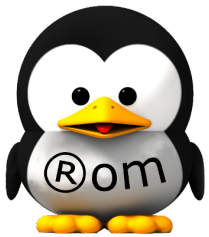
Lecture différée de la webcam d'un Rasberry Pi

Romain Vimont

12 février 2014 | `b2537fd`



Contexte

Raspberry Pi avec caméra

Les outils dédiés au *Raspberry Pi* :

- `raspivid` (capture)
- `omxplayer` (enregistrement)

Objectif

Lire le *direct* différé de quelques secondes.

Un tube

Le principe

- démarrer raspivid pour capturer l'image
- démarrer omxplayer quelques secondes plus tard
- brancher la sortie de raspivid sur l'entrée de omxplayer

Un simple *tube*

```
raspivid ... | { sleep 5; omxplayer ...; }
```

Un simple *tube*

```
raspivid ... | { sleep 5; omxplayer ...; }
```

Mais omxplayer ne sait pas lire son entrée standard.

FAIL #0

Ce n'est pas si simple.

Un simple *tube nommé*

```
# terminal 1
mkfifo /tmp/fifo
raspivid ... > /tmp/fifo

# terminal 2, quelques secondes plus tard
omxplayer /tmp/fifo
```

Un simple *tube nommé*

```
# terminal 1
mkfifo /tmp/fifo
raspivid ... > /tmp/fifo

# terminal 2, quelques secondes plus tard
omxplayer /tmp/fifo
```

Mais il y a plus simple...

Unwrapped

omxplayer est un script *wrapper* pour omxplayer.bin.

```
export LD_LIBRARY_PATH="/opt/vc/lib:/usr/lib/omxplayer"
omxplayer.bin ...
```

Entrée standard

omxplayer.bin peut lire sur son entrée standard.

Entrée standard

omxplayer.bin peut lire sur son entrée standard.

Mais pas directement, il attend un fichier en paramètre.

```
raspivid ... | omxplayer.bin -
```

Entrée standard

omxplayer.bin peut lire sur son entrée standard.

Mais pas directement, il attend un fichier en paramètre.

```
raspivid ... | omxplayer.bin -
```

Ne fonctionne pas !

Entrée standard

omxplayer.bin peut lire sur son entrée standard.

Mais pas directement, il attend un fichier en paramètre.

```
raspivid ... | omxplayer.bin -
```

Ne fonctionne pas !

`/dev/stdin` est un fichier !

```
raspivid ... | omxplayer.bin /dev/stdin
```

Capacité limitée

`man 7 pipe`

> *Un tube a une capacité limitée. [...] Différentes implémentations ont différentes limites de capacité des tubes.*
> *[...]*
> *Dans les versions de Linux antérieures à 2.6.11, la capacité d'un tube était la taille d'une page système (p.ex. 4096 octets sur i386). Depuis Linux 2.6.11, la capacité d'un tube est de 65536 octets.*

Résultat

```
export LD_LIBRARY_PATH="/opt/vc/lib:/usr/lib/omxplayer"
raspivid -t 0 -w 1280 -h 720 -fps 25 -n -o - |
    omxplayer.bin /dev/stdin
```

Résultat

```
export LD_LIBRARY_PATH="/opt/vc/lib:/usr/lib/omxplayer"
raspivid -t 0 -w 1280 -h 720 -fps 25 -n -o - |
    omxplayer.bin /dev/stdin
```

L'enregistrement se bloque immédiatement.

FAIL #1

Un *tube* ne suffit pas !

Un buffer

mbuffer

Nous avons besoin d'un buffer plus important.

mbuffer

Nous avons besoin d'un buffer plus important.

```
raspivid -t 0 -w 1280 -h 720 -fps 25 -n -o - |
    mbuffer -m 10m | # un buffer de 10Mo
    omxplayer.bin /dev/stdin
```

mbuffer

Nous avons besoin d'un buffer plus important.

```
raspivid -t 0 -w 1280 -h 720 -fps 25 -n -o - |
    mbuffer -m 10m | # un buffer de 10Mo
    omxplayer.bin /dev/stdin
```

On dirait que ça marche !

