

Introducción a la librería CImg y puesta en funcionamiento

Captura y Procesamiento Digital de Señales e Imágenes

Departamento de Informática, FICH - UNL

17 de marzo de 2009

1. Introducción

Este documento presenta una reseña de la librería CImg para procesamiento de imágenes en C++. Se explica brevemente su instalación, compilación y las funciones fundamentales para cargar una imagen de archivo, obtener información de la misma, visualizarla y otras cuestiones de manejo básico,

La librería y su documentación está disponible en:

<http://cimg.sourceforge.net>

2. Instalación

Previo a su instalación es necesario tener instalado:

- un compilador de C++ (paquete `gcc`),
- el programa ImageMagick (paquete `imagemagick`) para la creación de las ventanas, disponible en <http://www.imagemagick.org>. Este paquete está preinstalado en varias distribuciones de Linux, para saber si ya está instalado ejecutar la instrucción `display` desde un terminal (debe mostrar la ventana del programa),
- la librería de manejo de las X en Linux (paquete `libX11-dev`).

En Linux descarga la librería mediante

```
wget http://cimg.sourceforge.net/cimg-dev.deb
```

e instala con

```
dpkg -i cimg-dev.deb
```

o bien bajando los archivos del sitio web e instalando manualmente.

3. Compilación

Durante la compilación con `g++` de un código fuente `prog.cpp` en Linux es necesario indicar las librerías enlazadas, de la siguiente manera:

```
g++ -o prog prog.cpp -O2 -lm -lpthread -lX11
```

Si se edita el archivo fuente con Anjuta, es posible dejar como parámetros de la compilación las librerías, indicando en el menú “Opciones/Opciones del compilador y enlazador”, solapa “Opciones”, campo “Flags del compilador (CFLAGS)”:

```
-L/usr/X11R6/lib -lm -lpthread -lX11.
```

Para compilar en otras plataformas y compiladores ver:

http://cimg.sourceforge.net/reference/group__cimg__overview.html

4. Manejo básico de una imagen

- Crear una imagen vacía:

```
CImg<tipo> img(dx,dy,dz,dv,val);
```

El tipo de dato utilizado para guardar los valores de la imagen es variado: `unsigned char`, `int`, `float`,... El tipo por defecto es `float`, accesible mediante la instrucción `CImg<> img...`

El resto de los parámetros son:

- `dx`: ancho, número de columnas.
- `dy`: alto, número de filas.
- `dz`: profundidad, número de cuadros.
- `dv`: número de canales, ej 1 (escala de grises), 3 (color).
- `val`: valor de inicialización de los píxeles, ej: 0 (negro).

Ejemplos:

- `CImg<unsigned char>img(640,480)`: imagen en tonos de grises.
- `CImg<unsigned char>img(640,480,1,1,0)`: idem anterior, pero inicialmente negra.
- `CImg<>img(3,3,1,1,1/9)`: imagen de 3x3 en tonos de grises inicializada con valores 1/9 (utilizada como máscara para filtrado de promediado).
- `CImg<int>img(200,100,1,3)`: imagen color.

- Crear una imagen cargándola de un archivo:

```
CImg<tipo> img("archivo.ext");
```

La instrucción anterior equivale a:

```
CImg<tipo> img;  
img.load("archivo.ext");
```

- Campos de la estructura `CImg`: accesibles a través de las siguientes funciones, mediante la instrucción `img.funcion`:

- `dimx()`: ancho, número de columnas.

- `dimy()`: alto, número de filas.
- `dimz()`: profundidad, número de cuadros.
- `dimv()`: dimensión del pixel, número de canales.
- `ptr()`: datos, puntero a la matriz imagen.

La matriz se accede desde el elemento (0,0) en la esquina superior izquierda, hasta el elemento (ancho-1,alto-1) en la esquina inferior derecha.

