

# ■ 3 ■

## INTRODUCCIÓN AL TRATAMIENTO DIGITAL DE SONIDO

---

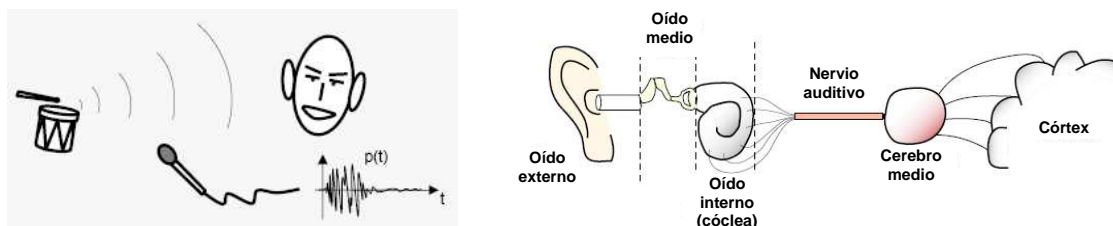
### INTRODUCCIÓN

El procesamiento de sonido es una de las aplicaciones digitales más extendidas. En esta práctica vamos a usar el programa MATLAB y sus herramientas de procesado de señal para editar sonidos, escucharlos y visualizarlos. En particular, vamos a grabar y sintetizar señales de sonido, y a identificar las notas de una melodía.

### DESCRIPCIÓN

#### 1. ¿Qué es el sonido y cómo se captura en un computador digital?

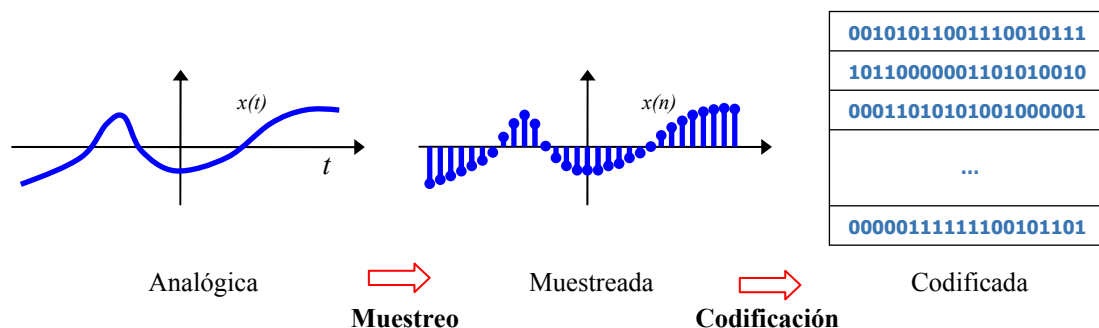
El sonido es una percepción humana que permite obtener gran cantidad de información de nuestro entorno. El fenómeno físico que lo produce es el movimiento del aire, o mejor dicho, el movimiento de una onda de presión (onda acústica).



**Figura 1.** Procesamiento del sonido por el oído humano.

El órgano del oído es un sistema muy sofisticado en el que se capta la onda acústica (oído externo y medio), se descompone frecuencialmente y se convierte en estímulos eléctricos (cóclea), se transmite al cerebro (nervio auditivo), y se procesa (cerebro) para construir la percepción subjetiva que llamamos sonido. La capacidad del cerebro para procesar sonidos es increíble y estamos lejos todavía de ser capaces de imitarlo.

Para capturar el sonido se utilizan micrófonos, que convierten la onda acústica (movimiento) en una señal eléctrica, y para generar sonido se utilizan altavoces, que realizan la operación contraria, convirtiendo la señal eléctrica en una onda acústica. Para poder ser usada en un computador digital la señal eléctrica procedente del micrófono debe ser digitalizada. Para ello, primero se muestrea y luego las muestras se cuantifican y codifican.



**Figura 2.** Proceso de conversión de una señal analógica en una señal digital.

Los parámetros fundamentales de la digitalización son: la frecuencia de muestreo, en muestras por segundo (hercios), y el número de bits empleado para codificar cada muestra. El resultado es una secuencia de códigos binarios manejable en un computador digital. Cuando el sonido es estéreo, hay dos señales (una por cada canal, izquierdo y derecho) que se digitalizan por separado.

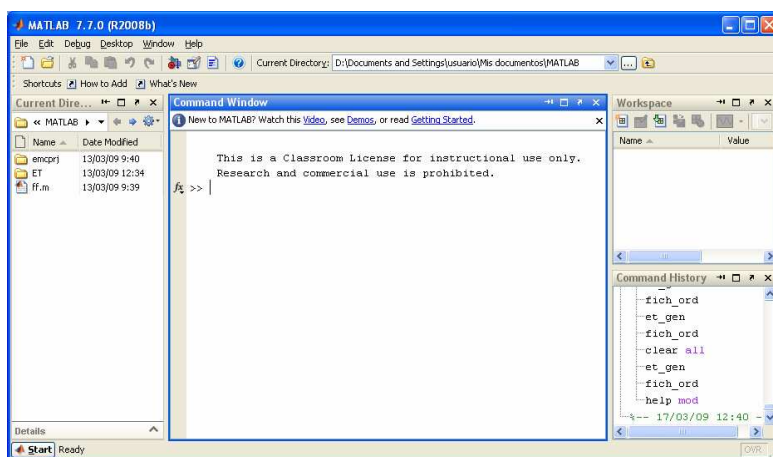
## 2. ¿Qué es MATLAB?