Décalage

Nous pouvons démarrer omxplayer.bin plus tard :

```
raspivid -t 0 -w 1280 -h 720 -fps 25 -n -o - |
    mbuffer -m 10m | # un buffer de 10Mo
    { sleep 5; omxplayer.bin /dev/stdin; }
```

Décalage

Nous pouvons démarrer omxplayer.bin plus tard :

```
raspivid -t 0 -w 1280 -h 720 -fps 25 -n -o - |
    mbuffer -m 10m | # un buffer de 10Mo
    { sleep 5; omxplayer.bin /dev/stdin; }
```

omxplayer.bin est tellement long à démarrer que c'est inutile...

Décalage non constant

- Le décalage augmente petit à petit au fil des minutes.
- Le buffer se remplit légèrement plus vite qu'il ne se vide.

Décalage non constant

- Le décalage augmente petit à petit au fil des minutes.
- Le buffer se remplit légèrement plus vite qu'il ne se vide.

Si on enregistre à 24 `fps` au lieu de 25 `fps`:

- Le décalage est rattrapé progressivement ($\rightarrow$ direct).
- Le buffer se vide plus vite qu'il ne se remplit.

Cause

$$fps\_réel_{enregistrement} > fps\_réel_{lecture}$$

Cause

$$fps\_réel_{enregistrement} > fps\_réel_{lecture}$$

Pourtant :

- raspivid est censé enregistrer à 25 fps (`-fps 25`).
- omxplayer nous indique dans la console qu'il lit à 25 fps.

