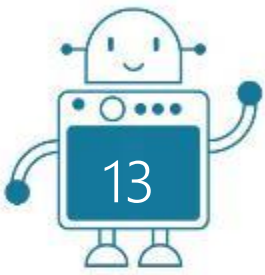


MOVIMIENTO ARMÓNICO CON ACELERÓMETRO

STEMJAM Teaching Guide

Developing make spaces to promote creativity around STEM in schools

Acronym: STEMJAM
Project no. 2016-1-ES01-KA201-025470

www.stemjam.eu

Co-funded by the Erasmus+ Programme of the European Union

MOVIMIENTO ARMÓNICO CON ACELERÓMETRO

RESUMEN

Esta actividad amplía la anterior sobre "Movimiento armónico" con el objetivo de ofrecer una observación directa de las variables cinemáticas de interés, como la aceleración. Las observaciones similares son útiles para los estudiantes mayores (15-16 años) para la comprensión de las ecuaciones de movimiento y sus soluciones. Para los alumnos más jóvenes (12-14 años), puede ser simplemente divertido observar el movimiento armónico del mBot en sí, como una medida complementaria con respecto a experimentos más estándar.

Para la plena explotación del giroscopio, también desarrollamos una breve guía sobre las mediciones de los ángulos de inclinación y de balanceo. Puede resultarle útil en las prácticas de física, p.ej. para medir la pendiente de un plano inclinado y, por supuesto, para ayudar a comprender el principio de control de navegación.

OBJETIVOS DIDÁCTICOS

Mientras realizas la actividad podrás …

- Física: explorar el movimiento armónico, identificar sus principales parámetros y ecuaciones.
- Física: observar directamente las oscilaciones en función del tiempo en los valores de aceleración.
- Tecnología: aprender sobre un giroscopio. El principio del control de navegación: cabeceo, balanceo y rumbo.

Mientras implementas el código aprenderás sobre ...

- Informática: desarrollo de algoritmos
- Matemáticas (avanzado, para alumnos de 17-18 años): derivadas numéricas.

Materias STEM: Ciencia☐ Tecnología ☒ Ingeniería☐ Matemáticas☒

Nivel Educativo: 12-14 años☒ 14-16 años☒

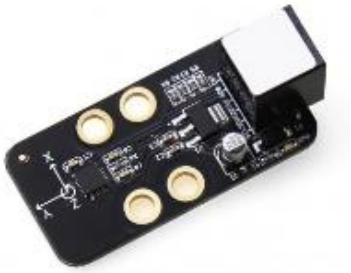
PLANTEAMIENTO DEL PROBLEMA

Cuelgue el mBot de un resorte y mida la aceleración y el período de su movimiento armónico, gracias a un acelerómetro de 3 ejes y un sensor de giro. Utilice el giroscopio con el propósito de controlar la navegación, aprendiendo sobre el ritmo y los ángulos de giro y rumbo.

LISTA DE MATERIALES NECESARIOS

- mBot **RANGER**:

- **Acelerómetro de 3 ejes y Sensor de giro**:



- **Matriz de LED 8 × 16** y/o **pantalla TFT LCD:**

- **Potenciometro**:

DESCRIPCIÓN DE LA ACTIVIDAD

El sensor del acelerómetro-giroscopio se utiliza para estudiar el movimiento armónico del mBot conectado a un resorte, así como para realizar mediciones de los ángulos de inclinación y balanceo.

Primera version

Inicialmente trabajamos las medidas de aceleración: enumeramos los pasos simples necesarios para realizar la actividad, analizamos las características del sensor de giro, ilustramos el algoritmo y el código y finalmente mostramos los resultados principales. Luego, en la segunda parte del documento, nos centramos en la aplicación estándar del giroscopio a la navegación (pitch and roll). Los ejemplos y las imágenes ayudarán a resaltar por qué aprender estos temas con los resultados de mBot es atractivo, simple y, por lo tanto, más efectivo.

A través del texto, los consejos útiles que conducen a mejores resultados son mencionados y resaltados en color naranja.

El mBot está adaptado para poder ser colgado de un resorte y realizar un movimiento armónico. A partir de las mediciones del giroscopio, proporcionales a la aceleración, se evalúa el período de oscilación. Ambas versiones mBlock (Scratch) y Arduino del código están disponibles. En Arduino, los datos del sensor se muestran en la salida en serie y se pueden trazar en un software externo para obtener una comprensión más profunda de las ecuaciones de movimiento.

Procedimiento experimental

1. Prepare el mBot con el acelerómetro / sensor de giro y la pantalla blanca. A pesar de que el sensor probablemente está diseñado para montarse debajo del chasis, al igual que el sensor de seguimiento de línea, aquí es conveniente montar el sensor en posición vertical en un lado del robot (en el ejemplo propuesto, el sensor se montó en el lado derecho del mbot), con las conexiones hacia arriba, con el eje x horizontal y apuntando hacia atrás, el eje y hacia abajo, el eje z horizontal y apuntando lejos del mBot). La razón de esta elección se detallará a continuación.

2. Cuelgue un resorte de un soporte adecuado y coloque el mBot en él (puede insertar el resorte en el orificio en forma de m en la parte posterior del chasis o usar un cordel. Por razones de seguridad, asegúrese de que el soporte sea estable y pesado. Tenga en cuenta que el mBot es pesado y si el resorte tiene una constante elástica baja, como se esperaba, el resorte se estirará considerablemente, en nuestro caso 1,5 m).

3. Encienda el mbot, precargado con el código, y comience un movimiento oscilatorio.

4. El período de oscilación se muestra en la pantalla de LED y la constante de resorte puede eventualmente evaluarse. Si el programa se ejecuta desde la interfaz estándar de Arduino con un cable usb, los datos de salida del sensor de tres giroscopios se mostrarán en la salida estándar (monitor de serie).

5. Discuta los resultados con los estudiantes (orientación y sugerencias a continuación).

(Mapa de conexión: 1. Pantalla LED 3. Sensor de giro. Consulte la hoja de electrónica para obtener más información)

El acelerómetro y el giroscopio de 3 ejes.

