PRÁCTICA 3

"Introducción a la adquisición de imágenes digitales en Matlab"

Objetivos.

Familiarizarse con las funciones básicas de adquisición de imágenes de Matlab (*toolbox Image Acquisition*). Diseñar secuencias de video que muestren el resultado de algún tipo de procesamiento de imagen.

Instrucciones: `imaqhwinfo, videoinput, getsnapshot, getdata, preview, start, stop, rgb2gray, bwlabel, regionprops, bwareaopen, roipoly.`

Ejercicios.– Conecte y configure un dispositivo de adquisición de imágenes (WebCam):

Primera Parte. – Sobre una imagen en color capturada con la WebCam:

1.- Utilizando la función de Matlab `subplot`, muestre en una misma ventana tipo *figure* la imagen capturada y distintas imágenes que resalten, sobre la imagen original, aquellos píxeles cuya intensidad sea mayor que un determinado umbral (asigne distintos valores de umbral para generar las distintas imágenes). La intensidad de un píxel se calculará como la media de los niveles de gris de las componentes roja, verde y azul.

2.- Para cada una de las imágenes generadas en el apartado anterior, localice a través de su centroide los distintos "objetos" (agrupaciones de píxeles conexos) detectados. Visualice el centroide del objeto de mayor área en otro color para distinguirlo.

Segunda Parte. - Visualice una secuencia de video que muestre:

3.- La escena inicialmente oscurecida y aclarándose progresivamente (utilizar la instrucción `imadjust`).

4.- Todos los píxeles que tengan una intensidad mayor que un determinado umbral. Asignar inicialmente el valor 0 a este umbral e ir aumentándolo progresivamente.

5.- Las diferencias que se producen entre los distintos frames que captura la webcam (utilizar la instrucción `imabsdiff`).

6.- El movimiento más significativo a partir de diferencias de imágenes de intensidad.

7.- El seguimiento del movimiento del objeto mayor detectado en las diferencias significativas de imágenes de intensidad. El seguimiento debe visualizarse a través de un punto rojo situado en el centroide del objeto.

OBSERVACIÓN:

Si el procesamiento de imágenes implementado mediante el uso de las funciones realizadas en la práctica anterior, `Funcion_Etiquetar , Calcula_Areas , Calcula_Centroides , Filtra_Objetos` , hace que las secuencias de video no se visualicen de forma correcta, puede a optimizar el rendimiento utilizando las siguientes funciones de Matlab `bwlabel`, `regionpropos` y `bwareaopen`:

```
[Ietiq N]=bwlabel(Ib); %Ib = matriz binaria ;Ietiq = matriz etiquetada

stats=regionprops(Ietiq,'Area','Centroid');

    % stats.Area % contiene información de las áreas de cada objeto

    % stats.Centroid - contiene información de la coordenada x (columna)
    % y la coordenada y (fila) del centroide de cada objeto

areas=cat(1,stats.Area); % vector columna con las áreas de cada objeto

centroides=cat(1,stats.Centroid); % matriz de dos columnas (x,y) y
    % tantas filas como objetos etiquetados distintos de cero haya


MatrizBinaria_ObjetosGrandes=bwareaopen(MatrizBinaria,NumPix); % se
eliminan de la MatrizBinaria todos las agrupaciones de píxeles a "1"
compuestas por menos de NumPix pixeles.
```

