

UNIVERSIDAD POLITÉCNICA DE MADRID



*DEPARTAMENTO DE INGENIERÍA
ELÉCTRICA, ELECTRÓNICA,
AUTOMÁTICA Y FÍSICA APLICADA*

Prácticas de Visión Artificial

Práctica 2

***Adquisición de imágenes y entornos
gráficos***

2	<u>ADQUISICIÓN DE IMÁGENES Y ENTORNOS GRÁFICOS</u>	3
2.1	ADQUISICIÓN DE LAS IMÁGENES	3
2.1.1	INSTALAR Y CONFIGURAR LA CÁMARA DE VÍDEO	4
2.1.2	OBTENER LA INFORMACIÓN DEL DISPOSITIVO DE VÍDEO	4
2.1.3	CREAR UN OBJETO DE VÍDEO DE ENTRADA	5
2.1.4	REALIZAR UN CANAL DE VÍDEO EN LÍNEA (OPTATIVO)	5
2.1.5	CONFIGURAR LAS PROPIEDADES DE ADQUISICIÓN DE LA SEÑAL DE VÍDEO (OPTATIVO)	5
2.1.6	ADQUIRIR LOS DATOS DE LA IMAGEN	6
2.1.7	LIBERAR LOS RECURSOS	6
2.2	INTERFAZ CON EL USUARIO MEDIANTE MATLAB	6
2.3	EJERCICIO	9

2 Adquisición de imágenes y entornos gráficos

El objetivo de esta práctica consiste en entender la etapa de adquisición de las imágenes. Se aprenderá a tomar imágenes desde una cámara de vídeo, indicando las habilidades necesarias para la formación de las imágenes, lo cual incluye el manejo de la óptica hasta la obtención de la imagen en una matriz de datos dentro del computador.

2.1 Adquisición de las imágenes

La primera tarea a realizar es la adquisición de las imágenes capturadas desde una cámara de vídeo. Para este objetivo se empleará “*Image Acquisition Toolbox*” de Matlab. Las etapas serán:

1. Instalar y configurar la cámara de vídeo en el computador.
2. Obtener la información de identificación del dispositivo dentro del entorno de Matlab.
3. Crear un objeto de vídeo de entrada.
4. Realizar un canal de vídeo en línea (optativo).

5. Configurar las propiedades de adquisición de la señal de vídeo (optativo).
6. Adquirir los datos de las imágenes.
7. Liberar los recursos.

En las siguientes secciones se implementan cada uno de los pasos indicados.

2.1.1 Instalar y configurar la cámara de vídeo

Con independencia del tipo de tarjeta de adquisición de vídeo deberá de seguirse las siguientes etapas:

1. Instalación de la tarjeta de adquisición de vídeo.
2. Instalación de los *drivers* de la tarjeta.
3. Conectar la cámara a la entrada de vídeo de la tarjeta.
4. Verificar que la cámara funciona correctamente mediante alguna aplicación que lo permita. Si emplea la cámara de su ordenador tendrá algún programa de vídeo. En la práctica se puede emplear la webcam Genius NB. Esta cámara tiene asociado el programa SKIN para visualizar la señal de vídeo.

Dispositivos genéricos de video, tales como WebCams y cámaras de video, generalmente no requieren grandes esfuerzos de instalación. Básicamente, la conexión al puerto USB o FireWire son suficientes.

Después de la configuración de la cámara y el sistema, se inicializará Matlab. No se necesitará ninguna configuración especial en Matlab para soportar las cámaras, siempre y cuando los *drivers* de las tarjetas de adquisición sean compatibles con esta herramienta de adquisición. Este toolbox es sólo válido para equipos de MATROX, DATA TRANSLATION y video de Windows genérico.