Este sensor, basado en el chip MPU6050, integra medidores de aceleración y velocidad angular y devuelve información sobre la orientación y el movimiento del robot. Aquí se ofrece una introducción al sensor [1]. Para conectar el sensor a Scratch y Arduino / C ++, la compañía Makeblock ofrece una biblioteca de programación basada en la clase de Gyro documentada aquí [2]. En realidad, esta biblioteca limita el potencial del MPU6050, en el sentido de que da acceso a algunas variables. Es decir, las tres salidas principales son (aparentemente) tres ángulos, el eje X, el eje Y y los ángulos del eje Z. La relación de estos resultados con las variables físicas de interés no está clara, ni en la documentación, ni en el foro de makeblock.

Resumimos aquí toda la información encontrada:

1. En el foro, el soporte técnico menciona que las 3 variables son los ángulos de Euler que definen la orientación del sensor con respecto a su posición inicial (se establece cuando se invoca la función gyro.begin ()). En muchos campos de aplicación, incluida la aeronáutica, estos ángulos también se conocen como inclinación, balanceo y rumbo (o guiñada): son los ángulos estándar dados por un giroscopio y se utilizan para fines de navegación. El sensor está diseñado esencialmente para la navegación robot.

2. Las tres salidas son efectivamente ángulos: los ángulos de los ejes x e y varían de -90 a +90, mientras que el ángulo del eje z varía de 0 a +360. El valor en el eje z también se desplaza de manera consistente y lineal con el tiempo (ver fig.3). Este es realmente un problema común en los giroscopios electrónicos y es por eso que sugerimos montar el sensor verticalmente para este experimento, de modo que el eje z permanezca horizontal durante el movimiento armónico y observemos las oscilaciones a lo largo de los ejes x e y, sin esta distorsión.

3. Sin embargo, la información en 1 es incorrecta. Las 3 salidas no son simplemente los ángulos de Euler que determinan la orientación, sino que cada una es proporcional a la componente relacionada del vector a - g, donde a es la aceleración del robot y g es la aceleración de la gravedad (por ejemplo, el "ángulo del eje x" es proporcional a la componente x de la aceleración ag). Cuando el robot está en reposo en una mesa, con el eje y hacia abajo, el ángulo del eje y es casi -90 °, detectando la gravedad (1 g corresponde a 90 grados) y cerca de cero en los otros ejes.

Flujo de trabajo del algoritmo y comentarios al código

El algoritmo se representa en el flujo de trabajo a continuación. Esencialmente, la señal, proporcional a la aceleración, se mide a lo largo de los 3 ejes y los máximos y mínimos de las oscilaciones se ubican por derivación. Se utiliza una derivada numérica de cinco puntos para el propósito [3]. Si no está familiarizado

con esta técnica, considere que se basa en la idea de tener en cuenta los valores de la función en algunos puntos antes y después del punto de interés, como se muestra en la siguiente figura.
La derivada en el punto n se evalúa como una combinación lineal de los valores de la función en los puntos vecinos, con coeficientes adecuados, de acuerdo con la fórmula.

$$f_n' = \frac{1}{12} f_{n-2} - \frac{8}{12} f_{n-1} + \frac{8}{12} f_{n+1} - \frac{1}{12} f_{n+2}$$

Para mayor explicación y derivación de esta fórmula, vea la referencia [3].

Cuando el signo de la derivada cambia, se produce un máximo o un mínimo, y el período puede evaluarse como el doble del tiempo entre dos de estos puntos críticos.

Un comentario técnico: En realidad, este algoritmo detecta el período con un error que debemos tener en cuenta: debido a que la derivada se hace como una diferencia finita de 5 puntos, los extremos (mínimo o máximo) se pueden detectar solo dos puntos después de que ocurran, lo que causa un retraso en el inicio y parada del temporizador. Este retraso siempre es ligeramente diferente debido a la frecuencia de muestreo finita, aquí elegida para ser f = 10 Hz. El error máximo que se puede cometer es menor que el tiempo de muestreo dt = 100 ms. Dado que el período medido es del orden de dos segundos, está dentro del 5% y podría considerarse aceptable y comparable a otras fuentes de error.

