UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Visor de anaglifos con OpenCV

Apellidos, nombre	**Agustí i Melchor, Manuel** (magusti@disca.upv.es)
Departamento	**Departamento de Informática de Sistemas y Computadores (DISCA)**
Centro	Universitat Politècnica de València

1 Resumen de las ideas clave

En el desarrollo de aplicaciones multimedia se puede ver habitualmente el uso de la **imagen 3D** que busca mostrar objetos volumétricos; esto es, representar sobre una superficie plana (2D) conceptos tridimensionales para mostrar lo que estamos acostumbrados a ver en el mundo real. Para ello se han venido utilizando los mecanismos que se han desarrollado durante siglos para el dibujo, la pintura o el grabado: p. ej. el uso de luces y sombras, la superposición o la perspectiva. La intención de estas técnicas es plasmar lo que cerebro está recibiendo continuamente: valores de distancia o **visión 3D**, al combinar las imágenes de los ojos que, como dos cámaras (véase la Figura 1[1]) captan dos imágenes ligeramente diferentes del mismo objeto: es la visión binocular, compare las diferencias en las zonas iluminadas y las más oscuras de las fotografías a los lados de la Figura 1.

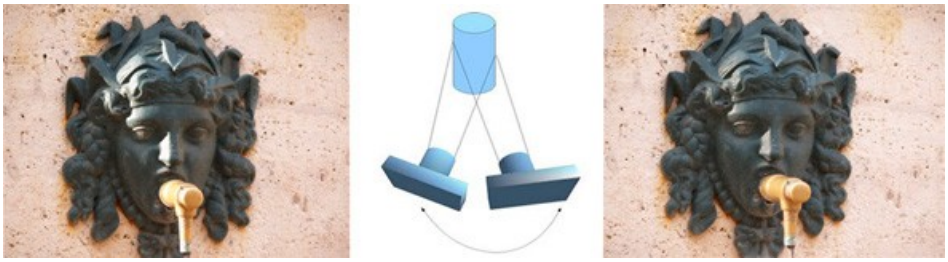
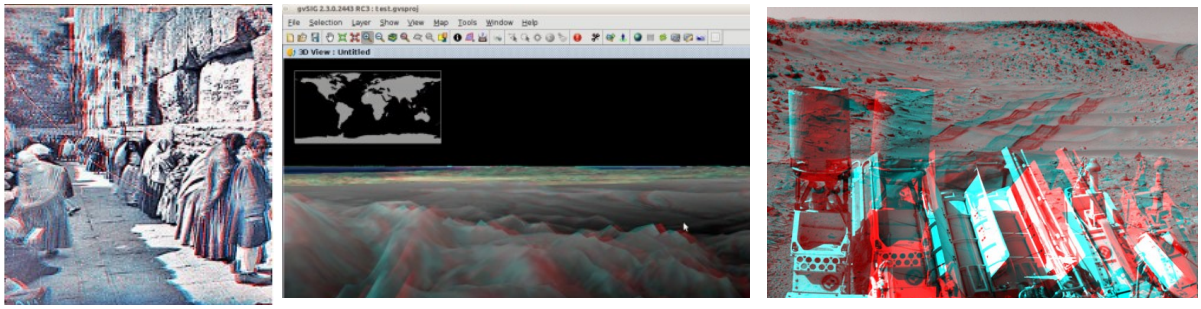


Figura 1: La estereoscopía es el principio básico de la imagen binocular.

¿Dos imágenes, eso cómo es? Se busca generar "imágenes 3D" que ofrezcan al usuario las sensaciones que su visión binocular le proporciona a diario. Para ello se utilizan diferentes técnicas que se denominan estereoscópicas y que se pueden ver utilizadas en diferentes dispositivos que ofrecen esa visualización 3D: desde hologramas (que tan familiares nos resultan gracias al cine), pasando por las videoconsolas, televisores, visores de Realidad Virtual o realizadas por software en aplicaciones de computadores de visualización de mapas, por citar algunos. En esta tarea se utilizan técnicas [1] como: multiplexación espacial (o *side by side*) que requiere gafas polarizadas o pasivas, multiplexación temporal que requiere gafas activas LCD, autoestereoscopía que no utiliza gafas (pero sí una barrera de paralaje o unas lentes semicilíndricas), los estereogramas que tampoco utilizan elementos externos (pero exigen del usuario un entrenamiento para focalizar y desenfocar la vista sobre las imágenes) y los **anaglifos**, que requieren de gafas con filtros de colores (habitualmente rojas y azules).



a) *b)* *c)*

Figura 2: Ejemplos de anaglifos: (a) Wikipedia, (b) gvSIG y (c) NASA.