Documentación y Programas MATLAB de Ejemplos de Adquisición de Imágenes

```
clear all
clc

% Función que devuelve una estructura con información del hardware de adquisición de imágenes
% disponible, incluyendo los adaptadores de video instalados
datos=imaqhwinfo;

%     InstalledAdaptors: {'coreco'  'demo'  'winvideo'}
%         MATLABVersion: '7.4 (R2007a)'
%           ToolboxName: 'Image Acquisition Toolbox'
%        ToolboxVersion: '2.1 (R2007a)'

% Función que devuelve una estructura con información del dispositivo de video instalado
datos=imaqhwinfo('winvideo');

%           DefaultFormat: 'RGB24_352x288'
%     DeviceFileSupported: 0
%                 DeviceName: 'ICatch (VI) PC Camera'
%                   DeviceID: 1
%       ObjectConstructor: 'videoinput('winvideo', 1)'
%         SupportedFormats: {1x10 cell}

%
% SupportedFormats
%
% 'I420_160x120'    'I420_176x144'    'I420_320x240'    'I420_352x288'    'I420_640x480'
%
% 'RGB24_160x120'    'RGB24_176x144'    'RGB24_320x240'    'RGB24_352x288'    'RGB24_640x480'
```

```
% Función para crear el objeto de video que contiene la configuración del
% dispositivo de adquisición de imágenes (WebCam, cámara...) y
% con el que Matlab se comunicará con
% el dispositivo de adquisición de imágenes (Webcam, cámara,...)

video=videoinput('winvideo',1,'RGB24_352x288'); %

% Para acceder a la información de este objeto Matlab:

get(video)
%   General Settings:
%     DeviceID = 1
%     DiskLogger = []
%     DiskLoggerFrameCount = 0
%     EventLog = [1x0 struct]
%     FrameGrabInterval = 1
%     FramesAcquired = 0
%     FramesAvailable = 0
%     FramesPerTrigger = 10
%     Logging = off
%     LoggingMode = memory
%     Name = RGB24_352x288-winvideo-1
%     NumberOfBands = 3
%     Previewing = off
%     ROIPosition = [0 0 352 288]
%     Running = off
%     Tag =
%     Timeout = 10
%     Type = videoinput
%     UserData = []
%     VideoFormat = RGB24_352x288
%     VideoResolution = [352 288]
%
```

```
%   Color Space Settings:
%     BayerSensorAlignment = grbg
%     ReturnedColorSpace = rgb
%
%   Callback Function Settings:
%     ErrorFcn = @imaqcallback
%     FramesAcquiredFcn = []
%     FramesAcquiredFcnCount = 0
%     StartFcn = []
%     StopFcn = []
%     TimerFcn = []
%     TimerPeriod = 1
%     TriggerFcn = []
%
%   Trigger Settings:
%     InitialTriggerTime = []
%     TriggerCondition = none
%     TriggerFrameDelay = 0
%     TriggerRepeat = 0
%     TriggersExecuted = 0
%     TriggerSource = none
%     TriggerType = immediate
%
%   Acquisition Sources:
%     SelectedSourceName = input1
%     Source = [1x1 videosource]

% Para ver una pequeña descripción de lo que es cada parámetro:
% imaqhelp videoinput

% Todos estos paráameros son modificables abriendo el objeto de video. Doble
% click en el workspace.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Para capturar una imagen independiente (no afecta el número de disparos y
% frames por disparos):

preview(video)
% se abre una pantalla gráfica que muestra lo que visualiza la cámara
I = getsnapshot(video);
% captura la imagen que se está visualizando la cámara
% en el momento de la llamada
% Antes de capturar hay que previsualizar (si no se captura una imagen en
% negro)
imtool(I) % para mostrar la imagen por imtool

% Hay cámaras que no ofrecen modelo RGB de salida, sino que ofrecen modelos
% de color basados en luminancia y dos componentes cromáticas YCbCr,
% YUY,...
% Hay que aplicar alguna función MATLAB que transforme el modelo de color a RGB
% Esta función es: ycbcr2rgb.m  para modelos YCbCr.

video=videoinput('winvideo',1,'I420_320x240'); %
preview(video)
I = getsnapshot(video);
image(I)
Imod=ycbcr2rgb(I);
imshow(Imod)

% Otra opción es editar el objeto video y seleccionar el modelo de color de
% salida de la imagen - En ReturnedColorSpace
% De esta forma, se hace la conversión de forma automática, sin necesidad
% de aplicar ninguna función.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% VIDEO: Adquisición de imágenes, frames, continuada. Parámetros de interés

video.TriggerRepeat=3; % set(video,'TriggerRepeat',Inf);
% número de disparos adicionales programados para el dispositivo.
% si tiene un valor 3, se ejecutan 4 disparos

% video.TriggerRepeat=inf; % set(video,'TriggerRepeat',Inf); Con esta
% configuración, infinitos disparos

video.FramesPerTrigger=3;
% Número de imágenes o frames que se capturan por disparo

video.FrameGrabInterval=3;
% Respecto a los frames que la camara puede adquirir a su máxima velocidad
% de captura (típicamente 30 fps), se almacenan en memoria el primero de cada tres hasta
% un total de (Número de disparos * Frames por Disparos).
% Este paramétro es importante porque determina los frames por segundo fps
% a la que se graba en memoria (para que un video se pueda ver de forma
% aceptable se deben mostrar al menos 1 fps).

% Si por ejemplo el dispositivo de video es capaz de capturar 4 fps
% y se fija el FrameGrabInterval a 2,