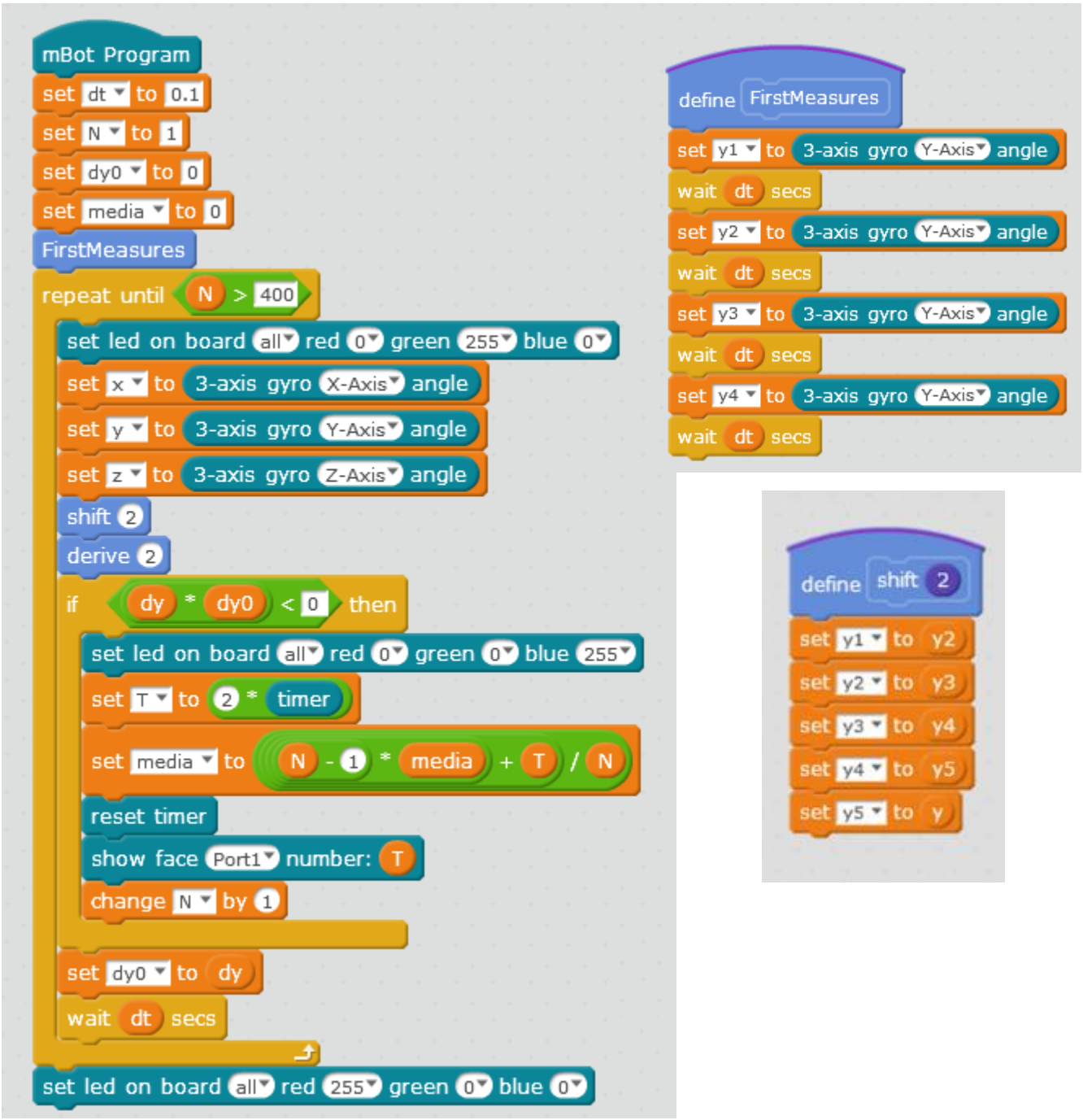
Código Arduino

```
//Librerías.   MeMCore es la biblioteca principal para controlar mBot y otros productos de Makeblock
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeMCore.h>

// Declaración de variables globales. Se ofrecen clases especiales para control de motores y sensores y otros componentes mBot.
En particular. MeRGBLed para el led de a bordo, MeGyro para el sensor de giro, MeLEDMatrix para la pantalla de led blanco
MeDCMotor motor_9(9);
MeDCMotor motor_10(10);
double x, y, z;  double A, B;
double T, t, t0;
double x1,x2,x3,x4,x5;      double y1,y2,y3,y4,y5;
double dt, dy, dy0;
double N, media;
MeRGBLed rgbled_7(7, 7==7?2:4);
```

```
MeGyro gyro;
MeLEDMatrix ledMtx_1(1);


void setup(){                       // Comandos que se ejecutan una vez al inicio del programa
  Serial.begin(9600);               // Establecer conexión serie / usb
  gyro.begin();                     //Inicializar el sensor gyro
  ledMtx_1.setColorIndex(1);        // Configuración de pantalla
  ledMtx_1.setBrightness(6);
  Serial.println("Data from mBot Gyro sensor");    // Encabezado para salida estándar.
  Serial.println(" xAngle  yAngle  zAngle ");
  N=1;
  dy0=0;
  FirstMeasures();                  // Consigue los primeros cuatro puntos (sin hacer nada más).
                                    // Por qué usamos una derivada de cinco puntos,
                                    // Tenemos que ingresar al bucle con los primeros 4 puntos ya disponibles.
}

void loop(){                        // Los comandos se ejecutan repetidamente hasta que el programa termina
  // Los leds a bordo se mantienen verdes mientras se ejecuta
  rgbled_7.setColor(0,0,255,0);
  rgbled_7.show();

  // Lea las salidas del sensor de giro y los valores de impresión en salida estándar
  // para visualización de datos y uso futuro
  x = gyro.getAngle(1);   y = gyro.getAngle(2);   z = gyro.getAngle(3);
  Serial.print(x);  Serial.print("  ");   Serial.print(y);   Serial.print("  ");   Serial.println(z);

  // Reorganizar los últimos 4 datos anteriores y el nuevo. Vea la definición de la función a continuación.
  shift(2);

  // Evaluar numéricamente la derivada en el punto n. Vea la definición de la función a continuación.
  dy=derive(2);

  // si se encuentra un mínimo o un máximo, el período se evalúa y se reinicia el temporizador.
  // Los leds de a bordo se ponen en azul durante un rato.
  if (dy*dy0<0){
    rgbled_7.setColor(0,0,0,255);
    rgbled_7.show();
    t = millis()/1000.0 - t0;       // Recuperar el valor del temporizador
    T=2*t;
    media=((N-1)*media+T)/N;        // promedio en los periodos medidos hasta ahora
```

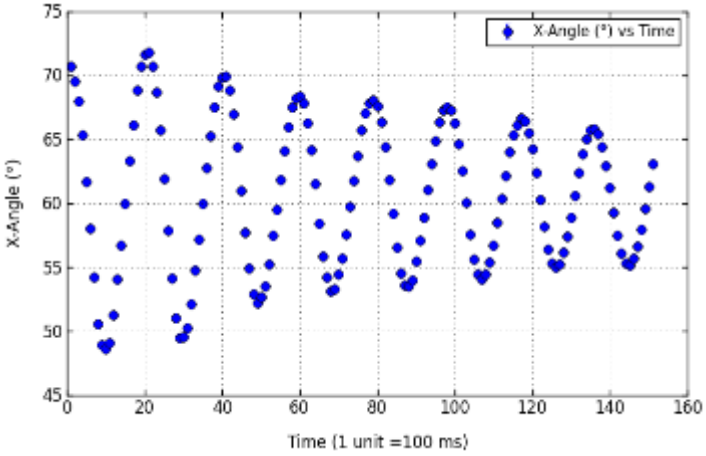
```
    t0 = millis()/1000.0;              //reinicio del temporizador
    N=N+1;                             // número de medias oscilaciones
    }

   dy0=dy;                             //almacenar la derivada actual para compararla con la siguiente.
   ledMtx_1.showNum(media,3);          // mostrar el período promedio en la pantalla de a bordo
   delay(100);                         //TIEMPO DE MUESTREO = 100 ms
   gyro.update();                      // recuperar datos actualizados del sensor gyro
}

// De los cinco puntos actuales de la función, descarte el anterior, desplace los otros y agregue uno nuevo (= lectura actual del sensor) en la posición 5
void shift(int i){
 if (i==1) {
   x1=x2; x2=x3; x3=x4; x4=x5; x5=x;   }

 else if (i==2) {
   y1=y2; y2=y3; y3=y4; y4=y5; y5=y;  }

 else {Serial.println("Possible input for the shift function are 1 and 2 only, corresponding to the x and y axes");}
}

// Evalúe numéricamente la derivada en el punto n de acuerdo con la fórmula dada en el texto
double derive(int i){
 double der;
 if (i==1) {
   der=(x1-8*x2+8*x4-x5)/12; }//cinco puntos de diferencias finitas

 else if (i==2) {
   der=(y1-8*y2+8*y4-y5)/12; }//cinco puntos de diferencias finitas

 else {Serial.println("Possible input for the shift function are 1 and 2 only, corresponding to the x and y axes");}
 return der;
}

// Lee los primeros puntos
void FirstMeasures(){
   gyro.update();   y1=gyro.getAngle(2);   delay(100);     //TIEMPO DE MUESTREO = 100 ms
   gyro.update();   y2=gyro.getAngle(2);   delay(100);
   gyro.update();   y3=gyro.getAngle(2);   delay(100);
   gyro.update();   y4=gyro.getAngle(2);   delay(100);   gyro.update();
}
```

