajouter l'utilisateur au groupe :

```
su -
usermod -a -G dialout xor
```

Démarrer arduino avec :

```
/home/xor/arduino-1.8.5/arduino &
```

Penser à régler le port `/dev/ttyUSB0` et le type de carte (arduino nano) avant de téléverser!

Installation de Processing

Installer processing (télécharger, décompresser).
Installer les lib. depuis le gestionnaire de bibliothèques : video, sound, controlP5.
Pour démarrer à partir d'un terminal :

```
/home/xor/processing-4.0b2/processing &
```

Installation de la webcam

Pour une webcam sony PS-Eye SLEH-00448 `lsusb` renvoie

```
Bus 002 Device 010: ID 1415:2000 Nam Tai E&E Products Ltd. or OmniVision Technologies, Inc. Sony Playstation Eye
```

Quelles définitions/framerates sont accessibles pour cette caméra ?

```
apt install v4l-utils
v4l2-ctl -d /dev/video1 --list-formats-ext # pour connaitre les formats possibles
```

Ce qui renvoie

```
ioctl: VIDIOC_ENUM_FMT
    Type: Video Capture

    [0]: 'YUYV' (YUYV 4:2:2)
        Size: Discrete 320x240
            Interval: Discrete 0.005s (187.000 fps)
            Interval: Discrete 0.007s (150.000 fps)
            Interval: Discrete 0.007s (137.000 fps)
            Interval: Discrete 0.008s (125.000 fps)
            Interval: Discrete 0.010s (100.000 fps)
            Interval: Discrete 0.013s (75.000 fps)
            Interval: Discrete 0.017s (60.000 fps)
            Interval: Discrete 0.020s (50.000 fps)
            Interval: Discrete 0.027s (37.000 fps)
            Interval: Discrete 0.033s (30.000 fps)
        Size: Discrete 640x480
            Interval: Discrete 0.017s (60.000 fps)
```

```
            Interval: Discrete 0.020s (50.000 fps)
            Interval: Discrete 0.025s (40.000 fps)
            Interval: Discrete 0.033s (30.000 fps)
            Interval: Discrete 0.067s (15.000 fps)
```

Dans Processing, on pourra initialiser la caméra avec :

```
cam = new Capture(this, 640, 480, "USB Camera-B4.09.24.1", 30);
```

Quelles valeurs pourra t'on modifier en ligne de commande (avec v4l2-ctl) ?

```
v4l2-ctl --all # renvoie les contrôles et le type de données
v4l2-ctl -l    # renvoie les contrôles modifiables par l'utilisateurice
```

Dans le cas de la caméra qu'on utilise (gamme PS-Eye), on peut lire :

```
User Controls
                     brightness 0x00980900 (int)    : min=0 max=255 step=1 default=0 value=0 flags=slider
                       contrast 0x00980901 (int)    : min=0 max=255 step=1 default=32 value=32 flags=slider
                     saturation 0x00980902 (int)    : min=0 max=255 step=1 default=64 value=64 flags=slider
                            hue 0x00980903 (int)    : min=-90 max=90 step=1 default=0 value=0 flags=slider
        white_balance_automatic 0x0098090c (bool)   : default=1 value=1
                       exposure 0x00980911 (int)    : min=0 max=255 step=1 default=120 value=253 flags=inactive, volatile
                gain_automatic 0x00980912 (bool)   : default=1 value=1 flags=update
                          gain 0x00980913 (int)    : min=0 max=63 step=1 default=20 value=31 flags=inactive, volatile
              horizontal_flip 0x00980914 (bool)   : default=0 value=0
                vertical_flip 0x00980915 (bool)   : default=0 value=1
           power_line_frequency 0x00980918 (menu)   : min=0 max=1 default=0 value=0
                     sharpness 0x0098091b (int)    : min=0 max=63 step=1 default=0 value=0 flags=slider
Camera Controls
                  auto_exposure 0x009a0901 (menu)   : min=0 max=1 default=0 value=0 flags=update
```

Par exemple, ou pourra faire un flip horizontal avec

```
v4l2-ctl -d /dev/video -c horizontal_flip=1
```

Installation d'openCV dans processing

La version d'openCV du gestionnaire de bibliothèques est compaticle avec Processing 3 mais pas Processing 4, pour arranger ça on peut installer la version alternative de jaegonlee qui utilise openCV for java version 4.0.0 (en remplacement de la version 2.4.5)

Mais ça ne fonctionne toujours pas car une dépendance requise n'est pas installée :

```
UnsatisfiedLinkError: /home/xor/sketchbook/libraries/opencv_processing/library/linux64/libopencv_java400.so: libtesseract.so.4: Ne peut
ouvrir le fichier d'objet partagé: Aucun fichier ou dossier de ce type
(...)
```

Pour régler le problème, installer **Tesseract** (Librairie de reconnaissance des caractères “OCR”)

```
sudo apt install tesseract-ocr
```

Ça ne fonctionne pas dans mon cas (debian 9.5), probablement car la version installée par apt est la version 3.04 de tesseract-ocr ...
En revanche sur Debian 10, ça fonctionne, la version de tesseract-ocr est la 4.0.0

