UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE LOS LAGOS / DIVISIÓN DE ESTUDIOS DE LA BIODIVERSIDAD E INNOVACIÓN TECNOLÓGICA / DEPARTAMENTO DE CIENCIAS EXACTAS Y TECNOLÓGICAS / LABORATORIOS DE INGENIERÍAS

Manual del kit LEGO[MR] 9797

Área
SISTEMAS ROBOTICOS

LABORATORIO DE MINI-ROBOTICA

ENRIQUE DÍAZ DE LEÓN S/N COL. PASEOS DE LA MONTAÑA, LAGOS DE MORENO, JALISCO.
TEL. Y Fax: +52 (474) 742 36 78, 742 43 14 Fax ext. 6527
www.lagos.udg.mx

Contenido

Introducción

La Robótica es hoy en día una de las áreas con más impulso dentro de las aplicaciones de Ingeniería, por lo que los conocimientos generales de la lógica en programación aplicada a algún mecanismo, resultan indispensables para todos aquellas personas cuya formación implica el generar procesos óptimos de automatización o control.

¿Qué necesitamos para construir un robot? Antes que nada paciencia y ganas de aprender. Una vez que contamos con los materiales necesarios y los conocimientos básicos de lo que será nuestro robot, no debemos desesperarnos cuando no funcione, debemos analizar las posibles causas para solucionar el problema y jamás perder el espíritu de búsqueda en nuevas opciones (construcción y programación).

Este manual tiene como finalidad dar a conocer a los alumnos de la materia Sistemas Robóticos del CULagos las especificaciones y características principales del kit LEGO^{MR} 9797, tales como son partes que lo componen, posibles estructuras a realizar, tipos de programación, etc.

Aunque se genera para los alumnos de una asignatura en particular, no se descarta la opción de que algún interesado en el área lo use, ya que el tipo de conocimiento que aquí se presenta puede ser utilizado por cualquier persona con inquietud hacia el área de la programación y la Mini-Robótica.

Ing. Diana Costilla López
Lab. Mini-Robótica

Generalidades

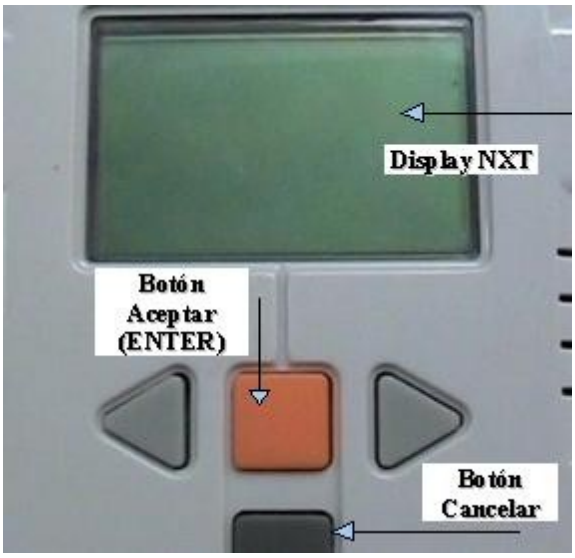
El kit LEGO^{MR} 9797 de la serie Mindstorms permite desarrollar las habilidades tanto de diseño mecánico como de programación estructurada y sobre todo análisis para resolución de problemas específicos con sus diversas restricciones. Está conformado por un cerebro programable al cual pueden conectarse motores y varios tipos de sensores. Las estructuras mecánicas a las cuales se adapte permitirán probar la programación generada para cumplir un objetivo determinado. Principalmente es utilizado para prácticas de Mecanismos, Diseño y Programación de Sistemas Robóticos.

Partes que lo componen

Cada kit trae consigo un listado de piezas, dentro de las cuales destacan, el cerebro NXT, tres motores, dos sensores de tacto, un sensor de luz, un sensor de sonido, un sensor de proximidad, cables, y una batería recargable, el resto son piezas tales como engranes, ruedas, ejes, coples, pelotas y un simpático ayudante.

NXT

El cerebro NXT es alimentado por 6 baterías AA de 1.5 v (no incluidas) o por 1 batería recargable. Tiene 3 puertos para conectar los motores (A, B y C) y 4 puertos para conectar los sensores (1, 2, 3 y 4). También tiene una entrada USB la cuál sirve para conectarlo a la PC y descargar de esta manera los programas generados.



El NXT tiene su pantalla display y además cuenta con 4 botones:

ENTER: naranja
(Este botón sirve para seleccionar y también es para el encedido)
CANCELAR: gris oscuro
(*Este botón sirve para el apagado)*
2 Botones Direccionales: gris claro

Una vez encendido se mostrará la pantalla con el logo de LEGO[MR] y a continuación se mostrará la carpeta **My Files**, la cual contendrá las siguientes carpetas:

Software Files, contiene los archivos que se transmitan de la PC al NXT.

Una vez dentro de la carpeta puede seleccionarse el archivo, y seleccionándolo se tiene la opción de **ejecutarlo** o **borrarlo**.

Si se ejecuta el programa la línea interna del ícono comenzará a girar.

En caso de querer borrar algún programa aparecerá la siguiente pantalla para confirmar si estás seguro de eliminar el archivo seleccionado, el NXT te preguntará 2 veces antes de eliminar cualquier programa.

La opción más apropiada para eliminar los programas que se encuentran en el NXT es conectarlo a la PC y desde el software explorar el contenido de la memoria, para borrar o renombrar.

Menús:

My Files:

Software Files
NXT Files
Sound Files

NXT Program

Please use ports:	Por favor utiliza los puertos:
1.- Touch sensor	1.- Sensor de Tacto
2.- Sound Sensor	2.- Sensor de Sonido
3.- Light Sensor	3.- Sensor de Luz
4.- Ultrasonic Sensor	4.- Sensor Ultrasónico
B/C Motors L/R	B/C – Motores Izquierdo y Derecho

View

(dentro de cada uno de estos podrás seleccionar el puerto a utilizar)

Sound DB	Sonido
Sound DBA	Sonido ambiente

Reflected light	Luz reflejada (*emitida por el sensor*)
Ambient light	Luz ambiental
Light Sensor	Sensor de luz
Temperature ºC*	Temperatura ºC*
Temperature ºF*	Temperatura ºF*
Rotation*	Rotación*
Motor Rotations*	Rotaciones del Motor*
Motor Degrees*	Giro del Motor (*en grados*)*
Touch	Tacto
Ultrasonic inch	Ultrasónico en pulgadas
Ultrasonic cm	Ultrasónico en centímetros

*Sensores del modelo RCX

Bluetooth

My contacts	Mis contactos
Connections	Conecciones
Visibility	Visibilidad
On/Off	Encendido/Apagado
Search	Búsqueda

Settings

Volume	Volumen (0-4)
Sleep	Hibernar
NXT version	Versión de NXT
Delete files	Borrar archivos

Try me

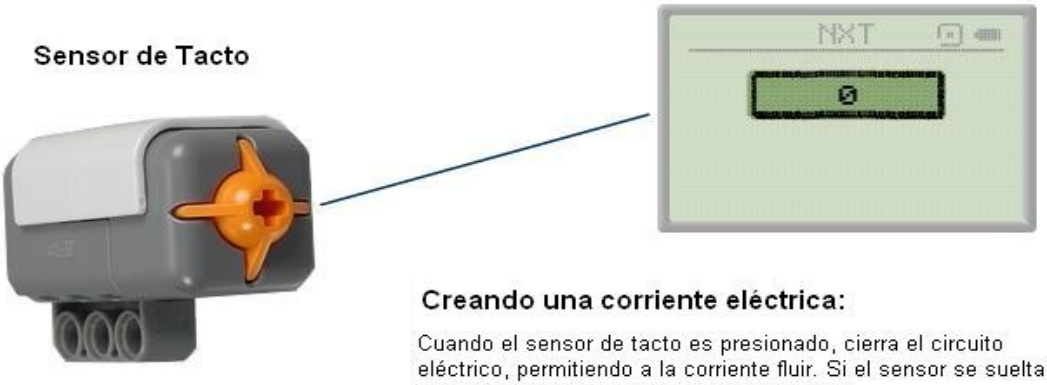
Sensores

Una de las partes fundamentales de la Robótica son los sensores ya que es mediante estos elementos que el sistema entra en contacto con el mundo exterior y permite al programador desarrollar respuestas a partir de las señales adquiridas. A continuación se muestra imagen y breve descripción del funcionamiento de cada uno de los sensores.

Los sensores que incluye el NXT son:

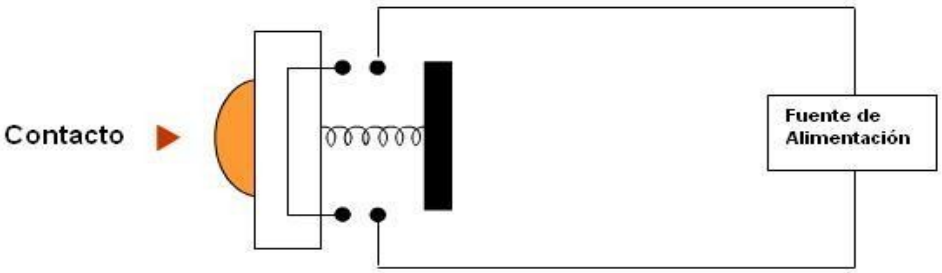
- Sensor de Tacto
- Sensor de Sonido
- Sensor de Luz
- Sensor Ultrasónico
- Sensor de Rotación (Servo-Motor)

Sensor de Tacto

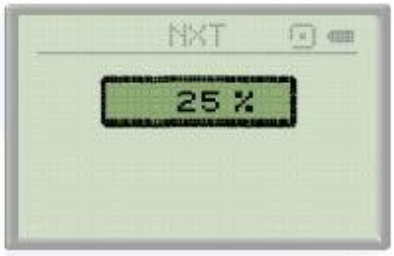


Creando una corriente eléctrica:

Cuando el sensor de tacto es presionado, cierra el circuito eléctrico, permitiendo a la corriente fluir. Si el sensor se suelta, el circuito se abre, y la corriente se detiene.

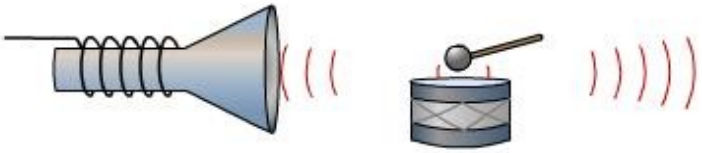


Sensor de Sonido



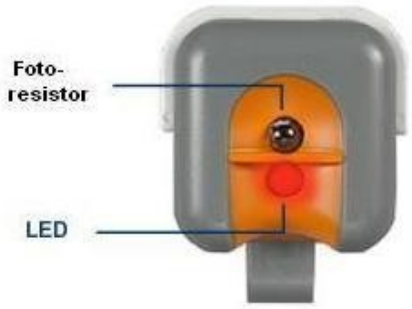
Creando una señal eléctrica:

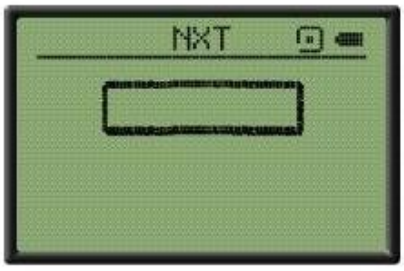
El diafragma está rodeado de imanes los cuales a su vez están rodeados de espiras. Las ondas provocan que el diafragma vibre con los imanes. Esto induce corriente a la bobina lo que la convierte en una señal, NXT recibe la señal y la usa para determinar el nivel del sonido.



El sonido entra al transductor y convierte la onda en una señal eléctrica

Sensor de Luz



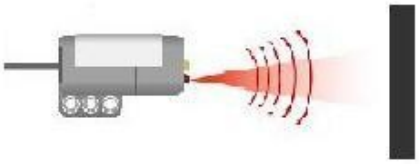


El sensor de luz, mide la luz reflejada o la ambiental. El sensor tiene un foto-resistor y un LED (diodo emisor de luz).

El foto-resistor convierte la luz en energía eléctrica y la envía al NXT. El LED puede ser encendido o apagado en el programa.

El sensor de luz es un sensor analógico. Provee retroalimentación en rango de números. NXT convierte la señal eléctrica desde 0 hasta 100.

Se tienen modos pasivo y activo:
Pasivo solo recibe la señal.
Activo emite y espera a que la señal sea reflejada.



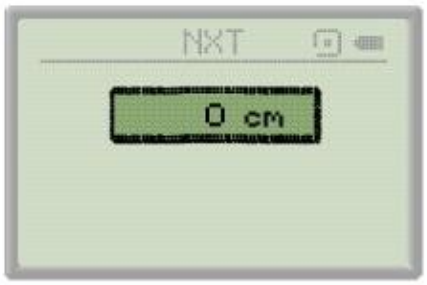
Sensor Ultrasónico

Utiliza la velocidad del sonido para medir la distancia a un objeto. El sensor tiene un emisor y un receptor.

Los científicos han determinado la velocidad del sonido en aproximadamente 341 m/s. El sensor ultrasónico para determinar la distancia a un objeto utiliza la siguiente ecuación.

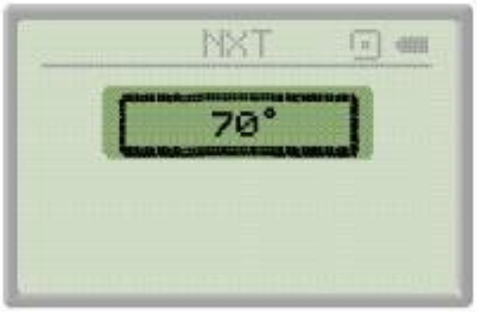
$$\frac{\text{T x Veloc. Sonido}}{2} = \text{Distancia al objeto}$$

T = tiempo entre que una onda ultrasónica es emitida y recibida



Servo-Motor Interactivo / Sensor de Rotación

El motor provee al robot la habilidad de saber qué tan lejos ir. Cada motor tiene incluido un Sensor de Rotación el cual le permite al programador controlar exactamente las rotaciones de la rueda.El servo-motor puede medir el movimiento de la rotación con margen de 2 grados.



Estructuras Básicas

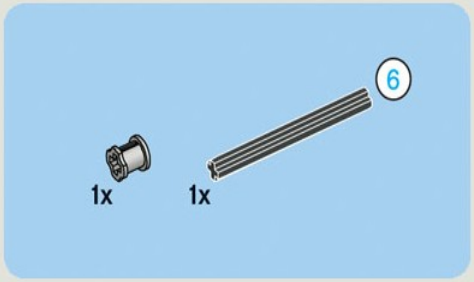
En cada kit se incluye un pequeño libro con la base para ensamblar un pequeño robot con estructura básica **TASKBOT**.

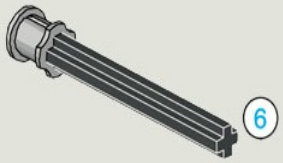
A continuación se anexan imágenes del **TASKBOT**, a partir de esta es posible conectar los sensores y partes mecánicas adicionales que le permitan al robot realizar sus tareas lo mejor posible.