2.1.2 Obtener la información del dispositivo de vídeo

Antes de empezar a adquirir las imágenes hay que saber cómo identificar al dispositivo de vídeo. Esta etapa es necesaria para crear el objeto de vídeo. Utilice el comando:

info = imaqhwinfo

Esta sentencia retornará varios parámetros, pero el más importante es el que indica los tipos de drivers con los que se puede trabajar: *InstalledAdaptors*. Para mayor información sobre cada uno de ellos indicar su identificador dentro del anterior comando. Por ejemplo, si se tratase de video de Windows genérico:

```
info = imaqhwinfo('winvideo')
```

En el campo DeviceIDs y DeviceInfo se puede obtener información detallada de los estándares de vídeo que soporta.

```
info.DeviceInfo
```

```
info_dis = imaqhwinfo('winvideo', 1)
```

2.1.3 Crear un objeto de vídeo de entrada

Para trabajar con la cámara de vídeo hay que crear un objeto que controle la cámara, conectando ésta con Matlab:

```
vidobj = videoinput('winvideo')
```

```
vidobj
```

Aparecerá un sumario de las características del objeto de vídeo creado, indicando el número de imágenes adquiridas, el tipo de disparo para la adquisición (*trigger*), el estado del objeto, etc.

2.1.4 Realizar un canal de vídeo en línea (optativo)

Antes de empezar a tomar imágenes es necesario ajustar la óptica, el diafragma, el sistema de iluminación, ... Para realizar estas tareas se requiere crear un canal de vídeo en línea, esto es, una ventana que muestre de forma continua la señal de vídeo:

```
preview(vidobj);
```

2.1.5 Configurar las propiedades de adquisición de la señal de vídeo (optativo)

Después de haber creado el objeto de vídeo y tener una ventana de vídeo en línea, se puede cambiar algunas características de la adquisición. Una lista de las propiedades de este proceso se puede ver con:

```
get(vidobj)
```

Especial atención habría que poner en el tipo de disparo para adquirir las imágenes. Por defecto está definido como inmediato. Sin embargo, se puede hacer manual que empiece a tomar imágenes después de un retardo, ... Un sin fin de posibilidades que se escapan del ámbito de esta práctica.

2.1.6 Adquirir los datos de la imagen

Con la creación y configuración del objeto de video se puede empezar a tomar imágenes. Tres pasos hay que seguir: 1. Inicializar el objeto de vídeo, 2. Configurar el disparo y 3. Convertir los datos en variables de imágenes:

```
start(vidobj);  
  
datos=getdata(vidobj,4);  
  
imaqmontage (datos);  
  
stop(vidobj);
```

La variables '*datos*' contiene las cuatro imágenes adquiridas. Sin embargo, para tomar una sola imagen, el proceso se puede simplificar a:

```
imgAdq = getsnapshot(vidobj);  
  
imshow(imgAdq)
```

En este caso la variable '*imgAdq*' es una matriz que contiene la información de cada píxel de la imagen adquirida.

2.1.7 Liberar los recursos

Una vez acabado de adquirir las imágenes, habrá de liberar los recursos de la memoria asociado a los objetos:

