Reporte de Proyecto
Análisis del porcentaje de cambio de tamaño de la pupila

Visión Computacional - Clave 2815 01

Período - Agosto/Enero 2020

Luis Nieto Palacios {251737}

13 de diciembre de 2020

Índice

1. Introducción **3**
1.1. Importancia del proyecto en la actualidad . . . 3
1.2. Por qué es importante hace primero el código en Octave y posteriormente en un lenguaje de alto nivel . . . 3
1.3. Análisis de la problemática . . . 3
1.4. Diseño de la problemática . . . 3

2. Técnicas actuales **4**
2.1. Compositing and matting . . . 4
2.1.1. Composición de filtro de reconocimiento . . . 4
2.2. Mejora de la imagen . . . 5
2.3. Correlación para identificar objetos . . . 6

3. Esquema general del proyecto **8**
3.1. Adquisición . . . 8
3.1.1. Ambiente . . . 8
3.1.2. Sujetos . . . 9
3.2. Preprocesamiento . . . 9
3.2.1. Código . . . 9
3.3. Segmentación . . . 10
3.3.1. Código . . . 10
3.4. Extracción de características . . . 11
3.4.1. Código . . . 12
3.5. Reconocimiento de objetos . . . 12
3.5.1. Código . . . 12
3.6. Cálculo del cambio de tamaño de la pupila . . . 13

4. Manual del programador **14**
4.1. main_3p.m . . . 14
4.2. funciones_p.m . . . 14

5. Manual de usuario **14**

1. Introducción

1.1. Importancia del proyecto en la actualidad

Crear una herramienta que ayude en el campo de la medicina. Mientras más herramientas existan, más opciones y avances se podrán construir sobre estas.

1.2. Por qué es importante hace primero el código en Octave y posteriormente en un lenguaje de alto nivel

A mí me toco desarrollar el proyecto solamente en Octave, pero de todos modos pondré mi opinión. Octave permite la experimentación de forma fácil; las funciones ya están hechas, y ver el resultado es muy sencillo. Una vez que tengamos una idea que como funcionan las cosas, ahora podemos pasar a implementarlas.

1.3. Análisis de la problemática

Una de las maneras de determinar la muerte cerebral es la ausencia de varios reflejos, entre estos el reflejo pupilar a la luz [2] (reflejo fotomotor). El reflejo fotomotor se clasifica de la siguiente manera: constricción (encogimiento) a la iluminación directa (respuesta directa) y hacia la iluminación del ojo opuesto (respuesta consensual) [3]. El examinador debe checar el tamaño, forma, igualdad, y posición de las pupilas, asi como su respuesta a la luz [3]. Podemos inferir que por la falta de mención de equipo especial, este método es el más utilizado. La observación de la constricción de la pupila es completamente subjetiva al examinador, lo cual no es ideal. Otro método es la toma de fotografías antes y después de estimular la pupila por medio de iluminación, donde cada una de las fotografías debe de tener una forma de determinar la escala (E.g. una regla) [1]. Mientras que este método es mas preciso, es necesario el uso de una escala (regla), que puede no estar presente.

1.4. Diseño de la problemática

Analizar video digital del ojo humano mientras es sometido al procedimiento clínico que produce el reflejo fotomotor.

- Reconocer la pupila humana en video digital.
- Determinar el porcentaje de contracción de la pupila a lo largo de cierto tiempo de latencia.
- NO intentar dar un diagnóstico, solamente crear una herramienta que obtenga información capaz de asistir en el diagnóstico de muerte cerebral.

2. Técnicas actuales

Las técnicas a continuación mostradas fueron implementadas en alguna sección del proyecto; se muestra una pequeña definición, la manera de utilizarla y el código.

2.1. Compositing and matting

Esta técnica consiste en en recortar algun primer plano de alguna imagen, y ponerlo sobre un fondo diferente [4]. Al procesar imágenes me di cuenta de que podia utilizar como filtros algunas secciones de estas imágenes.