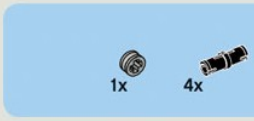
En cada imagen se muestran las piezas necesarias en número y tipo, así como el lugar donde ensamblar cada parte.

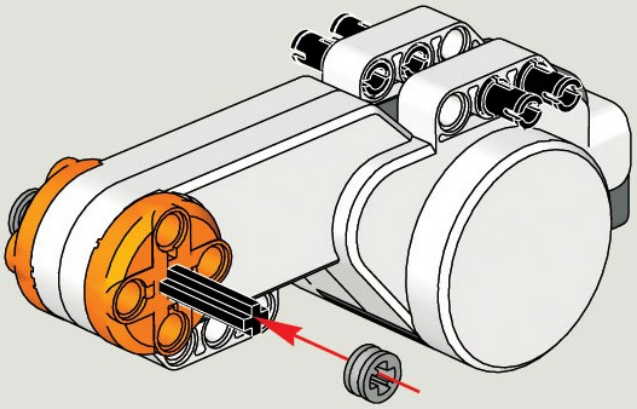
Cabe señalar que esta es una estructura de ejemplo y en la realidad la imaginación del usuario es la pauta a desarrollar mecanismos y elementos necesarios para cumplir las tareas asignadas.

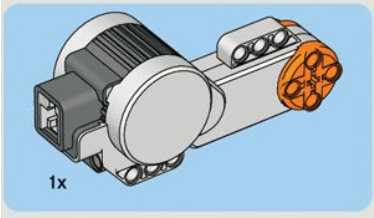
Es importante señalar deben colocarse las pilas o la batería recargable (previamente cargada) antes de montar el NXT ya que se evitará el tener que desarmar y armar la estructura.

