Pilotage en immersion avec les LEGO Mindstorms EV3

Comment piloter un véhicule LEGO Mindstorms EV3 en immersion avec un joystick de PlayStation, des caméras USB, un casque vidéo stéréoscopique à base de smart-phone, et le système d'exploitation alternatif ev3dev.

READ THE HYPERTEXT VERSION HERE:
http://www.pabr.org/bricks/brickfpv/brickfpv.fr.html

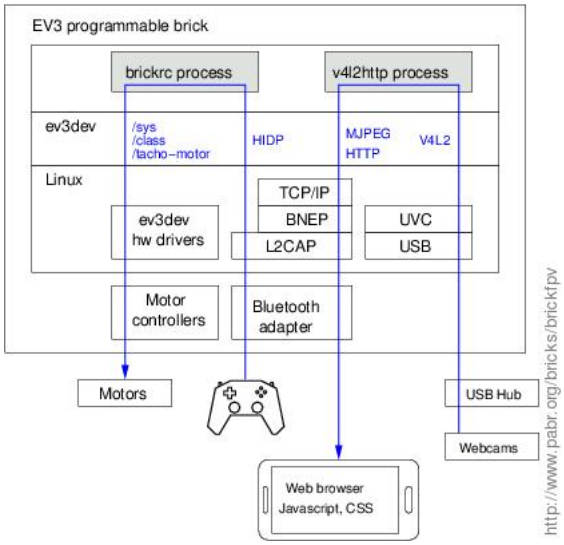
Historique des versions		
1.0	2014-12-14	Première publication.
1.1	2015-01-05	Procédure d'installation plus simple.

Table des matières

1. Introduction

Le pilotage en immersion (FPV) est en train de révolutionner le modélisme radiocommandé. Il est généralement réalisé à l'aide de matériels dédiés : caméras analogiques PAL/NTSC, transmetteurs 2.4/5.8 GHz, et lunettes vidéo analogiques avec récepteur intégré. On peut évidemment installer de tels équipements sur un modèle LEGO. Dans ce projet nous intégrons plutôt les fonctions de télécommande et de vue en immersion dans la plate-forme LEGO Mindstorms EV3 en exploitant son processeur ARM9 relativement puissant, son Linux embarqué, son port USB et sa connectivité sans fil. La vidéo numérique est reçue et affichée par un smartphone ordinaire installé dans un casque de type Google Cardboard.

Pour illustrer l'intérêt d'un tel système entièrement numérique et intégré, nous affichons des informations de télémétrie en surimpression à l'aide de quelques lignes de Javascript. Dans un système analogique ceci exigerait du matériel embarqué supplémentaire.



Dans l'implémentation actuelle un joystick Bluetooth se connecte directement à la brique EV3, et les flux vidéo MJPEG de caméras USB sont transmis également par Bluetooth (HTTP/TCP/IP/BNEP), en privilégiant la qualité d'image ou la vitesse de rafraîchissement selon les circonstances. Nous proposons plusieurs pistes pour améliorer la portée, la latence et la qualité vidéo.

2. Avertissements relatifs à la sécurité et à la santé

L'engouement récent pour les applications de réalité virtuelles conduit les autorités de santé à publier des préconisations pour l'utilisation des systèmes d'affichage immersifs. Voir par exemple :https://www.oculus.com/warnings/.

Ces recommandations ne sont pas à prendre à la légère. Des casques stéréoscopiques de mauvaise qualité, bricolés ou simplement mal réglés peuvent soumettre la vision binoculaire humaine à des stimuli qui n'existent pas dans la nature, ce qui peut provoquer une gêne persistante ou des blessures.

Les enfants de moins de 13 and et les personnes souffrant de troubles de la vision peuvent pratiquer le pilotage vidéo sur un écran ordinaire (non immersif).

3. Prérequis

3.1. Pièces LEGO

Le véhicule à chenilles de la vidéo est dérivé des modèles TRACK3R et GRIPP3R de la boîte Mindstorms EV3 de base (31313).