% las imágenenes se han grabado en memoria con una tasa de 2fps.

% LoggingMode = memory;
% el almacenamiento de los frames es en memoria -también puede ser en disco

% TriggerType = immediate
% El número de disparos programado es inmediato, uno detrás de otro
```

```
% La otra opción es disparar de forma manual, si lo permite el dispositivo.

% Video.FramesAcquired
% En esta variable se almacena el número de Frames que se han adquirido
% con getdata. La instrucción getdata permite guardar como variable matlab
% uno o varios frames guardados en memoria.

% Para cambiar estos parámetros, puede hacerse como se ha mostrado o
% haciendo doble click en el objeto video en el Workspace y modificar las
% opciones.


% Para comenzar a capturar una secuencia de frames:

start(video) % el dispositivo de video empieza a funcionar con la
% configuración almacenada en el objeto.

% Si el disparo es inmediato y el número de disparos infinito, está
% continuamente capturando fotos hasta que se llame la función stop(video).
% No todos los frames se guardan en memoria
% - sólo los que indica video.FrameGrabInterval

stop(video) % para detener una adquisición de video




% CREAR UNA SECUENCIA DE 50 FRAMES Y MOSTRARLA:

% 1 OPCIÓN: CAPTURARLOS Y GUARDARLOS EN MEMORIA TODOS, Y DESPUÉS MOSTRARLOS
% ESTA OPCIÓN NO SE APLICA PERO ES ILUSTRATIVA DEL FUNCIONAMIENTO

video.TriggerRepeat=1 % dos disparos programados para el dispositivo.
```

```
video.FramesPerTrigger=25;
% Número de imágenes o frames que se capturan por disparo

video.FrameGrabInterval=2;

% Con esta configuración se graban en memoria uno de cada dos frames
% que el dispositivo es capaz de capturar a su máxima velocidad.
% Se reducen los fps guardados en memoria a la mitad.
% El número de frames capturados siguen siendo 50, pero están más
% espaciados en el tiempo.

start(video)

% Cuando termina, se puede ver un reporte con video: se puede comprobar que
% hay 50 frames disponibles con getdata
% Para mostrarlos:
N=((video.TriggerRepeat+1)*video.FramesPerTrigger);
% N es el número de frames guardados en memoria

% Accedemos a la memoria para ir cogiendo frames de uno en uno y los vamos
% mostrando con imshow

figure, hold on
for i=1:N
    I=getdata(video,1);
    imshow(I)
end

start(video)
% Accedemos de golpe a toda la información de la memoria y la mostramos
I=getdata(video,N);
[Filas Columnas Bandas Imagenes]=size(I);
```

```
% Primera imagen: I(:,:,1,1)
% Última imagen: I(:,:,1,N)

for i=1:N
    imagen=I(:,:,1,i);
    figure,imshow(imagen) % se abren N imágenes pero podemos ver lo grabado
end


% SEGUNDA OPCIÓN QUE ES LA QUE SE UTILIZA:
% Se programan infinitos disparos y el video termina cuando se han
% adquirido de la memoria un número determinado de frames

video.TriggerRepeat=inf; % disparos continuados

video.FramesPerTrigger=1;
% Número de imágenes o frames que se capturan por disparo

video.FrameGrabInterval=1; % Hacer para un valor 10 y 20;

start(video)
while (video.FramesAcquired<50)

I=getdata(video,1); % captura un frame guardado en memoria. A medida que se va llamando
% a esta función se van capturando los frames en el mismo orden cronológico en que fueron
guardados
% Ver la ayuda de esta función: admite guardar simultánemente un número mayor de frames, en cuyo
% caso se almacena en I un vector de frames.


imshow(255-I) % para ir mostrando la secuencia de frames - en este caso se muestra la imagen
complemtaria
```

```
end

stop(video)

% Para ver el reporte de los frames que se han capturado con getdata
% y los que quedan por capturar guardados en memoria:

video


% SELECCIÓN DE video.FrameGrabInterval
% La función getdata permite guardar información temporal
% de cuando se han tomado los frames. Esto es importante porque, fijando
% video.FrameGrabInterval a 1 (es decir se guardan los frames a la máxima
% velocidad de captura en memoria),  permite tener una idea de los fps
% que nuestro dispositivo de video es capaz de capturar.
% En base a  ello podemos fijar
% el número de frames por segundo que queremos que
% se graben en memoria a través del parámetro video.FrameGrabInterval, para
% que la secuencia de video registrada se visualice con un mínimo de 1fps.

% Un Ejemplo sería:

video.FrameGrabInterval=1;
start(video)
TIEMPO=[];
while (video.FramesAcquired<100)
    % Como ahora se graban todos los frames a la velocidad de captura de la
    % cámara, varios frames por segundo,
    % para que la secuencia dure un poco más de tiempo hay que programar un
    % mayor número de frames adquiridos con getdata
[I TIME]=getdata(video,1);
TIEMPO=[TIEMPO ; TIME];
```

```
gamma=1.5;
I=imadjust(I,[],[],gamma);
imshow(I) % para ir mostrando la secuencia de frames - en este caso se muestra una secuencia más
"clara"
end
stop(video)
video

  % En este ejemplo se ha guardado en la variable TIEMPO los instantes de
% tiempo, contados desde el primer disparo, en los que se
% capturan los frames que se graban
```