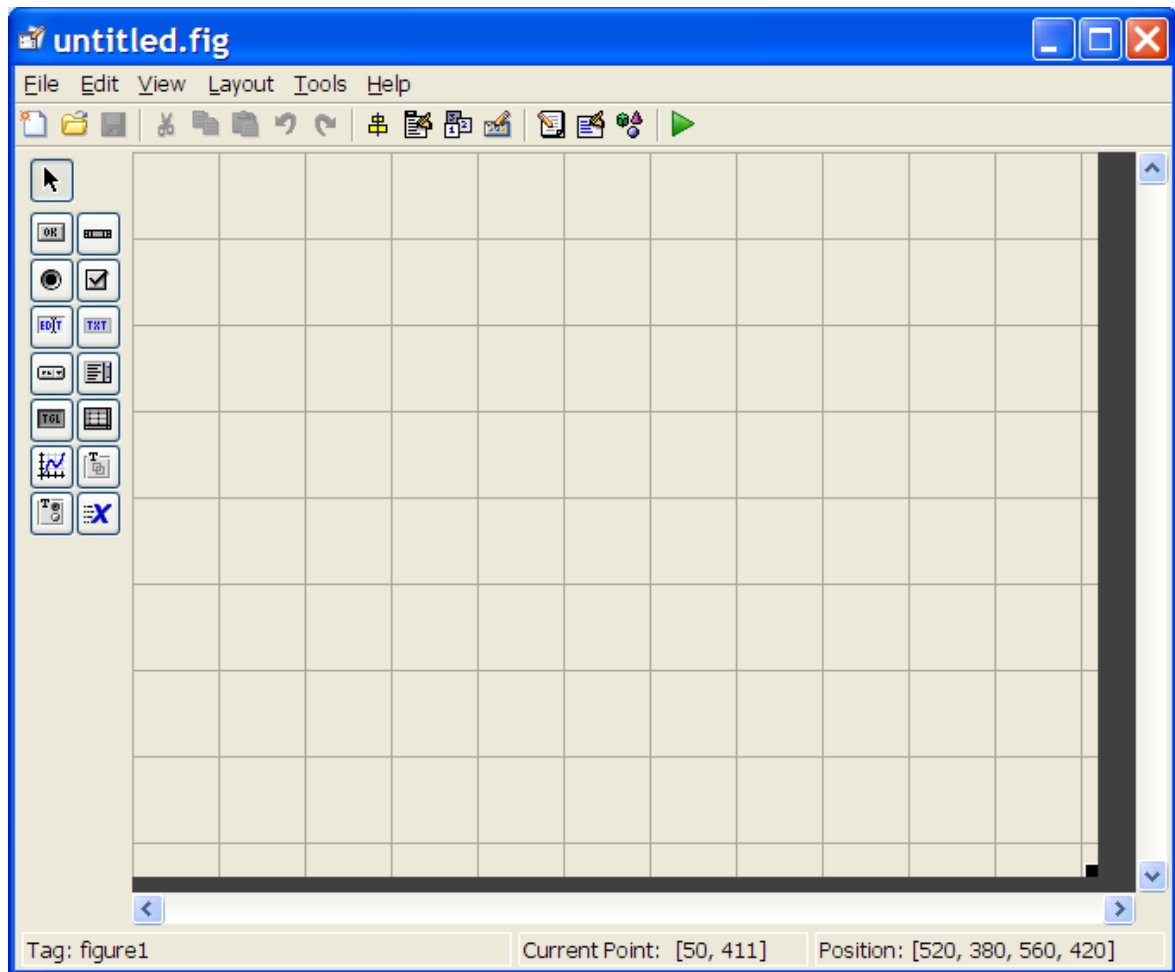
```
closepreview(vidobj)  
  
delete(vidobj)
```