Les caméras sont installées sur une tourelle motorisés réalisée avec des pièces de la boîte EV3 Education Expansion (45560) et un moteur EV3 moyen supplémentaire (45503).

3.2. Caméras

La plupart des caméras USB sont maintenant supportées sous Linux, mais les pilotes des *chipsets* propriétaires sont souvent basés sur un *reverse-engineering* partiel des pilotes Windows. Nous préférons donc utiliser des caméras qui respectent la norme USB video device class (UVC). Voici une liste [http://www.ideasonboard.org/uvc/] de caméras UVC courantes.

Pour ce projet la caméra doit impérativement fournir un flux compressé ; il est hors de question de traiter de la vidéo brute sur le processeur de la brique EV3. On trouve facilement des caméras MJPEG bon marché, et il est facile de décoder et d'afficher des flux MJPEG dans un navigateur web.

La caméra doit être compatible avec USB 1.1, car le port USB sur le côté de la brique EV3 n'est pas compatible 2.0.

Pour raccorder deux caméras à la brique EV3, un petit hub USB est nécessaire.

Nous utilisons actuellement une paire de caméras Logitech C170 (VGA, MJPEG) déjà réputées compatibles avec ev3dev (voir [AD4M4ST0R]). Nous transmettons par défaut un flux 120x160 mono pour maximiser la cadence de rafraîchissement, et l'utilisateur peut demander une photo stéréo de 2x480x640 pixels lorsque le véhicule est à l'arrêt.

3.3. Casque stéréoscopique

On peut piloter en FPV simplement en regardant un flux vidéo sur une tablette ou un smartphone, mais un casque vidéo améliore la sensation d'immersion. Un casque monoscopique (où les écran gauche et droit reproduisent la même image) est suffisant pour des vues aériennes, mais la stéréoscopie se justifie dans le contexte d'un véhicule terrestre évoluant à proximité d'obstacles.

N'importe quel clone de Google Cardboard peut être utilisé pour ce projet. Nous utilisons le casque Homido qui a quelques avantages notables par rapport aux versions plus rustiques :

- distance inter-lentilles ajustable à l'écart inter-pupillaire de l'utilisateur (plage d'environ 57 mm à 64 mm) ;
- trois réglages pour la distance entre les lentilles et l'écran (permet de corriger la vision de l'utilisateur) ;
- distance entre les lentilles et les yeux ajustable (pour optimiser le champ de vision) ;
- Lentilles larges (36 mm au lieu de 25 mm) ;
- supports coniques qui positionnent les lentilles près des yeux sans presser sur le nez ;
- bonne aération qui évite la formation de buée sur les lentilles.

3.4. Smartphone

L'appareil utilisé pour l'affichage doit gérer la connectivité IP par Bluetooth (BNEP, PAN ou simplement "Accès Internet" dans les paramètres Bluetooth).

Actuellement l'appareil idéal pour les applications de réalité virtuelle mobile est probablement le Samsung Galaxy Note 4 (2560x1440, 515 pixels par pouce), mais un appareil avec un écran de 1280x720 est largement suffisant pour un projet avec des caméras VGA.

3.5. Joystick Bluetooth

Le logiciel prend actuellement en charge le SIXAXIS, le DualShock 3 et le Move Navigation Controller. On pourra facilement obtenir ces périphériques d'occasion à faible coût au fur et à mesure que les nouvelles générations de consoles envahissent le marché.

4. Logiciels

brickfpv se compose de plusieurs petits programmes écrits en C (**brickrc**, **v4l2http**, **sixpair**) tournant sous [EV3DEV], un système d'exploitation dérivé de Debian pour la brique EV3 qui donne accès à toutes ses fonctionnalités de plate-forme Linux embarquée. Les programmes peuvent être configurés et combinés pour prendre en charge plusieurs types de véhicules (à chenilles ou à roues directrices), de modes de pilotage (caméra avant, avant+arrière, stéréo), de montages de caméra (azimut/élevation, azimut seul, fixe) et d'affichages (casque vidéo stéréoscopique ou écran ordinaire).