MATLAB (abreviatura de MATrix LABoratory) es una potente herramienta para el tratamiento matemático de datos en el computador. Nosotros emplearemos en esta práctica la versión 7.7.0 de

MATLAB y el *toolbox* (conjunto de funciones predefinidas) para procesamiento de señal. Las funciones de MATLAB pueden ejecutarse interactivamente mediante comandos, o utilizarse para escribir programas. Por ello, conviene conocer la variedad de funciones disponibles (el programa dispone de ayuda *on line*).

MATLAB existe en muchas plataformas y el arranque varía de unas a otras. En nuestro caso, vamos a utilizarlo en PCs con sistema operativo Windows, y el arranque se realizará desde el icono de acceso directo que se encuentra en el escritorio.

Al arrancarlo se abrirá la ventana de MATLAB, y dentro de ella la ventana de comandos (*Command Window*) en la que se pueden teclear los comandos que se desea ejecutar.



En esa ventana se escriben los comandos que se desea ejecutar. El símbolo » indica que el programa está esperando que introduzcamos un comando. Para comprobar lo que es capaz de hacer, prueba a teclear:

```
>> 2+2
ans =
4
```

Como puedes ver, MATLAB ¡sabe sumar! En esta práctica vamos a centrarnos en unas sencillas aplicaciones de sonido, y no vamos a considerar las capacidades de cálculo y programación.

### 3. Captura, visualización y reproducción de sonido en MATLAB.

Desde Windows es muy fácil grabar y reproducir sonido, si tenemos instalada una tarjeta de sonido y un micrófono, accediendo a Inicio/Programas/Accesorios/Entretenimiento/Grabadora de sonidos. Para que funcione, puede ser necesario configurar y habilitar la entrada por micrófono o la salida por los altavoces, utilizando el programa

Control de Volumen. El sonido grabado puede ser guardado en un fichero .wav. Compruébalo.

Pero también podemos grabar sonido desde MATLAB. Para comprobarlo, conecta el micrófono e intenta lo siguiente:


```
>> fs = 8000;  
>> frase = wavrecord(5*fs, fs, 'int16');  
>> wavplay(frase, fs)
```

Si todo ha ido bien, has grabado 5 segundos de sonido con frecuencia de muestreo de 8.000 Hz y 16 bits, utilizando el comando **wavrecord**. Además, lo has reproducido a través de los altavoces mediante **wavplay**. Si quieres visualizar la señal grabada, ejecuta:

```
>> plot(frase)
```

Se abre una ventana en la se representa gráficamente la señal grabada. MATLAB une con líneas las muestras que forman la señal, y en el eje horizontal vemos el número de muestra dentro del vector. Si queremos que el eje horizontal muestre el tiempo en segundos, podemos hacer:

```
>> N = 0:(length(frase)-1);  
>> tt = N/fs;  
>> plot(tt, frase)
```

Para ampliar un fragmento de la señal, puedes hacer zoom utilizando los botones  que aparecen en la barra de menús

Por último, en ocasiones conviene mostrar claramente las muestras sin dejar que **plot** una los puntos de forma automática. Por ejemplo:

```
>> plot(tt, frase, '.')
```

o las dos cosas a la vez:

```
>> plot(tt, frase, '-.')
```

MATLAB permite también leer y escribir ficheros .wav, que es el formato habitual en Windows para almacenar sonidos. Las funciones son **wavwrite** y **wavread**. En los ficheros .wav, además de las muestras de la señal de sonido, se almacenan algunos datos sobre el proceso de digitalización: frecuencia de muestreo, resolución, etc. En el siguiente ejemplo se lee un fichero que contiene una melodía, se representa gráficamente, y se escucha:

```
>> [melo fs bits] = wavread('melodia.wav');  
>> fs  
>> bits  
>> plot(melo)  
>> wavplay(melo, fs)
```

## 4. Síntesis de sonidos y percepción.

Vamos ahora a generar sonidos artificiales para experimentar cómo percibimos los humanos los sonidos. Debemos tener en cuenta que, para ello, vamos a asignar valores a las muestras, como si estuviéramos digitalizando una señal analógica que luego la tarjeta de sonido se encargará de reproducir. Por tanto, deberemos tener en cuenta la frecuencia de muestreo que queremos utilizar.

