

Installation de la librairie OpenCV sous Ubuntu

Par X. HINAULT - Septembre 2011 - www.mon-club-elec.fr

> Soit à partir des paquets sous Ubuntu

> Sous Ubuntu 10.04, les paquets installés sont ceux de la version openCV 2.0 qui sont au nombre de 4 :

libcv4, libhighgui4, libcvaux4, libcv-dev, libcvaux-dev, libhighgui-dev

> Ces librairies sont ensuite disponibles dans le répertoire /usr/lib

> Il peut être nécessaire ensuite de créer des liens symboliques pour que les programmes puissent utiliser la version installée...

```
$ sudo ln -s libxcv.so libxcv.so.1
$ sudo ln -s libcv.so libcv.so.1
$ sudo ln -s libhighgui.so libhighgui.so.1
$ sudo ln -s libcvaux.so libcvaux.so.1
$ sudo ln -s libml.so libml.so.1
```

Pour avoir la liste des liens en place :

```
$ ls -l /usr/lib/libcx*.*
```

> Soit installation à partir des sources et compilation

Intro :

> La version OpenCV disponible dans les dépôts Ubuntu, notamment pour la version LTS (10.04 à ce jour) n'est pas forcément la plus récente et il devient tentant d'installer la dernière version en la compilant directement depuis les sources. Suivez le guide !

Liens utiles :

- > http://opencv.itseez.com/doc/tutorials/introduction/linux_install/linux_install.html#linux-installation
- > <http://www.samontab.com/web/2011/06/installing-opencv-2-2-in-ubuntu-11-04/>

Paquets nécessaires au préalable :

voir > http://opencv.itseez.com/doc/tutorials/introduction/linux_install/linux_install.html#linux-installation

Notamment : GCC 4.x or later. This can be installed with

```
sudo apt-get install build-essential
```

- **CMake** 2.6 or higher
- Subversion (SVN) client
- GTK+2.x or higher, including headers
- pkgconfig
- libpng, zlib, libjpeg, libtiff, libjasper with development files (e.g. libjpeg-dev)
- **Python 2.3** or later with developer packages (e.g. **python-dev**)
- **SWIG** 1.3.30 or later (only for versions prior to OpenCV 2.3)
- libavcodec
- libdc1394 2.x

> Télécharger les sources :

>> <http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.3.1/OpenCV-2.3.1a.tar.bz2/download>

> Extraire l'archive

> se positionner dans le répertoire obtenu :

```
$ cd ~/Téléchargements/OpenCV-2.3.1
```

> créer un répertoire de compilation et se placer dedans :

```
$ mkdir rep_compil
```

```
$ cd rep_compil
```

> Lancer le Cmake (../ fait remonter d'un niveau..) :

```
$ sudo cmake ../
```

> Lancer ensuite la compilation :

```
$ sudo make
```

```
[ 1%] Generating precomp.hpp.gch/opencv_imgproc_Release.gch
[ 2%] Built target pch_Generate_opencv_imgproc
[ 2%] Generating opencv_core_pch_dephelp.cxx
Scanning dependencies of target opencv_core_pch_dephelp
[ 2%] Building CXX object modules/core/CMakeFiles/opencv_core_pch_dephelp.dir/opencv_core
Linking CXX static library ../../lib/libopencv_core_pch_dephelp.a
[ 2%] Built target opencv_core_pch_dephelp
Scanning dependencies of target pch_Generate_opencv_core
[ 2%] Generating precomp.hpp
[ 2%] Generating precomp.hpp.gch/opencv_core_Release.gch
[ 3%] Built target pch_Generate_opencv_core
Scanning dependencies of target opencv_core
[ 3%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/matop.o
[ 3%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/datastructs.o
[ 3%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/dxt.o
[ 3%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/convert.o
[ 4%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/matmul.o
[ 4%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/rand.o
[ 4%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/array.o
[ 4%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/matrix.o
[ 4%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/system.o
[ 5%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/mathfuncs.o
[ 5%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/drawing.o
[ 5%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/copy.o
[ 5%] Building CXX object modules/core/CMakeFiles/opencv_core.dir/src/lapack.o
```