Código mBlock scratch

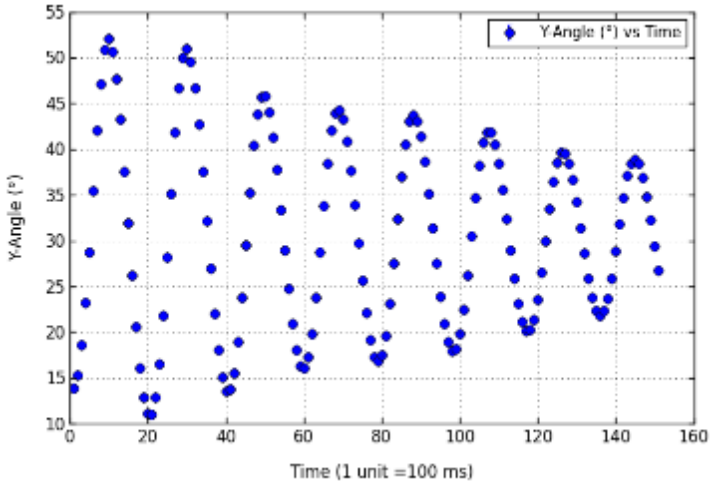


Resultados Experimentales

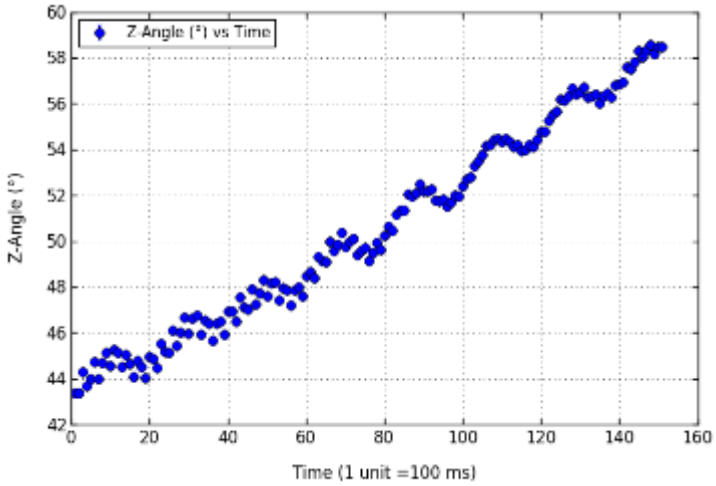
Los datos que se muestran en la figura 3 se obtuvieron de la salida estándar de Arduino y se representaron en un software externo. La oscilación armónica es claramente visible en el canal de los ejes x e y.
Como es evidente, en esta escala de tiempo (tiempo de muestreo 100 ms), el sensor genera datos de buena calidad con un nivel de ruido muy bajo.



En este experimento, el período estimado es de aproximadamente 2 segundos, pero se reduce lentamente con el tiempo a 1.75 segundos.



Un análisis similar es de interés para estudiantes de 15 años o más que estudian física: las oscilaciones de la aceleración con el tiempo se pueden apreciar fácilmente y ofrecen información útil sobre la ecuación de movimiento. Como desarrollo adicional, se podría agregar al programa la medición directa del desplazamiento de mBot a través del sensor ultrasónico.



En el canal del eje z se superpone una deriva de sensor considerable y lineal a las oscilaciones y, por lo tanto, el canal no es el más conveniente para el análisis de datos u otras observaciones. Por esta razón, sugerimos montar el sensor en posición vertical (o en cualquier posición que permita que el eje z sea perpendicular al movimiento armónico).

En general, el "Sensor de aceleración y giroscopio de 3 ejes" demostró ser un sensor de buena calidad, aunque no es tan fácil de usar debido a la documentación incompleta..

Segunda versión

Ángulo de balanceo y movimiento oscilante

Para llevar a cabo la actividad, y para poder calcular el movimiento oscilante al que se puede someter el mBot, necesitamos saber cuál es el ángulo de balanceo. Se define como el ángulo de inclinación lateral que forma la vertical del plano perpendicular del plano del vehículo. Cuando el mBot está en un plano horizontal, el ángulo de inclinación será aproximadamente 0, ya que tiene un pequeño margen de error.

Cuando el mBot está en un plano inclinado, el ángulo de inclinación será igual al de la superficie. Además, el Ranger mostrará gráficamente en la Matriz de LED su inclinación lateral, es decir, el ángulo que presenta con respecto al eje longitudinal (Y). Si este ángulo excede el valor establecido por el potenciómetro, lo advertirá por medio de una señal acústica y luminosa.

Dado que la placa Auriga que incorpora el Ranger tiene un módulo de medición inercial (IMU) integrado con un giroscopio y un acelerómetro de 3 ejes, es suficiente utilizar el valor del acelerómetro correspondiente al eje longitudinal del Ranger (eje Y) para obtener El ángulo lateral de inclinación directamente.

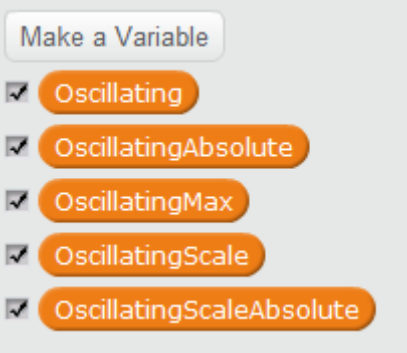
Además, en la pantalla TFT LCD mostraremos el ángulo del plano.