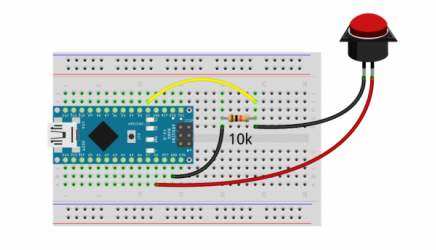
Rien à voir mais intéressant pour processing

https://github.com/cansik/deep-vision-processing
https://discourse.processing.org/t/deep-vision-machine-learning-computer-vision-for-processing-library/30118

Éléments de code

Code complet : https://github.com/lesporteslogiques/attraction_mutante

Bouton / arduino



bouton_attraction_mutante.ino (cliquer pour afficher le code)

bouton_attraction_mutante.ino

```
/*
 * Envoi d'un message série pour donner l'état d'un bouton
 * La Baleine, Quimper, Dour Ru, 20220201 / pierre@lesporteslogiques.net
 * Processing 4.0b2 / Arduino 1.8.5 @ kirin / Debian Stretch 9.5
 * + lib. OneButton v2.0.4 de Matthias Hertel https://github.com/mathertel/OneButton
 */

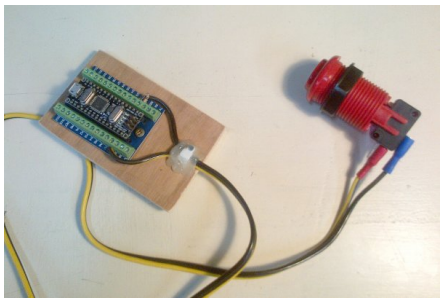
#include <OneButton.h>
#define BROCHE_BOUTON           2

OneButton bouton(BROCHE_BOUTON, false, false);

void setup() {
  pinMode(13, OUTPUT);      // sets the digital pin as output
  bouton.attachClick(clicBouton);
  Serial.begin(9600);
}

void loop() {
  bouton.tick(); // surveiller le bouton
  delay(10);
}

void clicBouton() {
  // Vérification visuelle, la LED change d'état quand on clique le bouton
  static int m = LOW;
  m = !m;
  digitalWrite(13, m);
  // Envoyer le message sur le port série
  Serial.println("1");
}
```



Configuration

Ajouter des raccourcis clavier

Rappel : système Debian 10.9 avec LXDE et openbox

Editer le fichier de configuration d'openbox :

```
nano /home/xor/.config/openbox/lxde-rc.xml
```

Dans la section `<!— Keybindings for running applications —>`, ajouter

```
<keybind key="C-A-t">
    <action name="Execute">
      <startupnotify>
        <enabled>true</enabled>
        <name>Terminator</name>
```

```
      </startupnotify>
      <command>terminator</command>
    </action>
</keybind>
```

/!\ conserver action name à Execute Autre exemple pour créer un raccourci vers un script

```
<keybind key="S-n">
  <action name="Execute">
    <startupnotify>
      <enabled>true</enabled>
      <name>RotationNormale</name>
    </startupnotify>
    <command>/home/xor/bin/ecran_rotation_normale.sh</command>
  </action>
</keybind>
```

Le script :

```
xrandr -o right
lxpanelctl restart # Compenser le bug d'affichage qui fait disparaître les fichiers de la barre des tâches
```

Le script doit être exécutable :

```
chmod +x /home/xor/bin/ecran_rotation_normale.sh
```

Puis redémarrer openbox

```
openbox --reconfigure
openbox --restart
```

Ressources sur le sujet

- http://openbox.org/wiki/Help:Bindings
- http://xahlee.info/linux/linux_lxde_add_key_shortcuts.html

Monitoring du wifi

TODO

GrandK propose :

Une solution, parmi d'autres, pour surveiller l'accessibilité réseau sur les machines.

https://www.supertechcrew.com/watchdog-keeping-system-always-running/

Attention à bien désactiver le watchdog avoir d'avoir finalié la configuration au risque de rentrer dans une boucle de reboot.

La configuration de base effectue un reboot de la machine, il est possible de faire plus subtil en relançant uniquement le réseau dans un premier temps

```
sudo apt install watchdog
sudo systemctl disable watchdog.service
sudo vi /etc/watchdog.conf # (décommenter une ligne ping et renseigner l'adresse)
sudo systemctl enable watchdog
sudo sstemctl start watchdog
```

Automatisation

Plusieurs étapes :

- démarrage de session automatique, sans login/mot de passe
- démarrer Processing en ligne de commande
- exécuter un script au démarrage
- raccourcis clavier utiles

Démarrage de session automatique, sans mot de passe

(D'après debian wiki).