[1] Imagen obtenida de "Cómo Realizar Fotos en 3D: Un Caso Práctico" <https://www.dzoom.org.es/como-realizar-fotos-en-3d-un-caso-practico/>.

Ejemplos de anaglifos se pueden ver en la Figura 2[2]. Los anaglifos constituyen una técnica de visualización de imágenes 3D a partir de mostrar superpuestas dos imágenes en una. Para poderlos ver es necesario utilizar un "visor": unas gafas que, con filtros de colores, proporcionan al usuario una visualización con mayor sensación de profundidad de la que se puede observar en una pantalla plana.

a) *b)*

Figura 3: Ejemplos de visores: (a) estereoscópicos (Wheatstone, Brewster o Holmes) y (b) de gafas para anagligos.

¿Gafas? ¿Qué gafas es necesario utilizar? Depende, pero desde el siglo XIX, los visores estereoscópicos[3] (Figura 3a) se han ido desarrollando y podemos encontrar diferentes modelos de gafas de anaglifos (en cartón o plástico, Figura 3b). Estas gafas son [1]: bastante más económicas que los visores que se necesitan en los otros tipos de técnicas estereoscópicas, su calidad de imagen también es menor, pero puede utilizarse en imágenes sobre cualquier soporte. ¿Y si no tengo gafas? En este artículo se va a utilizar OpenCV [3] para mirar dentro y fuera de un "anaglifo" y así entender cómo están formadas estas imágenes.

2 Objetivos

Una vez que el lector se lea con detenimiento este documento y explore el código que se adjunta, podrá ver el interior de un anaglifo en su computador, y con ello:

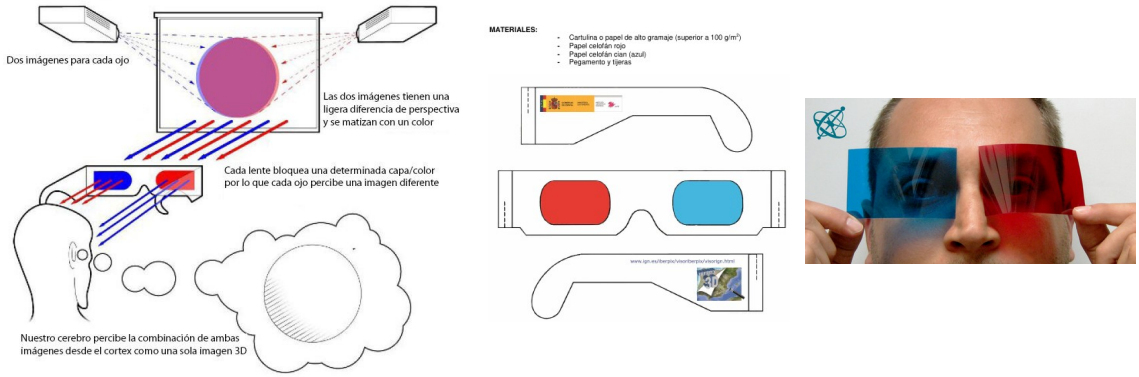
- Explicar cómo está construido un anaglifo.
- Explorar un ejemplo de visualización de anaglifos que permita comprobar empíricamente cómo es el contenido de los mismos.
- Ver el anaglifo sin necesidad de gafas de colores.

[2] Imágenes obtenidas de Wikipedia <https://en.wikipedia.org/wiki/Stereoscopy>, el proyecto gvSIG <https://blog.gvsig.org/2016/08/30/camino-a-gvsig-2-3-anaglifos/> o desde las páginas de la NASA <https://mars.nasa.gov/resources/6033/panoramic-view-from-west-of-dingo-gap-stereo/ > y existen colecciones muy interesantes en *Stereograph Cards* <https://www.loc.gov/pictures/collection/stereo/> y en *NASA Science MARS Exploration* <https://mars.nasa.gov/3d/images/?> .