2.1.1. Composición de filtro de reconocimiento

Este es un marco de un video (fig. 1) del cual yo extraje un filtros (fig. 2) para tratar de reconocer un ojo. En este caso yo recorté la imagen hasta que solamente quedó los bordes de la puipila y el iris, y traté de que el borde del iris quedara reconocible. El código utilizado solamente fue recortar la imagen.

Figura 1: Frame

Figura 2: Filtro 2

```
frame = aviread('AcostadoFijo/af_(3).avi', num_marco_filtro);
frame = imcrop( frame, [x,y,101,575]);
imwrite('frame', 'filtro.jpg');
```

2.2. Mejora de la imagen

Los videos están grabados a color (fig. 3), asi que los transformamos a escala de grises (fig. 4). También incrementamos el contraste (fig. 5) para que fuera más fácil obtener el decremento de tamaño de la pupila.

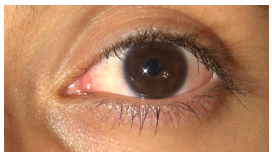


Figura 3: A color

Figura 4: Grises

Figura 5: Contraste

```
% Convertimos a escala de grises
i = rgb2gray( i );
% Incrementamos el contraste
i_a = imadjust( i, [0.2,0.8], [0,1] );
```

2.3. Correlación para identificar objetos

La correlación es un tipo de operador local, es decir, que el valor final de un pixel es influenciado por los valores de los pixeles vecinos. También es un operador linear, lo cual significa que el valor de salida del pixel es la *suma* sopesada de los valores entrantes de los pixeles vecinos [4]. Utilizé la correlación para obtener la localización de un objeto (ojo) en una imagen.

