First-person view and remote control with LEGO Mindstorms EV3

We add FPV and RC functionality to a LEGO Mindstorms EV3 vehicle with a PlayStation gamepad, USB webcams, a smartphone-based stereoscopic head-mounted display, and the alternative operating system ev3dev.

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

Revision History		
1.0	2014-12-14	Initial release.
1.1	2015-01-05	Simpler installation instructions.

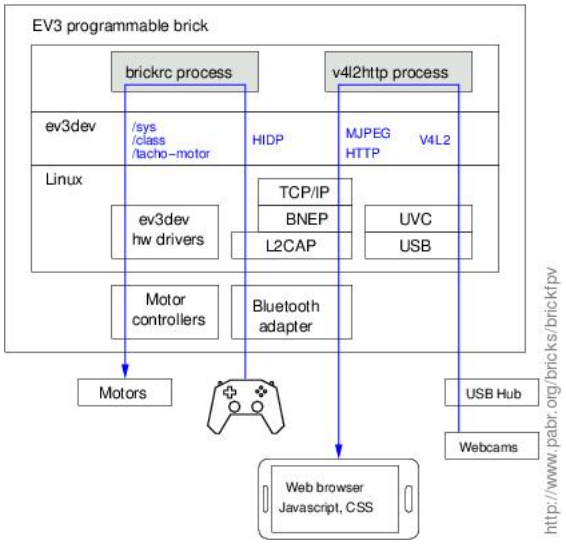
Table of Contents

1. Introduction 3
2. Health and safety warnings 3
3. Requirements 3
    3.1. LEGO parts 3
    3.2. Cameras 4
    3.3. Head-mounted display 4
    3.4. Smartphone 4
    3.5. Bluetooth gamepad 4
4. Software 5
    4.1. Remote control: brickrc 5
    4.2. Video streaming: v4l2http 5
    4.3. Client-side rendering: Javascript and CSS 5
    4.4. Bluetooth pairing: sixpair 5
5. Operation 5
    5.1. Installing and customizing ev3dev 5
    5.2. Downloading and running **brickfpv** 6
    5.3. Testing the gamepad 6
    5.4. Testing video streaming 6
    5.5. Installing **brickfpv** permanently 6
    5.6. Troubleshooting 7
6. Development 7
    6.1. Using gcc on the EV3 brick 7
    6.2. Customizing and recompiling the kernel 7
7. Known limitations 8
8. Perspectives 8
    8.1. Robotics 8
    8.2. Stereo pair rectification 8
    8.3. Using two USB webcams simultaneously 8
    8.4. Improving video compression 8
    8.5. Using WiFi instead of Bluetooth 9
    8.6. Head tracking 9
Bibliography 9

1. Introduction

First Person View has become popular among RC enthusiasts. FPV is typically implemented with dedicated equipment such as PAL/NTSC analog video cameras, 2.4/5.8 GHz transmitters, and analog video goggles with built-in receivers. Such equipment can of course be mounted on a LEGO model. In this project, instead, we merge radio control and FPV functionality into the LEGO Mindstorms EV3 platform, leveraging its relatively powerful ARM9 system-on-chip, embedded Linux OS, USB capabilities and wireless connectivity. Digital video is received and displayed by a generic smartphone in a Google Cardboard-style headset.

As an illustration of the benefits of a fully digital and integrated system, we overlay telemetry data with just a few lines of Javascript code. In a legacy analog system this would typically require dedicated on-board hardware.



In the current implementation a Bluetooth gamepad connects directly to the EV3 brick, and MJPEG video from USB webcams is also transmitted over Bluetooth (HTTP/TCP/IP/BNEP) with adjustable trade-offs between image quality and frame rate. We discuss ways to improve range, latency and video quality.

2. Health and safety warnings

As virtual reality applications become mainstream, medical authorities worldwide are developing guidelines for the safe use of immersive displays. See for example: https://www.oculus.com/warnings/.

These recommendations are not to be taken lightly. Low-quality, DIY or poorly configured head-mounted stereoscopic displays can exercise human binocular vision in ways that are completely unnatural, leading to long-lasting discomfort or even injury.

Children under 13 and anyone with impaired vision can still enjoy FPV on regular (non immersive) displays.

3. Requirements

3.1. LEGO parts

The tracked vehicle in the demo video is derived from the TRACK3R and GRIPP3R models from the Mindstorms EV3 base set (31313).

Cameras are mounted on a motorized turret which uses parts from the EV3 Education Expansion set (45560) and an extra EV3 medium motor (45503).

3.2. Cameras

Most USB webcams are supported under Linux nowadays, but many chipsets are not publicly documented and their Linux driver are based on partial reverse-engineering of Windows drivers. Thus, we favour webcams which comply with the USB video device class (UVC) standard. Here is a list [http://www.ideasonboard.org/uvc/] of common UVC webcams.

This project requires a webcam with built-in compression; we do not want to process raw video on the EV3 CPU. Inexpensive MJPEG webcams are now available, and it is easy to decode and display MJPEG streams in a web browser.