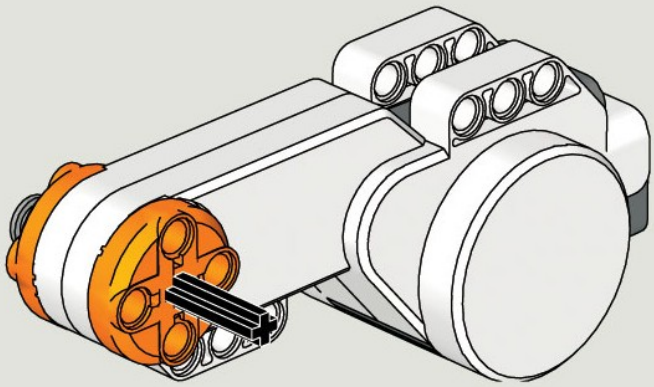

6
1x
1x

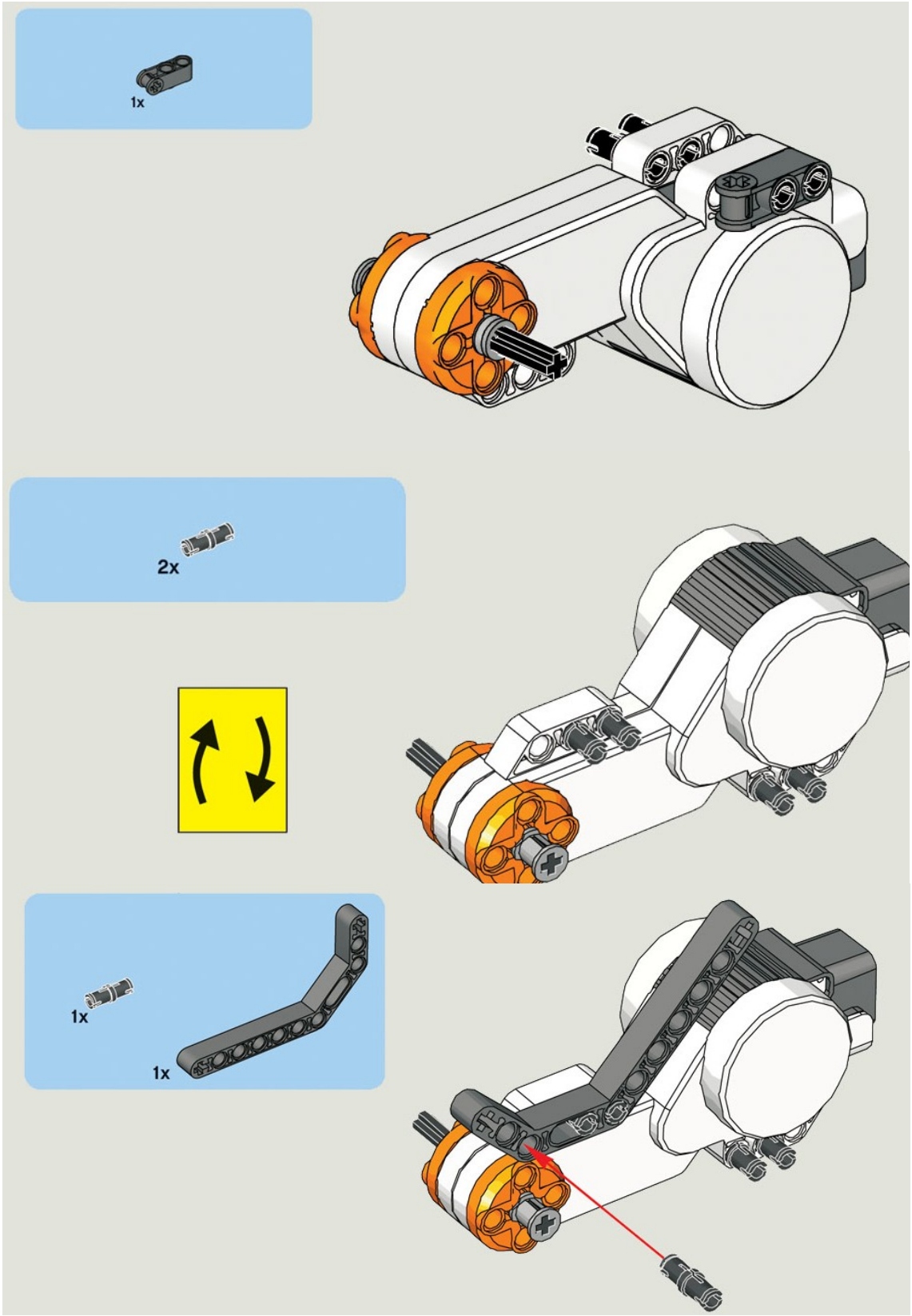

6

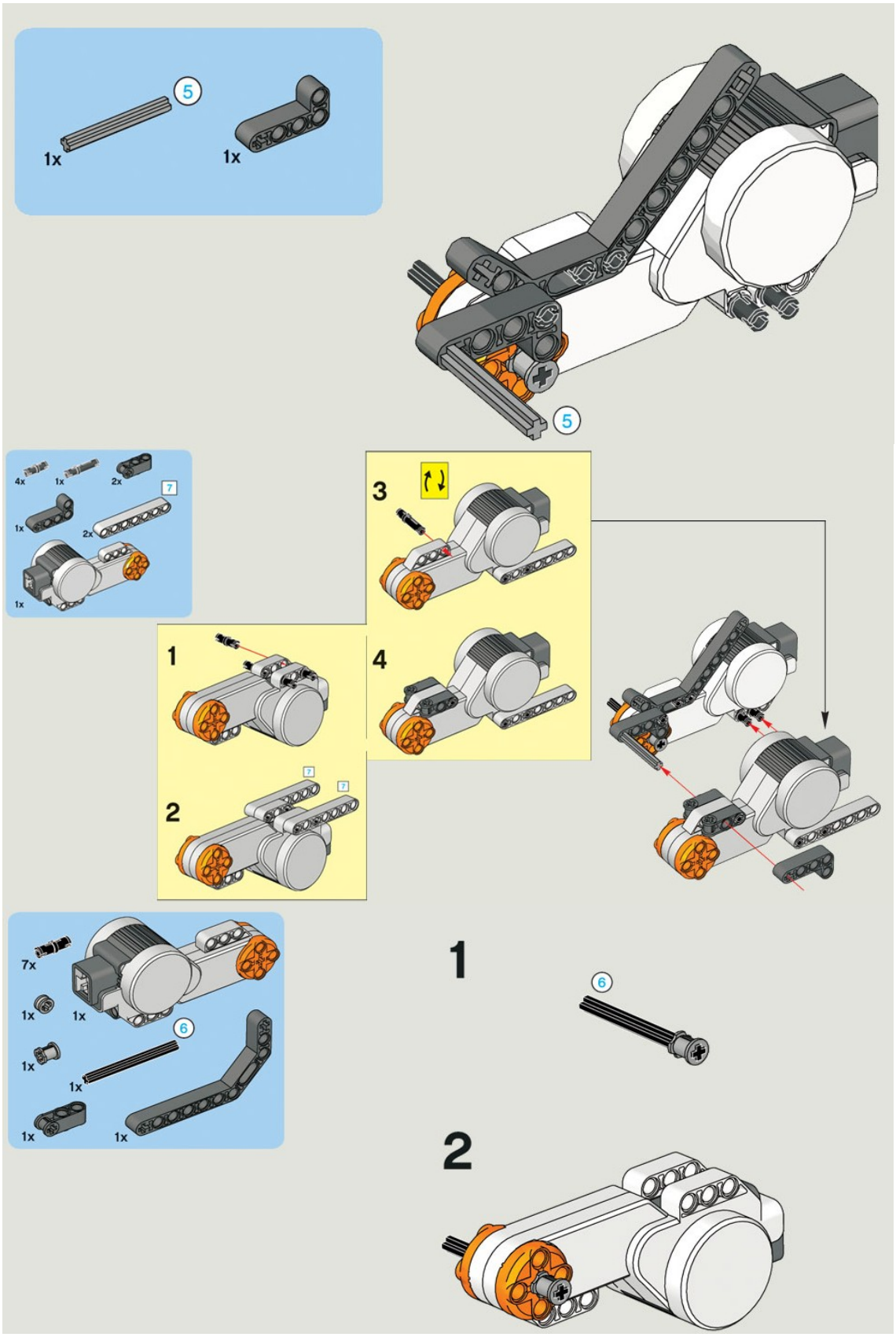

1x
4x

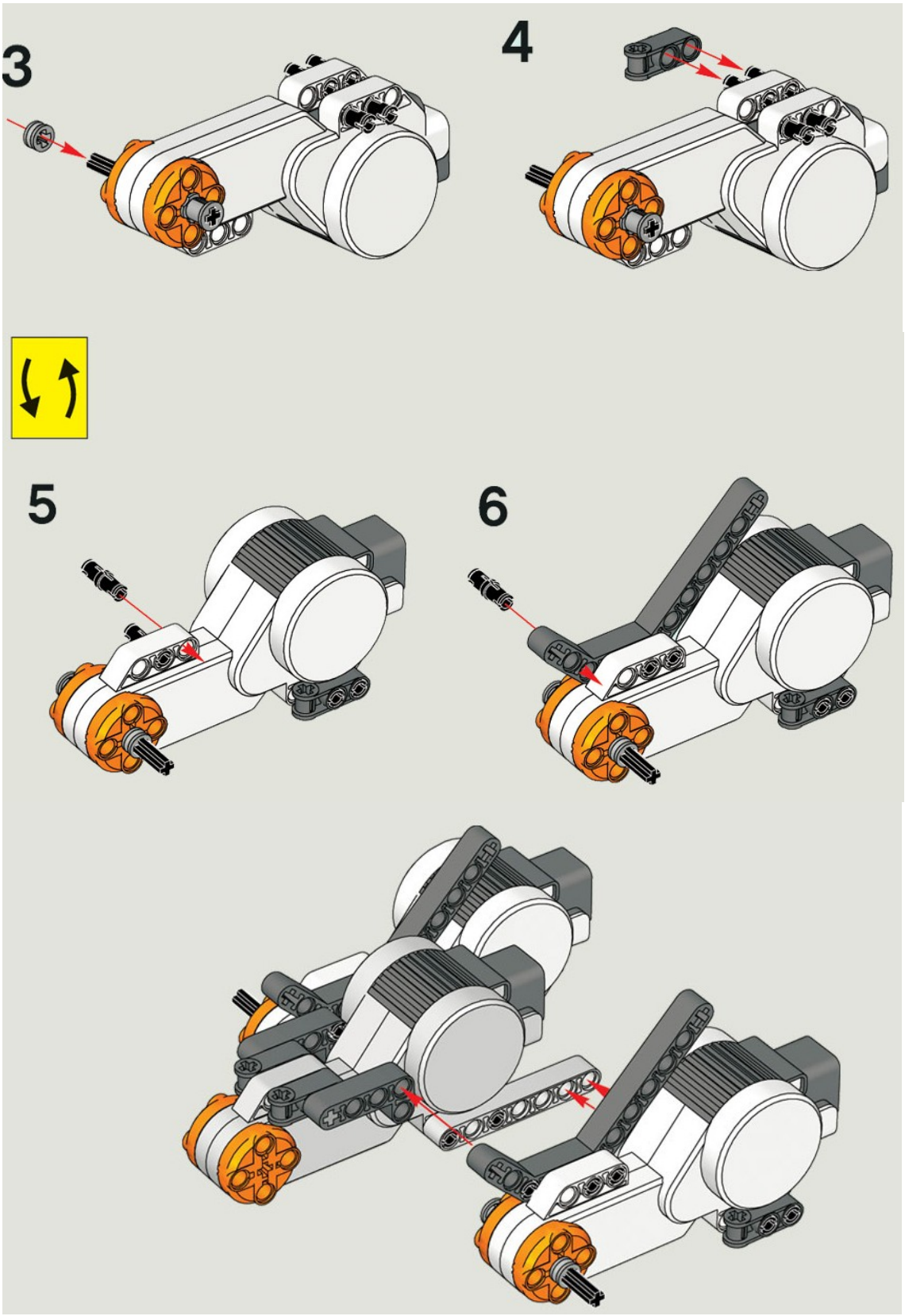


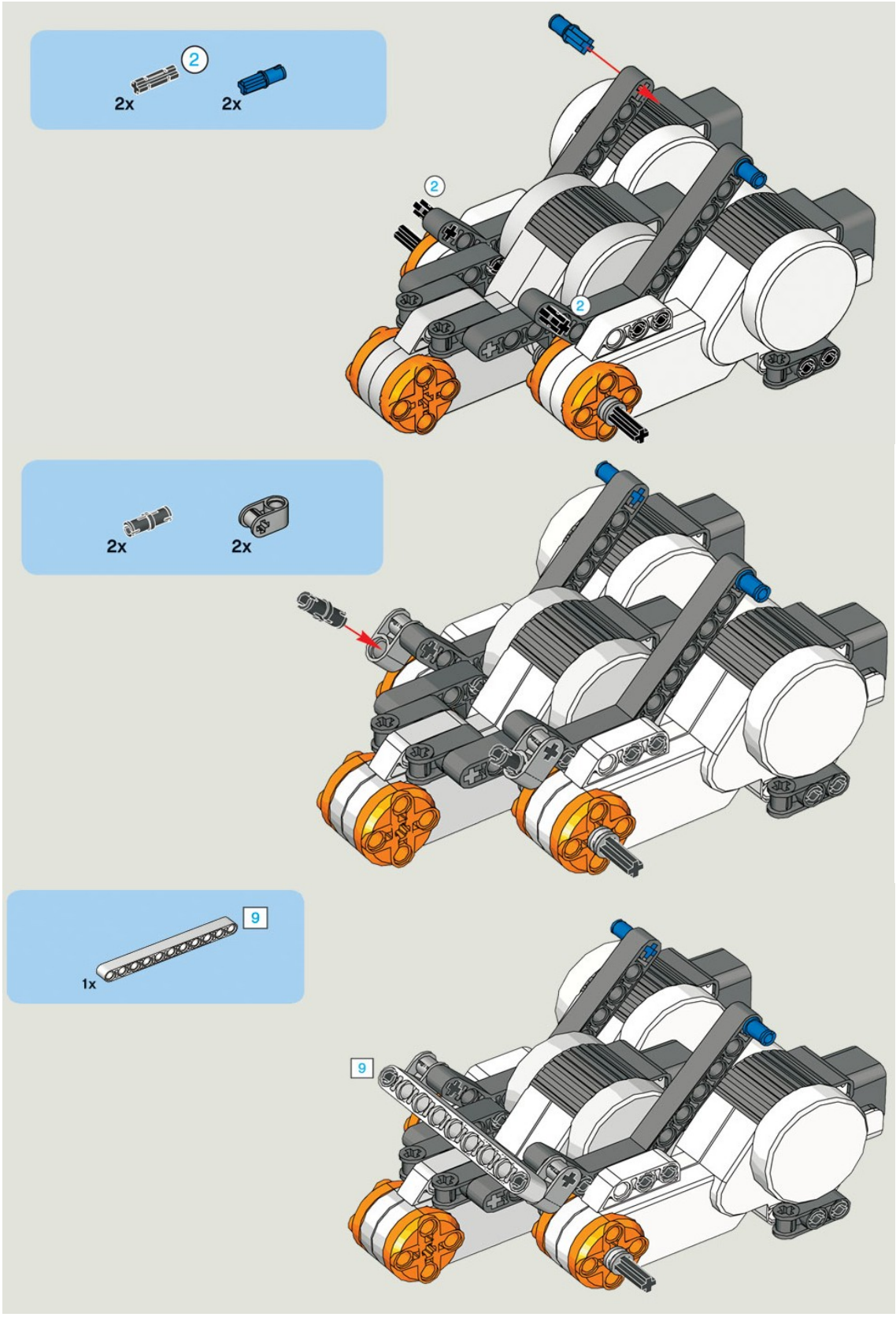

1x

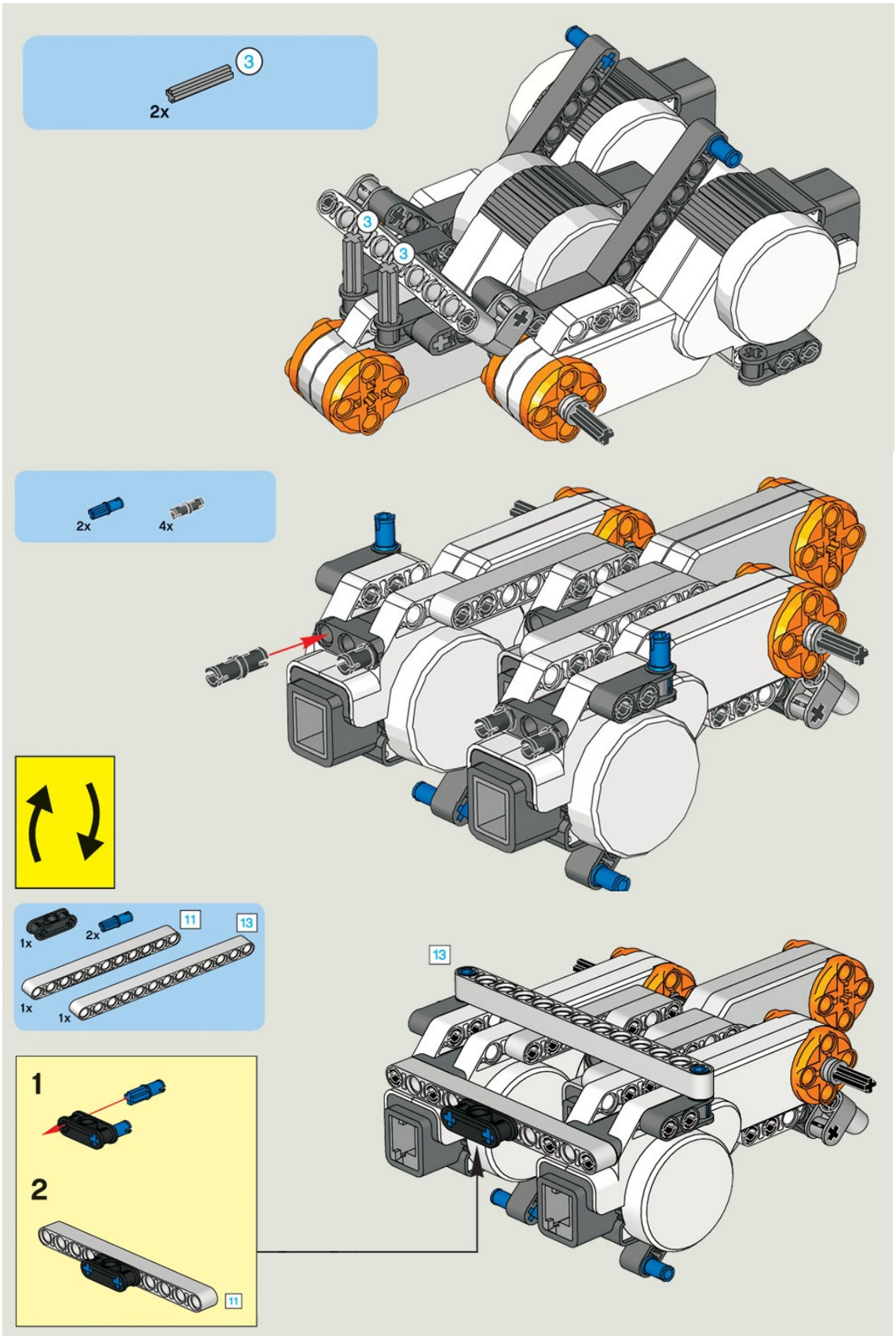


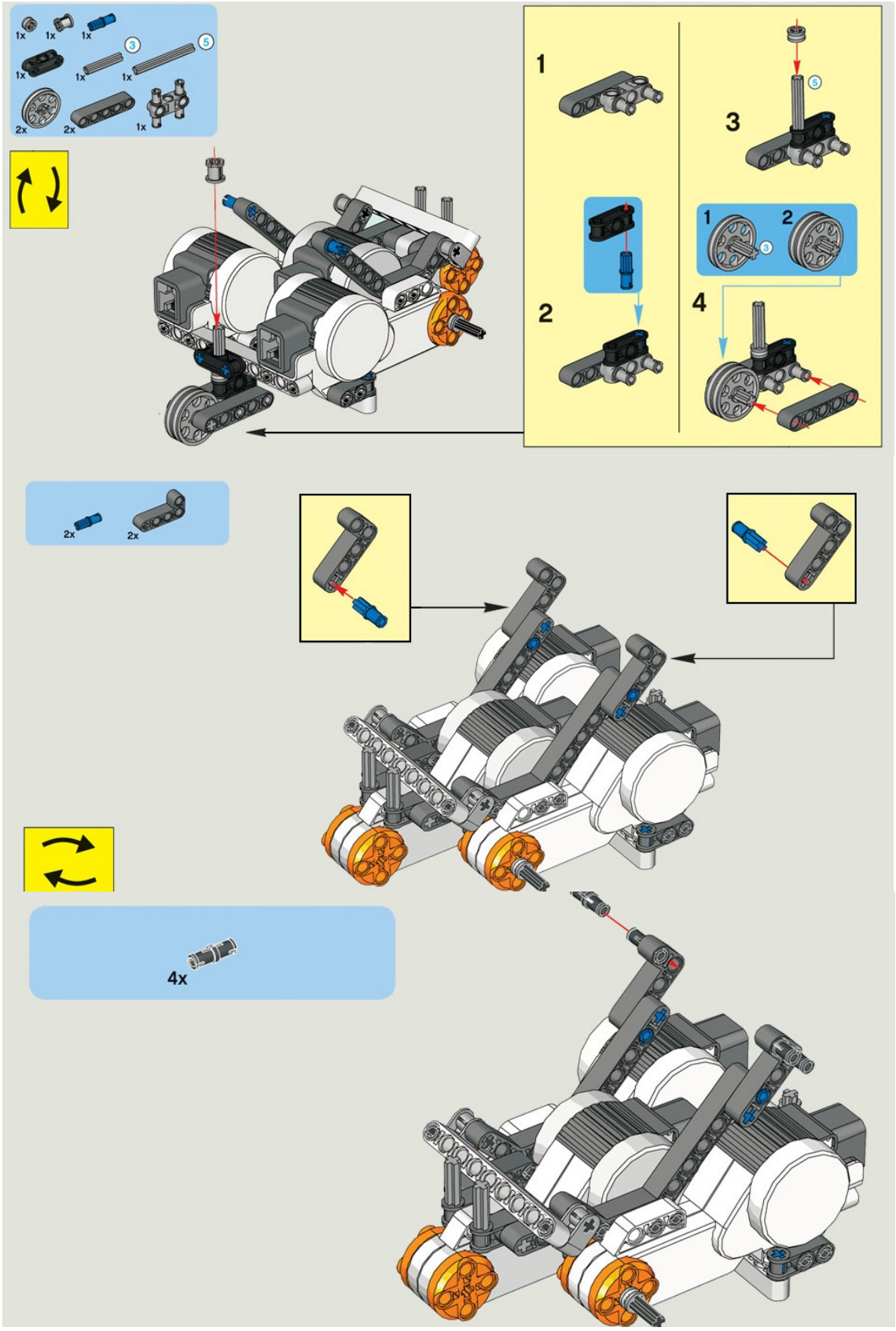

1x
2x
1x
1x

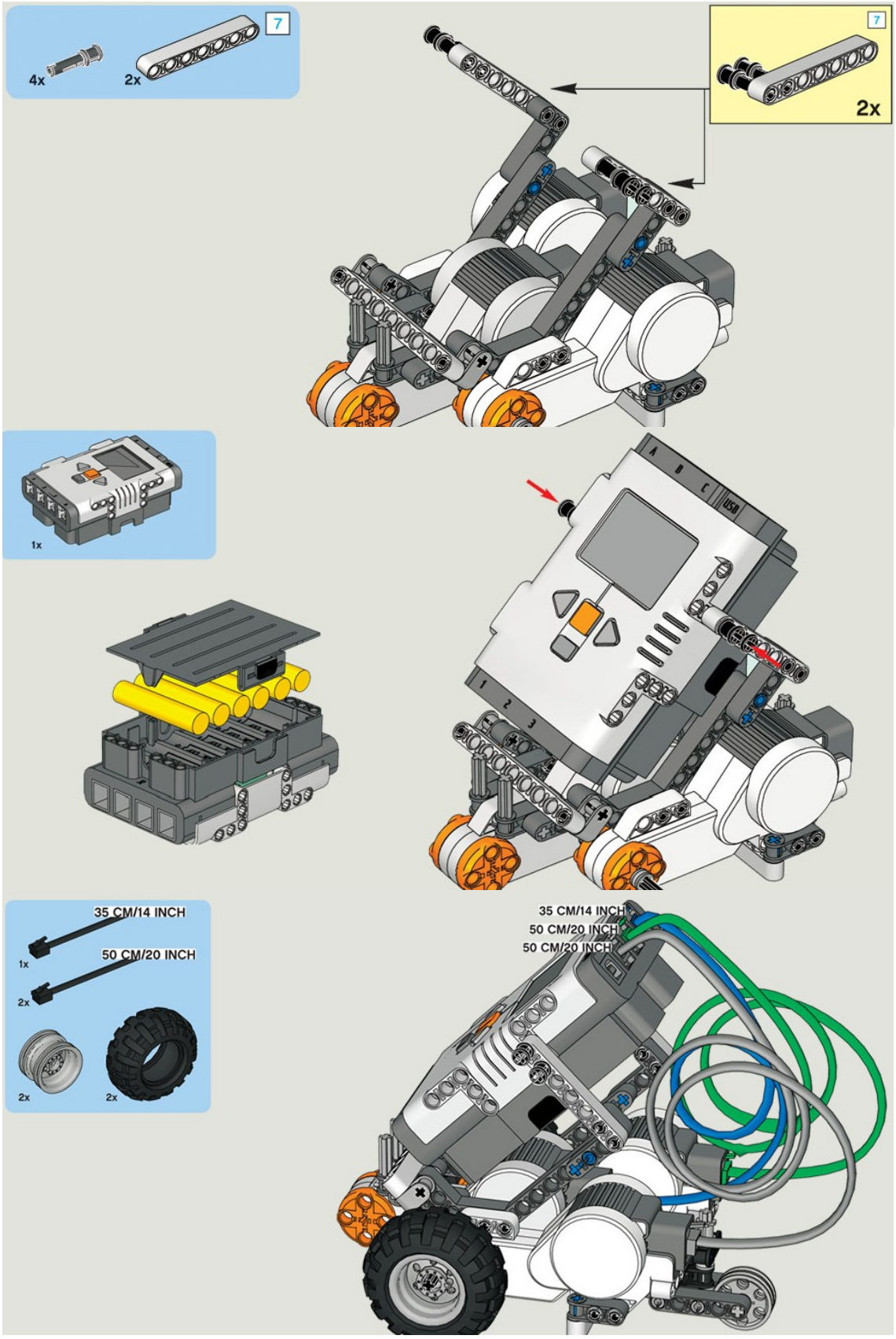

5
1x
1x
5
4x
1x
2x
7
1x
2x
1x
3
1
4
2
7
7
7x
1x
1x
6
1x
1x
1x
1x
1
6
2


3
4
5
6


2
2x
2x
2
2
2x
2x
9
1x
9


3
2x
3
3
2x
4x
1x
2x
11
13
1x
1x
13
1
2
11


1x
1x
1x
3
5
1x
1x
1x
2x
2x
1x
1
2
3
4
1
2
3
2x
2x
4x


7
4x
2x
7
2x
1x
35 CM/14 INCH
50 CM/20 INCH
1x
2x
2x
2x
35 CM/14 INCH
50 CM/20 INCH
50 CM/20 INCH

Lenguajes de Programación

Para programar los NXT abordaremos dos tipos de lenguajes, el visual del mismo nombre (NXT), basado en una plataforma LabVIEW, y el de código con base en C++. También existen lenguajes de código para el brick (cerebro) NXT basados en JAVA, pero no se tratarán en este manual, al final se anexan las ligas para los interesados en otros lenguajes de programación.

NXT:

El software LEGO[MR] NXT permite generar programas de una forma bastante fácil y amigable, ya que mediante un entorno gráfico podemos establecer una línea de comandos visuales (íconos) y definir especificaciones para cada uno de ellos.