### A. Señales sinusoidales puras.

Por ejemplo, para generar una senoide de 440 Hz que dure 1 segundo, contando con que vamos a reproducir el sonido a 22.050 Hz, bastaría hacer lo siguiente:

<code>&gt;&gt; tt = 0:1/22050:1;</code>	Vector con instantes de muestreo
<code>&gt;&gt; s = sin(2*pi*440*tt);</code>	Señal sinusoidal con $f = 440$ Hz y $fs = 22.050$ Hz
<code>&gt;&gt; plot(tt,s)</code>	Dibujamos la señal completa.
<code>&gt;&gt; plot(tt(1:500),s(1:500))</code>	Dibujamos un trozo.
<code>&gt;&gt; wavplay(s,22050)</code>	Escuchamos el sonido.

Para hacer más pruebas de forma cómoda, vamos a utilizar una función muy sencilla que hace algo parecido. Es una función que hemos escrito utilizando el editor de MATLAB, se encuentra en la carpeta de alumnos (**tono.m**), y se puede ejecutar desde la línea de comandos. Aquí puedes ver el contenido de la función:

```
function y = tono(f, d)
% TONO Genera sonido sinusoidal puro
% y = tono(f, d) genera muestras de una señal sinusoidal de frecuencia f
% en hercios y duración d en segundos, muestreada a fs = 22050 Hz
%
% tono(f, d) sin argumento de salida hace sonar la señal

fs = 22050;
tt = 0:1/fs:d;
s = cos(2*pi*f*tt);

if nargin == 0,
    wavplay(s, fs)
else
    y = s;
end
```

Ahora, ejecuta:

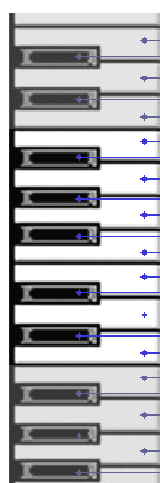
```
>> help tono
>> tono(440,1)
```

Genera y escucha señales correspondientes a diferentes frecuencias. Por ejemplo: 50, 100, 500, 1000, 2000... Hz. Podrás apreciar varias cosas:

- El tono o altura percibido depende de la frecuencia de la onda.
- Dependiendo de la frecuencia se percibe diferente volumen, a pesar de que las señales tienen la misma amplitud (se supone que la sensibilidad máxima se da entre 1 y 5 kHz).
- A mayor frecuencia, los incrementos de frecuencia deben ser mayores si queremos percibir subidas de tonos similares.

No subas la frecuencia por encima del valor crítico  $f_s/2$ , porque en ese caso se produce *aliasing* (la frecuencia de muestreo no es suficiente para capturar esas frecuencias) y la frecuencia aparente bajará en lugar de subir.

En música las notas de la escala utilizada corresponden a frecuencias concretas, (por ejemplo 440 Hz corresponde a la nota LA). En la figura puedes ver parte de la escala.



E	Mi	659.26
D# (Eb)	Re # (Mi b)	622.25
D	Re	587.33
C# (Db)	Do # (Re b)	554.37
C	Do	523.25
B	Si	493.88
A# (Bb)	La # (Si b)	466.16
A	La	440.00
G# (Ab)	Sol # (La b)	415.30
G	Sol	392.00
F# (Gb)	Fa # (Sol b)	369.99
F	Fa	349.23
E	Mi	329.63
D# (Eb)	Re # (Mi b)	311.13
D	Re	293.66
C# (Db)	Do # (Re b)	277.18
C	Do	261.63
B	Si	246.94
A# (Bb)	La # (Si b)	233.08
A	La	220.00
G# (Ab)	Sol # (La b)	207.65
G	Sol	196.00
F# (Gb)	Fa # (Sol b)	185.00

**Figura 3.** Frecuencias correspondientes a las notas de la escala musical.

Lee el código de los ficheros **ejemplo1.m** y **ejemplo2.m** que encontrarás en la carpeta de la práctica, ejecútalos y verás lo fácil que es generar música.

## B. Combinaciones de armónicos

En **ejemplo3.m** se generan para la frecuencia fundamental  $f_0 = 440$  Hz sus **armónicos s1, s2, s3, s4, s5**, es decir sinusoidales de frecuencias 440 Hz, 880 Hz, 1.320 Hz... y se escuchan por separado y sumadas con diferentes pesos. Podrás observar que la señal combinada es periódica, con el mismo periodo fundamental que s1, y al escucharla se percibe como un solo sonido con el mismo tono que s1 ( $f_0$ ) pero con distinto timbre.

De todos modos, esto es demasiado simplificar. La percepción de la frecuencia fundamental (*pitch*) de un sonido periódico es muy subjetiva. Normalmente corresponde a la frecuencia con más peso en la onda, pero hay casos en los que puede no ser así, y además el oyente puede percibir más de un tono dependiendo del contexto y del entrenamiento.