■ Visualización de una imagen:

La matriz de imagen se visualiza en una estructura de datos especial que crea la ventana y además maneja eventos de teclado y mouse:

```
CImgDisplay ventana(img, "titulo",n);
```

El parámetro `img` es la matriz con la imagen propiamente dicha, mientras que el parámetro `n` es la normalización de la visualización (0: sin normalización, 1(default): normalizada al intervalo [0,255] ó 2: cambia el comportamiento por defecto a normalizada de ahí en adelante)¹. La cadena de caracteres `"titulo"` fija el título que aparecerá en la barra superior de la ventana.

El ejemplo siguiente muestra la creación de la ventana y un bucle que la mantiene abierta hasta pulsar la tecla Q:

```
CImgDisplay ventana(img, "titulo",0);
while (!ventana.is_closed && ventana.key!=cimg::keyQ) { }
```

Cambia/vuelve a pantalla completa:

```
ventana.toggle_fullscreen();
```

Un comportamiento especial es considerado cuando se debe sobreimprimir alguna información o gráfica sobre la imagen, por ej., para mostrar un cuadro con las coordenadas del puntero del mouse. En estos casos, en el bucle de espera se re-dibuja la ventana mediante una re-definición de la estructura donde los paréntesis entre el nombre de la imagen hacen referencia al contenido previo (de esta manera no se pierde la imagen cargada). Las órdenes de graficación extras y una llamada a la visualización mediante una tubería se muestra en la línea siguiente²:

```
CImg<tipo> (img).draw_line(...).display(ventana);
```

Otras veces le resultará útil hacer simplemente

```
img.display("titulo");
```

sin necesidad de crear una ventana previamente. Esta función visualiza una ventana con la imagen hasta que se produzca un evento de teclado, mostrando los valores y posición del píxel sobre el cual se pasa el mouse. A diferencia de la visualización con ventanas permite mostrar sólo una ventana a la vez; no obstante, es cómoda para ver resultados previos y/o obtener valores de píxeles particulares.

■ Grabación de una imagen:

Para guardar en disco una imagen, simplemente se llama a la función:

```
img.save("nombre.ext");
```

¹Más información en página 13 del manual de referencia.

²La llamada a `draw_line()` es solamente un ejemplo de cualquier función utilizable.

5. Funciones adicionales

- Agregar un comentario de ayuda al invocar el programa mediante la línea de comandos con la opción -h:
`cimg_usage("Este programa realiza...");`
- Pasar parámetros por línea de comandos:
`const char* filename = cimg_option("-i", "cameraman.tif", "Input Image File");`
Pasando la opción -i por línea de comandos podemos indicar otra imagen para la cadena `filename`, que por defecto tomará `cameraman.tif`.

6. Ejemplo completo

A continuación se muestra un programa ejemplo del tipo "Hello World":

```
#include <CImg.h>
using namespace cimg_library;

int main(int argc, char *argv[]) {
    cimg_usage("Ejemplo básico.");
    const char* filename = cimg_option("-i", "cameraman.tif", "Image file\n");

    //Imagen color, 8 bits de profundidad por componente.
    CImg<unsigned char> img(640,400,1,3);
    img.fill(0); //Asigna 0 a todos los píxeles
    unsigned char rojo[] = { 255,0,0 }; //Define color rojo
    img.draw_text("Hello World",100,100,rojo); //Escribe en (100,100).
    img.display("Hola..."); //Visualiza

    CImg<unsigned char> img1(filename); //Carga de disco
    //Copia lo que hay en img1 pero como valores de punto flotante
    CImg<float> img2(img1);
    img2.normalize(0,1); //Normaliza en [0,1]
    img2.sqrt(); //Aplica raíz cuadrada a cada pixel
    img2.resize(250,150); //Redimensiona a 250x150
    //Visualiza ambas imágenes a la vez
    CImgDisplay vent1(img1,"original"), vent2(img2,"procesada");
    while(!vent1.is_closed){}

    return 0;
}
```

7. Comentarios finales

Este documento es de libre distribución y reproducción total o parcial por cualquier medio. Comentarios y sugerencias a los contactos de cpdsi-fich.wikidot.com.