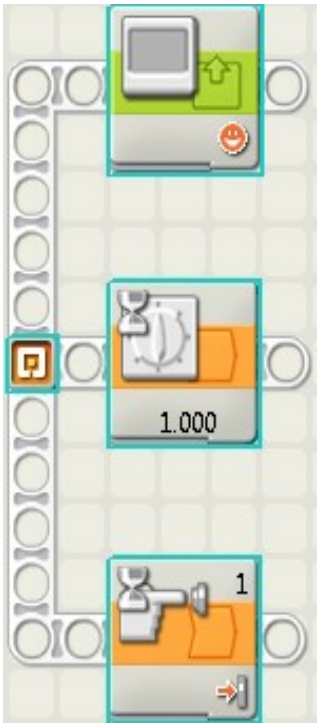
Además permite formar ciclos y condiciones a partir de las señales obtenidas por los sensores, para así poder controlar el comportamiento de los motores. Es posible manejar funciones lógicas, incluso crear secuencias de sonido. También es posible descargar de internet sonidos o programas ya elaborados y modificarlos de acuerdo a la tarea a realizar.

A continuación se anexa una lista de los principales íconos y su función:

MOTOR NXT, determina a que puerto ira conectado, así como la potencia con la que girará el motor, el sentido del giro, además de seleccionar el modo en que lo hará, pueden ser: grados, revoluciones, segundos o de forma infinita.

RECORD, para grabar sonidos en nuestra unidad principal.

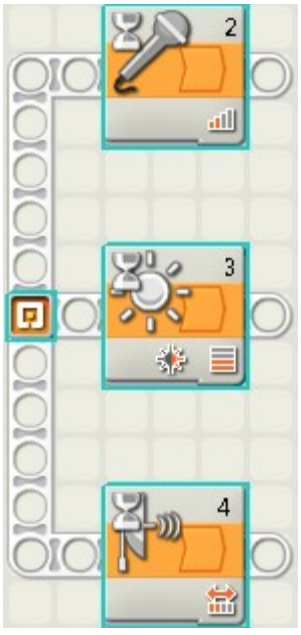
SONIDO, reproduce sonidos predeterminados que se encuentran en el programa NXT.



IMAGEN, proyecta una serie de imágenes en la pantalla del brick NXT.

TIEMPO DE ESPERA es un retardo, el cual se determina en el bloque de programación, y puede incluirse como retardo en cada acción del programa.

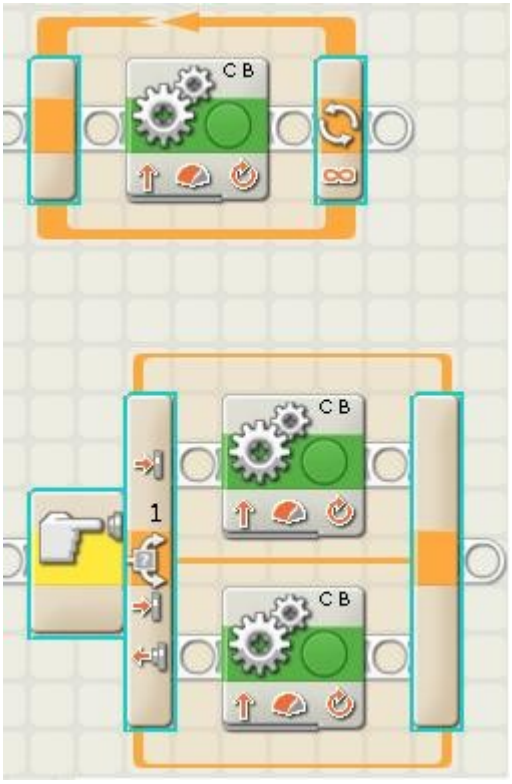
ESPERA AL SENSOR DE TACTO, espera la señal del sensor de tacto periférico, indica el puerto al que está conectado (en este ejemplo al 1), puede esperar señal de activación o desactivación.



ESPERA AL SENSOR DE SONIDO, espera la señal del sensor de sonido, indica el puerto al que está conectado (en este ejemplo 2), la señal es de valores mayor o menor a un nivel de sonido predeterminado.

ESPERA AL SENSOR DE LUZ, espera la señal del sensor de luz, indica el puerto al que está conectado (en este ejemplo 3), la señal es de valores mayor o menor de una determinada intensidad luminosa. Puede activarse la emisión de luz.

ESPERA AL SENSOR ULTRASÓNICO, espera la señal del sensor ultrasónico, indica el puerto al que está conectado (en este ejemplo 4), se mide un valor mayor o menor a una distancia predeterminado. La distancia puede medirse en pulgadas o centímetros.



CICLO LOOP, se utiliza para repetir secuencias de código. Se fija la condición que terminará el lazo: tiempo transcurrido, el número de repeticiones, una señal de la lógica o un sensor. También se incluye la opción de repetir el programa de forma infinita.

SWITCH (interruptor), este bloque permite elegir entre dos opciones de programa. La selección depende de la señal del sensor activado o desactivado.

MOTOR (RCX) controla la potencia y el sentido de giro, del motor. (Para utilizar el motor RCX se requiere el cable del convertidor.)

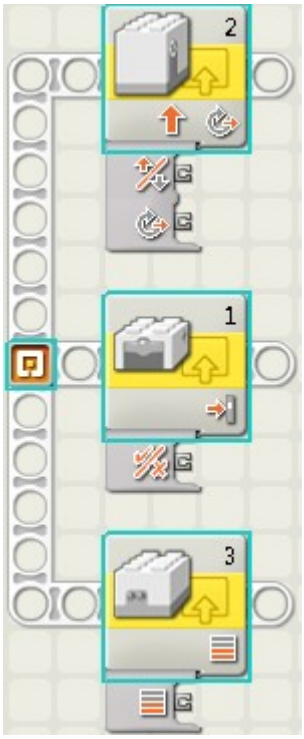
MENSAJE BLUETOOTH permite enviar un mensaje mediante conexión inalámbrica, debe elegirse el tipo de mensaje a enviar Texto, Numérico o Lógico, así como el número de conexión para cada dispositivo (NXT o celular) y el número del recipiente para el mensaje.

LÁMPARA RCX para controlar la lámpara se necesita un bloque para encender y un segundo bloque para apagar. (Para utilizar la lámpara RCX requiere un cable del convertidor.)

TEMPORIZADOR, cuando el programa comienza, los tres contadores incorporados del NXT comienzan automáticamente a contar. Con este bloque puede seleccionarse el valor actual de uno de los timers o hacer al contador comenzar a contar otra vez desde cero.

BOTONES DE NXT emite una salida a través de uno de los cables de datos cuando uno de los botones de NXT se activa. Debe seleccionarse el botón y la acción que enviará la señal "verdadera".

RECEPCION DE MENSAJE. para recibir mensajes inalámbricos, debe fijarse el tipo de mensaje y el número de la caja de entrada para hacer que coincidan con los datos del NXT que envía. La salida de este bloque puede ser el mensaje entrante en sí mismo o un valor "verdad / falso" lógico (si usted está comparando el mensaje entrante a un mensaje de prueba).



SENSOR DE ROTACIÓN RCX cuenta el número de las señales (16 para una rotación) a que el motor gira. A través de los alambres / cables de los datos, puede enviar el número de señales o pulsos del motor y una señal de la lógica (verdadero / falso) basado en si el número de pulsos es superior o inferior al de referencia. (Para utilizar el sensor RCX se requiere un cable del convertidor.)

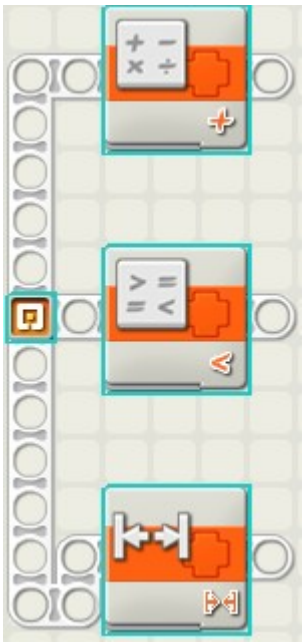
SENSOR DE TACTO RCX este bloque envía una señal Lógica (*verdadera o falsa*) a través de un alambre de datos. Si el sensor ha sido activado, emitirá una señal "verdadera", si no es activado enviará una "falsa". (Para utilizar el sensor RCX se requiere un cable del convertidor.)

SENSOR DE LUZ RCX a través de los alambres de datos, puede enviar el valor actual de luz y una señal lógica (*verdadera o falsa*) respecto a la lectura luminosa actual. (Para utilizar el sensor RCX se requiere un cable del convertidor.)

SENSOR DE TEMPERATURA RCX a través de alambres de datos, puede enviar el valor de temperatura actual o una señal Lógica (*verdadera o falsa*), basada en si la temperatura es superior o inferior a una previamente definida para activación. Pueden cambiarse las unidades de la temperatura de Centígrados a Fahrenheit (o viceversa). (Para utilizar el sensor RCX se requiere un cable del convertidor).

PARO o STOP parará el programa y cualquier motor, lámparas o sonido que estén funcionando.

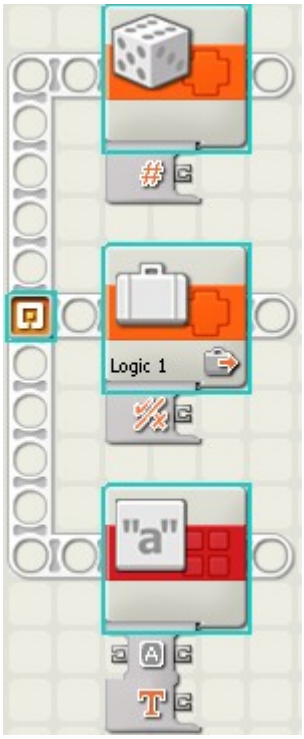
BLOQUE LÓGICO realiza una operación lógica en sus entradas y envía la respuesta verdadera/falsa por un alambre de datos. Utiliza solo dos posibles valores verdad y falso tanto para entradas como para salidas, frecuentemente escritos como números 1 y 0.



BLOQUE DE MATEMÁTICAS realiza operaciones aritméticas simples como la adición, substracción, multiplicación, y división. Los números de la entrada se pueden teclear o enviar mediante los alambres de datos.

BLOQUE DE COMPARACIÓN, puede determinar si un número es mayor que (>), menor que (<), o igual (=) a otro número. Los números de la entrada se pueden teclear o enviar mediante los alambres de datos.

BLOQUE DE RANGO, puede determinar si un número se encuentra dentro o fuera de un rango o intervalo de números. Los valores de la entrada pueden ser tecleados, o ser provístos mediante los alambres de datos. La señal lógica de salida (*verdadera o falsa*) será enviada por un alambre de datos.



BLOQUE DE AZAR, genera un número al azár. Se puede utilizar este número para generar comportamientos imprevistos en el robot. Los límites mínimo y máximo del rango que puede tomar este número se pueden teclear o enviar mediante los alambres de datos.

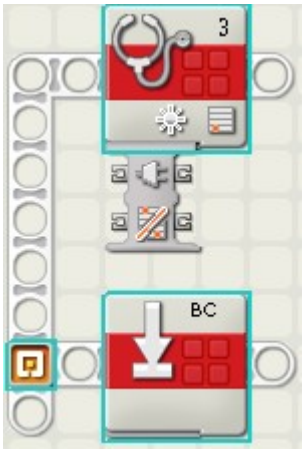
BLOQUE VARIABLE, una variable puede almacenar valores en la memoria del NXT, otros bloques de programa pueden leer el valor actual de dicha variable o incluso cambiarlo.

BLOQUE DE TEXTO, puede reunir grupos de caracteres para formar textos, una oración puede formarse utilizando tres cadenas pequeñas de texto. El texto de entrada se puede teclear directamente o enviarse mediante los alambres de datos.

CONVERTIDOR NUMERO A TEXTO, este bloque tomará un número (una lectura de un sensor) y lo convertirá a texto, el que será presentado en la pantalla del NXT. El número de la entrada puede ser tecleado o enviado mediante los alambres de datos.

MANTENTE VIVO, permite seguir funcionando al NXT, evitando que entre en modo del sueño (hibernación). Se recomienda utilizar este bloque si el programa requiere esperar un tiempo mayor al que necesita el NXT para entrar en modo de hibernación.

ACCESO AL ARCHIVO, permite guardar datos del robot en el NXT. Después de guardar el archivo, se debe utilizar otro bloque de acceso para cerrar el archivo antes de que se pueda leer o eliminar el archivo (lo que se hará usando un tercer bloque de acceso).



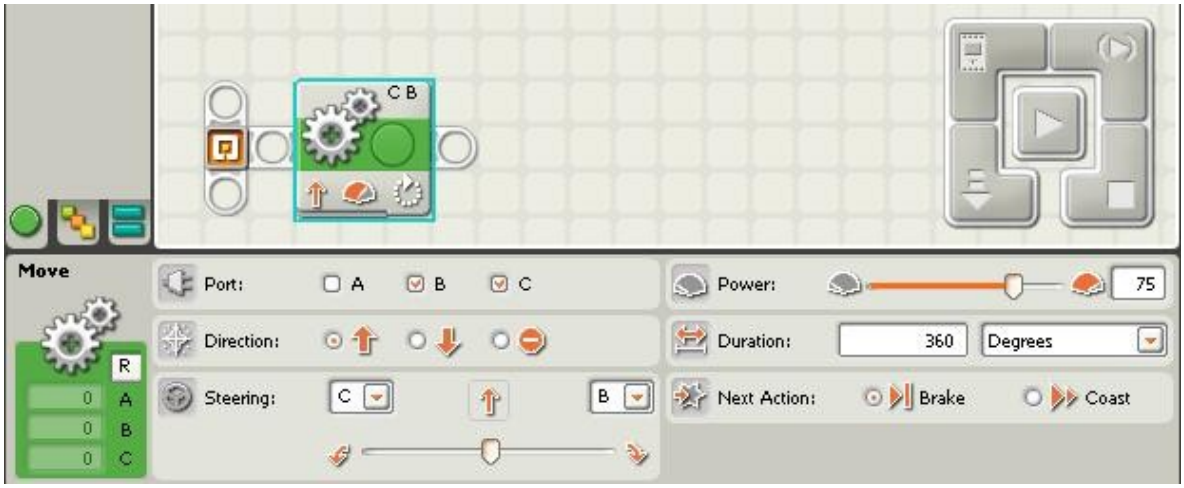
CALIBRACIÓN este bloque permite calibrar el valor mínimo (0%) o máximo (100%) detectado por un sensor de sonido o de luz. Se requieren dos bloques para calibrar completamente un sensor uno con el valor mínimo y otro con el valor máximo.

REAJUSTE DEL MOTOR, los servomotores interactivos tienen un mecanismo de corrección de error automático que perimiten que el robot se mueva de forma precisa. Sin embargo hay ocasiones donde se desea desabilitarla, este bloque lo permite.

Además existen los bloques Custom o diseñados por el usuario donde se pueden crear midis o diversas aplicaciones extra.

Cuando seleccionamos los íconos o bloques de programa, se despliega en la parte inferior una pantalla. Como ejemplo en la siguiente figura se presenta un motor, en la pantalla de configuración es posible indicar los puertos que se quieren controlar, la dirección, la potencia, la duración del giro en grados, tiempo, número de revoluciones, o si se trata de una acción por tiempo indefinido.