## 5. Identificar notas en una melodía

En el apartado anterior hemos realizado síntesis de señales de sonido, y ahora vamos a intentar lo contrario, el análisis de una señal para determinar algunas de sus características. Hemos visto que el tono percibido parece corresponder a la frecuencia o periodo fundamental de la señal. Este tono es muy importante en música, porque a cada nota le corresponde una frecuencia determinada. Para comprobar esto, vamos a utilizar el sonido que hemos cargado del fichero **melodia.wav**, que contiene un fragmento de una melodía interpretada al piano.

Primero, visualiza la señal total, marcando las posiciones de las muestras:

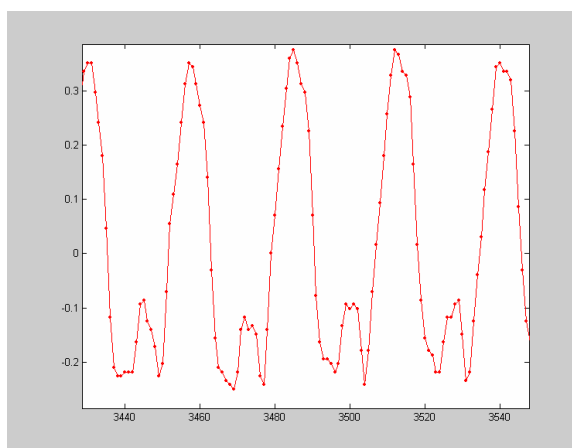
```
>> [melo fs] = wavread('melodia.wav');  
>> plot(melo,'r.-')  
>> wavplay(melo,fs)
```

Observando la amplitud de la señal, apreciarás claramente los instantes en que se pulsan las teclas. Con la ayuda del zoom, analiza fragmentos de señal más pequeños. Podrías medir (contando muestras) el periodo correspondiente a cada nota.

### >> EJERCICIO

Si visualizas un fragmento de la primera nota, verás que el periodo es aproximadamente 27 muestras.

Como la frecuencia de muestreo es de 11.025 Hz **¿Cuál es la frecuencia fundamental de esa nota? ¿A qué nota podría corresponder?**

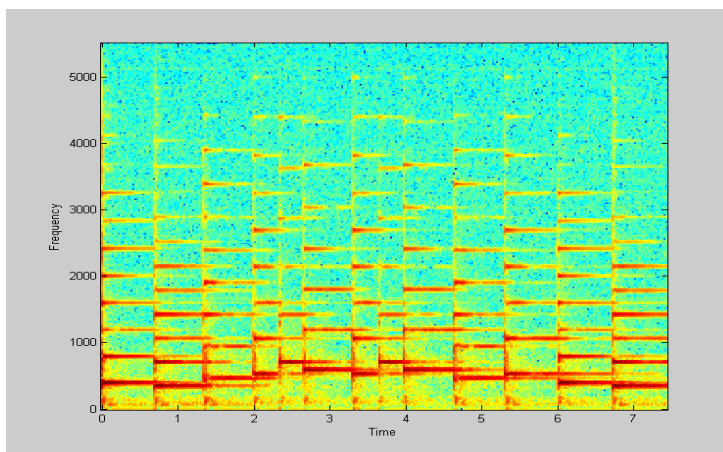


**Repite el procedimiento para la segunda nota.**

Otra manera de observar mejor la frecuencia de las notas de la melodía es calculando y visualizando su “**espectrograma**”. Ejecuta:

```
>> specgram(melo,512,fs,512,212)
```

En este caso, MATLAB divide la señal en partes (de 512 puntos cada una), y en cada una de ellas se extrae la información “frecuencial” asociada, mediante una operación muy utilizada en ingeniería llamada **Transformada de Fourier**. Esto imita de alguna manera la descomposición frecuencial que se realiza en la cóclea del oído humano.



El concepto es bastante complicado para explicarlo en profundidad en este documento, pero basta con pensar que el gráfico nos muestra la evolución en el tiempo (eje horizontal) de la composición frecuencial (eje vertical) de la señal. Los puntos rojos indican la presencia en un fragmento concreto de una senoide de frecuencia concreta. Las líneas horizontales de frecuencia más baja nos muestran la frecuencia fundamental de las notas y la evolución temporal de la melodía.

Pues bien, utilizando el zoom, trata de identificar las 13 notas (frecuencia y duración) tecleadas en esta melodía y de completar la siguiente tabla con las medidas realizadas:

	Frecuencia fundamental (Hz)	Duración (ms)
1	400	0.7
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		



Por último, podemos utilizar la función **tono** para sintetizar artificialmente la melodía utilizando sinusoides. Para ello, generamos el sonido correspondiente a cada nota:

```
>> N1 = tono(400,0.7);  
>> ...
```

Y una vez que tengamos las 13 notas, las concatenamos y las hacemos sonar:

```
>> wavplay([N1,N2,N3,N4,N5,N6,N7,N8,N9,N10,N11,N12,N13],22050)
```

¿Suenan igual?