Jusqu'à obtenir après quelques minutes voire dizaines de minutes :

```
Scanning dependencies of target opencv_stitching
[ 98%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/matchers.o
[ 98%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/exposure_compensate.o
[ 98%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/motion_estimators.o
[ 98%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/blenders.o
[ 99%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/warpers.o
[ 99%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/main.o
[ 99%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/seam_finders.o
[ 99%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/util.o
[ 99%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/autocalib.o
[100%] Building CXX object modules/stitching/CMakeFiles/opencv_stitching.dir/precomp.o
Linking CXX executable ../../bin/opencv_stitching
[100%] Built target opencv_stitching
hinault@hinault-desktop:~/Téléchargements/OpenCV-2.3.1/rep_compil$
```

> Ensuite, on finalise l'installation

```
$ sudo make install
```

> A présent, il faut figner l'installation. Editer le fichier :

```
$ sudo gedit /etc/ld.so.conf.d/opencv.conf
```

> Ajouter la ligne :

```
/usr/local/lib
```

et enregistrer

> Ensuite, lancer la ligne de configuration de la librairie :

```
$ sudo ldconfig
```

> Puis éditer le fichier suivant :

```
$ sudo gedit /etc/bash.bashrc
```

> Ajouter les lignes :

```
#----- opencv ----
```

```
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig
```

```
export PKG_CONFIG_PATH
```

> Enregistrer puis quitter/relancer un terminal pour prise en compte des chemins indiqués.

> Voilà, à priori, c'est bon pour l'installation de openCV...

Etape optionnelle :

> Au besoin, il reste éventuellement à mettre en place au besoin quelques liens symboliques pour rendre compatible la version compilée et la version utilisée avec javacv. Dans mon cas, javacv utilise la version 2.3 et la version installée est 2.3.1 (pas nécessaire dans mon cas - au besoin pour voir les liens en place, faire `ls /usr/local/lib/libopencv*`)

```
sudo ln /usr/local/lib/libopencv_core.so.2.3 /usr/local/lib/libopencv_core.so.2.3.1
```

```
sudo ln /usr/local/lib/libopencv_imgproc.so.2.3  
/usr/local/lib/libopencv_imgproc.so.2.3.1
```

```
sudo ln /usr/local/lib/libopencv_highgui.so.2.3  
/usr/local/lib/libopencv_highgui.so.2.3.1
```

```
sudo ln /usr/local/lib/libopencv_highgui.so.2.3  
/usr/local/lib/libopencv_highgui.so.2.3.1
```

```
sudo ln /usr/local/lib/libopencv_flann.so.2.2 /usr/local/lib/libopencv_flann.so.2.3
```

```
sudo ln /usr/local/lib/libopencv_calib3d.so.2.3  
/usr/local/lib/libopencv_calib3d.so.2.3.1
```

A présent, plusieurs possibilités se présentent :

> soit travailler en C/C++ : faut avoir l'habitude d'utiliser des outils C/C++ pour le faire... et franchement, c'est pas le plus simple... L'avantage par contre, c'est de pouvoir utiliser « tels que » les codes d'exemple fournis sur le site d'OpenCV.

> soit travailler en Java en se basant sur javacv, dans un IDE tel que Eclipse (pas forcément simple si on n'est pas habitué à travailler dans Eclipse. Sinon, ce n'est pas très compliqué...)

> soit travailler depuis Processing en appelant les fonctions Opencv « in line » dans le code Processing, ce qui nécessite au préalable d'installer la librairie javacv en tant que librairie utilisable avec Processing (simple à faire).

> soit d'utiliser ma librairie javacvPro qui permet l'utilisation simplifiée d'OpenCV dans Processing, à partir d'objet Pimage.

Il faudra aussi prendre le temps :

> de « découvrir » la structure générale de la librairie OpenCV, de son organisation en « modules thématiques », des objets fondamentaux de la librairie et des centaines de fonctions disponibles. Le mieux est de parcourir la documentation de la librairie OpenCV elle-même : <http://opencv.itseez.com/>

> de se familiariser avec les principales opérations possibles en traitement d'image et reconnaissance visuelle. Il sera sinon difficile de comprendre l'intérêt des fonctions fournies par OpenCV. Si Canny, Hough, Sobel ou encore « noyau de convolution » ne vous disent absolument rien, faites un tour ici :

>> http://en.wikipedia.org/wiki/Feature_detection_%28computer_vision%29 (excellente porte d'entrée pour avoir une vue d'ensemble des techniques possibles pour un traitement d'image avancé).

>> http://www.ensta-paristech.fr/~manzaner/Support_Cours.html (cours abordant tous les concepts du traitement d'image, très intéressant, mais ardu...)

Vous pouvez aussi choisir d'approfondir les différents concepts au fur et à mesure de leur découverte.