Para el caso de los sensores se podrá definir la intensidad que se espera como respuesta, si trabajará de forma activa o pasiva, y en qué puerto estará conectado. (Los puertos mencionados anteriormente son los de base, pero pueden modificarse mediante el software).



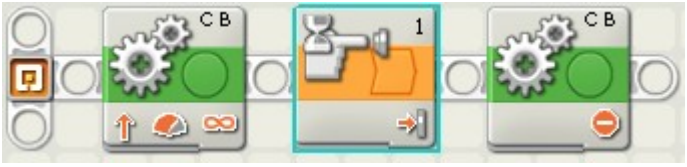
Programa Ejemplo:

DETECTA Y PARA

SENSOR DE TACTO

Un programa sencillo, al accionar este programa el robot avanza y al detectar una barrera se para hasta que volvamos a accionar el programa.

Para lograr el funcionamiento arriba mencionado se presenta el siguiente programa, en el cual comenzamos por colocar el ícono correspondiente a los motores (se seleccionan los puertos B y C) y se indica que se moverán hacia adelante indefinidamente. Posteriormente se especifica que deberá continuar con esta acción hasta que sea accionado el sensor de tacto, para pasar al siguiente bloque en el cual los motores conectados a los puertos B y C se detienen. Este programa solo volverá a funcionar hasta que vuelva a seleccionarse desde el NXT y se le indique arrancar nuevamente.

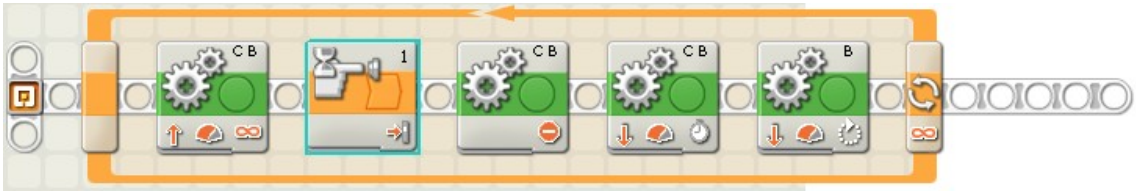


Programa Ejemplo:

DETECTA Y GIRA

Si queremos que el programa siga corriendo indefinidamente debemos introducir los bloques de programa dentro de un ciclo (loop), y de esa forma el programa correrá una y otra vez.

En la siguiente figura se muesta el programa anterior al que se le han agregado dos indicaciones más, para que los motores giren y el robot siga en movimiento hasta que vuelva a detectar algo con su sensor de tacto.



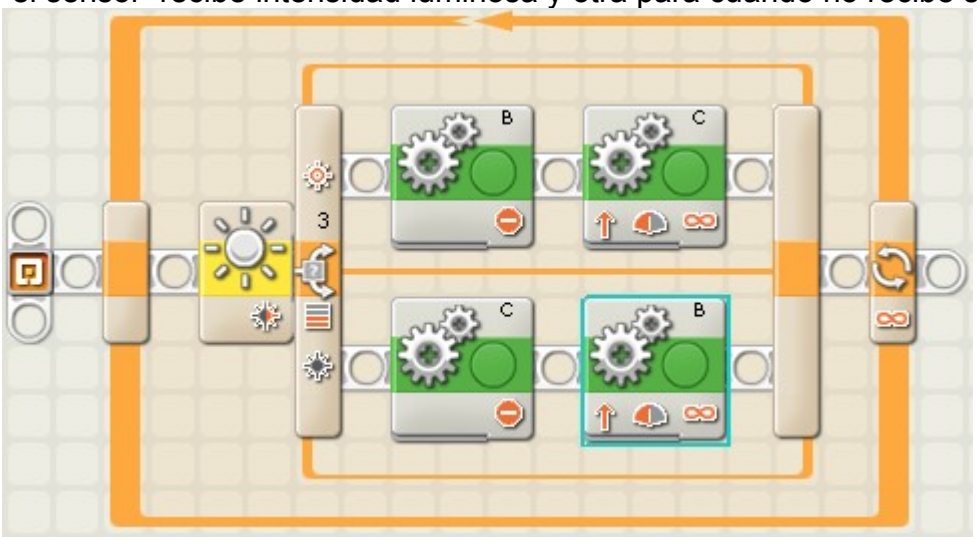
Programa Ejemplo:

SEGUIDOR DE LINEA

SENSOR DE LUZ

Este programa permite que el robot detecte una línea trazada previamente, y sea capaz de seguirla sin salirse del curso.

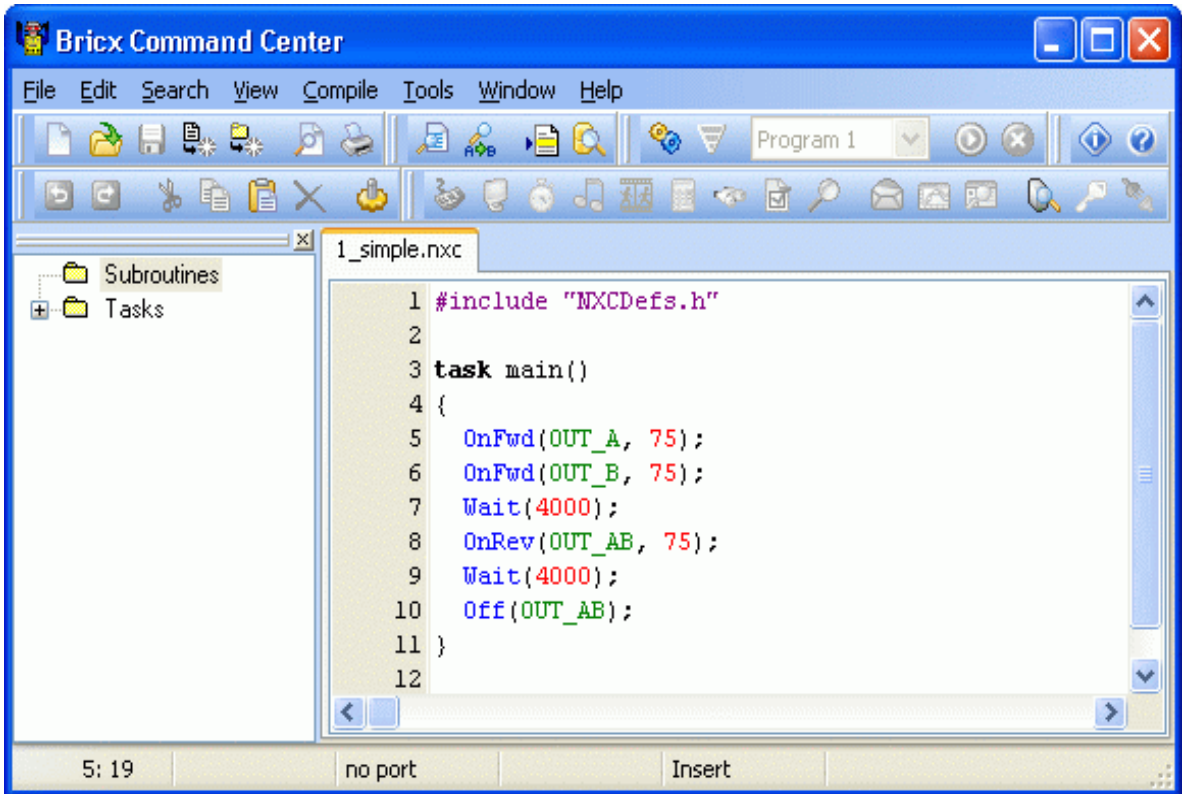
En este programa se están utilizando un ciclo infinito (loop) y además un interruptor o switch (de selección) con el sensor de luz. Dependiendo de la señal que reciba será la respuesta, se tienen dos posibles secuencias a seguir una para cuando el sensor recibe intensidad luminosa y otra para cuando no recibe señal.



De esta forma pueden generarse programas y tareas cada vez más complejas.

NXC: Lenguaje en código basado en el NBC que corre sobre C++, será necesario el software BricxCC para poder compilar y guardar con la extención correspondiente (*.nxc)

Al iniciar el programa debe indicarse qué modelo de brick (cerebro) se va a utlizar ya que es posible compilar en lenguajes diferentes, también debe seleccionarse el puerto mediante el cual se hará la conexión con el cerebro NXT para cargarle el programa correspondiente. Generalmente se utiliza el puerto USB. En la ventana del BricxCC podremos generar el código y compilarlo para ver que no tenga errores en la sintaxis.



Programa Ejemplo: **Movimiento hacia adelante, y hacia atrás**

```
#include "NXCDefs.h"

task main()
{
OnFwd(OUT_A, 75);
OnFwd(OUT_C, 75);
Wait(4000);
OnRev(OUT_AC, 75);
Wait(4000);
Off(OUT_AC);
}
```

En el cual, la sentencia ***#include "NXCDefs.h"*** es necesaria para poder compilar el programa, si se tiene la versión 3.3.7.17, ya no será indispensable, pero para versiones anteriores debes ponerla siempre al principio de cualquier programa. Ya que define el lenguaje para la programación en NXC.

1. Se indica la primera tarea **task main ()**
2. Se abre la llave de la tarea.
3. La instrucción **OnFwd** (hacia adelante) debe ir seguida por la salida (motor en el puerto, A, B, o C) que va realizar la función, también es necesario indicar el porcentaje de potencia que deseamos utilizar.
4. Para que espere durante un lapso de tiempo se emplea **Wait** y se indican los milisegundos, para el ejemplo anterior 4000 quiere decir 4 segundos completos.
5. La instrucción **OnRev** (en reversa) debe ir seguida por la salida (motor en el puerto, en este caso A y C). *Nota:* pueden indicarse ambas salidas en la misma instrucción.
6. Nuevamente un tiempo de espera con **Wait** durante 4 segundos.
7. La función **Off** será quien determine el paro de los motores, en el programa se indican las salidas A y C (puertos del motor) en una sola indicación (OUT_AC).
8. Se cierra la llave de la tarea **task main ()**
9. Cada renglón debe concluir con punto y coma.

Programa Ejemplo: **Ciclo "repeat"**

```
#include "NXCDefs.h"
#define TURN_TIME 360

int move_time;                  // define variable
task main()
{

     move_time = 200;           // indica el valor
                                inicial
     repeat(50)

          {
          OnFwd(OUT_AC, 75);
          Wait(move_time);      // usa la variable
                                para detenerse
          OnRev(OUT_C, 75);
          Wait(TURN_TIME);
          move_time += 200;     // incrementa la
                                variable
          }
```

```
Off(OUT_AC);
}
```

1. Se define el término **TURN_TIME** como 360
2. Se define una variable **move_time** del tipo entero **int**
3. Se indica la primera tarea **task main ()**
4. Se abre la llave de la tarea.
5. Se indica el valor inicial de la variable move_time
6. Se agrega el ciclo **repeat**, la cual repetira la operación 50 veces
7. La instrucción **OnFwd** (hacia adenlante) debe ir seguida por la salida (motor en el puerto, A, B, o C) que va realizar la función, tambien es posible indicar el porcentaje de potencia que deseamos utilizar.
8. Para que espere un periodo de tiempo **Wait** y se llama la variable definida move_time que equivale a 200.
9. La instrucción **OnRev** (en reversa) debe ir seguida por la salida (motor en el puerto C).
10. Para que espere nuevamente **Wait** y se llama el valor definido TURN_TIME que equivale a 360.
11. Se incrementa la variable move_time
12. Se cierra la llave del ciclo **repeat.**
13. La función **Off** será quien determine el paro de los motores, en el programa se indican las salidas A y C (puertos del motor) en una sola indicación (OUT_AC).
14. Se cierra la llave de la tarea **task main ()**
15. Cada renglón debe concluir con punto y coma.

Programa Ejemplo: **Ciclo “repeat” anidado**

A continuación se muestra un ejemplo de como anidar un ciclo dentro de otro. Es importante el tener cuidado con la ubicación de las llaves.

```
#define MOVE_TIME 1000
#define TURN_TIME 500
task main()
{
     repeat(10)
     {
          repeat(4)
               {
               OnFwd(OUT_AC, 75);
               Wait(MOVE_TIME);
               OnRev(OUT_C, 75);
               Wait(TURN_TIME);
               }
```

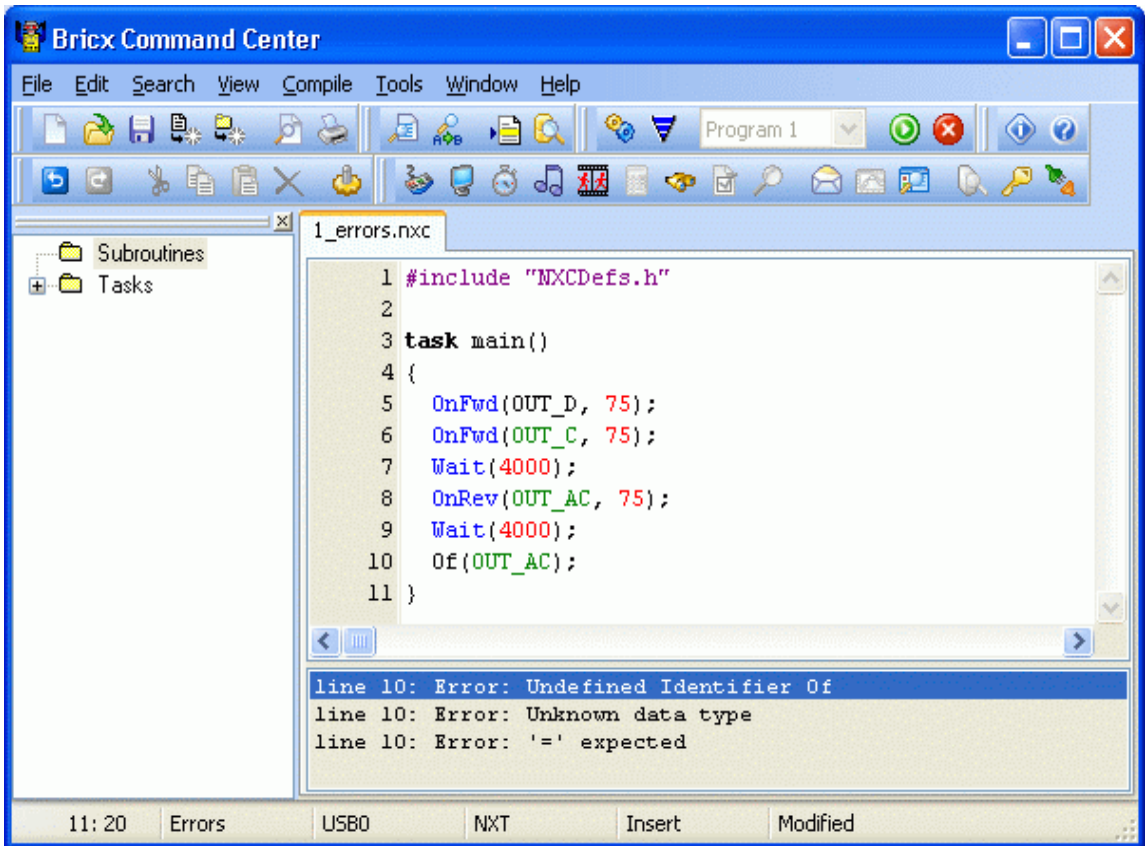
```
        }
Off(OUT_AC);
}
```

Para compilar y bajar el programa

En la imagen anterior se muestran los controles (iconos) para compilar, bajar al NXT, correr y detener el programa, respectivamente.