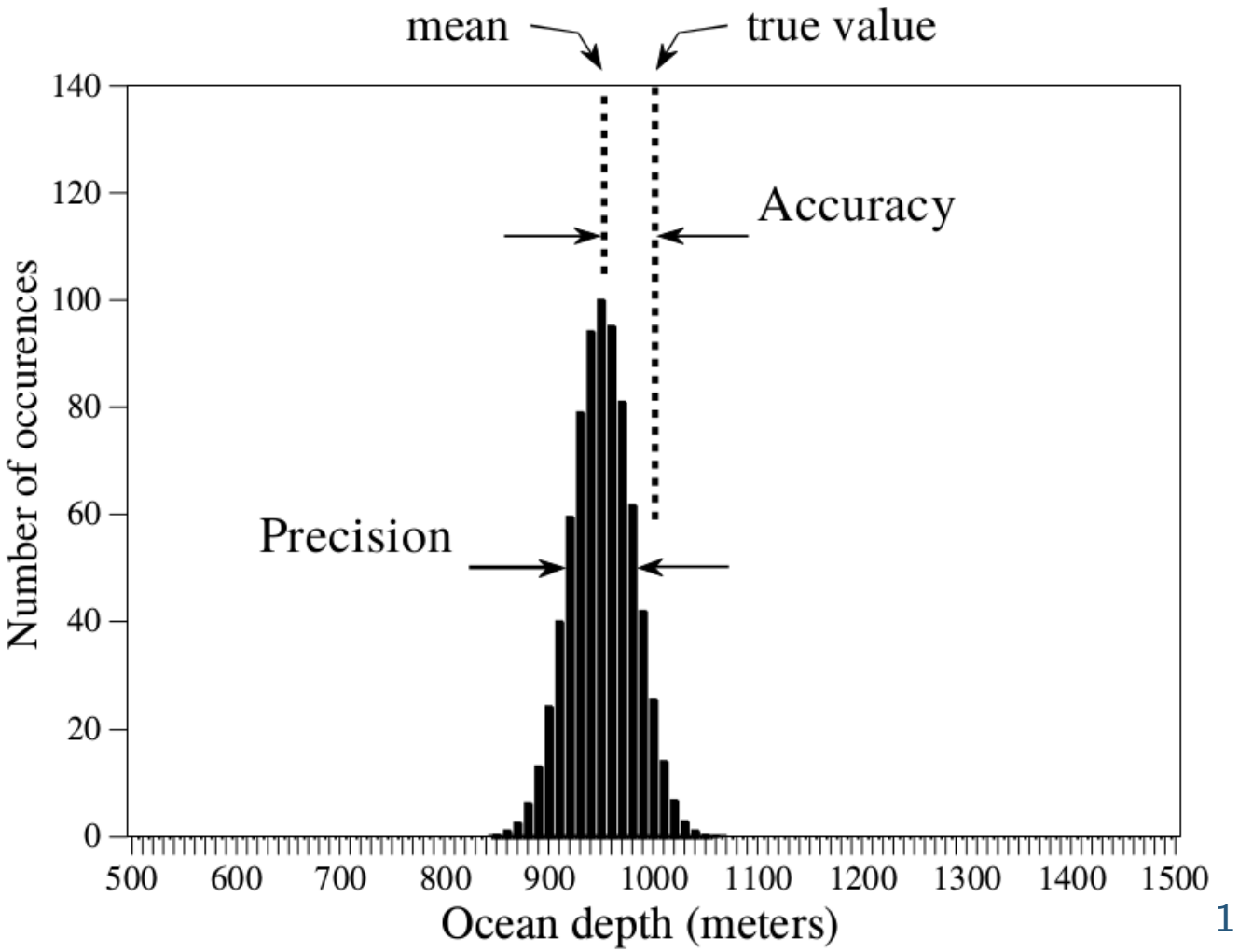
Inexactitude

- soit le débit d'images réel de l'**enregistrement** est légèrement **supérieur** à 25 ;
- soit le débit d'images réel de la **lecture** est légèrement **inférieur** à 25 ;
- soit les deux. . .

Inexactitude

- soit le débit d'images réel de l'**enregistrement** est légèrement **supérieur** à 25 ;
- soit le débit d'images réel de la **lecture** est légèrement **inférieur** à 25 ;
- soit les deux...

La valeur est *inexacte*, pas *imprécise*.



[1]

[1] http://www.dspguide.com/ch2/7.htm

FAIL #2

Un *buffer* ne suffit pas !

Informations temporelles

Nombre d'images par seconde

- omxplayer.bin produit un flux H.264 brut
  - pas de *timestamps*
  - pas de *fps*

Nombre d'images par seconde

- omxplayer.bin produit un flux H.264 brut
  - pas de *timestamps*
  - pas de *fps*
- Nous connaissons le nombre de *fps* de l'enregistrement et de la lecture
  - mais sa valeur est *inexacte*

Nombre d'images par seconde

- omxplayer.bin produit un flux H.264 brut
  - pas de *timestamps*
  - pas de *fps*
- Nous connaissons le nombre de *fps* de l'enregistrement et de la lecture
  - mais sa valeur est *inexacte*

Nous ne pouvons donc rien faire !?

Le direct

Une autre information temporelle : le flux est **en direct**.

Comment l'exploiter

Pour le comprendre :

- Enregistrement à 5 fps
- Lecture à 25 fps

Comment l'exploiter

Pour le comprendre :

- Enregistrement à 5 fps
- Lecture à 25 fps

À partir d'un **fichier**, la lecture sera à 25 fps en accéléré.

Comment l'exploiter

Pour le comprendre :

- Enregistrement à 5 fps
- Lecture à 25 fps

À partir d'un **fichier**, la lecture sera à 25 fps en accéléré.

En **direct**, la lecture sera à 5 fps à vitesse réelle.

Synchronisation

Sous ces conditions :

- l'enregistrement est en **direct** ;
- le débit d'images d'enregistrement est **inférieur** à celui de lecture ;

sans même connaître sa valeur exacte, nous parvenons à obtenir le **même débit d'images à l'enregistrement qu'à la lecture**.

Synchronisation

Sous ces conditions :

- l'enregistrement est en **direct** ;
- le débit d'images d'enregistrement est **inférieur** à celui de lecture ;

sans même connaître sa valeur exacte, nous parvenons à obtenir le **même débit d'images à l'enregistrement qu'à la lecture**.

☹ Par contre, le retard sera rattrapé ($\neq$ délai constant).

Un délai

On avance

Si nous pouvions retarder la sortie de raspivid de x secondes, alors omxplayer.bin rattrapera le retard pour parvenir au direct...
décalé de x secondes.

On avance

Si nous pouvions retarder la sortie de raspivid de x secondes, alors omxplayer.bin rattrapera le retard pour parvenir au direct. . . décalé de x secondes.

- débit d'images strictement égal entre l'enregistrement et la lecture
- décalage constant configurable

Commande shell ?