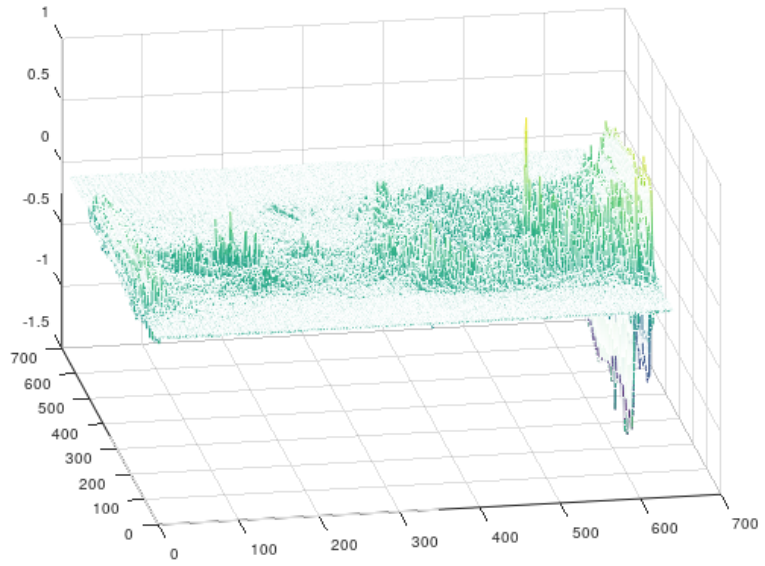
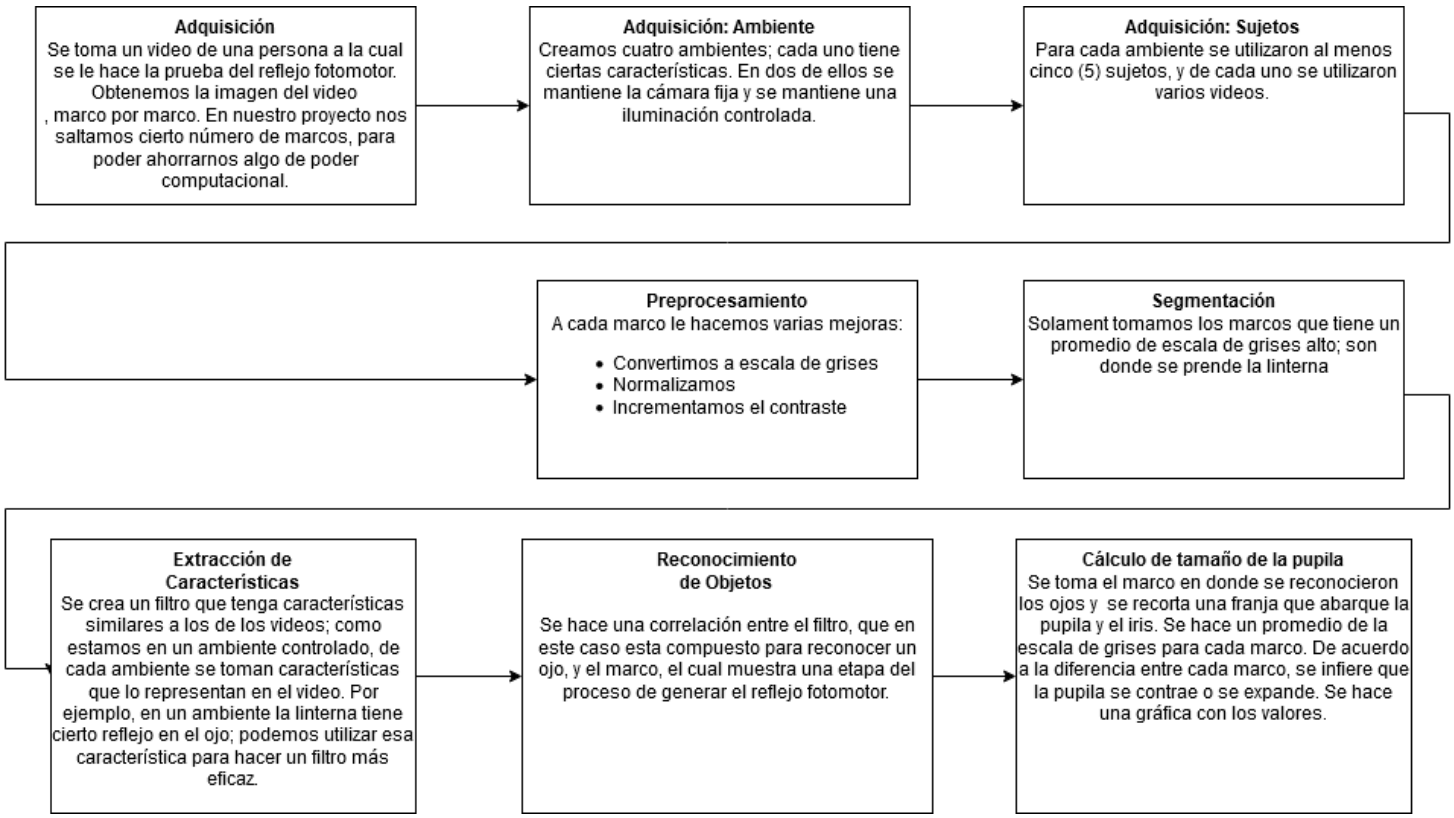


Figura 6: Correlación

```
['Inicio corr de frame(' mat2str(j) ') del video(' mat2str(i) ")"]
cc = normxcorr2( filtro , frame );
['Fin corr de frame(' mat2str(j) ') del video(' mat2str(i) ")"]
%figure , imshow( cc );
[max_c, imax] = max(abs(cc(:)));
[ypeak, xpeak] = ind2sub(size(cc),imax(1));

%yy y xx son las coordenadas del objeto.
xx = (xpeak * 100) / size( cc, 2 );
yy = (ypeak * 100) / size( cc, 1 );

xx = floor( size( frame, 2 ) * ( xx / 100 ) );
yy = floor( size( frame, 1 ) * ( yy / 100 ) );

%Anchura de la banda a tomar
anchura = 50 ;
```

```
%Medias para econtrar el borde
media_limite = 0.35 ;
%Nos movemos hasta encontrar los bordes del ojo.
media_left = 0.0 ;
media_right = 0.0 ;
offset_left = 0 ;
offset_right = 0 ;
left_border = false ;
right_border = false ;
right_limit = size( frame, 2 );
while( !(left_border && right_border)  )
    if( (xx - offset_left) == 1 || (xx + offset_right) == right_limit - 1 )
    break ;
    endif
    media_left = mean( frame(yy-anchura:yy+anchura,xx - offset_left) );
    media_right = mean( frame(yy-anchura:yy+anchura,xx + offset_right) );
    if( media_left > media_limite )
    left_border = true ;
    else
        ++offset_left ;
    endif
    if( media_right > media_limite )
    right_border = true ;
    else
    ++offset_right ;
    endif
endwhile

%Este es la que da las coordenadas correctas.
frame([yy-18:yy+18],[xx-18:xx+18]) = 0 ;
frame([yy-12:yy+12],[xx-12:xx+12]) = 1 ;
frame([yy-3:yy+3],[xx-3:xx+3]) = 0 ;

imwrite( frame, [ruta_img mat2str(i) '/frame' mat2str(j) '.jpg']);
imwrite( cc, [ruta_img mat2str(i) '/cc' mat2str(j) '.jpg']);
imwrite( frame(yy-anchura:yy+anchura,xx-offset_left:xx+offset_right),
            [ruta_img mat2str(i) '/franja' mat2str(j) '.jpg']);

['Frame(' mat2str(j) ')_del_video(' mat2str(i) ')_procesada']

video.franja_prom(j) = mean( mean( frame(yy-anchura:yy+anchura,xx-offset_left:xx
```