[3] Las imágenes de la Figura 3 han sido obtenidas de <https://en.wikipedia.org/wiki/Stereoscope> y <https://en.wikipedia.org/wiki/Anaglyph_3D>.

3 Introducción

Aunque de manera informal ya se ha introducido el concepto de anaglifo, se puede formalizar como un sistema visual que permite superponer dos imágenes estereoscópicas de colores complementarios y ligeramente separadas en el eje horizontal, que al ser vistas, producen la sensación de relieve. Para verlos, un visor [1] ofrece a cada ojo una imagen algo diferente (véase la Figura 4a) a través de los dos filtros de color con que está equipado: en el caso de rojo-cian, el ojo cubierto por el filtro rojo ve las partes azules de la imagen y el ojo cubierto por el filtro azul ve el efecto opuesto. El resto de la composición es percibida de igual manera por los dos ojos. El cerebro fusiona las imágenes recibidas de cada ojo y las interpreta como una imagen con profundidad. Si no se tienen unas gafas de anaglifos puede construirse unas a partir de una plantilla[4] (Figura 4b) o, por lo menos, sujetando con las manos[5] (Figura 4c) dos filtros (plásticos) de los colores que se utilicen en la imagen de anaglifo. Ah, y si le da la vuelta al plástico de color en cada ojo, no pasa nada.



a) *b)* *c)*

Figura 4: Anaglifos: (a) planteamiento, (b) plantilla para fabricar unas gafas (imágenes de [1]) y c) ¿solo con las manos? (imagen de Ciensación).

Imágenes de gris: $Gris_{izqda}$ y $Gris_{dcha}$	**Imágenes de color: RGB_{izq} y RGB_{dcha}**		
	Método 1	Método 2	Método 3
$R_{final} = Gris_{izqda}$ $G_{final} = (Gris_{izda} + Gris_{izqda})/2$ $B_{final} = Gris_{dcha}$	$R_{final} = R_{izqda}$ $G_{final} = 0$ $B_{final} = B_{dcha}$	$R_{final} = R_{izqda}$ $G_{final} = (G_{izqda} + G_{dcha}) / 2$ $B_{final} = B_{right}$	$R_{final} = R_{izqda}$ $G_{final} = G_{dcha}$ $B_{final} = B_{dcha}$

Tabla 1. Estrategias para la combinación de dos imágenes en anaglifos: una posible para grises y tres posibles para imágenes de color.

Para generar los anaglifos[6], manualmente o con un software, se procesan las imágenes y se combina ciertos canales RGB (que designan la componente de rojo, verde y azul respectivamente) de cada una. El resultado dependerá de cómo hayamos posicionado las cámaras, de nuestro conocimiento del software de procesado para componer anaglifos e, importante, que el tono de los

[4] Puedes utilizar la plantilla del Instituto Geográfico Nacional: <http://www.ign.es/3d-stereo/docStereo.pdf >.

[5] Imagen obtenida de "Ciensación experimentos «para poner las manos en la masa»: Anaglifos – profundidad en colores". Disponible en la URL <https://ciensacion.org/experimento_manos_en_la_masa/e6101bp_anaglyph.html>.

[6] Véase <https://dpmultimedia.art/renders-3d-proyectos/anaglifos-imagen-estereo/ >.

filtros de las gafas ha de coincidir con el empleado para la generación de las imágenes. Y también puede ser generado en vivo con el uso de OpenCV. En [2] se proponen diferentes estrategias, están resumidas en la Tabla 1, que muestra una de combinación de dos imágenes de niveles de gris ($Gris_{izqda}$ y $Gris_{dcha}$) y tres para imágenes de color (RGB_{izqda} y RGB_{dcha}), para generar una única imagen (el anaglifo) combinando los canales de color (RGB) de las imágenes de partida.

Para poder ver qué ha sucedido con una imagen que llegue a nuestras manos, se va a emplear OpenCV para analizar la imagen: descomponerla y permitir que el usuario pueda ajustar el tono del filtro de rojo y azul para acertar con los empleados en la generación del anaglifo.