Para correr el programa de forma automática se debe oprimir **Ctrl+F5**.

En caso de generar algun error, se mostrará como en la siguiente pantalla:



Al igual que la mayoría de los compiladores, se muestra la línea en la que existe algún error de programación o comunmente tipográfico.

A continuación se presentan algunos de los Macros o terminos principales, para ver el resto consultar el manual (**NXC N**ot e**X**actly **C**).

Macros de Motor API

OnFwd(puertos, potencia): Arranca los motores gira al nivel de potencia determinado.
OnRev(puertos, potencia): Igual a OnFwd pero en la dirección opuesta.
OnFwdReg(puertos, potencia, modo de regulación): Igual a OnFwd pero permite seleccionar el modo de regulación.
OnRevReg(puertos, potencia, modo de regulación): Dirección opuesta a OnFwdReg.
OnFwdSync(puertos, potencia,% de giro): Igual a OnFwd pero permite sincronizar múltiples motores.
OnRevSync(puertos, potencia,% de giro): En dirección opuesta a OnFwdSync.
RotateMotor(puertos, potencia, ángulo): Girar los motores a un ángulo determinado. Potencia negativa o valor negativo de potencia gira en la dirección contraria.
RotateMotorEx(puertos, potencia, ángulo,% de giro, sincronía boleana): Igual que RotateMotor pero con habilidad adicional de sincronizar los motores y especificar un radio de giro.
Coast(puertos): Para los motores sin frenar.
Off(puertos):Para los motores frenando.

Macros de Sensor API

El puerto debe ser una constante numérica
SetSensorType(puerto, tipo): Fija el tipo de sensor.
SetSensorMode(puerto, modo): Fija el modo de sensor.
ClearSensor(puerto): Limpia los datos del sensor (Ajusta a cero).
SetSensorTouch(puerto): Fija el tipo y modo como sensor de tacto.
SetSensorLight(puerto): Fija el tipo y modo como sensor de luz.
SetSensorSound(puerto): Fija el tipo y modo como sensor de sonido.
SetSensorUltrasonic(puerto): Fija el tipo y el modo como sensor ultrasónico.
El puerto puede ser una variable o una constante
ResetSensor(puerto): Espera a que el valor del sensor sea válido.
ReadSensor(puerto, valor): Convierte la lectura de un sensor a un valor.
ReadSensorUS(puerto, valor): Convierte la lectura de un sensor ultrasónico a un valor.

Macros de Dibujando API

TextOut(x, y, cls, text): Dibuja el texto en (x,y) y opcionalmente limpia la pantalla antes.
NumOut(x, y, cls, numeric): Dibuja el número en (x, y) con opción de limpiar pantalla.
PointOut(x, y, cls): Dibuja un punto en (x,y) con opción de limpiar pantalla.
LineOut(x1, y1, x2, y2, cls): Dibuja una línea de (x1,y1) a (x2,y2) con opción de limpiar pantalla.
RectOut(x, y, width, height, cls): Dibuja un rectángulo en (x,y) con ancho y altura especificada. Limpiar pantalla antes de dibujar (opcional).
CircleOut(x, y, radius, cls): Dibuja un círculo centrado en (x,y) con radio específico
GraphicOut(x, y, filename, variables, cls): Dibuja el archivo ícono RIC especificado por el nombre de archivo en (x,y). Limpiar pantalla antes de dibujar (opcional) . Usa el arreglo de valores de 16 como una función de la transformación para los comandos en el archivo de RIC.

Macros de Sonido API

PlayTone(frecuencia, duración): Toca un tono. La frecuencia esta en Hz. La duración está en milisegundos.
PlayToneEx(frecuencia, duración, volumen, ciclo): Reproduce un tono de volumen previamente especificado. Cicla el valor boleano indicando cuando se repite el tono.
PlayFile(nombre de fila): Reproduce un archivo de sonido (.rso) o melodía (.rmd)
PlayFileEx(nombre del archivo, volumen, ciclo): Reproduce un archivo de sonido o melodía a un volumen específico. Cicla el valor boleano indicando cuando se repite el archivo.
GetSoundState(estado, marcas): Obtiene el estado actual del módulo de los audio .
SetSoundState(estado,marcos, resultado): Fija el estado actual del módulo de audio.

Prácticas Propuestas

A continuación se presentan prácticas para utilizar lo explicado anteriormente a fin de que el manejo y programación del kit LEGO^{MR} 9797 se domine para tener la capacidad de realizar diversas tareas y aplicaciones, que pueden surgir de un desafío o como solución a una problemática en particular.

Las primeras diez están enfocadas al uso del lenguaje gráfico NXT y las otras diez son para adentrarse en la esctructura de programación y control del robot mediante el lenguaje NXC.

Cabe mencionar que estas prácticas son ejemplos básicos los que permitirán adquirir habilidades. para posteriormente realizar construcciones, programas y robots mucho más complejos.

PRACTICA 1

Movimiento hacia adelante

Objetivo:

El alumno identificará la programación gráfica mediante NXT, al programar el mini-robot para que se desplace hacia adelante.

Teoría preliminar:

La programación mediante NXT se basa en LabVIEW y se presenta en un entorno gráfico, es requisito conocer los íconos y sus respectivas tareas así como las variables que cada uno presenta. Para una información más detallada puede consultarse el apartado respectivo a **Generalidades**.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir la estructura básica **TASKBOT.**
- En el NXT, debe programarse la secuencia mostrada a continuación.
- Se Baja / Carga el programa en el cerebro (brick) NXT.
- Se Inicia / Corre el programa, se deberá analizar el comportamiento del mini-robot.

Esquema gráfico del Programa:

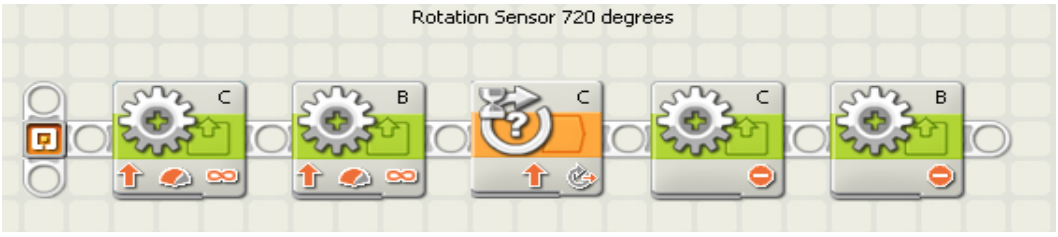


Fig. 1-1 Programa de movimiento hacia el frente.

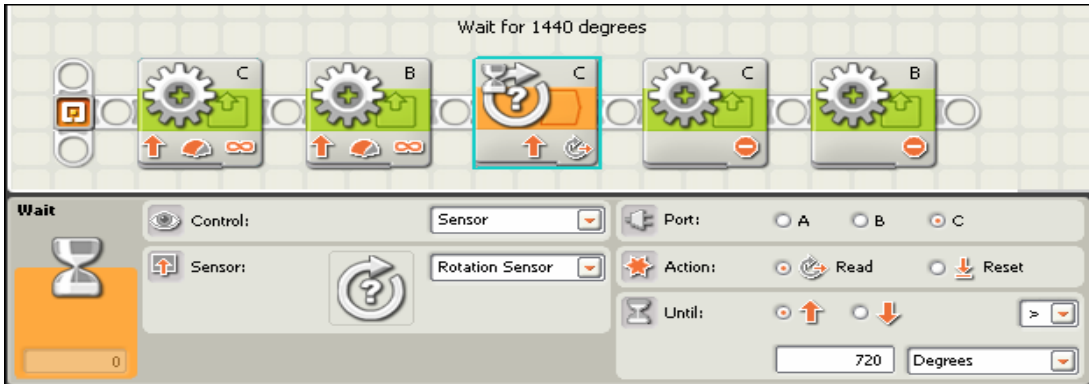


Fig. 1-2 Modificación de tiempos.

Problemas Propuestos:

- A continuación, se plantea modificar el programa de acuerdo a la figura 1-2. Pensando en el comportamiento que se obtiene, se debe modificar el programa para que avance el triple de la distancia que recorría originalmente en el primer programa.
- Ahora se propone modificar el programa para que se mueva hacia atrás en lugar de hacia adelante, pero solo la mitad de la distancia que recorría anteriormente.
- ¿Es posible generar un programa para que avance 20 cm, retroceda 10 cm y avance nuevamente 10 cm?
- ¿Qué aplicaciones tienen este tipo de funcionamientos?
- Generar un programa para un funcionamiento distinto, y entregar su justificación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

Fig 1-3 TASKBOT

PRACTICA 2

Ajuste de posición

Objetivo:

El alumno programará al mini-robot para que cambie de orientación, dependiendo de su posición original.

Teoría preliminar:

La orientación es una de las partes primordiales de la Robótica, la cual puede ser previamente programada o dependiente de alguna señal adquirida, en esta práctica se programará un cambio en la posición del robot.

En la Robótica movil es de suma importancia el poder determinar la hubicación final del robot lo que se consigue mediante sensores, y giros correcto de los motores.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir la estructura básica **TASKBOT.**
- En NXT, programar la secuencia mostrada a continuación.
- Bajar / Cargar el programa en el cerebro (brick) NXT.
- Iniciar / Correr el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

Fig. 2-1 Programa de movimiento para cambiar de orientación.

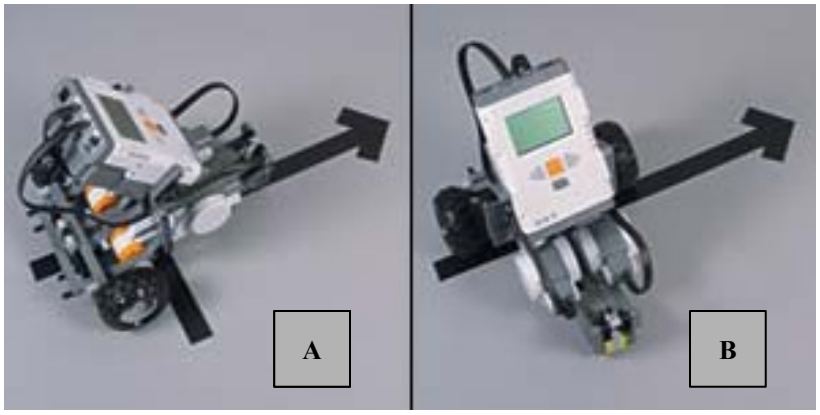


Fig. 2-2 Posición original (A) y Posición una vez que ha girado (B).

Problemas Propuestos:

- En la figura 2-2, se muestran las posiciones A y B:
- Modificar el programa indicado en la figura 2-1. Para que el robot gire desde A hacia B primero hacia la derecha. Y posteriormente para que gire de A hacia B girando hacia la izquierda.
- Ahora se deberá cambiar el programa para que haga los giros mientras avanza hacia atrás.
- ¿Es posible generar un programa para que al llegar al punto B, espere 10 segundos y gire nuevamente hacia el punto A?
- ¿Qué aplicación puede darse a este tipo de funcionamiento?.
- Generar un programa para un funcionamiento distinto y entregar su justificación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 3

Paro mediante sensor de tacto

Objetivo:

El alumno analizará el funcionamiento de los sensores de proximidad, detectando obstáculos mediante tacto, a la vez que propone aplicaciónes para los mismos.

Teoría preliminar:

Una de las partes fundamentales de la Robótica son los sensores ya que es mediante estos elementos que el sistema en contacto con el mundo exterior y permite al programador desarrollar respuestas a partir de las señales adquiridas. Para una información más detallada se recomienda consultar la parte correspondiente a sensores de la Guía y de la parte de Generalidades.

Equipo Requerido:

1 Kit LEGO 9797
2 sensores de tacto

Procedimiento:

- Construir **TASKBOT** básico.
- En el NXT, programar la secuencia mostrada acontinuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

El primer programa, es para probar el sensor de tacto.

Fig. 3-1 Programa para detener motores mediante sensor de tacto.

Problemas Propuestos:

- Debe observarse el comportamiento del robot ante el primer programa.
- Deberá analizarse el cambio en el funcionamiento cuando se modifica el programa de avance, para que se mueva hacia adelante hasta que encuentre algún obstáculo, con el cual haga contacto, una vez que lo detecte, debe detenerse, retroceder un poco y girar.
- Deberán adaptase al TASKBOT 2 sensores de tacto y generar un programa para que detecte tanto los obstáculos al frente como en la parte de atrás. Para esto se utilizarán dos puertos.
- Proponer una aplicación distinta a la anterior, generar el programa y entregar su justificación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 4

Movimiento y paro mediante sensor de sonido

Objetivo:

El alumno analizará el funcionamiento del sensor de sonido, a la vez que propone aplicaciónes para él mismo.

Teoría preliminar:

Una de las partes fundamentales de la Robótica son los sensores ya que es mediante estos elementos que el sistema en contacto con el mundo exterior y permite al programador desarrollar respuestas a partir de las señales adquiridas. Para una información más detallada se recomienda consultar la parte correspondiente a sensores de la Guía y de la parte de Generalidades.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de sonido

Procedimiento:

- Construir **TASKBOT** básico.
- En el NXT, programar la secuencia mostrada acontinuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

El primer programa, es para probar el sensor de sonido.

Fig. 4-1 Programa para detener motores mediante sensor de sonido.

El siguiente es el programa modificado:

Fig. 4-2 Programa para movimiento y paro mediante sensor de sonido.

Problemas Propuestos:

- Debe observarse el comportamiento del robot ante el primer programa.
- Deberá analizarse el cambio en el funcionamiento cuando se modifica el programa, explicar los cambios y como se indican mediante los íconos.
- Genera un programa que al detectar sonido el robot se detenga, retroceda un poco, gire 180 grados, y se mueva hacia adelante otra vez, en caso de escuchar el sonido nuevamente debe realizar el ciclo otra vez.
- Proponer una actividad distinta a la anterior, generar su programa y entregar su justificación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 5

Movimiento y paro mediante sensor de luz

Objetivo:

El alumno analizará el funcionamiento del sensor óptico, sus estados activo y pasivo, a la vez que propone aplicaciones para su uso.

Teoría preliminar:

Una de las partes fundamentales de la Robótica son los sensores ya que es mediante estos elementos que el sistema en contacto con el mundo exterior y permite al programador desarrollar respuestas a partir de las señales adquiridas. Para una información más detallada consulta la parte correspondiente a sensores.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de luz

Procedimiento:

- Construir **TASKBOT** básico.
- En el NXT, programar la secuencia mostrada acontinuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

El programa siguiente es para probar el sensor de luz, el cual puede ajustarse a trabajar en modo pasivo o activo.

Fig. 5-1 Programa para detener motores mediante sensor de luz (modo activo).

Fig. 5-2 Programa para detener motores mediante sensor de luz (modo pasivo).

