Bitácora de trabajo
Gustavo Ramírez Hidalgo
Proyecto final - SP2118

---

1. Descripción general:

El proyecto consiste en el empleo de OpenCV para el estudio de cierto tipo de videos generados a partir de la técnica de física experimental llamada LEED (low-energy electron diffraction), y generar a partir de ellos, en última instancia, una visualización tridimensional de las capas de material que pudieron ser accesadas por las ondas electromagnéticas, y que por lo tanto son las que se pudieron visualizar (hay que tener claro que esta es una visualización a través de modelaje).

2. Sobre LEED:

3. Procedimientos iniciales:

Para la instalación y puesta en marcha de la biblioteca de Python llamada OpenCV, se hizo uso de la documentación oficial de dicha biblioteca.[1]
De dicha documentación oficial, los tutoriales útiles fueron los 3 primeros, a saber: *Introduction to OpenCV*, *Gui features in OpenCV* y *Core operations*.
Se dará a continuación una descripción general de cada uno de esos tutoriales; principalmente se hará mención de los aspectos más importantes para el desarrollo de este proyecto:

3.1. *Introduction to OpenCV*

Antes que todo, es importante hacer mención de algunas generalidades de OpenCV. OpenCV fue iniciado en Intel, en 1999. OpenCV-Python es el API de Python para OpenCV. Este API hace uso de la capacidad que se tiene en Python de crear *wrappers*, los cuales consisten en código de C o C++, el cual luego se empaqueta y es llamado desde Python. Así, básicamente lo que se hace en OpenCV-Python es un uso directo del API de OpenCV para C++, y se aprovechan además ciertas características importantes de Python.
Un buen conocimiento de Numpy es requerido[2], en caso de que se quiera hacer un uso provechoso (desde el punto de vista de la escritura de códigos optimizados) de OpenCV-Python.
En cuanto a la versión de Python que se utilizará en este proyecto, es la 2.7.4. En Ubuntu (el sistema operativo sobre el cual se desarrolló este programa), Python viene instalado por defecto, así como python-numpy.
Por otro lado, es necesario instalar OpenCV:

```
$ sudo aptitude install libopencv-dev
```

Pero una instalación de este tipo no funcionó. Se procedió a desinstalar:

```
$ sudo aptitude purge libopencv-dev
```

Y después, se optó por la instalación a través del código fuente[3]. Se descargó la versión 2.4.8[4]. Luego, en el directorio /home/user/Downloads, se ejecutaron los siguientes comandos:

[1] http://docs.opencv.org/trunk/doc/py_tutorials/py_tutorials.html
[2] http://wiki.scipy.org/Tentative_NumPy_Tutorial
[3] http://stackoverflow.com/questions/15790501/why-cv2-so-missing-after-opencv-installed
[4] http://opencv.org/downloads.html

```
$ unzip opencv-2.4.8.zip
$ cd opencv-2.4.8
$ mkdir release
$ cd release
$ cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local -D
BUILD_NEW_PYTHON_SUPPORT=ON -D BUILD_EXAMPLES=ON ..
$ make
$ sudo make install
```

Incluso después de esto, parece ser necesario llevar a cabo un par de instalaciones más:

```
$ sudo aptitude install python-dev
$ sudo aptitude install python-opencv
```

Luego, basta con ejecutar en la terminal:

```
$ python
```

y en el intérprete interactivo de Python que sale a continuación, se escribe:

```
>> import cv2
```

y si no da error, quiere decir que todo está bien con la instalación de OpenCV.
Además, es importante instalar la siguiente biblioteca de Python (la cual se complementa muy bien con OpenCV, y permite, entre otras cosas, desplegar imágenes y llevar a cabo ciertas operaciones especiales sobre ellas, como por ejemplo aplicar zoom):

```
$ sudo aptitude install python-matplotlib
```