3. Esquema general del proyecto



3.1. Adquisición

La entrada de los scripts de Octave es un video en donde se realiza la actividad de generar el reflejo fotomotor en una persona; un ojo aparece en la imagen, y se ilumina por medio de una fuente de luz dirigida. Del video leemos las imágenes que nos interesan.

3.1.1. Ambiente

Tenemos cuatro ambientes, segmentados por la forma de utilizar el equipo, asi como por la posición del sujeto. En los primeros dos, se toman videos a mano libre, con el sujeto parado y acostado. Se trata de que la cámara encuadre al ojo de manera correcta. En el tercer ambiente (figs. 7 y 8) se tiene la cámara y la fuente de luz fijas, donde el sujeto se acuesta y la adquisición de video es llevada a cabo. En el cuarto ambiente, el equipo sigue fijo, pero ahora el sujeto está contra la pared, sentado (figs. 9 y 10).

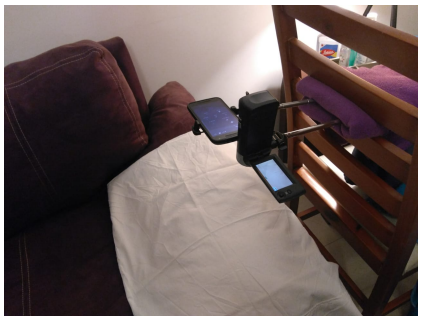


Figura 7: Acostado sin flash

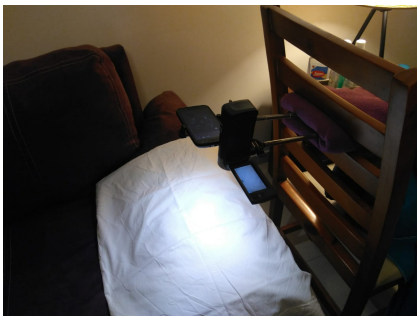


Figura 8: Acostado con flash

Figura 9: Sentado sin flash

Figura 10: Sentado con flash

3.1.2. Sujetos

Se les pidió a cinco (5) sujetos que participaran en el experimento. Las edades varían desde 4 a 50 años.

3.2. Preprocesamiento

Utilizamos dos preprocesamientos; convertimos la imagen del video a escala de grises, y luego le incrementamos el contraste.

3.2.1. Código

```
% Obtenemos el frame.
frame = CuadraImagen( rgb2gray( aviread( nom_video, i )));
% Normalizamos
frame = frame ./ max( frame(:));
% Incrementamos el contraste.
frame = imadjust( frame );
...
```

3.3. Segmentación

Una vez obtenidos los videos, se transforman a escala de grises, y se calcula la media de la escala de grises por cada video (fig. 11). Ya que el cambio de tamaño de la pupila no es instantáneo, cuando se enciende la linterna para activar el reflejo fotomotor, la pupila se encuentra en su tamaño pre-contracción por la iluminación controlada del ambiente.