## **NOTAS**

# ■ 4 ■

## SERVIDORES WEB Y CONEXIONES INALÁMBRICAS

---

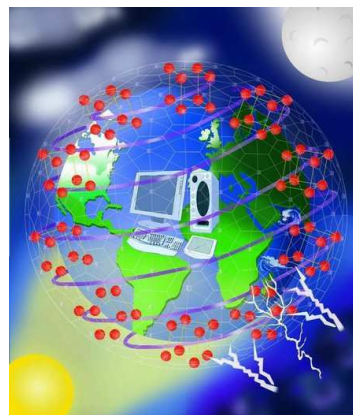
### INTRODUCCIÓN

¿Cómo funciona la *World Wide Web*? Probablemente has usado alguna vez un navegador, y te has "conectado" a la Web; en esta práctica vamos a conocer los pasos que hay que dar para que todo funcione. Durante la misma veremos: (1) qué aspecto tiene un servidor Web; (2) cómo se crean y ubican las páginas Web en el mismo; y (3) cómo se accede a dichas páginas desde un cliente o navegador. Se experimentará tanto con redes cableadas como inalámbricas (WiFi).

## DESCRIPCIÓN

### 1. Introducción

*Internet* es algo de lo que seguramente todos nosotros oímos hablar o hablamos a menudo. Para unos se ha convertido en una herramienta de trabajo, para otros de ocio, e incluso ambas cosas. ¿Pero qué es en realidad Internet? Se trata de un conjunto de redes de ordenadores interconectadas, de ámbito mundial, que permite compartir información y comunicarse de manera sencilla.



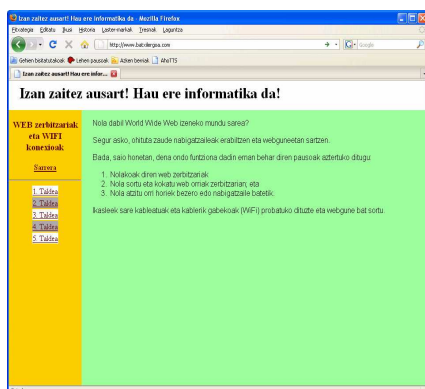
Fuente: <http://movilidad.universiablblogs.net/files/internet.jpg>

Uno de los servicios que más éxito ha tenido en Internet es la *World Wide Web* (“la Web”), hasta tal punto que es habitual confundir los términos Internet y Web. La Web es un servicio implantado sobre Internet, que se apoya en cuatro ideas fundamentales:

- programas servidores de contenidos en forma de páginas Web (servidores Web),
- programas que permiten acceder a dichos contenidos (clientes Web, también llamados “navegadores”),
- un lenguaje para la construcción de páginas Web (HTML, *HyperText Markup Language*),
- y un protocolo o conjunto de reglas para la transferencia de las páginas Web entre clientes y servidores (HTTP, *HyperText Transfer Protocol*).

Si bien esta práctica está orientada a la Web, existen otros servicios de amplio uso en Internet, como por ejemplo el correo electrónico o *e-mail*, las aplicaciones de conversación o *chats*, el intercambio de archivos, la telefonía a través de Internet...

### 2. Creación de una página Web



A partir de la plantilla que os proporcionamos, debéis crear una o varias páginas Web enlazadas, usando un programa de edición de páginas Web, por ejemplo **nvu-kompozer**. Os proponemos que incluyáis texto, alguna imagen y algún enlace a otras páginas, tal vez ya existentes.

El tema de la(s) página(s) lo eliges tú: aficiones (monte, fotografía, cine...), debates (crisis, ¿subirá la Real?...), etc.

### 3. “Publicación” de la página Web creada

Para que la página Web esté accesible, es necesario “subirla” al servidor Web. Para ello, debéis copiar los ficheros que habéis creado en la carpeta correspondiente del servidor.

### 4. Visualización de las páginas

Desde tu navegador Web (*Firefox*, *Internet Explorer*...), accede a la página Web que has publicado. Haz lo mismo con las páginas de los demás grupos. El servidor Web será una máquina distinta a la nuestra, por lo que el acceso deberemos realizarlo a través de la red. En nuestro caso, utilizaremos comunicación inalámbrica (WiFi).

### 5. Informe de accesos al servidor

¿Quién ha accedido a nuestras páginas? ¿A cuáles? ¿Cuándo? El servidor Web almacena información relativa a todo acceso al mismo en los llamados ficheros de **LOG**. En este apartado, visualizaremos el contenido de dicho fichero correspondiente a la práctica que acabamos de realizar.

### 6. ¿Quiénes hacen que todo esto sea posible?

A lo largo de esta práctica aparecen diferentes roles relacionados con la ingeniería informática:

- la persona que diseña y desarrolla los sitios Web (páginas dinámicas, acceso a bases de datos...)
- la persona que programa las diferentes herramientas y aplicaciones que has usado: editor de páginas Web, servidor Web, navegador Web...
- la persona que administra la red y los ordenadores conectados a la misma, incluido el servidor Web.