4 Desarrollo

En este apartado se va a mostrar un ejemplo de código que permitirá explorar el contenido de un anaglifo que puede estar formado a partir de alguna de las técnicas que se han expuesto en la Tabla 1. Se va a implementar con OpenCV, lo que hará posible cargar una imagen de color, mostrar sus componentes y simular el efecto que tiene utilizar un par de filtros de color rojo y azul sobre esas imágenes. Así se podrá "ver" el anaglifo sin las gafas, esto es, observar el efecto que tiene que, a cada ojo, se le filtre parte del contenido de una imagen. Al tiempo que será posible "ver" qué sucede dentro de la misma, como qué técnica se ha utilizado o cómo influye el color del filtro.

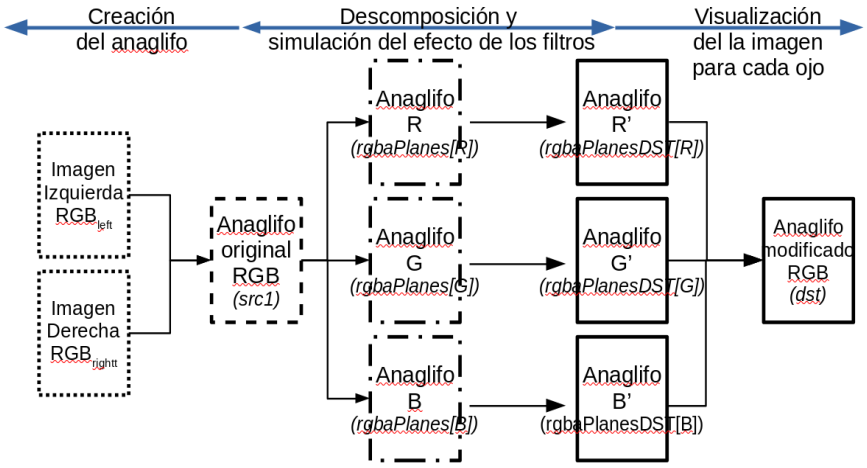


Figura 5: Esquema del proceso de análisis y simulación de visualización de anaglifos.

La Figura 5 muestra la idea de conjunto y dónde están las etapas del código realizado para poder proporcionar la visualización del anaglifo:

- Creación. Esta nos viene dada, así que es previa al desarrollo presentado en este artículo. El proceso de simulación del visor de anaglifos ayudará a ver cuál de las técnicas enunciadas (Tabla 1) se ha utilizado. El anaglifo del que parte la aplicación tiene tres componentes (R, G y B), tanto si las imágenes de partida son de niveles de gris o en color, cuyos valores son el resultado de alguna de las posibles combinaciones que crean un anaglifo.
- Descomposición y simulación del efecto de los filtros. La imagen en color es descompuesta en sus canales y estos son modificados en función de los valores de los controles de la aplicación para la simulación del efecto de los filtros de color.
- Visualización. La imagen resultante de recombinar los planos de color modificados es mostrada. El usuario decide, con los controles, si quiere ver la imagen del ojo tras el filtro rojo

(subiendo el control B) o el que está tras el azul (subiendo el control R). Si se observa todavía una imagen doble, es porque es necesario desacoplar la componente de verde (control G) que puede ser solo obtenida a partir de una de las dos imágenes de partida (derecha o izquierda).

El ejemplo se llama *opencv_anaglif* [4] y lo podemos compilar y ejecutar con las órdenes:

```
$ g++ opencv_anaglif.c -o opencv_anaglif `pkg-config opencv4 --cflags –libs`
$ opencv_anaglif 6101bp_anaglyph_4.jpg
```

Así, para una imagen[7] como la de la Figura 6a, opencv_anagligif mostrará su descomposición en canales (R, G y B); más otra ventana (Figura 6b) que, al manipular los controles que hay en la parte superior permitirá ver lo que le llega a cada ojo, así se puede ver la que se vería al otro lado del filtro azul a la izquierda y la que se vería tras el rojo a la derecha. El control indica el rango de niveles de ese color que se simula filtrar o apagar. Al variar los controles, se pondrán a negro los puntos de las componentes de color que resultarían alterados por el uso de un filtro real en el rango de cero al valor escogido. Se puede observar como en la Figura 6c todavía se ve mucha de las componentes R y B que aparecen con los puntos de las mismas en negro, que tiene un nivel de gris inferior al escogido en el control de barra deslizante, porque el valor del filtro está en 253 y no en 255, como era el caso de la Figura 6b.

a) Original (obtenida de "Ciensación") y canales de color (R, G y B)

b) Imágenes para cada ojo

c) Efecto del color del filtro

Figura 6: Salida de la ejecución de opencv_anaglif: (a) original, (b) las imágenes tras simular el efecto de los filtros y (c) cómo influye la elección el color para simular el filtro.