Consulter le code source : http://www.pabr.org/bricks/brickfpv/brickfpv-1.0/

4.1. Télécommande : brickrc

brickrc est un successeur de [BRICKHID] avec prise en charge des pilotes de moteurs de ev3dev (`/sys/class/tacho-motor/*`).

4.2. Vidéo : v4l2http

4.3. Affichage côté client : Javascript et CSS

Les flux vidéo sont affichés sous forme d'images JPEG dans un navigateur web. Du code Javascript reçoit les images depuis la brique EV3 par HTTP. Des transformations CSS permettent de pivoter et redimensionner les images en fonction de l'orientation des caméras.

4.4. Appariement Bluetooth : brickrc

5. Mise en oeuvre

5.1. Installation et configuration d'ev3dev

Nous utilisons ev3dev-jessie-2014-12-01 [https://github.com/ev3dev/ev3dev/releases/tag/ev3dev-jessie-2014-12-01], la première version qui permet d'exploiter facilement la connectivité IP sur Bluetooth.

Sous Linux l'installation se résume à flasher une carte microSD de 1 Go ou plus :

```
xzcat ev3dev-jessie-2014-12-01.img.xz [https://github.com/ev3dev/ev3dev/releases/downlo
```

(Pour de plus amples explications, voir http://www.ev3dev.org/docs/getting-started/#step-2-copy-the-image-on-to-the-sd-card.)

Après avoir insérer la carte microSD et booté la brique EV3, il faut la mettre en réseau. La solution la plus simple consiste semble-t-il à utiliser un adaptateur USB-ethernet. La solution la plus économique

consiste à configurer une connexion réseau sur USB entre la brique EV3 et un PC Linux. Voir http://www.ev3dev.org/docs/getting-started/#step-4-setup-a-network-connection.

Après avoir obtenu un shell root, commençons par installer quelques outils :

```
root@ev3dev:~# apt-get update
root@ev3dev:~# apt-get install bluez-hcidump bluez-tools bridge-utils iptables netca
```

Enfin, il faut que le gestionnaire Bluetooth laisse **brickrc** s'occuper des périphériques HID :

```
root@ev3dev:~# cp /lib/systemd/system/bluetooth.service /etc/systemd/system/bluetoot
```

Avec par exemple **vi** ou **nano**, éditer `/etc/systemd/system/bluetooth.service` pour modifier la ligne **ExecStart** comme suit :

```
ExecStart=/usr/lib/bluetooth/bluetoothd -P input
```

Puis redémarrer le gestionnaire Bluetooth avec la configuration modifiée :

```
root@ev3dev:~# systemctl daemon-reload
root@ev3dev:~# systemctl restart bluetooth.service
```

5.2. Téléchargement et test de brickfpv

```
root@ev3dev:~# wget http://www.pabr.org/bricks/brickfpv/brickfpv-1.0.tar.gz
root@ev3dev:~# tar zxvf brickfpv-1.0.tar.gz
root@ev3dev:~# cd brickfpv-1.0
```

Le joystick PS3 doit être apparié avec la brique EV3. Il faut donc brancher le joystick PS3 à la brique EV3 avec un cordon USB mini-B vers Type-A avant de continuer. Puis :

```
root@ev3dev:~/brickfpv-1.0# ./brickfpv.init start
root@ev3dev:~/brickfpv-1.0# tail -f /tmp/brickfpv.log
```

5.3. Test du joystick

Débrancher le joystick et presser son bouton PS. Les quatre LEDs vont clignoter brièvement, puis la première va s'allumer en rouge fixe. À ce stade le joystick contrôle les moteurs.