A continuación enunciamos los pasos necesarios para llevar a cabo la actividad completa:

a. En primer lugar, conectamos la matriz de Led y la pantalla TFT LCD al mBot Ranger.

b. También conectamos el sensor del potenciómetro.

c. Y luego, comenzamos a programar el mBot Ranger:

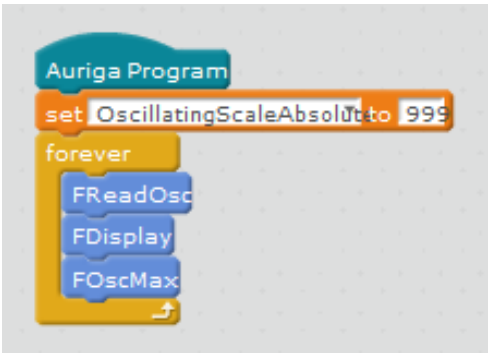


The most significant variables are:

- Oscillating: Ángulo IMU del eje Y
- OscillatingAbsolute: Valor absoluto de giro.
- OscillatingMax: Valor máximo del ángulo
- OscillatingScale: Valor de ángulo escalado.
- OscillatingScaleAbsolute: Valor absoluto de ángulo escalado

Definimos tres funciones de la siguiente manera:

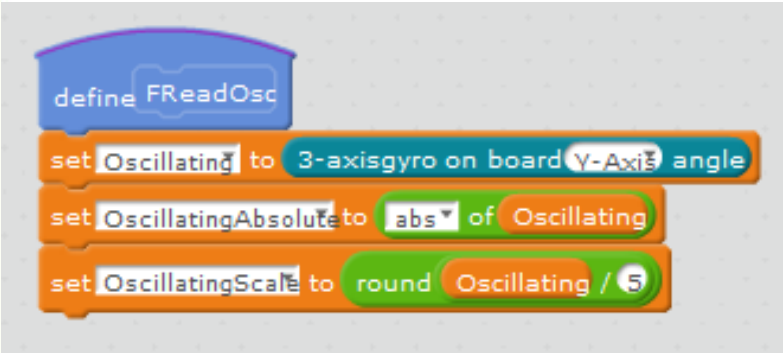
- FDisplay: la matriz LED y la pantalla LCD TFT mostrarán todos los datos al instante.
- FReadOsc: esta función es responsable de recolectar el ángulo de la placa Auriga.
- FOscMax: Función que mantendrá el ángulo máximo establecido por el potenciómetro.



Cuando iniciamos el programa, se establecerá un valor predeterminado en la variable "OscillatingScaleAbsolute", que es el valor del indicador de ángulo de guardabosques.

Luego, el código repetirá el ciclo ingresando a cada función en busca de cambios en el eje longitudinal.

Aquí se muestra lo que realmente hace cada función:



El valor de la variable "Oscillating" será el valor obtenido por el eje longitudinal Y.

El valor de la variable "OscillatingAbsolute" es el valor absoluto de la variable "Oscillating".

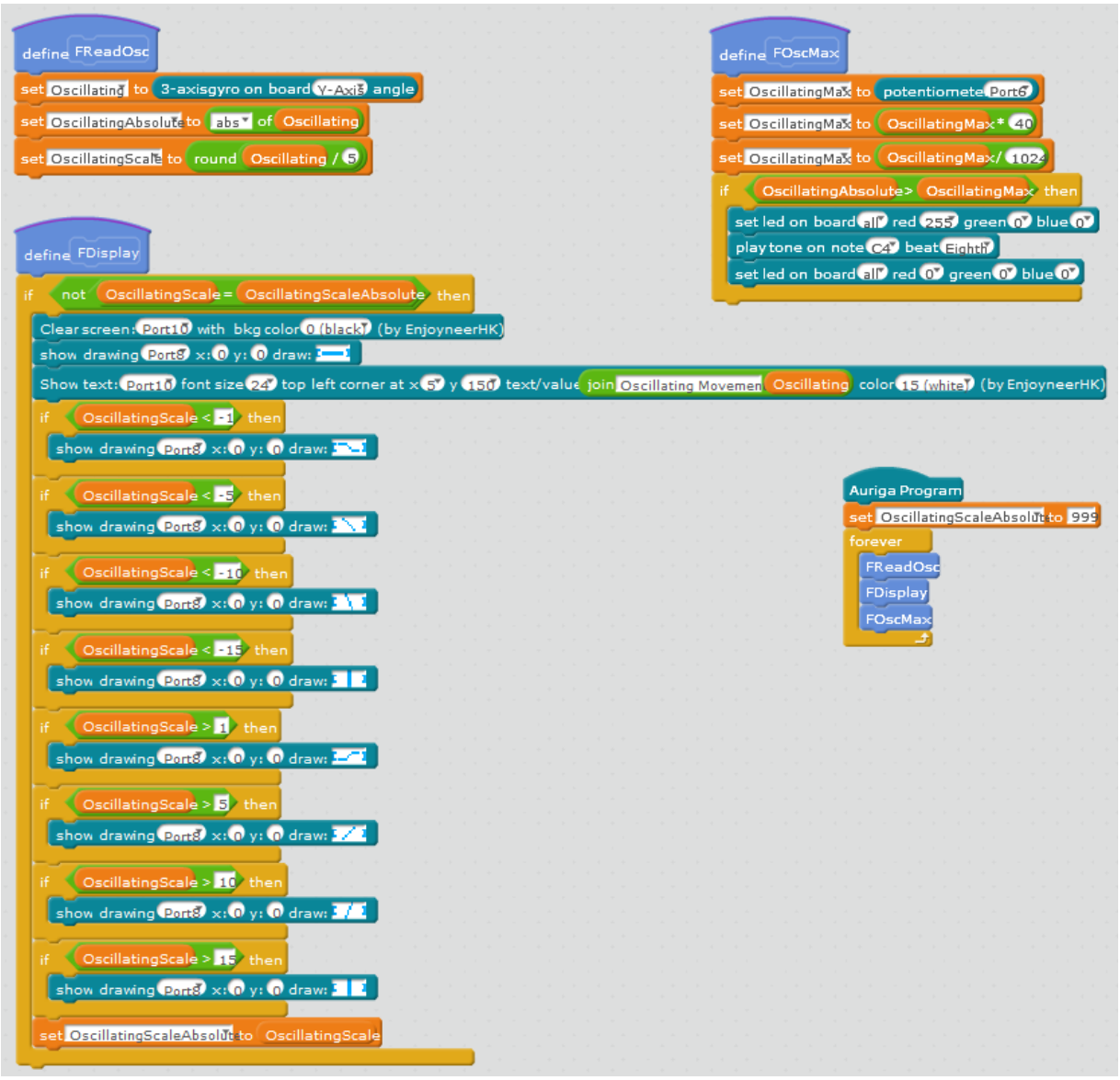
Y el valor de la variable "OscillatingScale" es el valor de la variable "Oscillating" redondeada y dividida por 5..