Problemas Propuestos:

- Generar un programa que detecte un cambio de color en el piso, (puede ser de blanco a negro o cualquier color), en cuanto reciba la señal deberá retrocer un poco y girar. Así sucesivamente.
- Se debe trazar un cuadrado amplio con cinta y deja al mini-robot dentro funcionando, la idea es que no salga del margen establecido.
- Deberá proponerse un ejercicio distinto al anterior, generar su programa y entregar su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Elaborar una lista de los valores arrojados por el sensor para los colores básicos.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 6

Movimiento y paro mediante sensor ultrasónico

Objetivo:

El alumno analizará el funcionamiento del sensor ultrasónico, a la vez que propone aplicaciónes para los mismos.

Teoría preliminar:

Una de las partes fundamentales de la Robótica son los sensores ya que es mediante estos elementos que el sistema en contacto con el mundo exterior y permite al programador desarrollar respuestas a partir de las señales adquiridas. Para una información más detallada consulta la parte correspondiente a sensores.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de ultrasónico

Procedimiento:

- Construir **TASKBOT** básico.
- En el NXT, programar la secuencia mostrada acontinuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

El programa siguiente permite probar el sensor de ultrasónico, es posible ajustar la distancia del objeto a detectar en pulgadas o en centímetros.

Fig. 6-1 Programa para detener motores mediante sensor ultrasónico.

Problemas Propuestos:

- Generar un programa que al detectar un objeto a 30 cm el robot se detenga, retroceda un poco, gire 180 grados, y se mueva hacia adelante otra vez, en caso de escuchar sonido realizará el ciclo nuevamente.
- Construir un laberinto y programa la ruta para que el mini-robot lo atraviese sin hacer contacto con las paredes.
- Proponer un ejercicio distinto a los anteriores, generar su programa y entrega su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 7

Golpear la pelota de acuerdo al color detectado

Objetivo:

El alumno utilizará varios sensores usados anteriormente y controlará 3 motores a fin de que el Mini-Robot realice la tarea propuesta.

Teoría preliminar:

Para implementar más motores o más sensores solo debe tenerse en cuenta indicar en el programa el puerto en el que se encuentran conectados. Los sensores pueden colocarse en cualquier puerto no necesariamente el que se marca por defecto, siempre y cuando sea indicado correctamente en la programación.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de luz
1 sensor ultrasónico
2 pelotitas de distinto color

Procedimiento:

- Construir **TASKBOT** básico, y adaptar la paleta para golpear las pelotitas.
- Construir las bases para colocar pelotitas.
- En NXT, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

El programa se plantea de la siguiente forma, es conveniente recordar que es solo una propuesta, se debe generar un programa propio, con los ajustes que se crean convenientes, el objetivo deberá cumplirse una vez que se presente el Mini-Robot.

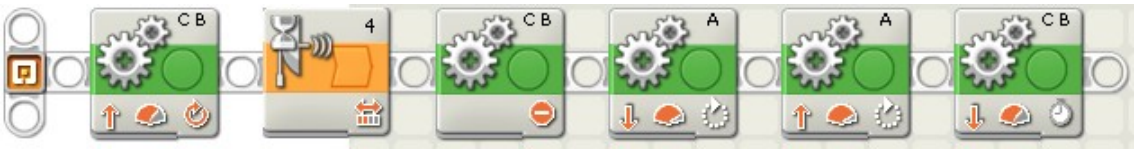


Fig. 7-1 Programa para detener motores mediante sensor ultrasónico, y golpear la pelota.

Problemas Propuestos:

- Generar un programa que sea capaz de detectar las bases donde se colocarán dos pelotitas de distinto color, el sensor de luz deberá leer el color de la pelota, si es roja deberá golpearla y girar 180 grados, si encuenta una pelota azul deberá ignorarla y girar 180 grados nuevamente.
- Deberá proponerse un ejercicio distinto al anterior, generar su programa y entregar su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 8

Seguidor de línea

Objetivo:

El alumno aplicará diversas estructuras de programación para controlar el movimiento del Mini-Robot para que siga la línea.

Teoría preliminar:

El realizar varias actividades mediante una secuencia es primordial para un excelente desempeño en la Robótica. Esto puede lograrse mediante el uso de ciclos y condiciones lógicas. La programación puede variar dependiendo de la trayectoria que se desea recorrer o la actividad que se quiere cumplir, la estructura principal la decide el propio programador.

Se presenta un programa con un sensor de luz, y se propone generar un programa que dependa de dos sensores de luz, lo que implica manejar dos puertos y la lectura de dos variables.

Los colores de las pistas son opcionales, es recomendable trazar pistas usando colores que contrasten con la base, debe también tenerse en cuenta la iluminación del lugar ya que influye en la lectura del sensor.

Equipo Requerido:

1 Kit LEGO 9797
1 o 2 sensores de luz

Procedimiento:

- Construir **TASKBOT** básico.
- En el NXT, programar la secuencia mostrada acontinuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

El programa se plantea de la siguiente forma, debe recordarse que es solo una propuesta, se debe generar un programa propio, con los ajustes que se crean convenientes, el objetivo deberá cumplirse una vez que se presente el Mini-Robot.

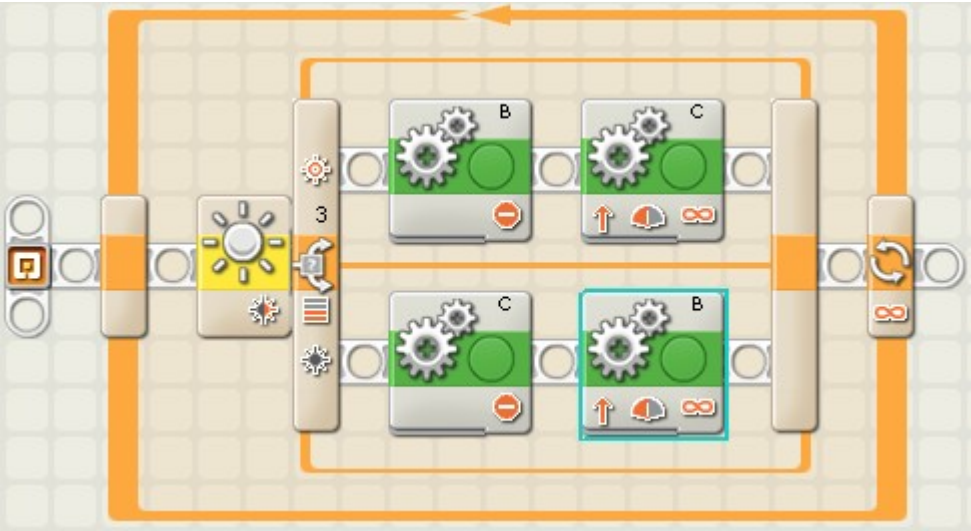


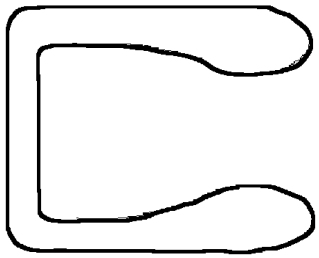
Fig. 8-1 Programa seguidor de línea.

Problemas Propuestos:

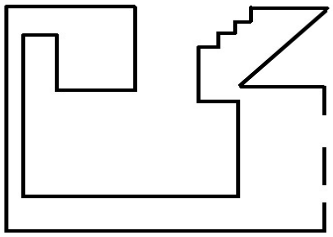
- Generar un programa que sea capaz de detectar la línea y seguir usando 2 sensores de luz.
- Modificar el programa para que el Mini-Robot sea capaz de seguir cualquiera de las pistas que se muestran a continuación.
- Deberá proponerse un ejercicio distinto al anterior, generar su programa y entregar su aplicación.

Producto:

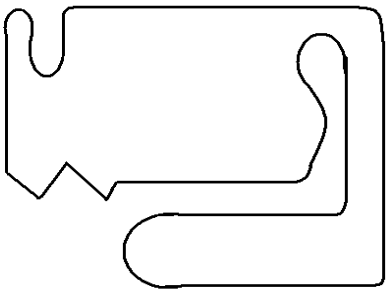
- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.



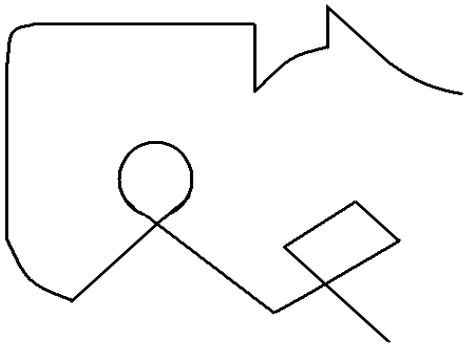
Pista ejemplo 1



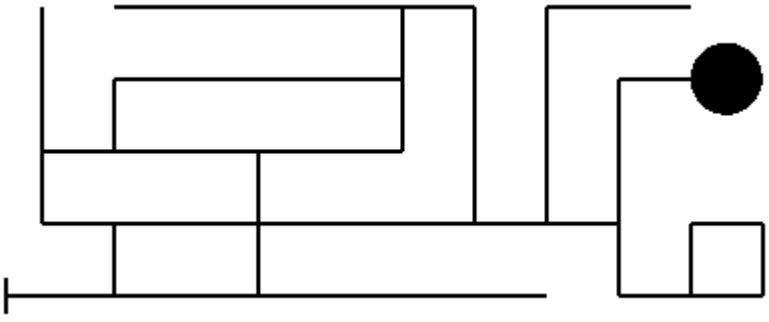
Pista ejemplo 2



Pista ejemplo 3



Pista ejemplo 4



Laberinto, la idea es que el robot siga la línea y llegue hasta un punto determinado.

PRACTICA 9

Grúa selectora fija

Objetivo:

El alumno generará tanto el programa como la estructura de una grúa fija capaz de separar objetos de por lo menos dos colores diferentes.

Equipo Requerido:

1 Kit LEGO 9797
1 o 2 sensores de luz
1 sensor ultrasónico o tacto (opcionales)

Procedimiento:

- Construir una grúa con diseño propio o siguiendo una estructura básica.
- En NXT, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

Podrán utilizarse los sensores que se crean convenientes tanto en número como en tipo. El programa para esta práctica es totalmente libre.

Problemas Propuestos:

- Generar un programa para que una grúa fija sea capaz de separar objetos de distinto color. Dos o tres colores diferentes como mínimo.
- La grúa deberá tomar de un contenedor un objeto y de acuerdo al su color separarlos en contenedores que estarán a su izquierda y derecha.
- Deberá proponerse un ejercicio distinto al anterior, generar su programa y entregar su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 10

Grúa selectora móvil

Objetivo:

El alumno generará tanto el programa como la estructura de una grúa móvil capaz de separar objetos de por lo menos dos colores diferentes y depositarlos en un lugar en específico.

Equipo Requerido:

1 Kit LEGO 9797
1 o 2 sensores de luz
1 sensor ultrasónico o tacto (opcionales)

Procedimiento:

- Construir una grúa con diseño propio o siguiendo una estructura básica.
- En NXT, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los programas propuestos que se solicitan.

Esquema gráfico del Programa:

Podrán utilizarse los sensores que se crean convenientes tanto en número como en tipo. El programa para esta práctica es totalmente libre .

Problemas Propuestos:

- Generar un programa para que una grúa móvil sea capaz de separar objetos de distinto color. Dos o tres como mínimo.
- Deberá tomar de un contenedor un objeto y de acuerdo al su color desplazarse hacia los contenedores correspondientes, los que podrán tener una marca del color o alguna otra seña para indicar donse se colocarán los objetos.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 11

Movimiento hacia adelante y atrás

Objetivo:

El alumno identificará la programación mediante NXC, al programar en lenguaje código el mini-robot para que se desplace hacia adelante y hacia atrás.

Teoría preliminar:

En la programación NXC se define la tarea a realizar mediante comandos o palabras definidas, también debe definirse quien realiza la acción en este caso los motores, los que se manejan como salidas OUT_A y OUT_B, si se requiere pueden escribirse en una sola instrucción tal como se hacia en los bloques NXT, OUT_AB. Deberá especificarse el tiempo de espera o el número de rotaciones, dependiendo la secuencia a realizar. Para más detalles de las estructuras y comandos básicos consulta la parte de Generalidades.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

task main()
{
  OnFwd(OUT_A, 50);
  OnFwd(OUT_B, 50);
  Wait(6000);
  Off(OUT_AB);
}
```

Segundo programa

```
#include "NXCDefs.h"

task main()
{
      OnFwd(OUT_A, 75);
      OnFwd(OUT_C, 75);
      Wait(4000);
      OnRev(OUT_AC, 75);
      Wait(4000);
      Off(OUT_AC);
}
```

En estos programas, la sentencia ***#include "NXCDefs.h"*** es necesaria para poder compilar el programa, si se cuenta con la versión 3.3.7.17, ya no será indispensable pero para versiones anteriores debe ponerse siempre al principio de cualquier programa, ya que es la librería que le da la extencion nxc.

Problemas Propuestos:

- A continuación, se deberá modificar el programa, para que avance el triple de la distancia que recorre el primer programa.
- Cambiar el programa para que se mueva hacia atrás en lugar de hacia adelante, pero solo la mitad de la distancia original.
- ¿Puede generarse un programa para que avance 20 cm, retroceda 10 cm y avance nuevamente 10 cm?
- Proponer una aplicación a este tipo de ejemplos.
- Plantear un ejercicio distinto, generar su programa y entregar su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los códigos en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 12

Ciclo repetitivo

Objetivo:

El alumno programará en lenguaje NXC ciclos que se repitan, para realizar funciones específicas tales como giros y trayectorias con el Mini-Robot.

Teoría preliminar:

Definir variables es fundamental en la programación de cualquier lenguaje código, el tener que aplicar ciertas operaciones en varias ocasiones, puede evitarse si se nombran los resultados, ya que pueden llamarse más adelante, permitiendo agilizar el proceso y la ubicación de las partes dentro del programa, además de que sirven como referencia para el programador.

El generar ciclos que pueden repetirse, permite llevar a cabo tareas mientras se incrementan las variables hasta obtener un valor deseado.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

#define MOVE_TIME 1000    // define la variable con nombre y valor
#define TURN_TIME 360


task main()
{
```

```
        OnFwd(OUT_AC, 75);
        Wait(move_time); // usa la variable para detenerse
        OnRev(OUT_C, 75);
        Wait(TURN_TIME);
        Off(OUT_AC);
}
```

Segundo programa

```
#include "NXCDefs.h"

#define MOVE_TIME 500     // define la variable con nombre y valor
#define TURN_TIME 500

    task main()
    {
        repeat(4)         // comienza un ciclo que se repetiá 4 veces
        {
        OnFwd(OUT_AC, 75);
        Wait(MOVE_TIME);
        OnRev(OUT_C, 75);
        Wait(TURN_TIME);
        }
    Off(OUT_AC);
    }
```

Problemas Propuestos:

- A continuación, se deberá modificar el primer programa. Para que el robot avance 20 cm y realice un giro de 180 grados hacia la izquierda, avance otros 20 cm y gire de 90 hacia la derecha.
- Modificar ahora el programa para que haga los giros mientras avanza hacia atrás.
- Generar un programa para que el Mini-Robot se mueva en un cuadrado perfecto, repitiendo el ciclo 5 veces.
- Plantear un ejercicio distinto, generar el programa y entregar su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los códigos en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad. mostrar el mini-robot funcionando.