## PREGUNTAS

- > Aparte de la Web, ¿qué otros servicios de Internet conoces?
  
  
  
  
  
  
  
  
  
  
- > Una tendencia actual es dar acceso a otros servicios de Internet a través de la Web. Un ejemplo de servicio muy popular es el correo electrónico. ¿Conoces algún sitio Web que ofrezca servicio de correo electrónico a través de sus páginas?
  
  
  
  
  
  
  
  
  
  
- > Cada vez que accedemos a una página Web con contenido multimedia, es necesario más de un acceso al servidor (accesos que gestiona automáticamente el navegador). Compruébalo a partir del fichero de **LOG** de la práctica, e indica cuántos accesos necesita tu página Web.

## NOTAS





# ■ 5 ■

## VISIÓN ARTIFICIAL EN SISTEMAS ROBÓTICOS

---

### INTRODUCCIÓN

Vamos a trabajar con un sistema de visión artificial integrado en el controlador de un robot, a través de la interfaz gráfica de la aplicación instalada en el PC al que se conecta la cámara. La interfaz gráfica permite obtener en tiempo real el video capturado por la cámara.

Para integrar el sistema de visión con el sistema robótico hay que ejecutar un procedimiento de calibración a fin de coordinar sus referencias espaciales. Tras esta preparación inicial, con el sistema de visión artificial se construirán de manera automática modelos de objetos (un lápiz, una llave) que después el sistema será capaz de reconocer en posiciones distintas, y con el robot se pueden realizar operaciones sencillas como recoger el objeto identificado.

## DESCRIPCIÓN

Se trabaja con el sistema robótico SCORBOT formado por un brazo robótico guiado por un controlador. Este controlador está conectado a un PC donde está instalado el software que sirve de interfaz entre el operador y el sistema robótico. Asimismo se dispone del sistema de visión que consta de una cámara conectada al PC y el software ViewFlex, instalado en el PC, software que procesa la imagen captada por la cámara y por otra parte actúa de interfaz con el controlador robótico, integrando de esta manera el sistema de visión y el controlador robótico.

### 1. Inicialización del sistema

La práctica comienza configurando los elementos físicos que conforman el sistema robótico y de visión, poniendo en marcha el controlador y arrancando el software ViweFlex. Se examinan algunas utilidades del software de visión, como la ventana de visionado de la imagen captada por la cámara y la terminal ACL para el controlador robótico. Se inicializa el controlador a través del terminal.

Para integrar el sistema de visión y el controlador robótico se realiza una operación de calibrado a fin de que haya una correspondencia entre las coordenadas espaciales que maneja el sistema de visión y las coordenadas espaciales de utiliza el controlador. Dos puntos de referencia sobre la mesa de trabajo permiten establecer esta correspondencia con el ViewFlex a partir de la imagen captada por la cámara y el desplazamiento del brazo robótico a estos puntos.

### 2. Generar el modelo de un objeto

La segunda parte de la práctica consiste en crear el modelo de un objeto de forma que el sistema de visión sea capaz posteriormente de reconocer ese objeto, u objetos similares, en distintas posiciones. Para esto se sitúa el objeto cerca del brazo robótico y se capta una imagen del objeto. Seguidamente se accede a la herramienta de procesado de imagen con el que, tras encuadrar la imagen, se crea el modelo del objeto.

Una vez creado el modelo, se realizan varios experimentos de reconocimiento del objeto, colocándolo en distintas posiciones y probando con otros objetos más o menos similares. En esta experimentación se modifican varios parámetros del algoritmo de reconocimiento y se analiza su efecto sobre los resultados obtenidos.

### **3. Verificación del funcionamiento del sistema completo**

La última parte de la práctica consiste en comprobar que la integración del sistema de visión y el controlador robótico funciona correctamente.

El software ViewFlex dispone de una utilidad que permite realizar sencillamente esta operación: una vez que se ha reconocido el objeto, situado en un determinado punto, se ordena al brazo robótico dirigirse hacia el objeto, de forma que si se ha realizado correctamente el reconocimiento del objeto y la calibración previa del sistema, el brazo robótico se situará sobre el objeto identificado. A través de la terminal ACL se ordenarán al robot operaciones adicionales, como recoger el objeto y trasladarlo a otro punto.



## NOTAS



# ■ 6 ■

## CONTROL DEL MOVIMIENTO DE UN ROBOT

---

### INTRODUCCIÓN

Un pequeño robot móvil debe seguir una trayectoria dibujada en el suelo. El vehículo dispone de:

- 2 ruedas a la izquierda y 2 a la derecha, controladas por sendos motores.
- 3 sensores de infrarrojos que detectan una línea marcada en el suelo.
- 1 circuito integrado que proporciona la potencia a los motores.
- 1 microcontrolador que ejecuta el programa.