EJEMPLO:

1.- Visualizar una secuencia de video que muestre el seguimiento de una determinada zona de la escena. Esta zona será proporcionada al proceso mediante una imagen almacenada en el ordenador. Utilizaremos la correlación normalizada para realizar el seguimiento.

```
clear all
clc

% Función que devuelve una estructura con información del hardware de adquisición de imágenes
% disponible, incluyendo los adaptadores de video instalados
datos=imaqhwinfo;

% Función que devuelve una estructura con información del dispositivo de video instalado
datos=imaqhwinfo('winvideo');


% Función para crear el objeto de video que contiene la configuración del
% dispositivo de adquisición de imágenes (WebCam, cámara...) y
% con el que Matlab se comunicará con
% el dispositivo de adquisición de imágenes (Webcam, cámara,...)


video=videoinput('winvideo',1,'YUY2_320x240'); %
video.ReturnedColorSpace='rgb';

% CAPTURAMOS UNA IMAGEN PARA EXTRAER LA PLANTILLA

preview(video) % se abra una pantalla gráfica que muestra lo que visualiza la cámara (1fps)
```

```
I = getsnapshot(video); % captura la imagen que se está visualizando la cámara en el momento de la
llamada
% antes de capturar hay que previsualizar (si no se captura una imagen en
% negro)
% Pasamos a una imagen intensidad
I=RGB2gray(I);

% De forma manual
imtool(I) % para mostrar la imagen por imtool y sacar las coordenadas de la plantilla
imtool close all
fila1=50; fila2=75; columna1=155; columna2=180;
Plantilla=I(fila1:fila2,columna1:columna2);
imshow(Plantilla)

% De forma automatizada
% Utilizamos la instrucción roipoly para seleccionar un área de interés
% Pinchamos cuatro veces crear el polígono de interés y doble click.

sample_regions(:,:) = roipoly(I); % Matriz lógica, donde a 1 se marcan
%los píxeles de interés
[filas columnas]=find(sample_regions==1); % Coordenadas de los pixeles que integran
% la region de interés
fila1=min(filas); fila2=max(filas);
columna1=min(columnas); columna2=max(columnas);
Plantilla=I(fila1:fila2,columna1:columna2);
imshow(Plantilla)

[NT MT]=size(Plantilla);
```

```
% Para capturar una secuencia de frames:
video.TriggerRepeat=inf; % set(video,'TriggerRepeat',Inf);
% número de disparos programados para el dispositivo.

video.FrameGrabInterval=5; % de todos los frames que se capturan, sólo se van grabando de 5 en 5.

start(video) % el dispositivo de video empieza a funcionar con la configuración almacenada en el
objeto.

while (video.FramesAcquired<150)

I=getdata(video,1); % captura un frame guardado en memoria.
I=rgb2gray(I);
ncc = normxcorr2(Plantilla,I);
[Nncc Mncc]=size(ncc);
ncc=ncc(1+floor(NT/2):Nncc-floor(NT/2),1+floor(MT/2):Mncc-floor(MT/2));
[i j]=find(ncc==max(ncc(:)));


imshow(I),hold on, plot(j,i,'R*'),hold off

end

stop(video)

delete(video);
clear video;
```