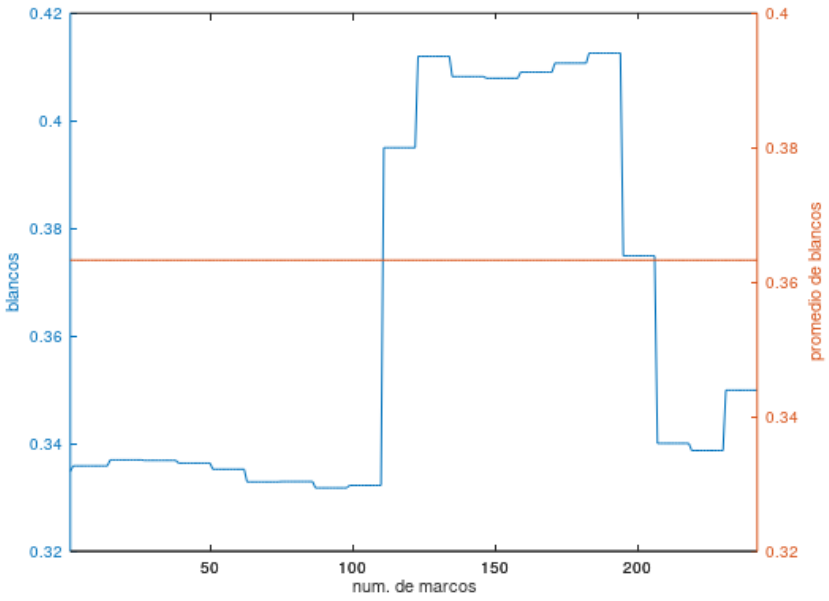


Figura 11: Escala de grises del video

La figura 11 muestra un ejemplo de la escala de grises del video; la línea naranja representa el promedio de la escala de grises del video entero, y la azul representa el promedio de la escala de grises de cada marco. Tomamos solamente los marcos que se encuentren por encima de la media.

3.3.1. Código

```
% Calculamos cada media de grises de cada marco
video.frames = FramesConMediaDeBlancos( nom_video );
% Calculamos la media de grises del video
video.media_video = mean( [video.frames.media] );
% Escogemos los marcos m_s brillantes
video.frames_con_luz = FramesConLuz( nom_video, video );
...
```

```
%-------------------------------------------------------
% Encuentra el promedio de la escala de grises del
% video.
function ret_val = FramesConMediaDeBlancos( nom_video )
    clear('marcos');
    video_info = aviinfo( nom_video );
    for j = 1:video_info.NumFrames
        try
        marco = rgb2gray( aviread( nom_video, j ));
        marcos(j).media = mean( marco(:) );
        marcos(j).num_marco = j ;
        catch
        %A veces no quiere leer el _ltimo marco, aunque
        %se encuentre dentro del rango.
        'EJECUCI_N COMPLETADA, PERO: El  ltimo marco no se pudo conseguir.'
        break ;
        end_try_catch
    endfor
    ret_val = marcos ;
endfunction

%-------------------------------------------------------
% Discrimina los marcos basandose en la media de su
% escala de grises,
%para que solo tomemos el inicio
%de cuando la pupila se empieza a dilatar.
function ret_val = FramesConLuz( nom_video, video )
    clear('frames');
    m = 0 ;
    for i = 1:size( video.frames, 2)
        media_marco = video.frames(i).media ;
        if( media_marco > video.media_video )
        ++m ;
        marco = double( rgb2gray( aviread( nom_video, i )));
        frames(m).imagen = marco ./ max( marco(:));
        frames(m).num_marco = i ;
        endif
    endfor
    ret_val = frames ;
endfunction
```

3.4. Extracción de características

Se crea un filtro que tenga las características para cada ambiente.