PRACTICA 13

Haciendo espirales

Objetivo:

El alumno programará en lenguaje código NXC espirales que se repitan, y manejará variables aleatorias con el Mini-Robot.

Teoría preliminar:

Es común definir variables, pero ¿qué sucede cuando le pedimos al robot que genere valores aleatorios?, el programa se portará de una forma que no esperamos, o al menos que no podemos predecir.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

#define TURN_TIME 360

int move_time;  // define la variable de tipo entero
task main()
{
      move_time = 200;  // indica el valor inicial
      repeat(50)
      {
      OnFwd(OUT_AC, 75);
      Wait(move_time);  // usa la variable para detenerse
      OnRev(OUT_C, 75);
```

```
      Wait(TURN_TIME);
      move_time += 200; // incrementa la variable
      }
Off(OUT_AC);
}
```

Segundo programa

```
int move_time, turn_time;
task main()
{
      while(true)
      {
      move_time = Random(600);
      turn_time = Random(400);
      OnFwd(OUT_AC, 75);
      Wait(move_time);
      OnRev(OUT_A, 75);
      Wait(turn_time);
      }
}
```

Problemas Propuestos:

- Debe analizarse el segundo programa y aplicar este tipo de comportamiento al primer programa.
- Proponer una aplicación a este tipo de ejercicios, y entregar su justificación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los códigos en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 14

Control y Lógica

Objetivo:

El alumno programará activicades utilizando funciones lógicas en lenguaje código NXC que se repitan, además de que manejará variables aleatorias con el Mini-Robot.

Teoría preliminar:

Las secuencias pueden programarse mediante ciclos y funciones lógicas, aquí se presenta unas de las principales, las condiciones (if-else), (while), (do) las cuales permiten desarrollar actividades cada vez más complejas.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código del Programa:

Primer programa

```
#define MOVE_TIME 500
#define TURN_TIME 360

     task main()
     {

          while(true)
          {
               OnFwd(OUT_AC, 75);
               Wait(MOVE_TIME);
               if (Random() >= 0)
               {
                    OnRev(OUT_C, 75);
```

```
                }
                else
                {
                      OnRev(OUT_A, 75);
                }
                Wait(TURN_TIME);
          }
     }
```

Segundo programa

```
int move_time, turn_time, total_time;
     task main()
     {
          total_time = 0;
          do
     {
          move_time = Random(1000);
          turn_time = Random(1000);
          OnFwd(OUT_AC, 75);
          Wait(move_time);
          OnRev(OUT_C, 75);
          Wait(turn_time);
          total_time += move_time;
          total_time += turn_time;
          }
     while (total_time < 20000);
     Off(OUT_AC);
}
```

Problemas Propuestos:

- Deberán analizarse estos programas y aplicar este tipo de estructura de código a programas generados anteriormente en NXT, por lo menos a dos de ellos.
- Proponer una aplicación a este tipo de ejemplos, y entregar su aplicación.

Producto:

- Debe mostrarse el mini-robot funcionando.
- Entregar los códigos en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 15

Sensor de Tacto

Objetivo:

El alumno programará en lenguaje código NXC el uso del sensor de tacto para el Mini-Robot.

Teoría preliminar:

Para el lenguaje código NXC se deben indicar los puertos donde están conectados los sensores, se especifican como IN_1, IN_2, IN_3 y finalmente IN_4. En este caso el sensor de tacto estará conectado en el puerto 1.

Equipo Requerido:

1 Kit LEGO 9797

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

task main()
{
      SetSensor(IN_1,SENSOR_TOUCH);
      OnFwd(OUT_AC, 75);
      until (SENSOR_1 == 1);
      Off(OUT_AC);
}
```

Segundo programa

```
#include "NXCDefs.h"

task main()
{
      SetSensorTouch(IN_1);
      OnFwd(OUT_AC, 75);
      while (true)
            {
            if (SENSOR_1 == 1)
            {
            OnRev(OUT_AC, 75); Wait(300);
            OnFwd(OUT_A, 75); Wait(300);
            OnFwd(OUT_AC, 75);
            }
      }
}
```

Problemas Propuestos:

- Analizar estos programas y aplicar este tipo de estructuras de código a las actividades realizadas previamente con el sensor de tacto programado para NXT, por lo menos a dos de ellos.
- Proponer una aplicación a este tipo de ejemplos, entregar la justificación.

Producto:

- Deberá mostrarse el mini-robot funcionando.
- Generar el código en NXC de los programas realizados anteriormente para NXT.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 16

Sensor de Sonido

Objetivo:

El alumno programará en lenguaje código NXC el uso del sensor de sonido para el Mini-Robot.

Teoría preliminar:

Para el lenguaje código NXC se deben indicar los puertos donde están conectados los sensores, se especifican como IN_1, IN_2, IN_3 y finalmente IN_4. En este caso el sensor de sonido estará conectado en el puerto 2.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de sonido

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

#define AUDIO 40
#define MIC SENSOR_2
task main()
{
     SetSensorSound(IN_2);
     while(true){
     until(MIC > AUDIO);
     OnFwd(OUT_AC, 75);
```

```
      Wait(300);
      until(MIC > AUDIO);
      Off(OUT_AC);
      Wait(300);
      }
}
```

Problemas propuestos:

- Analizar este programas y aplicar este tipo de estructura de código a las actividades realizadas previamente con el sensor de sonido para NXT, por lo menos a dos de ellos.
- Proponer una aplicación a este tipo de ejemplo, y entregar la justificación.

Producto:

- Deberá mostrarse el mini-robot funcionando.
- Generar el código de los programas en NXC, realizados anteriormente con NXT.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 17

Sensor Ultrasónico

Objetivo:

El alumno programará en lenguaje código NXC el uso del sensor ultrasónico para el Mini-Robot.

Teoría preliminar:

Para el lenguaje código NXC se deben indicar los puertos donde están conectados los sensores, se especifican como IN_1, IN_2, IN_3 y finalmente IN_4. En este caso el sensor ultrasónico estará conectado en el puerto 4.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor ultrasónico

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

#define CERCA 15 //cm

task main(){
      SetSensorLowspeed(IN_4);
      while(true){
      OnFwd(OUT_AC,50);
      while(SensorUS(IN_4)>CERCA);
      Off(OUT_AC);
      OnRev(OUT_C,100);
```

```
      Wait(800);
      }
}
```

Problemas Propuestos:

- Analizar estos programas y aplicar este tipo de estructuras de código a las actividades realizadas previamente para NXT con el sensor ultrasónico, por lo menos a dos de ellos.
- Proponer una aplicación a este tipo de ejemplos, y entregar la justificación.

Producto:

- Deberá mostrarse el mini-robot funcionando.
- Generar el código en NXC de los programas realizados anteriormente con NXT.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 18

Sensor de Luz

Objetivo:

El alumno programará en lenguaje código NXC el uso del sensor de luz para el Mini-Robot.

Teoría preliminar:

Para el lenguaje código NXC se deben indicar los puertos donde están conectados los sensores, se especifican como IN_1, IN_2, IN_3 y finalmente IN_4. En este caso el sensor de luz estará conectado en el puerto 3.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de Luz

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan..

Código del Programa:

Primer programa

```
#include "NXCDefs.h"

#define INTENSIDAD 40

task main()
{
     SetSensorLight(IN_3);
     OnFwd(OUT_AC, 75);
     while (true)
     {
     if (Sensor(IN_3) > INTENSIDAD)
```

```
            {
            OnRev(OUT_C, 75);
            Wait(100);
            until(Sensor(IN_3) <= INTENSIDAD);
            OnFwd(OUT_AC, 75);
            }
      }
}
```

Para cambiar el modo del sensor de luz de pasivo a activo y viceversa

```
SetSensorType(IN_3,IN_TYPE_LIGHT_INACTIVE);
SetSensorMode(IN_3,IN_MODE_PCTFULLSCALE);
ResetSensor(IN_3);
```

Problemas Propuestos:

- Analizar este programa y aplicar este tipo de estructura de código a las actividades realizadas previamente para NXT con el sensor de luz, por lo menos a dos de ellos.
- Proponer una aplicación a este tipo de ejemplo, y entregar la justificación.

Producto:

- Deberá mostrarse el mini-robot funcionando.
- Generar el código para NXC de los programas realizados anteriormente con NXT.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 19

Rutinas y Subrutinas

Objetivo:

El alumno aplicará diversas estructuras de programación (Rutinas y Subrutinas) en código NXC para controlar el movimiento del Mini-Robot.

Teoría preliminar:

Tareas-Rutinas: En la programación NXC se pueden encontrar hasta 255 tareas (rutinas) diferentes, cada una tiene un nombre único. La Tarea ***main*** debe estar siempre al principio, el resto se ejecutarán solamente cuando se indique, la tarea principal deberá concluir antes que las demás comiencen.

Subrutinas: Algunas veces se requiere una parte del código varias veces dentro de un programa, en este caso se coloca una subrutina y se le da un nombre. Se podrá ejecutar esta parte de código con solo llamarla dentro de una rutina.

Equipo Requerido:

1 Kit LEGO 9797
1 sensor de tacto

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código NXC:

Primer Programa

```
     #include "NXCDefs.h"
mutex moveMutex;
task move_square()
{
     while (true)
```

```
      {
      Acquire(moveMutex);
      OnFwd(OUT_AC, 75); Wait(1000);
      OnRev(OUT_C, 75); Wait(500);
      Release(moveMutex);
      }
}

task check_sensors()
{
      while (true)
      {
      if (SENSOR_1 == 1)
      {
      Acquire(moveMutex);
      OnRev(OUT_AC, 75); Wait(500);
      OnFwd(OUT_A, 75); Wait(500);
      Release(moveMutex);
      }
      }
}

task main()
{
      Precedes(move_square, check_sensors);
      SetSensorTouch(IN_1);
}
```

Segundo Programa

```
                  #include "NXCDefs.h"

sub turn_around(int pwr)
{
      OnRev(OUT_C, pwr); Wait(900);
      OnFwd(OUT_AC, pwr);
}

task main()
{
      OnFwd(OUT_AC, 75);
      Wait(1000);
      turn_around(75);
      Wait(2000);
      turn_around(75);
      Wait(1000);
```

```
    turn_around(75);
    Off(OUT_AC);
}
```

Problemas Propuestos:

- Generar un programa que utilice al menos 2 subrutinas, puede ser usando cualquiera de los sensores.
- Modificar el programa para que el Mini-Robot sea capaz de usar 3 subrutinas.
- Proponer una aplicación industrial para este tipo de sistema y entrega su justificación.

Producto:

- Deberá mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.

PRACTICA 20

Seguidor de línea

Objetivo:

El alumno aplicará diversas estructuras de programación en código NXC para controlar el movimiento del Mini-Robot para que siga la línea.

Equipo Requerido:

1 Kit LEGO 9797
1 o 2 sensores de luz

Procedimiento:

- Construir **TASKBOT** básico.
- En el BricxCC, programar la secuencia mostrada a continuación.
- Cargar el programa en el cerebro NXT.
- Ejecutar el programa y analizar su comportamiento.
- Realizar los problemas propuestos que se solicitan.

Código NXC:

```
#include "NXCDefs.h"


#define SpeedSlow 50
#define SpeedFast 100

int SV;
int LoopCount;

sub FollowLine(int loopTime)
{
  int Threshold1=600;
  int Threshold2=630;
  int theSpeed;
  long t;

  // ajusta el tipo y modo del sensor
  SetSensorType(IN_3, IN_TYPE_LIGHT_ACTIVE);
```

```
  SetSensorMode(IN_3, IN_MODE_RAW);


  // comienza el ciclo
      t = CurrentTick() + loopTime;
      while (t > CurrentTick())
  {
    // toma la lectura del sensor de luz
    SV = SensorRaw(IN_3);

   // ajusta la velocidad del motor 1
    if (SV < Threshold2)
      OnFwd(OUT_A, SpeedFast);
    else
      OnFwd(OUT_A, SpeedSlow);

  // ajusta la velocidad del motor 2
    if (SV > Threshold1)
      OnFwd(OUT_B, SpeedFast);
    else
      OnFwd(OUT_B, SpeedSlow);

  // muestra el valor del sensor
//    NumOut(0, LCD_LINE1, false, SV);

    LoopCount++;
  }
 // el ciclo de 10 segundos está hecho
  return;
}

task main()
{
  string lcStr;
  string svStr;
  string msg;

 // llama a la subroutina
  FollowLine(10000);

 // output results
 lcStr = NumToStr(LoopCount);
 svStr = NumToStr(SV);
  msg = svStr + " - " + lcStr;
     TextOut (0, LCD_LINE1, msg);

 // detener ambos motores
```

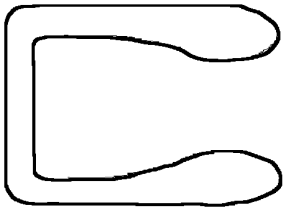
```
  Off(OUT_AB);

 // permite al usuario ver el último mensaje
  Wait(10000);
}
```

Problemas Propuestos:

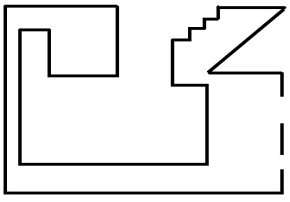
- Generar un programa que sea capaz de detectar la línea usando 2 sensores.
- Modificar el programa para que el Mini-Robot sea capaz de seguir cualquiera de las pistas que se muestran a continuación.
- Proponer una aplicación industrial para este tipo de sistema y entrega su justificación.

Producto:

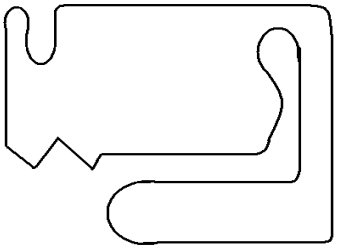
- Deberá mostrarse el mini-robot funcionando.
- Entregar los esquemas en archivo a más tardar la próxima sesión y conservarlos para impresión en el Trabajo Integrador al final de la Unidad.



Pista ejemplo 1



Pista ejemplo 2



Pista ejemplo 3

Para un siguiente nivel de capacidad en el manejo del sistema NXT se proponen Desafíos tales como utilizar el sistema de sonidos para crear melodías, de dibujo, comunicación mediante BlueTooth, y otras formas de controlar y sincronizar motores, para los usuarios expertos.

Referencias

Para mayor información respecto a los temas aquí mencionados:

Using NXT Introduction to Robotics
(Robotics Academy 2006) by Carnegie Mellon

Programming LEGO NXT Robots using NXC (beta 27 or higher)
(Version 2.1, Apr 9, 2007) by Daniele Benedettelli

Not eXactly C (NXC) Programmer's Guide
Version 1.0.1 b25 by John Hansen

www.legomindstorms.com

Para temas avanzados y diversas construcciones podrás encontrar varios sitios y foros dedicados a proyectos de Mini-Robótica.

Con la colaboración de:

Jose Alberto Martínez Torres

Fidencio Costilla Hermosillo

Adrián Ortíz Cano

Moisés Martínez Márquez