3.2. *Gui features in OpenCV*

Al ir desarrollando el programa, se fueron presentando problemas en la ejecución (principalmente con el método VideoCapture del objeto cv2). Después de varias búsquedas, se encontró que la solución es hacer una instalación adecuada de ffmpeg[5]. Así, se llevaron a cabo los siguientes pasos (se descargó la última versión de ffmpeg a través de un repositorio git [6]):

```
$ git clone git://source.ffmpeg.org/ffmpeg.git ffmpeg
$ sudo aptitude install libfaac-dev
$ wget http://downloads.sourceforge.net/project/lame/lame/3.98.4/lame-3.98.4.tar.gz?
r=http%3A%2F%2Fffmpeg.zeranoe.com%2Fforum%2Fviewtopic.php%3Ff%3D5%26t%3D94&ts=133914
0293&use_mirror=ignum
$ tar -xvzf lame-3.98.4.tar.gz
$ cd lame-3.98.4
$ ./configure
$ make
$ sudo make install
$ sudo aptitude install libopencore-amrwb-dev
$ sudo aptitude install libopencore-amrnb-dev
$ sudo aptitude install libtheora-dev
$ sudo aptitude install libvorbis-dev
$ sudo aptitude install libx264-dev
$ sudo aptitude install libxvidcore-dev
```

[5] http://answers.opencv.org/question/263/videocapture-is-not-working-in-opencv-242/
[6] http://www.ffmpeg.org/download.html

```
$ sudo aptitude install libxext-dev
$ sudo aptitude install libxfixes-dev
$ cd ffmpeg
$ ./configure --enable-gpl --enable-libfaac --enable-libmp3lame --enable-libopencore
-amrnb --enable-libopencore-amrwb --enable-libtheora --enable-libvorbis --enable
-libx264 --enable-libxvid --enable-nonfree --enable-postproc --enable-version3 --enable
-x11grab --disable-yasm
$ make
$ sudo make install
$ cd opencv-2.4.8/release
$ cmake -D CMAKE_BUILD_TYPE=RELEASE ..
$ make
$ sudo make install
```

Pero se presentó un error en la recompilación. En específico, el error tiene que ver con: /usr/local/lib/libavcodec.a: could not read symbols: Bad value. La solución se encuentra en línea[7]. Para poder solucionar todos los errores, se tuvo que desinstalar todo nuevamente y hacer una instalación prácticamente desde cero.

```
$ sudo aptitude remove ffmpeg x264 libx264-dev
$ sudo aptitude update
$ sudo apt-get install build-essential checkinstall git cmake libfaac-dev libjack-
jackd2-dev libmp3lame-dev libopencore-amrnb-dev libopencore-amrwb-dev libsdl1.2-dev
libtheora-dev libva-dev libvdpau-dev libvorbis-dev libx11-dev libxfixes-dev
libxvidcore-dev texi2html yasm zlib1g-dev
$ sudo apt-get install libgstreamer0.10-0 libgstreamer0.10-dev gstreamer0.10-tools
gstreamer0.10-plugins-base libgstreamer-plugins-base0.10-dev gstreamer0.10-plugins-
good gstreamer0.10-plugins-ugly gstreamer0.10-plugins-bad gstreamer0.10-ffmpeg
$ cd opencv-2.4.8
$ sudo make uninstall
```

y se optará por trabajar con la versión 2.4.2 de OpenCV:

```
$ sudo apt-get install libgtk2.0-0 libgtk2.0-dev
$ sudo apt-get install libjpeg8 libjpeg8-dev
$ mkdir src
$ cd src
$ wget ftp://ftp.videolan.org/pub/videolan/x264/snapshots/x264-snapshot-20120528-2245
-stable.tar.bz2
$ tar xvf x264-snapshot-20140206-2245.tar.bz2
$ cd x264-snapshot-20140206-2245/
$ ./configure --enable-shared --enable-pic
$ make
$ sudo make install
$ mkdir src0
$ cd src0
$ wget http://ffmpeg.org/releases/ffmpeg-0.11.1.tar.bz2
$ tar xvf ffmpeg-0.11.1.tar.bz2
$ cd ffmpeg-0.11.1
$ ./configure --enable-gpl --enable-libfaac --enable-libmp3lame --enable-libopencore
-amrnb --enable-libopencore-amrwb --enable-libtheora --enable-libvorbis --enable-libx264
```

[7] http://www.ozbotz.org/opencv-installation/

```
--enable-libxvid --enable-nonfree --enable-postproc --enable-version3 --enable
-x11grab --enable-shared --enable-pic
$ make
$ sudo make install
```