4.1 El código

En el repositorio creado [4] se puede encontrar el fichero de código que se muestra aquí, junto con las referencias a las imágenes para empezar a utilizarlo. Ahora se entrará a ver el detalle de la implementación. En el Listado 1 se puede ver las declaraciones previas, junto con un par de funciones auxiliares. En concreto, respecto a los previos, se ven: las cabeceras (líneas 1 a la 4) del código en C++, junto a un par de OpenCV, se han definido algunas constantes como macros de C para facilitar la legibilidad del código (entre las líneas 6 a la 16) y las declaraciones de los espacios de nombres (en las líneas 18 y 19).

[7] Imagen obtenida de "Ciensación experimentos «para poner las manos en la masa»: Anaglifos – profundidad en colores". Disponible en la URL <https://ciensacion.org/experimento_manos_en_la_masa/e6101bp_anaglyph.html>.

```
1.  #include <stdio.h>
2.  #include "opencv2/imgcodecs.hpp"
3.  #include "opencv2/highgui.hpp"
4.  #include <iostream>
5.
6.  #define MAXPIXEL (double)255
7.  #define R 2
8.  #define G 1
9.  #define B 0
10. #define OR ||
11. #define AND &&
12. #define TRUE 1
13. #define FALSE 0
14. #define ESC 27
15. #define MODO_FINESTRA WINDOW_AUTOSIZE
16. #define NOM_FINESTRA "Anaglifo"
17.
18. using namespace std;
19. using namespace cv;
20.
21. const int alpha_slider_max = 255;
22. int alpha_slider;
23. Mat dst1, dst2, dst;
24.
25. Mat rgbaPlanes[3];
26. Mat rgbaPlanesDST[3];
27.
28. static void on_trackbarR( int, void* ) {
29.     rgbaPlanes[R].copyTo(rgbaPlanesDST[R]);
30.     threshold( rgbaPlanes[R], rgbaPlanesDST[R],
31.               (double) alpha_slider, //double        thresh,
32.               MAXPIXEL,              //double        maxval,
33.               THRESH_TOZERO          // int    type
34.        );
35.    imshow( "R", rgbaPlanesDST[R] );
36.    merge(rgbaPlanesDST, 3, dst);
37.    imshow( NOM_FINESTRA, dst );
38.    moveWindow( NOM_FINESTRA, dst.cols+10, 0 );
39.
40. }
41. static void on_trackbarB( int, void* ) {
42.     rgbaPlanes[B].copyTo(rgbaPlanesDST[B]);
43.     threshold( rgbaPlanes[B], rgbaPlanesDST[B],
44.               (double) alpha_slider, // thresh,
45.               MAXPIXEL,              // maxval,
46.               THRESH_TOZERO          // type
47.        );
48.    imshow( "B", rgbaPlanesDST[B] );
49.    merge(rgbaPlanesDST, 3, dst);
50.    imshow( NOM_FINESTRA, dst );
51.    moveWindow( NOM_FINESTRA, dst.cols+10, 0 );
52. }
    ...
```