Existe-t-il une commande shell `X` telle que, quand on exécute :

```
command_A | X 5000 | command_B
```

la sortie standard de `command_a` soit écrite sur l'entrée standard de `command_b` (au moins) 5 secondes plus tard ?

Commande shell ?

Existe-t-il une commande shell `X` telle que, quand on exécute :

```
command_A | X 5000 | command_B
```

la sortie standard de `command_a` soit écrite sur l'entrée standard de `command_b` (au moins) 5 secondes plus tard ?

```
raspivid -fps 24 ... | X 5000 | omxplayer.bin ...
```

stackoverflow Questions Tags Users Badges Unanswered

Is there a shell command to delay a buffer?

6

3

I am looking for a shell command X such as, when I execute:

```
command_a | X 5000 | command_b
```

the `stdout` of `command_a` is written in `stdin` of `command_b` (at least) 5 seconds later.

A kind of delaying buffer.

As far as I know, `buffer` / `mbuffer` can write at constant **rate** (a fixed number of bytes per second). Instead, I would like a constant **delay** in time (t=0 is when `X` read a `command_a` output chunk, at t=5000 it must write this chunk to `command_b`).

[edit] I've implemented it: https://github.com/rom1v/delay

bash shell

share | edit | delete | flag

edited Jan 13 at 19:26

asked Jan 7 at 19:04
rom1v
83 ●6

[2]

[2] http://stackoverflow.com/questions/20979694/is-there-a-shell-command-to-delay-a-buffer

Fail #3

Cette commande n'existe pas.

delay

Je l'ai donc implémentée[3] (licence GPLv3) :

```
delay [-b <dtbufsize>] <delay>
```

[3] `git clone http://git.rom1v.com/delay.git`
ou https://github.com/rom1v/delay.

delay

Je l'ai donc implémentée[3] (licence GPLv3) :

```
delay [-b <dtbufsize>] <delay>
```

Principe :

- *timestamp* sur chaque paquet lu
- enregistrement dans un buffer circulaire
- écriture "au bon moment" sur la sortie standard

[3] `git clone http://git.rom1v.com/delay.git`
ou https://github.com/rom1v/delay.

Débit max

`delay [-b <dtbufsize>] <delay>`

Choisir `dtbufsize` pour avoir un bitrate suffisant :

$$bitrate_{max} = \frac{dtbufsize}{delay}$$

Exemple

Ça marche (sur mon laptop) :

```
ffmpeg -an -s 320x240 -f video4linux2 \
       -i /dev/video0 -f mpeg2video -b 1M - |
  delay -b10m 6s |
  vlc -
```

Pour omxplayer

Il suffit donc d'exécuter :

```
raspivid -t 0 -w 1280 -h 720 -fps 24 -n -o - |
  delay -b10m 6s |
  omxplayer.bin /dev/stdin
```

Pour omxplayer

Il suffit donc d'exécuter :

```
raspivid -t 0 -w 1280 -h 720 -fps 24 -n -o - |
  delay -b10m 6s |
  omxplayer.bin /dev/stdin
```

En fait, non.

- omxplayer est (très) long à s'initialiser
- il ne commence son initialisation qu'à partir du moment où il a *assez* de contenu vidéo

FAIL #4

La commande `delay` ne suffit pas !

Initialisation immédiate

On veut initialiser le player sans délai.

Initialisation immédiate

On veut initialiser le player sans délai.

- passer les premiers (méga-)octets immédiatement
- passer le reste décalé du délai souhaité

Initialisation immédiate

On veut initialiser le player sans délai.

- passer les premiers (méga-)octets immédiatement
- passer le reste décalé du délai souhaité

```
raspivid -t 0 -w 1280 -h 720 -fps 24 -n -o - |
  { head -c10M; delay -b10m 6s; } |
  omxplayer.bin /dev/stdin
```

WIN

Ça marche !

Liens

Des liens

ces slides	http://dl.rom1v.com/delayconf.pdf
leurs sources	`git clone http//git.rom1v.com/delayconf.git`
un billet de blog	http://blog.rom1v.com/2014/01/lecture-differee-de-la-webcam-dun-rasberry-pi/