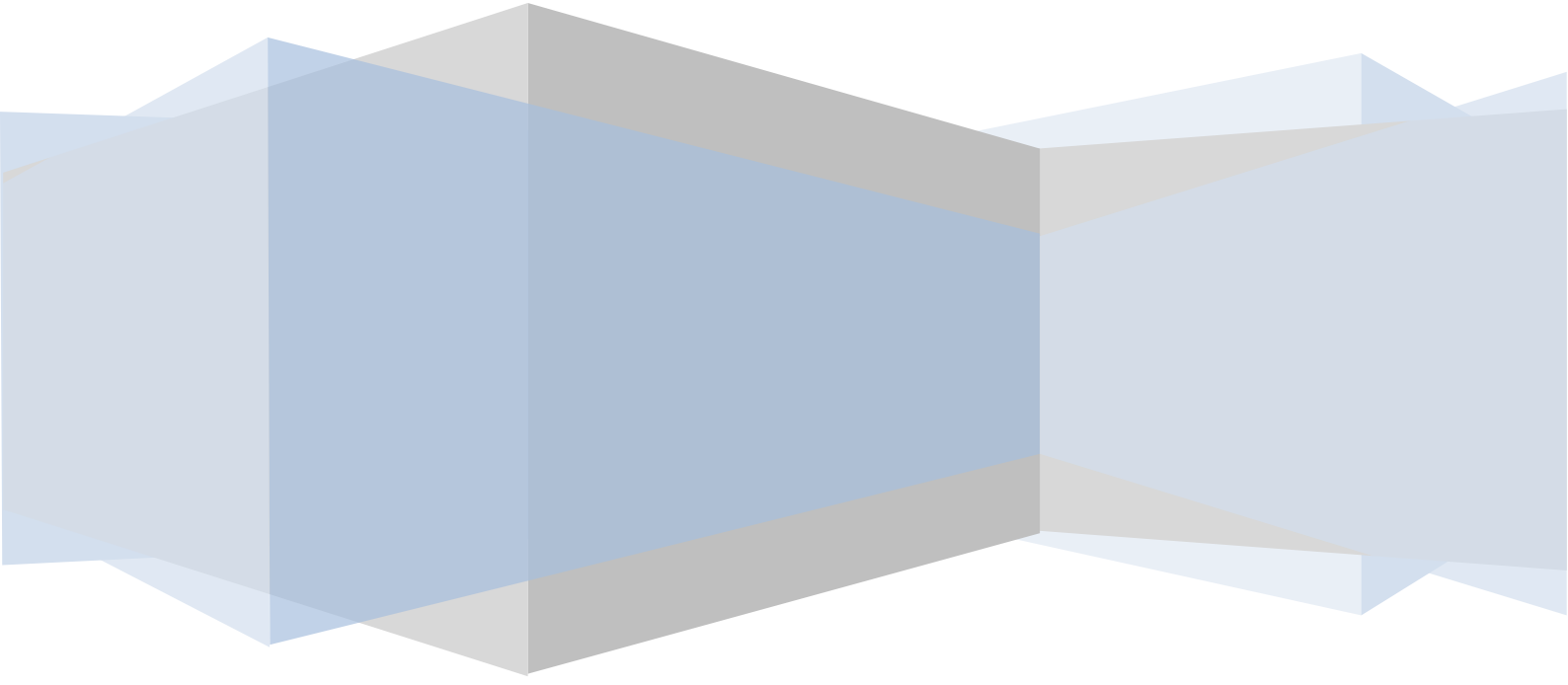
ROBÓTICA
CUARTO CURSO DEL GRADO EN
INGENIERÍA INFORMÁTICA

Universidad de Huelva

PRÁCTICA 1

INTRODUCCIÓN AL SISTEMA DE CONTROL Y SENSORES DEL EV3



CARACTERÍSTICAS DEL SISTEMA NXT

El LEGO ® Mindstorms EV3 es un kit de construcción de robots creativos e individuales que se pueden programar en lenguajes de programación diferentes. EL kit contiene una batería recargable, el 'ladrillo' Inteligente EV3 (unidad principal), tres motores, tres lámparas, dos sensores táctiles, un sensor de luz, un sensor de sonido, un sensor ultrasónico, cables y piezas de LEGO ®. Estos elementos pueden ser utilizado para construir un robot que interactúa con su entorno.