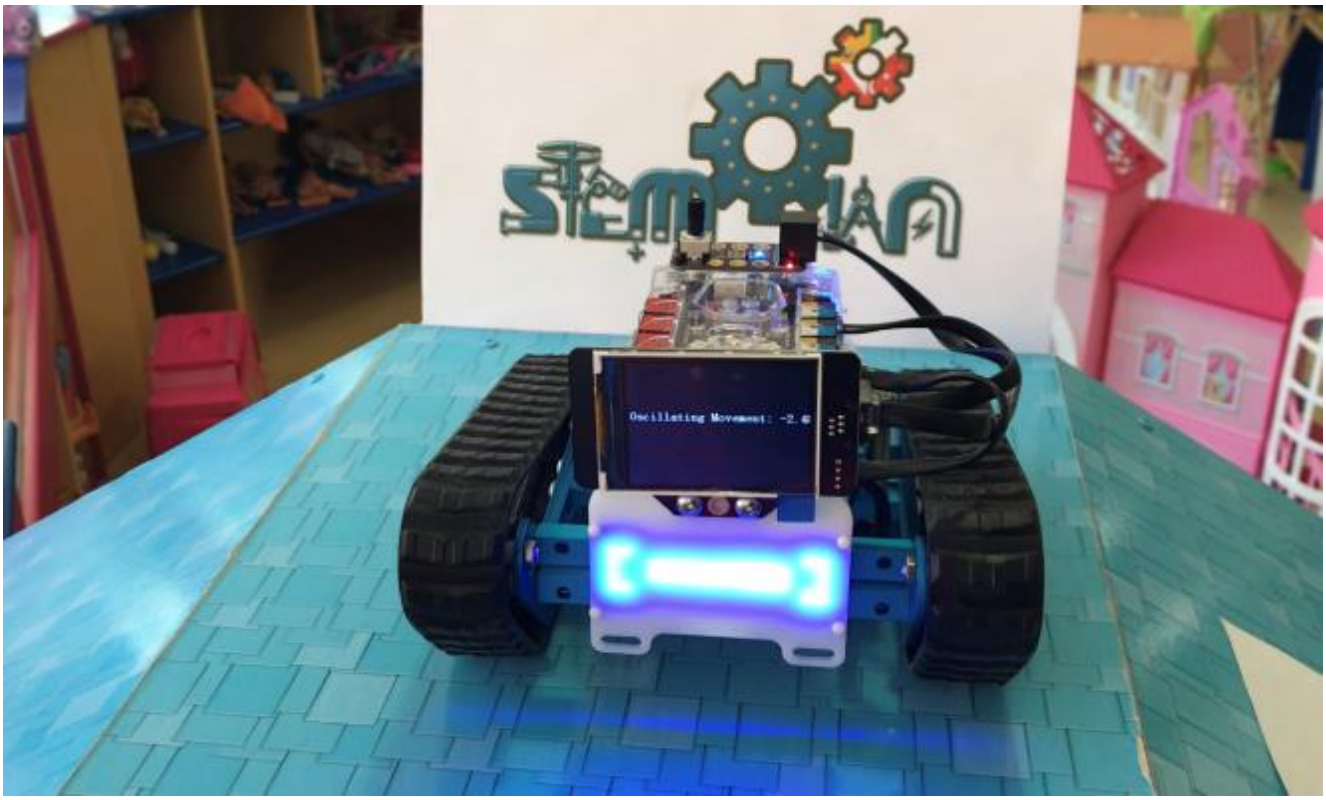
Aquí, podemos ver cómo la variable "OscillatingMax" recoge el valor del potenciómetro y establece el valor máximo al que se puede inclinar el mBot Ranger.