2.2 Interfaz con el usuario mediante Matlab

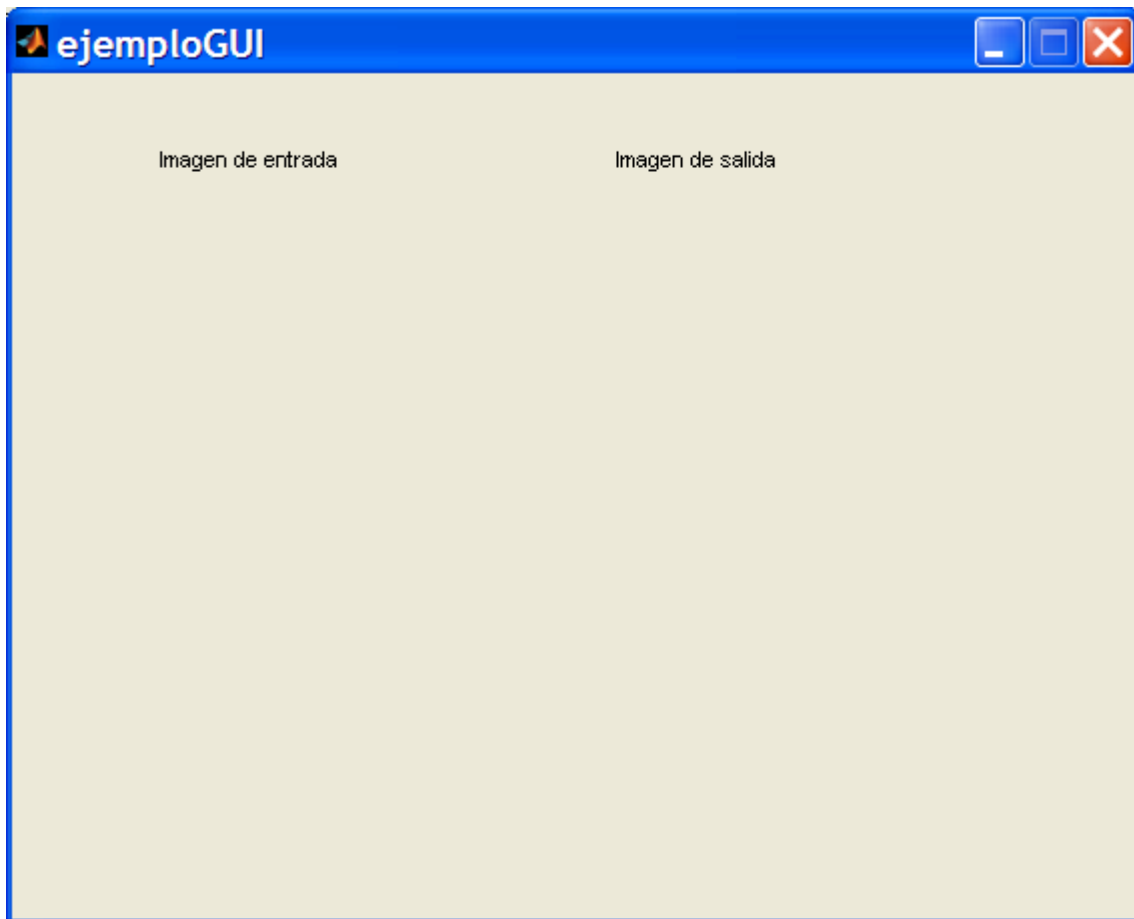
Matlab tiene una aplicación para la creación de interfases gráficos con el usuario (*Graphical User Interface*, GUI):

```
guide
```

Eligiendo una plantilla en blanco aparecerá el editor gráfico:



Se colocará en primer lugar dos textos estáticos (Imagen de Entrada / Imagen de Salida) y se grabará el proyecto GUI en dos ficheros, uno con extensión *.m y otro *.fig (por ejemplo: ejemploGUI) . El icono de “Run Figure” ejecutará el resultado:



Se añadirá dos zonas de dibujos (axes), uno para visualizar la imagen de entrada y otro para ver el la imagen de resultado. Se añadirá un botón para ejecutar el algoritmo de Visión Artificial. Este procedimiento se basará en leer la imagen de entrada ('cameraman.tif') y posteriormente se aplicará una umbralización. El valor de la umbralización quedará reflejado en el interfaz.



El código de la función callback del botón será:

```
data = guihandles(gcf);
imgEnt = imread('cameraman.tif');
umbral=graythresh(imgEnt);
axes(handles.axes1);
imshow(imgEnt);

set(data.edit1,'Enable','on');
set(data.edit1,'String',num2str(umbral*255));
set(data.edit1,'Enable','inactive');

axes(handles.axes2);
imshow(im2bw(imgEnt,umbral));
```

2.3 Ejercicio

Solicitar al usuario el nivel de umbral para realizar la umbralización.