Figura 1.- Sistema Mindstorms EV3

En los experimentos que realizaremos a lo largo del curso utilizaremos fundamentalmente los motores (como sistema de actuación), los sensores de contacto, y el sensor de ultrasonido. Cada motor está dotado de un sistema "encoder" al cuál se tendrá acceso como un sensor más. Existen un conjunto de sensores extra, con los que también se puede experimentar, como son giróscopos, acelerómetros y brújulas.

CARACTERÍSTICAS DEL "LADRILLO" EV3

En primer lugar, describiremos las características de la unidad principal del sistema, también llamada 'ladrillo'. En la siguiente imagen podemos observar cada uno de los puertos de entrada y salida disponibles y los distintos pulsadores e indicadores de estado del sistema.

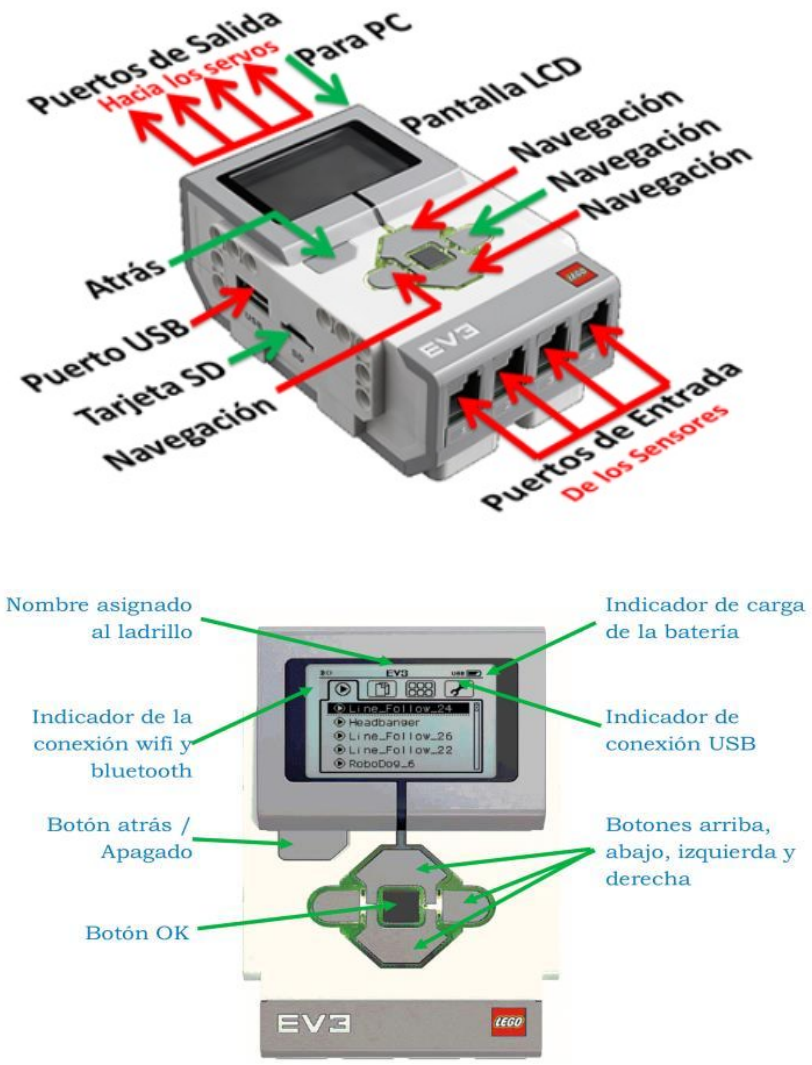


Figura 2.- El ladrillo EV3

A continuación, se describen cada una de las partes que lo componen:

- **Puertos de salida**: donde conectaremos los motores. El dispositivo está dotado con cuatro puertos de salida (A, B, C, D).
- Puertos de entrada, se conectarán los sensores. El dispositivo está dotado con tres puertos de entrada (1,2,3 y 4)
- **Display:** Es la pantalla del ladrillo. Donde nos indicará en la parte superior: la conexión de bluetooth, la serie del programa de este ladrillo y el nivel de batería. En el centro el programa que está corriendo. En la parte inferior distintos iconos pertenecientes a menús para configurar, guardar, exportar, importar, etc.