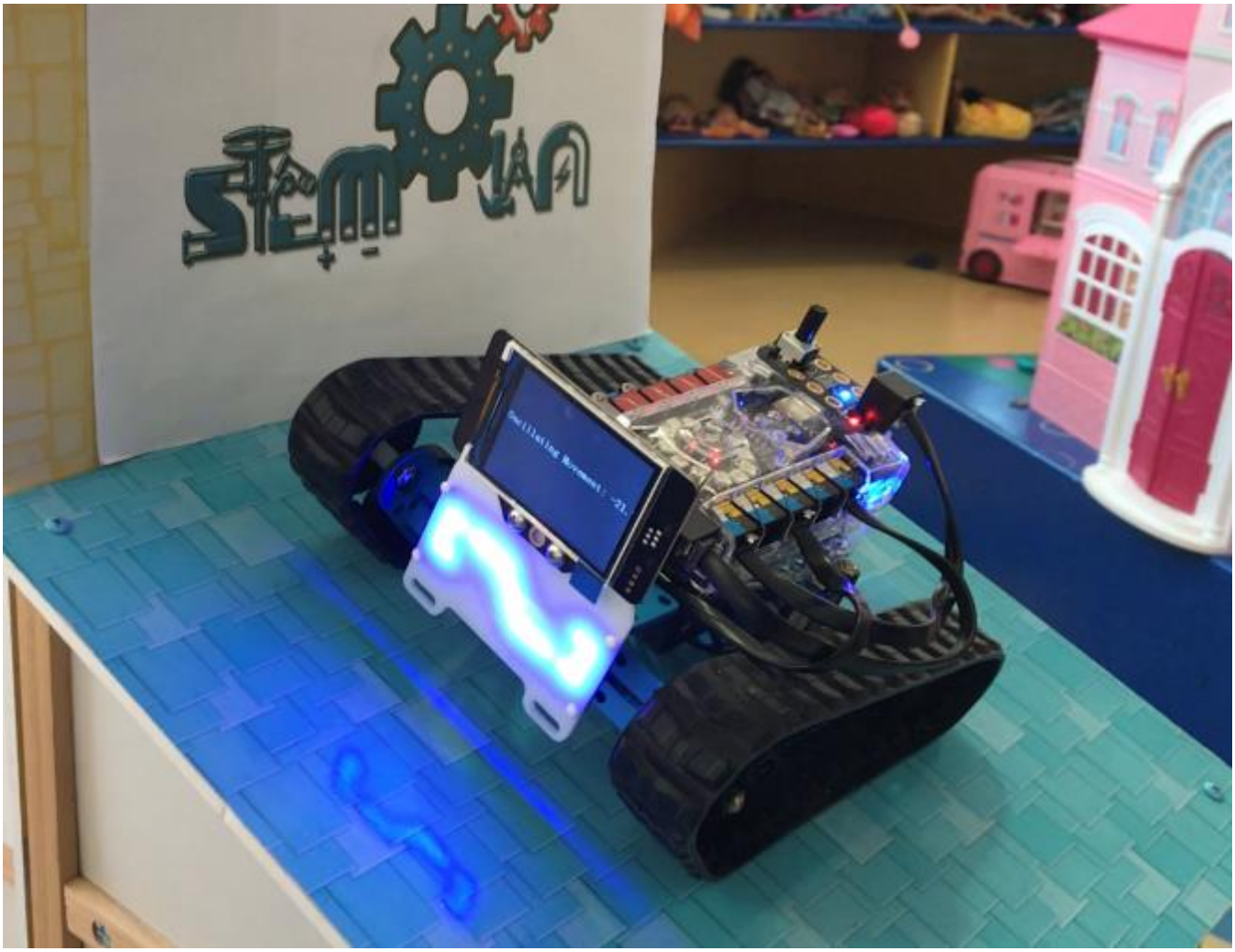
Si el mBot excede ese ángulo máximo, comenzará a emitir una señal acústica y luminosa.

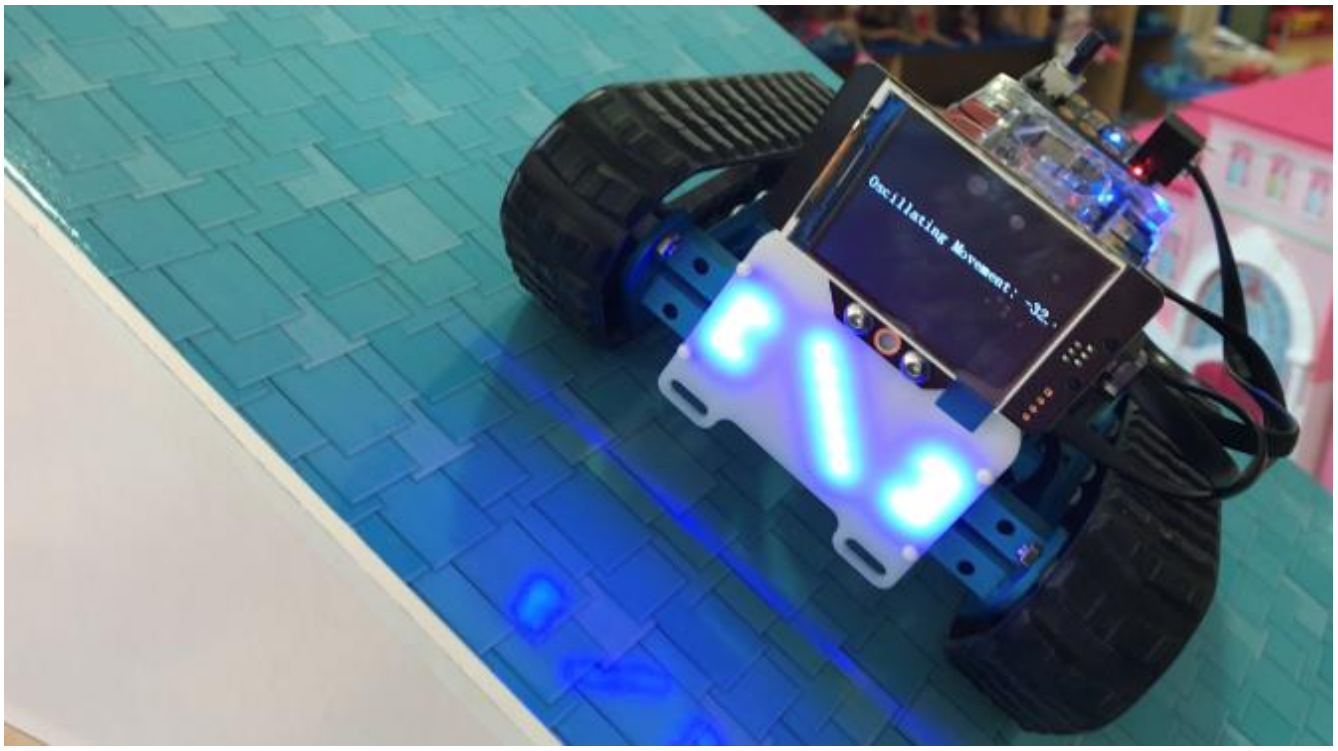
Finalmente, la función "FDisplay" muestra en tiempo real la inclinación de la matriz de LED y el valor del ángulo de balanceo en la pantalla LCD TFT.