En algún momento, la instalación se complicó, debido a ciertos llamados inadecuados entre funciones de ciertos archivos, todo esto relacionado con libavcodec y ffmpeg. Se fue al código fuente original de ffmpeg, y se ejecutó en la terminal:

```
$ sudo make uninstall
```

Los segmentos de línea –enable-shared y –enable-pic son muy importantes, ya que previenen los errores que generaron toda la reinstalación de OpenCV (aquellos de no poder utilizar VideCapture).
Continuando:

```
$ wget http://www.linuxtv.org/downloads/v4l-utils/v4l-utils-1.0.1.tar.bz2
$ tar xvf v4l-utils-1.0.1.tar.bz2
$ cd v4l-utils-1.0.1
$ ./configure
$ make
$ sudo make install
```

Y finalmente, se hace la instalación de OpenCV versión 2.4.2:

```
$ wget http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.4.2/OpenCV
-2.4.2.tar.bz2
$ tar xvf OpenCV-2.4.2.tar.bz2
$ cd OpenCV-2.4.2/
$ mkdir build
$ cd build
$ cmake -D CMAKE_BUILD_TYPE=RELEASE ..
$ make
$ sudo make install
```

Después de la instalación, hay que llevar a cabo algunas configuraciones en linux (para que identifique de manera adecuada todos los elementos de OpenCV):

```
$ export LD_LIBRARY_PATH=/usr/local/lib
(agregar este de arriba a .bashrc)
```

Además, hay que agregar las siguientes dos líneas a /etc/bash.bashrc:

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

Y listo! Con estas configuraciones ya es posible hacer uso de OpenCV sin problemas.

Ahora bien, en cuanto tamaño de video, es mejor hacer un reajuste antes de ejecutar el programa. Todos los videos son pasados al mismo tamaño (se ha elegido XxX, de manera levemente arbitraria). La transformación se hace básicamente con el siguiente comando (se agrega aquí con un ejemplo incorporado):

```
/home/gustavo/Downloads/src0/ffmpeg-0.11.1/ffmpeg -i LEEDgraphite2.wmv -s 640x480
-b:v 512k -vcodec mpeg1video -acodec copy LEEDtransformed2.wmv
```

Además del archivo llamado 'programa.py', que es el script en Python, llamado a su vez por el script en Bash 'general.py', se hizo un script (también en Bash) para transformar todos los archivos que vayan a estar adentro de 'videos/'.

Aparte de todo lo mencionado anteriormente, cabe rescatar algunos puntos importantes de esta (segunda) sección de tutoriales.
Los detalles de los scripts mismos se pueden encontrar en ellos, los cuales ya vienen con su correspondiente documentación. De esta sección, es importante mencionar lo siguiente:

- Los son tratables exactamente igual que las imágenes que se leen de archivos .png, .jpg, etc., y se les puede asignar eventos.
- El método waitKey(time) resulta ser muy útil, especialmente cuando se quiere detener un video, dibujar sobre él, y luego retomar la reproducción del video en ese punto de detención previo.
- La reproducción de videos con OpenCV es trucosa, y hay que tener en cuenta que el último frame al que se accesa (en caso de hacer uso de un while, combinado con el método isOpened del objeto generado a partir del llamado de la función cv2.VideoCapture(str)) no existe, por lo que hay que hacer una validación en ese caso.
- La combinación de waitKey(time) y destroyWindow(str), hacen posible una finalización del programa sin problema alguno, a pesar de interrumpirse incluso.
- El uso de círculos y polígonos como objetos para dibujo, resulta muy útil en el trazo de lazos cerrados sobre frames. Pero hay que hacer un llamado directo del frame (es decir, pasarlo por parámetro a la función empleada en la implementación del evento) o tenerlo como variable global. Si no se tiene acceso al frame sobre el que se quiere dibujar, es claro que no se podrá dibujar sobre él.
- El uso de listas anidadas resulta muy útil, e incluso indispensable, en el redibujado de lazos cerrados sobre los frames posteriores al frame sobre el que se dibujó o dibujaron el o los lazos.
- El dibujado de figuras sobre imágenes, así como la implementación de eventos (como click derecho, por ejemplo) en las mismas, resultan las herramientas básicas e indispensables para el uso del mouse como herramienta de dibujo sobre un video.

3.3. *Core operations*