Listado 1: Código del visor de anaglifos: opencv_anaglif.c (1).

```
...
int main( int argc, char* argv[]) //void ){
  Mat src1;
  int tecla, // Tecla leída
      salir; // Controla la condición de terminación de la aplicación
  char nomDst[1024];

  if (argc < 2)  {
    printf("Faltan paràmetros: %s ruta_imagen\n", argv[0]);
    exit( 1 );
  }

  src1 = imread( argv[1] );
  if( src1.empty() ) { cout << "Error loading src1 \n"; return 1; }
  namedWindow( "IMG1", MODO_FINESTRA );
  imshow( "IMG1", src1 );
  moveWindow( "IMG1", 0, 0 );
  split(src1, rgbaPlanes);
  rgbaPlanes[R].channels() );
  rgbaPlanes[R].copyTo(rgbaPlanesDST[R]);
  rgbaPlanes[G].copyTo(rgbaPlanesDST[G]);
  rgbaPlanes[B].copyTo(rgbaPlanesDST[B]);

  namedWindow( "R", MODO_FINESTRA );
  imshow( "R", rgbaPlanes[R] );
  moveWindow( "R", 0, src1.rows+70 );
  namedWindow( "G", MODO_FINESTRA );
  imshow( "G", rgbaPlanes[G] );
  moveWindow( "G", 1*(src1.cols+10), src1.rows+70 );
  namedWindow( "B", MODO_FINESTRA );
  imshow( "B", rgbaPlanes[B] );
  moveWindow( "B", 2*(src1.cols+10), src1.rows+70 );
  imshow( NOM_FINESTRA, src1 );
  moveWindow( NOM_FINESTRA, src1.cols+10, 0 );

  alpha_slider = 0;
  namedWindow( NOM_FINESTRA, MODO_FINESTRA);
  createTrackbar( "R", NOM_FINESTRA, &alpha_slider, alpha_slider_max,
on_trackbarR );
  createTrackbar( "B", NOM_FINESTRA, &alpha_slider, alpha_slider_max,
on_trackbarB );
  on_trackbarR( alpha_slider, 0 );
  salir = FALSE;
  while ( !salir )  {
   tecla = waitKey(25) & 255;
    switch ( tecla )  {
     case ESC:  case 'q':   case 'Q':    // Si 'ESC', q ó Q, ¡acabar!
      salir = TRUE;
      break;
    default: ;
    } // Fi de "switch ( tecla )"
  }// Fi de  "while ( !salir )"
  return( 0 );
}
```

Listado 2: Código del visor de anaglifos: opencv_anaglif.c (y 2).

Después viene el bloque de las funciones auxiliares, son las encargadas de recibir los eventos de respuesta al uso de los controles en forma de barras de deslizamiento, que permitirán simular el color de los filtros de las gafas. Para lo cual, primero se definen, líneas 21 a la 26, las variables que guardan los valores y las imágenes. Después ya están las tres funciones que hacen el filtrado para cada componente de color. El funcionamiento es el mismo para las tres, así que solo se comentará para la primera:

- Se copia el valor del plano original, para permitir que el usuario pueda modificar en cualquier momento, cualquiera de los controles, línea 29.
- Se simula el efecto del filtro (mediante la umbralización, poniendo a negro los píxeles de una componente cuyos valores no sobrepasen el nivel indicado por el usuario con los controles), línea 30, y se actualiza la visualización de la componente, línea 35, para poder apreciar el efecto de un valor de filtro acorde con la imagen o no.
- Se vuelve a reconstruir la imagen de resultado y se actualiza la visualización en pantalla, línea 36 a la 38,

En el Listado 2 se puede ver el programa principal que se encarga de cargar la imagen que se recibe como parámetro en memoria, línea 64, para crear con ella el sencillo interfaz de la aplicación. En primer lugar visualizando la imagen, línea 65 a la 68, sino ha habido algún error al abrirla. La imagen se mostrará en la ventana de nombre "IMG1" para poder compararla con las modificaciones que se realicen después. La imagen se mueve a unas coordenadas para poder tener una visualización global. Después visualizando los tres planos de color (R, G y B), entre las líneas 75 y la 85, para observar sobre ellos el efecto de los filtros, conforme los vaya proponiendo el usuario. Entre las líneas 69 a la 73 se obtienen los tres canales y se copian en variables diferentes para poder volver a los valores originales, cuando se quiera simular otros valores de filtros de color. Las tres componentes también se muestran en pantalla en coordenadas que dependen del tamaño de la imagen. El resto del código, entre las líneas 87 y 91, se encarga de crear la ventana principal donde se muestra la imagen recombinada y donde se simula el efecto del filtro; también tiene los controles para modificar los colores de aquellos y las funciones que deben encargarse de aplicar los valores si se modifican los controles. El programa se mantiene en un bucle de espera, líneas 92 a la 102, gestionando el flujo del programa con las funciones de *callback*.

4.2 Algunas pruebas

¿Qué es lo que se puede esperar ver con este visor software que se propone? Mejor unas imágenes de ejemplos que más palabras. ¿Verdad? Vamos allá.