5.4. Test de la video

Sur le smartphone :

- accéder au menu des réglages Bluetooth ;
- rechercher des périphériques ;
- sélectionner le périphérique nommé "ev3dev" ;
- sélectionner "Utiliser pour l'accès Internet" ;
- vérifier l'apparition d'un dialogue sur l'écran LCD et confirmer avec le bouton OK ;
- dans le navigateur web, ouvrir **http://192.168.0.1/**

5.5. Installer brickfpv de façon permanente

Pour démarrer **brickfpv** automatiquement à l'allumage de la brique EV3 :

```
root@ev3dev:~/brickfpv-1.0# cp brickfpv.service /etc/systemd/system/
root@ev3dev:~/brickfpv-1.0# systemctl daemon-reload
root@ev3dev:~/brickfpv-1.0# systemctl enable brickfpv.service
```

brickfpv apparie automatiquement les joysticks qu'il détecte sur le bus USB au démarrage. Il faut brancher le joystick, allumer la brique, puis attendre que la LED s'allume en vert fixe et que l'écran LCD affiche le menu **brickman**) avant de débrancher.

5.6. En cas de problème

Consulter les messages d'erreur dans `/tmp/brickfpv.log`.

Arrêter les éventuels composants de **brickfpv** en cours d'exécution :

```
root@ev3dev:~# cd /root/brickfpv-1.0
root@ev3dev:~/brickfpv-1.0# ./brickfpv.init stop
```

Exécuter **sixpair** et **brickrc** manuellement depuis la ligne de commande :

```
root@ev3dev:~/brickfpv-1.0# ./sixpair
root@ev3dev:~/brickfpv-1.0# ./brickrc -v ps3_tank.config
```

puis lire les messages qui s'affichent lorsqu'on presse le bouton PS.

Vérifier que le joystick essaie d'ouvrir une connexion Bluetooth :

```
root@ev3dev:~# hcidump
```

Activer la connectivité IP sur Bluetooth en mode NAP (point d'accès) :

```
root@ev3dev:~# connmanctl tether bluetooth on
```

Exécuter le serveur vidéo manuellement depuis la ligne de commande :

```
root@ev3dev:~/brickfpv-1.0# ./streamer.sh 81 "./v4l2http -c /dev/video0 -r 15 --mjpe
root@ev3dev:~/brickfpv-1.0# ./httpd.sh  &
```

6. Développement

6.1. Utiliser gcc sur la brique EV3

Pour développer du logiciel embarqué, on recourt généralement à un compilateur croisé. Pour ev3dev, voir https://github.com/ev3dev/ev3dev/wiki/Using-brickstrap-to-cross-compile-and-debug.

Mais on peut aussi compiler de petits programmes (tels que **brickfpv**) directement sur la brique EV3.

ev3dev est livré avec des compilateurs et interpréteurs pour plusieurs langages exotiques, mais pas pour C. Il faut donc faire un peu de place et installer **gcc** :

```
root@ev3dev:~# dpkg -P golang-go-linux-arm golang-go golang-src
root@ev3dev:~# dpkg -P guile-1.8 guile-1.8-libs
root@ev3dev:~# apt-get install gcc libusb-dev libbluetooth-dev
```

6.2. Recompilation du noyau

Certaines fonctionnalités potentiellement utiles pour ce projet sont désactivées dans le noyau ev3dev. Pour les activer, il faut recompiler le noyau (ou au moins certains modules).

Comme précédemment, nous pouvons compiler le noyau Linux directement sur la brique EV3. La première compilation prend un certain temps, mais il est ensuite possible de patcher, recompiler et tester des modules en temps raisonnable. La seule difficulté est le manque de RAM. Il faut donc ajouter du swap (nécessite une carte microSD de plus de 1 Go):

```
root@ev3dev:~# lvcreate ev3devVG --size 512M --name swap
root@ev3dev:~# mkswap /dev/ev3devVG/swap
root@ev3dev:~# swapon /dev/ev3devVG/swap
```

7. Problèmes connus

- BlueZ 5.12+ est censé prendre en charge les joysticks de PS3. Ceci ne semble pas fonctionner. C'est pourquoi **brickfpv** recourt à **sixpair** et des *L2CAP*.
- **brickfpv** utilise **netcat-traditional** et des scripts shell comme un serveur HTTP minimaliste. Il faudrait plutôt utiliser des *sockets* **systemd**.