Figura 12: Acostado fijo

Figura 13: Acostado libre

Figura 14: Parado fijo

Figura 15: Parado libre

3.4.1. Código

```
%Se elige una imagen que represente el ambiente.
filtro = imread('acostadoLibre (30).jpg');
%Se recorta y se escribe.
imwrite( imcrop( filtro, [x,y,xx,yy]), 'filtro.jpg');
```

3.5. Reconocimiento de objetos

Cada marco le hacemos una correlación con el filtro de su ambiente. Luego, encontramos el valor más alto de la correlación e inferimos que ahí se encuentra el ojo. En la figura 17 podemos observar un pequeño recuadro que marca el

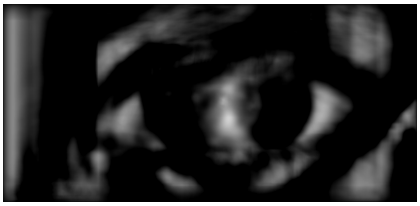


Figura 16: Correlación

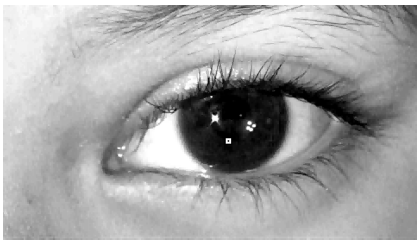


Figura 17: Valor máximo

valor máximo de la correlación.

3.5.1. Código

```
cc = normxcorr2( filtro, frame );
['Fin corr de frame(' mat2str(j) ') del video(' mat2str(i) ")"]
%figure, imshow( cc );
[max_c, imax] = max(abs(cc(:)));
[ypeak, xpeak] = ind2sub(size(cc),imax(1));

%yy y xx son las coordenadas del objeto.
```

```
xx = (xpeak * 100) / size( cc, 2 );
yy = (ypeak * 100) / size( cc, 1 );

xx = floor( size( frame, 2 ) * ( xx / 100 ) );
yy = floor( size( frame, 1 ) * ( yy / 100 ) );
```

3.6. Cálculo del cambio de tamaño de la pupila

Para determinar el procentaje del cambio de tamaño, se toma una banda de 100 pixeles de anchura, con el centro en la marca de la correlación. Después, se busca hacia los lados hasta encontrar el borde del iris, que contrasta con el blanco. Se toma la media de grises de esa banda y se compara con la banda de todos los marcos. Se toma la media de grises más alta como el 100 % y la diferencia con la media menor es lo que nos da el porcentaje de cambio 18.

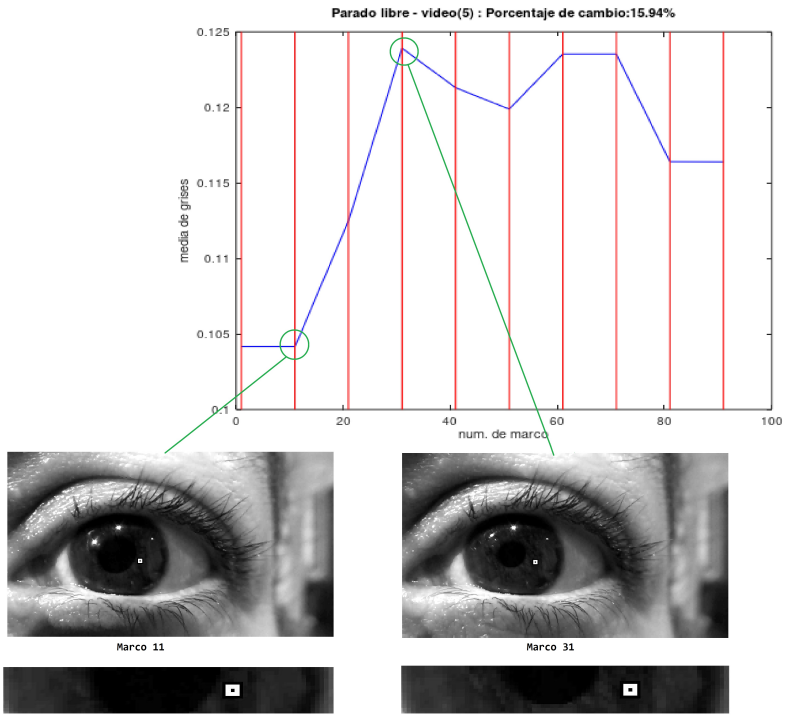


Figura 18: Resultados

4. Manual del programador

Dentro del código fuente tiene explicaciones para cada archivo, y casi para cada línea. Aquí solamente se pone una visión general.