- **OK**: El botón central de color gris oscuro es el que permite encender y apagará el ladrillo y se usa también para validar las opciones elegidas en el menú.
- **Botones de navegación:** Son los cursores que me ayudarán a moverme dentro de los menús.
- **Atrás/Apagado**: El botón que sirve para anular alguna acción del menú y regresar al inicio. También sirve para seleccionar la opción de apagado.
- **Usb pc:** Es un puerto usb que permite conectar el ladrillo con computador, para cambiar la configuración, enviar datos o programas o para mandar comandos de control.
- **Usb port:** Es un puerto usb que permite conectar periféricos al ladrillo, como por ejemplo un adaptador WIFI.

	EV3
Procesador	64 bits, 300 MHz 16 MB Flash 64 MB RAM
Sistema Operativo	LINUX
Sensores	4 A/D 460 Kbit/s
Puerto Motores	4
USB (Com)	480 Mbit/s
USB (Host)	Daisy-chain (3 niveles) Adaptador WIFI
Tarjeta SD	Ampliable 32GB
Comunicación	Apple y Android
Interfaz	6 botones incluyendo depuración y estado
Comunicación	Bluetooth v2.1DER USB 2.0 (conexión con PC) USB 1.1 (conexión en cadena)

Figura 3.- Características técnicas EV3

CONEXIÓN DEL LADRILLO CON MATLAB

Para controlar el comportamiento del robot desde Matlab, vamos a utilizar el paquete MATLAB para LEGO MINDSTORMS EV3. En el caso de no tener instalado este paquete en la versión de Matlab que se esté corriendo será necesario instalarlo:

(https://es.mathworks.com/matlabcentral/fileexchange/47619-matlab-support-package-for-lego-mindstorms-ev3-hardware)

Si la instalación es correcta, el comando 'legoev3' permite instanciar un objeto que permitirá comunicarnos con el ladrillo desde Matlab.

Matlab soporta esta comunicación utilizando distintos medios y protocolos.

Por ejemplo:

`mi_Robot=legoev3('USB');->` permite la comunicación con el ladrillo que se encuentre conectado al computador con un cable USB.

`mi_Robot=legoev3('WiFi','192.168.80.151','001653814f40');->` permite la comunicación por wifi con un ladrillo que tenga una dirección IP y número de serie determinados.

`mi_Robot=legoev3('Bluetooth','COM3');->` permite la comunicación utilizando bluetooth y un puerto determinado.

Para más información puede consultarse la documentación que proporciona Matlab tecleando en la ventana de comandos: >> help legoev3.

Una vez creado el objeto, si todo es correcto la comunicación arrancará sin ningún problema, si detrás del comando no se ha escrito',' aparecerá en la ventana de comandos la información del establecimiento de la misma. A partir de aquí estaremos en disposición de comunicarnos con el robot.

Elementos sensoriales y de actuación y su control desde Matlab

Los motores.

El sistema EV3 dispone de tres motores con encoders (dos grandes y uno mediano) con los que construir el sistema de tracción de robots móviles. En nuestro caso también vamos a utilizar un motor grande de la anterior versión de Mindstorms NXT. Estos motores son compatibles con el ladrillo EV3. Cada motor dispone de un sistema de engranajes que permiten ajustar la velocidad y par generado por el motor a la velocidad y par necesarios en nuestras aplicaciones. Mediante un *encoder* interno es posible conocer el ángulo girado por el eje del motor.