8. Perspectives

8.1. Robotique

Tel que décrit ci-dessus, **brickfpv** ne fait que reproduire (plutôt maladroitement) les fonctionnalités de systèmes de radiocommande et de pilotage en immersion. Mais comme nous avons intégré toutes ces fonctions sur un ordinateur unique, nous pouvons faire beaucoup plus :

- Traitement d'images : Système anti-collision, estimation de distances.
- Pilote automatique simple : Aller jusqu'à un endroit sélectionné par l'utilisateur sur l'affichage vidéo.
- Comportements autonomes : Navigation visuelle, cartographie, reconnaître et ramasser des objets, etc.

8.2. Rectification des paires d'images stéréo

Pour la vision stéréoscopique, les paires d'images doivent être alignées à quelques pixels près. Nous devons donc orienter les caméras très précisément à la main. Une meilleure approche consiste à calibrer les caméras et à reprojeter les images. Cette technique s'appelle rectification. On en trouve une implémentation dans [OPENCV]. On peut envisager de faire effectuer la reprojection en temps réel par le GPU d'un smartphone moderne, en compensant la distorsion des lentilles du casque vidéo par la même occasion.

8.3. Utiliser deux caméras USB simultanément

Lorsqu'on essaie d'utiliser simultanément deux périphériques `/dev/video` sous Linux, on obtient souvent l'erreur ENOSPC (bande passante insuffisante). Ceci se produit même en USB 2.0, quelles que soient la résolution, la cadence et la compression.

Pour les formats non compressés, **modprobe uvc_video quirks=128** règle parfois le problème.

Pour les formats compressés, il n'y a actuellement pas de solution simple. Cette discussion [http://stackoverflow.com/questions/9781770/capturing-multiple-webcams-uvcvideo-with-opencv-on-linux] propose un patch. N.B. : Ceci implique de recompiler le module noyau **uvc_video**.

8.4. Améliorer la compression vidéo

On trouve maintenant des caméras USB avec compression H.264. D'un côté, la norme H.264 est plus économe en bande passante que MJPEG, et elle traite non seulement la vidéo mais aussi le son. D'un autre côté les décodeurs H.264 des smartphones ne sont généralement pas très flexibles, et il reste à déterminer s'il serait facile d'afficher deux flux gauche/droite synchronisés côte à côte.

D'après cette discussion [http://stackoverflow.com/questions/11695508/how-to-control-bitrate-media-subtype-h264-directshow], des caméras Logitech C920 pourraient peut-être transmettre deux flux Full HD à 30 im/s sur USB 1.1. Mais une liaison Bluetooth de 1 Mbit/s ne pourrait supporter que deux flux 360x180 à 5 im/s. Il reste à déterminer si le taux de compression peut être modifié.

8.5. Utiliser WiFi au lieu de Bluetooth

On pourrait améliorer la portée en diffusant la vidéo par WiFi au lieu de Bluetooth. Même avec deux caméras et un adaptateur WiFi tous connectés sur le port USB 1.1, la bande passante pourrait s'avérer meilleure que dans l'implémentation actuelle.

On connecterait le joystick Bluetooth à une app tournant sur le smartphone qui relaierait les commandes à la brique EV3 par WiFi.

8.6. Suivi des mouvements de la tête

Avec une app dédiée sur le smartphone, on pourrait facilement faire pivoter la tourelle de la caméra en réponse aux mouvements de rotation ou d'inclinaison du casque.

Bibliographie

[EV3DEV] *Debian on LEGO MINDSTORMS EV3* . Ralph Hempel et David Lechner. http://www.ev3dev.org/.

[AD4M4ST0R] *AD4M4ST0R – um rover LEGO*. Jorge Pereira. http://ofalcao.pt/blog/pt/2014/ad4m4st0r-um-rover-lego.

[OPENCV] *OpenCV (Open Source Computer Vision*. http://opencv.org.

[BRICKHID] *brickhid - Connexion directe entre un joystick Bluetooth et les LEGO Mindstorms EV3* . http://www.pabr.org/bricks/brickhid/brickhid.fr.html .

[SIXLINUX] *Utilisation du joystick de la PlayStation 3 en mode Bluetooth avec Linux* . http://www.pabr.org/sixlinux/sixlinux.fr.html .