The webcam should be backward-compatible with USB 1.1, as the Host port on the side of the EV3 brick does not support USB 2.0.

To connect two cameras to the EV3 brick, a small USB hub is required.

We use a pair of Logitech C170 webcams (VGA, MJPEG) which were already known to work with ev3dev (see [AD4M4ST0R]). We stream at 120x160 mono by default to maximize frame rate, and the user can manually request a stereo snapshot at 2x480x640 when the vehicle is stationary.

3.3. Head-mounted display

FPV can be as simple as looking at a video feed on a tablet or smartphone, but a head-mounted display improves the feeling of immersion. While a monoscopic head mount (with left and right screens showing the same image) is sufficient for aerial imagery, a stereoscopic view is definitely useful when driving a ground vehicle in a cluttered environment.

Any clone of Google Cardboard should be suitable for this project. We use the Homido headset which has significant advantages over plain implementations:

- Inter-lens distance adjustable from about 57 mm to 64 mm (to match the user's IPD)
- Three lens-to-screen distance settings (useful for vision-impaired users)
- Adjustable lens-to-eye distance (to maximize the field of view)
- Wide lenses (1.4" vs 1")
- Conical mounts that bring the lenses close to the eyes without pressing against the nose
- Good airflow which prevents fogging of the lenses.

3.4. Smartphone

The device must support IP networking over Bluetooth, also known as BNEP or PAN or simply "Internet access" in the Bluetooth settings menu.

We currently use an old Samsung Galaxy S3 phone, but any recent Android device is likely to be suitable. Even iOS might work as well.

At the time of writing the best available device for mobile VR is probably the Samsung Galaxy Note 4 (2560x1440, 515 pixels per inch), but a device with a 1270x720 display is sufficient for a project with VGA cameras.

3.5. Bluetooth gamepad

Our software currently supports the SIXAXIS, DualShock 3 and Move Navigation Controller. Second-hand units should be massively available and inexpensive in the next few years as newer gaming consoles reach the market.

4. Software

brickfpv consists of small independent C programs (**brickrc**, **v4l2http**, **sixpair**) running on top of [EV3DEV], a Debian-based third-party operating system for the EV3 brick which unlocks its full potential as an embedded Linux platform. The programs can be configured and combined to handle various types of vehicles (tracked, steered), view configurations (single front camera, front/rear cameras, stereo cameras), camera gimbals (pan/tilt, pan only, fixed) and displays (stereoscopic HMD or regular screen).

Browse the source code: http://www.pabr.org/bricks/brickfpv/brickfpv-1.0/

4.1. Remote control: brickrc

brickrc is a successor to [BRICKHID] with support for the motor drivers of ev3dev (`/sys/class/tacho-motor/*`).

4.2. Video streaming: v4l2http

4.3. Client-side rendering: Javascript and CSS

Video is simply rendered as JPEG images in a web browser. Javascript code receives images from the EV3 brick over HTTP. CSS transforms rotate and scale the images to account for various camera mounting options.

4.4. Bluetooth pairing: sixpair

The PlayStation gamepads require a special pairing procedure over USB before they can be used wirelessly. See [SIXLINUX] for background information.

5. Operation

5.1. Installing and customizing ev3dev

We use ev3dev-jessie-2014-12-01 [https://github.com/ev3dev/ev3dev/releases/tag/ev3dev-jessie-2014-12-01], apparently the first release with user-friendly IP networking over Bluetooth.

On Linux, installation boils down to flashing a microSD card (1 GB or more):

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

(See http://www.ev3dev.org/docs/getting-started/#step-2-copy-the-image-on-to-the-sd-card for details.)

After inserting the microSD card and booting the EV3 brick, we need to connect it to a network. The simplest way is reportedly with a USB-to-ethernet dongle. The least expensive (but not so simple) way is to set up networking over USB between the EV3 and a Linux PC. See http://www.ev3dev.org/docs/getting-started/#step-4-setup-a-network-connection.

Once we have a root shell, we can install a few useful packages:

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

Finally, we must tell the Bluetooth HID subsystem to let **brickrc** handle HID peripherals:

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

Using e.g. **vi** or **nano**, edit `/etc/systemd/system/bluetooth.service` to change the **ExecStart** line to:

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

Then restart the Bluetooth subsystem: with the new configuration:

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

5.2. Downloading and running 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
```

We need to pair the PS3 gamepad with the EV3 brick. So, connect the PS3 gamepad to the EV3 brick with a USB mini-B to Type-A cable before proceeding. Then:

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

5.3. Testing the gamepad

Unplug the gamepad and press its PS button. The four LEDs will flash briefly, then one will turn solid red. At this point the gamepad controls the motors.

5.4. Testing video streaming

In the smartphone:

- Enter the Bluetooth settings menu
- Scan for devices
- Select the device named "ev3dev"
- Select "Use for Internet access"
- Watch for a pop-up dialog on the LCD screen and confirm with the OK button.
- In the web browser, open **http://192.168.0.1/**

5.5. Installing brickfpv permanently

To start **brickfpv** automatically when the EV3 brick is powered up:

```
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 will automatically pair any gamepad found on the USB bus when it starts. Plug the gamepad, turn the brick on, then wait at least until the LED turns solid green and the LCD shows the **brickman** menu before unplugging.