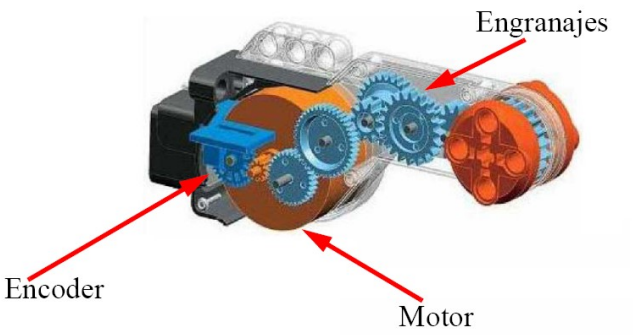


Figura 7.-Motor Mindstorms

Para inicializar los motores usamos la orden 'motor' y entre paréntesis el nombre del objeto que nos permite comunicarnos con el robot, además del puerto del ladrillo donde está conectado el motor, asociándole un nombre al objeto motor creado:

```
motor_A = motor(mi_Robot,'A');

motor_B = motor(mi_Robot,'B');
```

Antes de trabajar con los motores hay que activarlos con el comando start:

```
start(motor_A);

start(motor_B);
```

Para establecer la velocidad de giro de los motores basta con realizar la asignación:

```
motor_A.Speed=50; % velocidad a un 50% del valor máximo
```

Podemos utilizar un numero entero con signo entre -100 y 100 dependiendo de la velocidad de giro deseada y de su sentido. Si vamos a mandar a los motores un valor de velocidad procedente de una operación en la que hemos utilizado una variable 'float' o de otro tipo distinto al de entero con signo, deberemos hacer un cambio de tipo:

```
motor_A.Speed=int8(variable);
```

Por último, para finalizar el uso de los motores se utilizan las ordenes:

```
stop(motor_A)
```

Lectura del *encoder* del motor.

Si queremos leer el encoder basta con invocar la función:

```
giro_A=readRotation(motor_A);
```

Este comando introduce en la variable 'giro_A' el número de grados sexagesimales girado por el motor_A.

Para poner a 0 el contador del *enconder* debemos escribir la orden:

```
resetRotation(motor_A);
```

Sensores del sistema NXT.

Para utilizar cualquier sensor, es necesario que previamente lo declaremos y que al final de nuestro programa lo cerremos. A continuación, definiremos las instrucciones de declaración de los sensores más usuales:

```
Pulsador=touchSensor(mi_Robot,1); %define un sensor de contacto, que está conectado en el puerto 1, del objeto definido con el nombre 'mi_robot'. Es posible obviar el puerto de conexión y esperar que el propio ladrillo encuentre el puerto de conexión.


Sonar = sonicSensor(mi_Robot,2); %definición del sensor de ultrasonido conectado en el puerto 2. Es posible obviar el puerto de conexión y esperar que el propio ladrillo encuentre el puerto de conexión.

Sensorcolor= colorSensor(mi_Robot,3); %definición del sensor de color conectado en el puerto 3. Es posible obviar el puerto de conexión y esperar que el propio ladrillo encuentre el puerto de conexión.

Gyrosensor = gyroSensor(mi_Robot,4); %definición del sensor de giróscopo conectado en el puerto 4. Es posible obviar el puerto de conexión y esperar que el propio ladrillo encuentre el puerto de conexión
```

En el caso de que haya más de un sensor de un tipo conectado al ladrillo y se obvie el puerto de conexión, el comando hará referencia al sensor conectado al puerto de menor número.

Sensor de Contacto. El sensor de contacto permite detectar si el bloque que lo posee ha colisionado o no con algún objeto que se encuentre en su trayectoria inmediata. Al tocar una superficie, una pequeña cabeza externa se contrae, permitiendo que una pieza dentro del bloque cierre un circuito eléctrico.

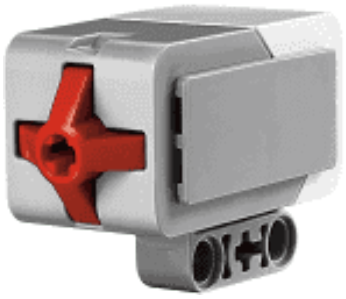


Figura 8.- Sensor de contacto

Para conocer la lectura del sensor de contacto utilizaremos el comando:

```
pulsacion = readTouch('nombre_del_sensor')
```

Entre paréntesis hay que figurar el nombre asociado al sensor en el comando de definición. La variable pulsación valdrá '1' si el pulsador está actuado y 0 si no lo está.