4.2.1 Imágenes estereográficas

La Figura 7[8] muestra un ejemplo de imagen de anaglifo formado con el método tres de color de la Tabla 1. A la derecha se ve la imagen que llega al ojo tras el filtro azul (apagando B), es de color amarillento, porque vemos mezclado rojo y verde[9]. A continuación está la imagen que llega tras el filtro de rojo (apagando R), de color cian (azul más verde). Las últimas dos imáges son para comprobar que la componente de verde estaba presente en los dos ojos, se pueden apagar las componentes por parejas: B+G (que nos deja solo el rojo) y R+G (que nos deja solo el azul).

[8] La imagen "800px-Stereograph_as_an_educator_-_anaglyph.jpg" ha sido obtenida de Wikipedia.

[9] Véase la URL <https://es.wikipedia.org/wiki/Color_primario>.

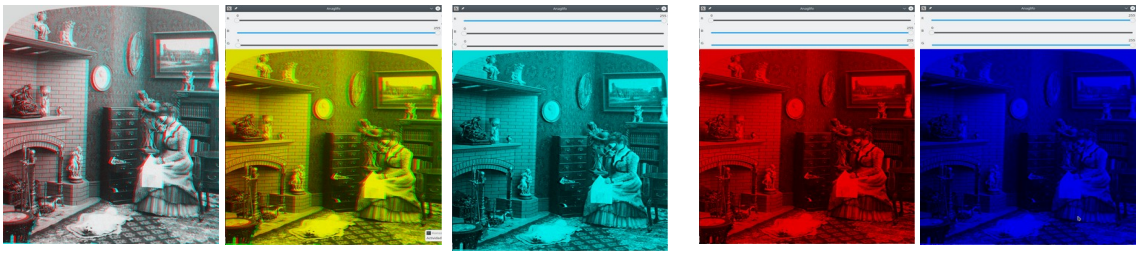


Figura 7: Ejemplo de anaglifo y su descomposición en canales vista a través de opencv_anaglif.

La Figura 8 muestra otros ejemplos extraídos del sitio web de Ciensación (como en la Figura 6) y uno de Mullor[10].

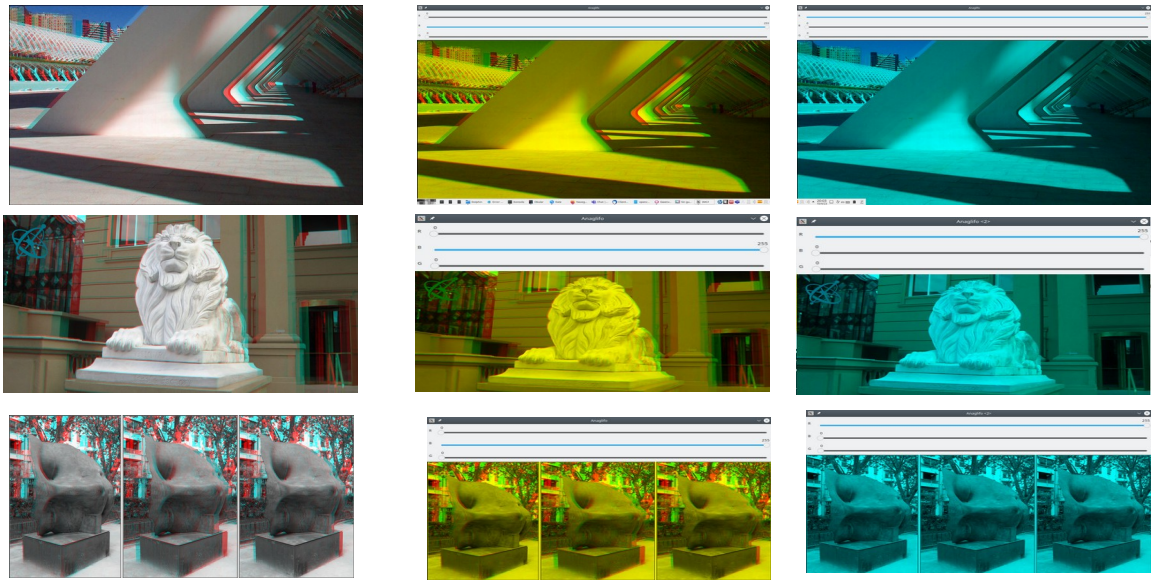


Figura 8: Salida de opencv_anaglif: original, tras el filtro azul y rojo, respectivamente.