5.6. Troubleshooting

Look for error messages in `/tmp/brickfpv.log`.

Stop any lingering component of **brickfpv**:

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

Run **sixpair** and **brickrc** manually from the command-line:

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

then watch for messages when the PS button is pressed.

Check whether the gamepad tries to establish a Bluetooth connection:

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

Enable IP networking over Bluetooth in NAP (Network Access Point) mode:

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

Run the video server manually from the command-line:

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

6. Development

6.1. Using gcc on the EV3 brick

The traditional way to develop embedded software involves cross-compilation toolchains. For ev3dev, see https://github.com/ev3dev/ev3dev/wiki/Using-brickstrap-to-cross-compile-and-debug.

Alternatively, small programs (such as **brickfpv**) can be compiled directly on the EV3 brick.

ev3dev ships with compilers and interpreters for several exotic languages, but not for C. So we free some space and install **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. Customizing and recompiling the kernel

The ev3dev kernel configuration lacks a few features which might benefit this project. To enable them, the kernel (or at least certain loadable modules) must be recompiled.

Again, we can compile the whole Linux kernel on the EV3 brick. The initial build takes some time, but then individual drivers can be patched, recompiled and tested reasonably quickly. The only difficulty is that RAM is too small, so we add swap (requires a microSD card larger than 1 GB):

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

7. Known limitations

- In theory BlueZ 5.12+ has support for PS3 controllers. For some reason this does not work. This is why **brickfpv** uses the old **sixpair** tool and raw L2CAP sockets.
- **brickfpv** uses **netcat-traditional** and shell scripts as a lightweight HTTP server. This should probably be done with **systemd** sockets.

8. Perspectives

8.1. Robotics

As described above, **brickfpv** merely replicates (and poorly so) the functionality of dedicated RC and FPV systems. However, having integrated all these functions digitally on a single computer, we can do much more:

- Image processing: Collision avoidance, range estimation.
- Simple autopilot: Navigate to a location that the user has selected on the video display.
- Autonomous behaviour: Visual navigation, mapping, recognizing and picking objects, etc.

8.2. Stereo pair rectification

For proper stereoscopic viewing, image pairs must be aligned to within a few pixels. Therefore we have to orient the cameras very carefully. Alternatively, cameras can be calibrated and images can be reprojected in software. This process is known as image rectification. Implementations can be found e.g. in [OPENCV]. It might be possible to perform reprojection in real-time on the GPU of a modern smartphone, and even compensate for HMD lens distortion in the same rendering pass.

8.3. Using two USB webcams simultaneously

Trying to stream from two `/dev/video` devices simultaneously on Linux often fails with ENOSPC (not enough bandwidth). This happens even on USB 2.0, regardless of resolution, frame rate and compression.

For uncompressed formats, **modprobe uvc_video quirks=128** sometimes fixes the problem.

For compressed formats, there is currently no simple workaround. This discussion [http://stackoverflow.com/questions/9781770/capturing-multiple-webcams-uvcvideo-with-opencv-on-linux] provides a patch. Note that this requires recompiling the **uvc_video** kernel module.

8.4. Improving video compression

USB webcams with built-in H.264 compression are now available. On the one hand, H.264 is more efficient than MJPEG, and it handles not only video but also audio. On the other hand, H.264 decoders in smartphones tend to be proprietary and it is unclear whether two synchronized left/right streams could be displayed side by side easily.

According to this discussion [http://stackoverflow.com/questions/11695508/how-to-control-bitrate-mediasubtype-h264-directshow], Logitech C920 webcams might be able to stream two Full HD streams at 30 fps over USB 1.1. But a 1 Mbit/s Bluetooth link could only carry two 360x180 streams at 5 fps. It is unclear whether the compression level can be adjusted.

8.5. Using WiFi instead of Bluetooth

Range could be extended by streaming video over WiFi instead of Bluetooth. Even with two webcams and a WiFi adapter all connected to the USB 1.1 port, bandwidth might be better than in the current implementation.

The Bluetooth joystick would connect to an app running on the smartphone, and commands would be relayed to the EV3 brick over WiFi.

8.6. Head tracking

With an app running on the smartphone, it would be easy to pan the camera turret in response to rotation or tilt of the HMD.

Bibliography

[EV3DEV] *Debian on LEGO MINDSTORMS EV3* . Ralph Hempel and 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 - Direct connection between Bluetooth gamepads and LEGO Mindstorms EV3* . http://www.pabr.org/bricks/brickhid/brickhid.en.html .

[SIXLINUX] *Using the PlayStation 3 controller in Bluetooth mode with Linux* . http://www.pabr.org/sixlinux/sixlinux.en.html .