4.1. main_3p.m

Es el archivo principal. Va llamando archivos de script para el procesamiento del video.

4.2. funciones_p.m

Funciones varias, desde hacer cuadrada una imagen rellenando alrededor hasta determinar si un ojo se encontró analizando el resultado de una correlación.

5. Manual de usuario

Para utilizar este proyecto, se necesita tener instalado Octave versión 4.4.1, la cual se puede descargar del siguiente link: `https://www.gnu.org/software/octave/download`. También se necesitan los paquetes *image* y *video*. Una vez descargado e instalado Octave, se pueden instalar por medio de los comandos

```
pkg install image
pkg install video
```

Una vez instalado estos requerimientos, dentro del GUI de Octave se navega hasta la carpeta *source*. Dentro de esta, se ejecuta *main_3p.m* dentro de Octave, por medio del comando

```
main_p
```

Si se desea utilizar otro video, primero se guarda el nuevo video dentro de la carpeta videos, y luego se modifica *main_3p.m* de manera que la siguientes lineas contenga la ruta del nuevo video.

```
%*****************************************
%En esta secci_n el usuario puede configurar
%los valores para que se utilizen sus videos.
%***
%Nombre de los videos
carac_videos = 'ParadoLibre';
ruta_videos = ['videos/' carac_videos '/'];
prefijo_videos = 'pl_(';
sufijo_videos = ').avi';
i = 3 ;

%Por ejemplo, esta configuraci_n:
```

```
%
% carac_videos = 'AcostadoLibre';
% ruta_videos = ['videos/' carac_videos '/'];
% prefijo_videos = 'al (';
% sufijo_videos = ').avi';
% i = 15 ;
%
% busca los videos en:
%
% videos/AcostadoLibre/al (i).avi
%
% y donde i es la variable para indexar
% el video.

% ***
% Filtro a usar.
filtro_ojo = double( imread( ['imagenes/' carac_videos
                    '/filtro.jpg'] ));


% Esta configuraci_n:
%
% filtro_ojo = double( imread( ['imagenes/'
%                         carac_videos '/filtro.jpg'] ));
%
% busca un filtro en:
% imagenes/AcostadoLibre/filtro.jpg

% ***
% Ruta de las im_genes a guardar
%IMPORTANTE: ya debe de existir el directorio.
ruta_img = ['imagenes/' carac_videos '/video_('];

% Esta configuraci_n:
%
% ruta_img = ['imagenes/' carac_videos '/video ('];
%
% guarda las im_genes resultantes en:
%
% imagenes/AcostadoLibre/video (i)
%
% ******************************************
```

Los resultados se muestran automáticamente

```
% Plot
figure, plot( [1:offset:ultimo_marco],
```

```
                [media_blancos(1:offset:ultimo_marco)],
                    'color', 'b');
xlabel('num. de marco');
ylabel('media de grises');
title([ carac_videos '- video(' mat2str(i) ') :
          Porcentaje de cambio:' sprintf("%.2f",per) '%']);
ax = axis();
hold on ;
%Agregamos lineas para cada marco
for j = 1:offset:ultimo_marco
    plot( [j j],[ax(3) ax(4)], 'color', 'red');
    hold on ;
endfor
```

Referencias

[1] Michael D. Twa OD MS, Melissa D. Bailey OD MS, John Hayes PhD, and Mark Bullimore McOptom PhD. Estimation of pupil size by digital photography. *Journal of Cataract & Refractive Surgery*, 30(2):381–389, Febrero 2004.

[2] National Institute of Neurological, Communicative Disorders, and Stroke. *NINCDS collaborative study of brain death*, volume 24 of *NINCDS Monograph*. Bethesda, Md. : The Institute, 1980.

[3] Spector R.H. *Clinical Methods: The History, Physical, and Laboratory Examinations*. Butterworths, Boston, 3rd edition, 1990.

[4] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer London Dordrecht Heidelberg New York, 1st edition, 2011.