Sensor de Ultrasonido. El sensor Ultrasónico tiene como principal función detectar las distancias respecto de un objeto que se interponga en el camino del robot. Este sensor es capaz de detectar objetos que se encuentren desde 0 a 255 cm. El sensor funciona mejor cuando las señales ultrasónicas que recibe provienen de objetos que sean grandes, planos o de superficies duras. Los objetos pequeños, curvos o suaves, como pelotas pueden ser muy difíciles de detectar. Si en el cuarto se encuentra más de un sensor ultrasónico, los dispositivos pueden interferir entre ellos, resultando detecciones pobres.

Figura 9.- Sensor de ultrasonido

Para conocer la lectura del sensor de ultrasonido utilizaremos el comando:

distancia = readDistance('nombre_del_sensor)

Entre paréntesis hay que figurar el nombre asociado al sensor en el comando de definición. La medida devuelve un valor entero que representa la distancia en cm al objeto sobre el que rebotó la señal sonora.

Sensor Giróscopo. Este módulo es un sensor giroscópico de un eje, que permite medir el ángulo y la velocidad de rotación del sistema sobre el que esté situado. Devuelve el número de grados girados desde la creación de la conexión al sensor

Figura 10.- Sensor Giróscopo

El siguiente comando permite obtener dicho valor.

angulo = readRotationAngle ('nombre del sensor')

Se puede utilizar la función

resetRotationAngle(‘nombre del sensor’)

para restablecer a cero el valor del ángulo girado .

Para conocer la velocidad de giro utilizaremos el comando:

velocidad_giro = readRotationRate(‘nombre del sensor’)

Devuelve la velocidad de giro en el instante que se ejecuta el comando, en grados/s.

La figura 10 ilustra el eje de giro sobre el que sensor mide la velocidad de rotación.

Figura 11**.- Sistema de referencia local del sensor giróscopo.**

Sensor de Color. Este sensor le permite a nuestro robot distinguir entre colores y medir la intensidad de luz. Concretamente el comando:

color = readColor (‘nombre_del_sensor)

lee el color de un objeto frente al sensor de color y lo devuelve como una cadena, como rojo, verde o azul.

Figura **12**.-Sensor de color

Para obtener la intensidad de luz recibida se utilizará el comando:

intensidad = readLightIntensity (‘nombredelsensor, modo)

mide la intensidad de la luz que llega al sensor de color y la devuelve como un valor de 0 a 100 (oscuro a claro). Donde 'modo' puede ser: 'ambient' o 'reflected'. El modo predeterminado de esta función ('ambient') mide la luz ambiental del entorno. Cuando se configura el modo 'reflected' para medir la luz reflejada, el sensor emite luz de un LED rojo y mide la cantidad de luz reflejada por los objetos cercanos.

Enunciados de Prácticas Propuestas

1) Realice un Script que realice medidas con el sensor de ultrasonido y acumule las medidas en una matriz. Deberá registrase el tiempo de forma que al final pueda realizarse un gráfica de la evolución temporal de la medida. Experimente variando la distancia de los objetos frente al sensor.

2) Implemente en un Script un controlador para cambiar el ángulo de giro del motor sobre el que está situado el sensor de ultrasonido. Para este ejercicio la referencia ha de ser un valor constante.

3) Utilizando el mismo controlador, de la actividad anterior, realice un Script que lea el ángulo girado (manualmente) por parte del eje del motor A y haga girar el mismo ángulo el motor sobre el que está situado el sensor de ultrasonido. El programa ha de comportarse de forma que parezca que la cabeza del robot "sigue" el movimiento del motor A.

4) A partir del código anterior, programe un controlador que haga que la cabeza del robot siga una referencia senoidal, de forma que la cabeza se mueva entre pi/2 y –pi/2 radianes. Realice dentro del bucle medidas con el sensor sonar y construya un mapa local del entorno. Utilice las funciones de representación gráficas desarrolladas en clase para visualizar el mapa y el movimiento de la cabeza a la vez que se realiza la tarea programada.