## DESCRIPCIÓN

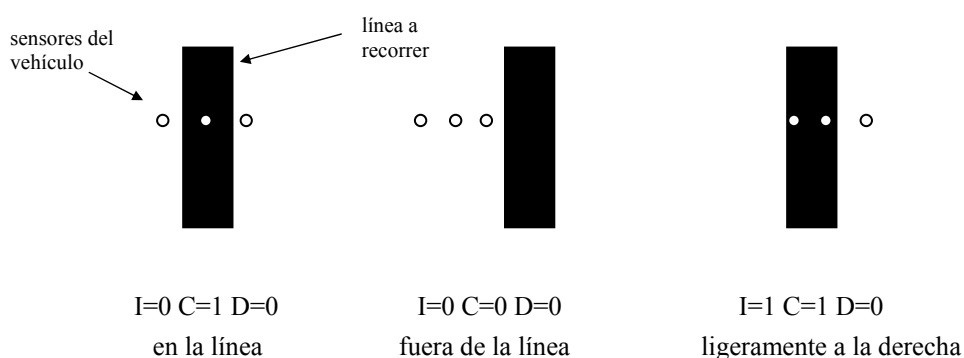
Queremos que un pequeño robot móvil siga una trayectoria dibujada en el suelo. El robot dispone de **tres sensores de infrarrojos** –uno a la izquierda, uno en el centro y uno a la derecha– que nos van a permitir detectar la trayectoria que hay que recorrer.

El circuito dibujado en el suelo es de color oscuro, mientras que el suelo es de color claro, por lo que la luz se refleja de manera diferente en cada superficie. Esa propiedad física es utilizada por los sensores, que **devuelven el valor binario 1 cuando detectan el circuito a recorrer** y el valor 0 cuando no lo detectan.

En función de los códigos devueltos por los tres sensores, tenemos que hacer que el robot se mantenga y avance dentro del recorrido prefijado, para lo que tenemos que controlar la velocidad de giro de los motores. Así, el microcontrolador del vehículo debe leer periódicamente el valor de los sensores y aplicar a cada motor a velocidad correspondiente. Recuerda que un vehículo al moverse tiene una cierta inercia, por lo que si va demasiado rápido y se sale de la trayectoria prefijada costará volver a recuperarla, pero tampoco tiene que ir demasiado lento!

### Interpretación de los valores devueltos por los sensores

Al disponer el vehículo de tres sensores, I, C y D, y devolver cada uno de ellos uno de dos valores, podemos tener 8 combinaciones diferentes que el controlador del robot tiene que interpretar. Por ejemplo:



Las ocho combinaciones posibles y su interpretación son las que aparecen en la siguiente tabla:



Sensor Izquierdo	Sensor Central	Sensor Derecho	Posición del vehículo
0	0	0	Fuera de la línea
0	0	1	A la izquierda de la línea
<b>0</b>	<b>1</b>	<b>0</b>	<b>En línea (situación óptima)</b>
0	1	1	Casi centrado, ligeramente a la izquierda
1	0	0	A la derecha de la línea
1	0	1	Situación dudosa, posible cruce
1	1	0	Casi centrado, ligeramente a la derecha
1	1	1	Situación dudosa, posible cruce

La práctica consiste en **decidir qué velocidad** tiene que aplicar el microcontrolador a cada motor, derecho e izquierdo, en cada uno de los 8 casos anteriores, para que siga la trayectoria de manera adecuada. Hemos dividido el rango de velocidades en 16 casos: el valor 0 corresponde al motor parado, y el valor 15 a la velocidad máxima.

Rellena la tabla adjunta con la velocidad que propones, de 0 a 15, para cada motor.

Sensor Izquierdo	Sensor Central	Sensor Derecho	Posición del vehículo	Motor izquierdo	Motor derecho
0	0	0	Fuera de la línea		
0	0	1	A la izquierda de la línea		
<b>0</b>	<b>1</b>	<b>0</b>	<b>En línea (situación óptima)</b>		
0	1	1	Casi centrado, ligeramente a la izquierda		
1	0	0	A la derecha de la línea		
1	0	1	Situación dudosa, posible cruce		
1	1	0	Casi centrado, ligeramente a la derecha		
1	1	1	Situación dudosa, posible cruce		

¿Funcionó como esperabas? ¿Por qué? Si no fue bien, modifica tus propuestas en base a lo observado y comprueba los resultados. ¡Suerte!

## PREGUNTAS

> Si el motor está parado y le aplicamos poca velocidad, ¿arranca o no arranca el motor?  
¿Por qué?

> ¿Qué velocidades aplicas si se sale de la línea? ¿Por qué?

## NOTAS



# ■ 7 ■

## ADAPTA TU WEB A TUS NECESIDADES (introducción)

---

### INTRODUCCIÓN

En el sector del automóvil está extendida la práctica de adaptar el coche propio a los gustos específicos del propietario. Se le llama tuneado. El conductor melómano y duro de oído, se instalará potentes baffles, mientras que el conductor amante de la pesca tendrá un maletero especialmente adaptado para llevar las cañas. Todos son conductores, pero cada uno ha adaptado su vehículo a sus peculiaridades. De la misma forma, todos tenemos acceso a las aplicaciones disponibles en la Web, pero nuestras aficiones, intereses y frecuencias de acceso, hacen deseable que podamos también tunear nuestras aplicaciones favoritas a nuestro propio perfil. ¿Cómo se tunea entonces una aplicación web? Esto es precisamente de lo que trata este laboratorio.