4.2.2 Imágenes no estereográficas

La técnica de los anaglifos se utilizó hace algún tiempo, no para imágenes en 3D, sino para "ocultar mensajes" y divertir a los pequeños de la casa: los "cromos mágicos"[11] traían una pequeña historieta en dos viñetas, dibujadas en tinta roja y azul. Así que con las gafas de anaglifos, al cerrar un ojo, veías una de las escenas y al cerrar el otro, la segunda. Ahora se pueden volver a ver con este visor, como se ve en la Figura 9. A diferencia del caso anterior, ahora se apagará también la componente de verde (G), puesto que no se utiliza en los cromos.

[10] Extraído de la URL <http://personales.upv.es/rmullor/Comparativa.html>.

[11] Véase la URL <https://memoriasdeplasticoypapel.blogspot.com/2011/10/cromos-magicos-de-cropan.html>.

5 Conclusión y cierre

Después de haber examinado el proceso de descomposición y recombinación de las componentes de color el lector se puede hacer una idea del contenido de un anaglifo en su computador, y con ello:explorar y comprobar empíricamente cómo es, ¡Ah, y ver el anaglifo sin necesidad de gafas! ;-) Si tienes curiosidad sobre el tema, puede ampliarlo[12] con el trabajo que empresas como Dolby o Panavisión desarrollaron para el cine. Espero que, a estas alturas, tenga ejecutándose el código propuesto y comprobando que puede ver cómo se han formado algunos anaglifos. Puede descargar el ejemplo del repositorio creado a tal efecto en *GitHub* [4] y no cierre este documento sin haberlo comprobado antes. ¡¡ÁNIMO!!

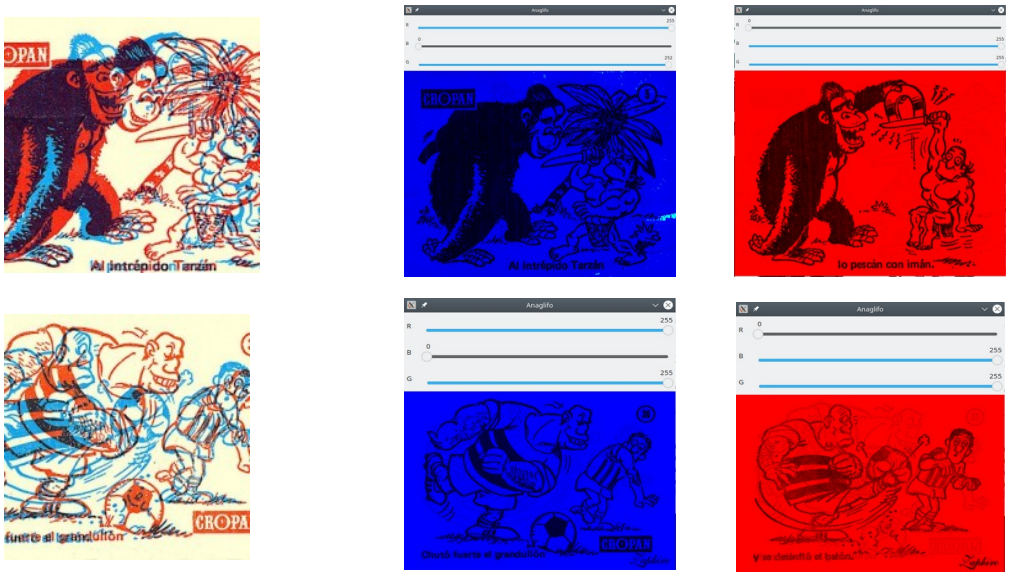


Figura 9: Salida de opencv_anaglif: original, R y B.

6 Bibliografía y referencias

[1] Anagligos. Televisión digital. Disponible en la URL <https://televisiondigital.mineco.gob.es/>.

[2] Bourke, P. (2000). Creating and Viewing Anaglyphs. Disponible en la URL <http://paulbourke.net/stereographics/anaglyph/>.

[3] OpenCV <http://www.opencv.org>.

[4] Agustí, M. (2023). opencv_anaglifo. Github. <https://github.com/magusti/OpenGL_examples/opencv_anaglifo/>.

[12] Veasé "3D 3º: Sistemas de Proyección y Visionado - SUPER ANAGLIFOS". Disponible en la URL <http://parallax3d.blogspot.com/2013/07/1.html>.