En root, éditer `/etc/lightdm/lightdm.conf`

```
su -
nano /etc/lightdm/lightdm.conf
```

Modifier les entrées suivantes

```
[Seat:*]
autologin-user=indiquer-nom-utilisateur
autologin-user-timeout=0
```

NB : Pourquoi `su -` ? : su (substitute user), le tiret permet de se placer dans le répertoire de l'utilisateur auquel on se substitue, ici root, de mettre à jour les variables d'environnement, donc les chemins vers les commandes. Sans le tiret, ces variables ne sont pas mises à jour et on reste dans le même répertoire.

Démarrage de Processing en ligne de commande

Processing en ligne de commande :

```
/home/xor/processing-4.0b2/processing-java --sketch= --output= --run  # à compléter
```

Lancer un script au démarrage

Créer le script et le rendre exécutable (on verra son contenu plus loin)

```
pwd    # renvoie /home/xor
touch ./script_demarrage_qiff.sh
chmod +x ./script_demarrage_qiff.sh
```

Éditer `/home/xor/.config/lxsession/LXDE/autostart`

```
@lxpanel --profile LXDE
@pcmanfm --desktop --profile LXDE
@xscreensaver -no-splash
/home/xor/script_demarrage_qiff.sh
```

Raccourcis clavier utiles

Une fois le script de démarrage automatique lancé en plein écran, ça peut-être utile...
SUPER + D : revenir au bureau
CTRL + ALT + T : démarrer Terminator
CTRL + ALT + V : placer l'écran en mode vertical (rotation droite)
CTRL + ALT + N : placer l'écran en mode normal

Script de démarrage

script_demarrage_qiff.sh

```
v4l2-ctl -d /dev/video0 -c vertical_flip=1
/home/xor/processing-4.0b2/processing-java --sketch=/home/xor/QIFF-2022/attraction_mutante/qiff_app_001 --run
```

Nécessitera une mise à jour manuelle pour lancer le bon script P5!

Installation

Sur clé USB, tous les fichiers nécessaire sur l'ordi, la récupération des scripts à jour se fera par git clone. (dossier avec tous les fichiers sur ordi CL23)

install_root.sh

```
# à exécuter en root

# installer git
apt install git-all
```

```
# installer tesseract-ocr pour opencv_processing
apt install tesseract-ocr

# démarrage sans login EN ROOT
cp ./lightdm.conf /etc/lightdm/lightdm.conf
```

install.sh

```
# Script d'installation pour les ordis du QIFF
# tous les fichiers sont sur clé usb
# Ouvrir un terminal dans le dossier de la clé
# certaines opérations sont à réaliser en root

# install processing
cp -RT ./processing-4.0b2 /home/xor/processing-4.0b2
chmod +x /home/xor/processing-4.0b2/processing
mkdir /home/xor/sketchbook
mkdir /home/xor/sketchbook/examples
mkdir /home/xor/sketchbook/libraries
mkdir /home/xor/sketchbook/modes
mkdir /home/xor/sketchbook/templates
mkdir /home/xor/sketchbook/tools
cp -RT ./controlP5 /home/xor/sketchbook/libraries/controlP5
cp -RT ./opencv_processing /home/xor/sketchbook/libraries/opencv_processing
cp -RT ./sound /home/xor/sketchbook/libraries/sound
cp -RT ./video /home/xor/sketchbook/libraries/video

# install arduino
cp -RT ./arduino-1.8.5 /home/xor/arduino-1.8.5
mkdir /home/xor/Arduino
mkdir /home/xor/Arduino/libraries
cp -RT ./OneButton /home/xor/Arduino/libraries/OneButton

# install script de démarrage
cp ./script_demarrage_qiff.sh /home/xor/script_demarrage_qiff.sh
chmod +x /home/xor/script_demarrage_qiff.sh
cp ./autostart /home/xor/.config/lxsession/LXDE/autostart

# raccourcis clavier
mkdir /home/xor/bin
cp ./ecran_rotation_droite.sh /home/xor/bin/ecran_rotation_droite.sh
cp ./ecran_rotation_normale.sh /home/xor/bin/ecran_rotation_normale.sh
chmod +x /home/xor/bin/ecran_rotation_droite.sh
chmod +x /home/xor/bin/ecran_rotation_normale.sh
cp ./lxde-rc.xml /home/xor/.config/openbox/lxde-rc.xml
openbox --reconfigure
openbox --restart

# clone du git avec les applications
mkdir /home/xor/QIFF-2022
git clone https://github.com/lesporteslogiques/attraction_mutante.git /home/xor/QIFF-2022/attraction_mutante
```

TODO

Reprendre * mise à jour du squelette qif_app_001

- OK ajouter mode diagnostic (dans la version temp)
- OK ajouter nocursor (ds keyPressed) selon mode param ou non
- (à faire) tester micro
- (à faire) image inversée par p5 optionnelle
- OK rotation de l'image avant upload

Mettre à jour le script d'install pour copier la clé d'API dans chaque dossier

ajouter timeDisplacement aux scripts

Article extrait de : http://lesporteslogiques.net/wiki/ - **WIKI Les Portes Logiques**
Adresse : **http://lesporteslogiques.net/wiki/openatelier/projet/attraction_mutante_qiff?rev=1646160016**
Article mis à jour: **2022/03/01 19:40**