## DESCRIPCIÓN

Se está produciendo un cambio en la forma en la que trabajamos con la Web. Primero sólo leíamos, como mucho, rellenábamos formularios. Esto es lo que se llama Web 1.0. Desde hace algunos años, hemos adoptado un rol más activo: escribimos en blogs, colaboramos en las wikis, puntuamos qué nos parecen las películas, los libros o las novedades de nuestros cantantes favoritos. La red ya no es sólo una gran enciclopedia donde encontramos todo tipo de información, sino que se ha convertido en un espacio de interacción social: “chateamos”, quedamos para ir a la playa, o intercambiamos fotos o experiencias. Esto es lo que se llama Web2.0.

La Web ha pasado de ser una herramienta más, a ser LA herramienta. ¿Has pensado alguna vez en cuánto tiempo pasas pegado al ordenador? No eres el único. ¿Sabías que en USA la gente pasa más tiempo salseando en la Web que viendo la televisión, que ya es decir? ¿y que en el año 2008 las páginas sociales como *twitter*, *facebook*, etc. superaron por primera vez las de otros contenidos como los casinos *on-line* o las páginas para adultos? ¿y que el número de trabajadores en el sector servicios que trabaja con un ordenador ha crecido en un 70% en los últimos años?

Este uso creciente de la Web, tanto para el trabajo como para el ocio, hace que los usuarios seamos cada vez más exigentes en nuestra interacción con la Web. En concreto, al pasar tantas horas en la Web, la posibilidad de adaptar la interacción con las aplicaciones Web a nuestra propias necesidades resulta especialmente atractiva.

Deja que te dé algunos ejemplos. ¿Conoces SKYPE? Es una aplicación que te permite llamar a teléfonos desde el ordenador a costes muy razonables. Pues bien, si utilizas frecuentemente esta aplicación, podrías tunear las aplicaciones Web de forma que cuando aparezca un número de teléfono en una página Web, en vez de ser un simple número, se convierta en un enlace a SKYPE. Al hacer doble clic en el enlace, en vez de navegar a otra página, estarías llamando directamente a través de SKYPE y escucharías la voz de tu amigo

Otro ejemplo. Imagina que estás aprendiendo inglés o euskara. Por ello, haces un esfuerzo por leer la Web en estos idiomas. Lamentablemente, con cierta frecuencia encuentras palabras que no conoces. Ir al diccionario, incluso si éste está en la propia Web, es farragoso: abre una nueva pestaña, carga la página del diccionario, busca la palabra... vamos, que ya no te acuerdas de lo que estabas leyendo. Solución: adapta la Web para que sólo con posicionarte con el ratón en la palabra que ignoras, obtengas la traducción al castellano, y todo ¡sin salir de la página que estás leyendo!

Y hay muchos más ejemplos. Cada uno tiene sus sitios preferidos o necesita acceder a Web concretas con cierta frecuencia. Adaptar, tunear la Web permite relacionarse con la Web de forma más cómoda y eficiente.

En este laboratorio vamos a ver un par de herramientas que hacen esto posible: JavaScript y Greasemonkey. Sólo necesitas tener unos conocimientos básicos de programación. El objetivo del laboratorio no es tanto que conozcas estas tecnologías en profundidad, sino que vislumbres hacia dónde avanza la Web con las técnicas semánticas y de microformatos: la Web3.0

### **Desarrollo de la sesión**

El proceso que vamos a seguir es el siguiente (puedes hacerlo directamente en tu portátil):

- > Para mostrar la utilidad, veremos algunos ejemplos en el ordenador del profesor.
- > Seguidamente, instalarás en tu portátil Greasemonkey .
- > Identifica una aplicación a la que accedas frecuentemente. Piensa en cómo te gustaría adaptarla
- > Intenta buscar un script ya disponible que consiga algo parecido (acceder a <http://userscripts.org/>)
- > Instala este script en Greasemonkey. Mira cómo funciona. Mira su código.
- > Terminaremos comentando cómo estas utilidades son la antesala de la Web3.0

## PREGUNTAS

- > ¿Qué implicar tunear la Web?
  
  
  
  
  
  
  
  
  
  
- > ¿Cuál es el aspecto distintivo de la Web Semántica?

## INFORMACIÓN COMPLEMENTARIA

- Breve tutorial sobre Greasemonkey:  
<http://extensionesfirefox.blogspot.com/2008/02/manual-o-tutorial-de-greasemonkey.html>
- Lista de Scripts para uso publico: <http://userscripts.org/>