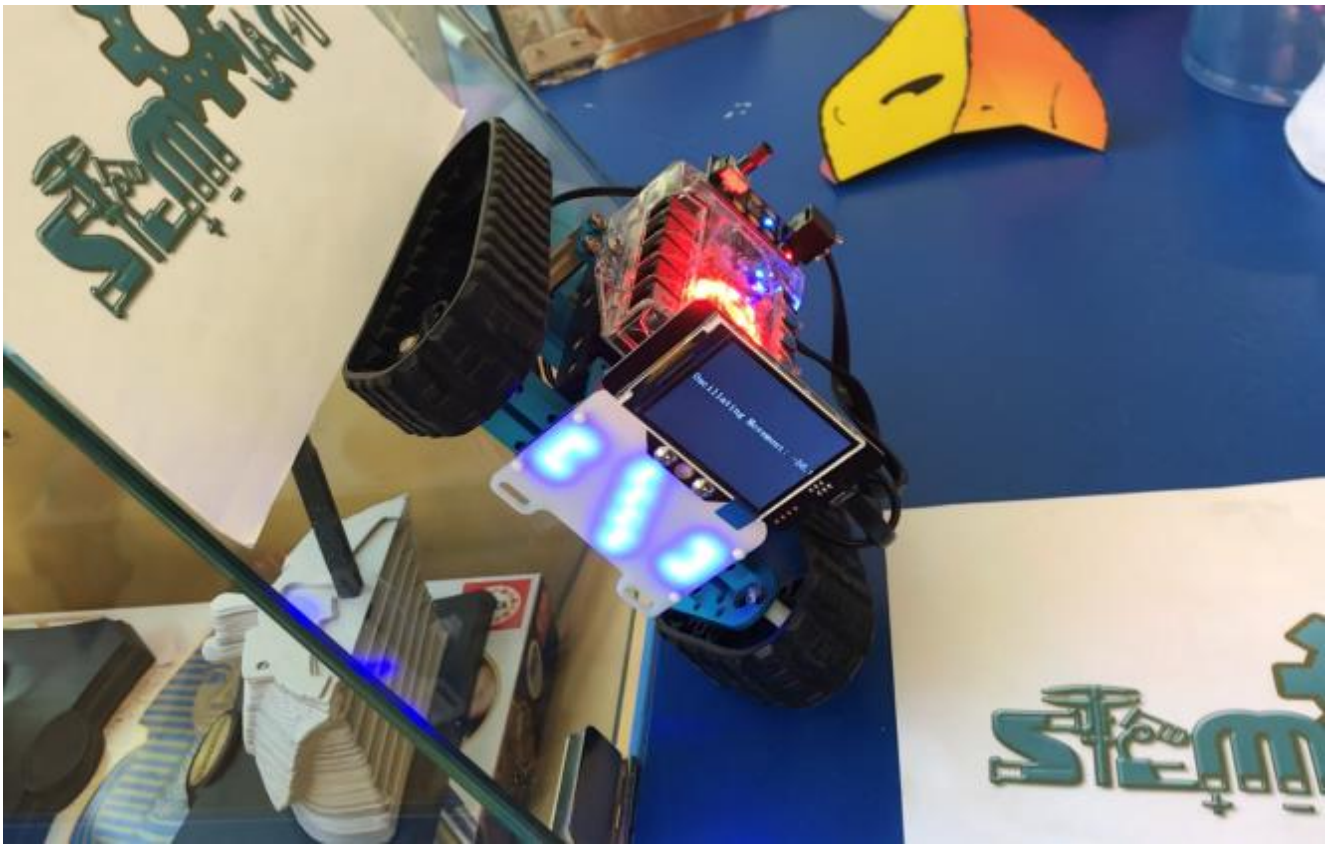
La versión completa del programa se puede encontrar a continuación y en las siguientes páginas se muestran algunas fotos de la actividad.




Oscillating Movement: -2.4




Oscillating Movement: -32



Ángulo de inclinación

El Movimiento de Inclinación, cuyo nombre es de origen náutico, se utiliza en el mundo automotriz para indicar la inclinación longitudinal de un vehículo con respecto al plano horizontal. En particular, el ángulo de inclinación es el que forma el eje longitudinal del vehículo con el plano horizontal.

Cuando se viaja en una carretera horizontal, el ángulo de inclinación será aproximadamente 0, ya que hay una pequeña desviación del sensor. Al subir una pendiente, tendremos un ángulo de inclinación positivo. Durante la actividad, el Ranger mostrará gráficamente en la Matriz de LED su inclinación longitudinal, es decir, el ángulo de inclinación o giro que presenta con respecto al eje transversal (X). Si este ángulo excede el valor establecido por el potenciómetro, lo advertirá con una señal acústica y luminosa. Además, en la pantalla TFT LCD mostraremos el ángulo del plano.

Dado que la placa Auriga que incorpora el Ranger tiene un módulo de medición inercial (IMU) integrado con un giroscopio y un acelerómetro de 3 ejes, es suficiente utilizar el valor del acelerómetro correspondiente al eje transversal del Ranger (eje X) para obtener el ángulo de inclinación directamente.

a. A continuación, enunciamos los pasos necesarios para realizar la actividad completa.
   Al igual que en el caso anterior, primero conectamos al mBot Ranger tanto a la Matriz de Led como a la Pantalla LCD TFT, así como al Potenciómetro.

b. Y luego, comenzamos a programar el mBot Ranger::

Las variables más significativas son:

- Pitch: ángulo IMU del eje X.
- PitchAbsolute: Valor absoluto de la inclinación.
- PitchMax: Valor máximo del ángulo.
- PitchScale: Valor del ángulo escalado.
- PitchScaleAbsolute: Valor absoluto del ángulo escalado.

Definimos tres funciones de la siguiente manera:

- FPitchMax: Función que mantendrá el ángulo máximo establecido por el Potenciometro.

- FDisplay: la matriz LED y la pantalla LCD TFT mostrarán todos los datos al instante.

- FReadPitch: esta función es responsable de recoger el ángulo de la placa Auriga.

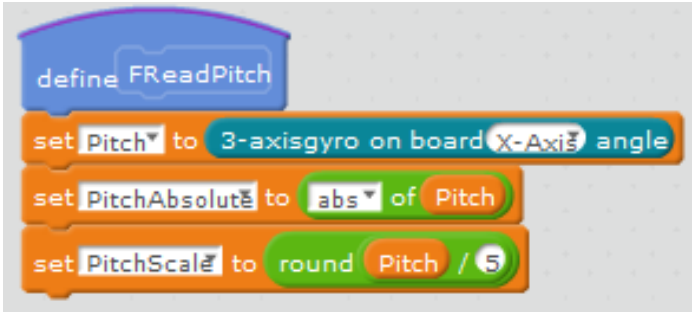
Cuando iniciamos el programa, un valor predeterminado se establecerá en la variable "PitchScaleAbsolute", que es el valor del indicador de ángulo del Ranger.

Luego, el código repetirá el ciclo ingresando a cada función en busca de cambios en el eje transversal.

Aquí se muestra lo que realmente hace cada función:

El valor de la variable "Pitch" será el valor obtenido por el eje transversal X.

El valor de la variable "Pitch Absolute" es el valor absoluto de la variable "Pitch".



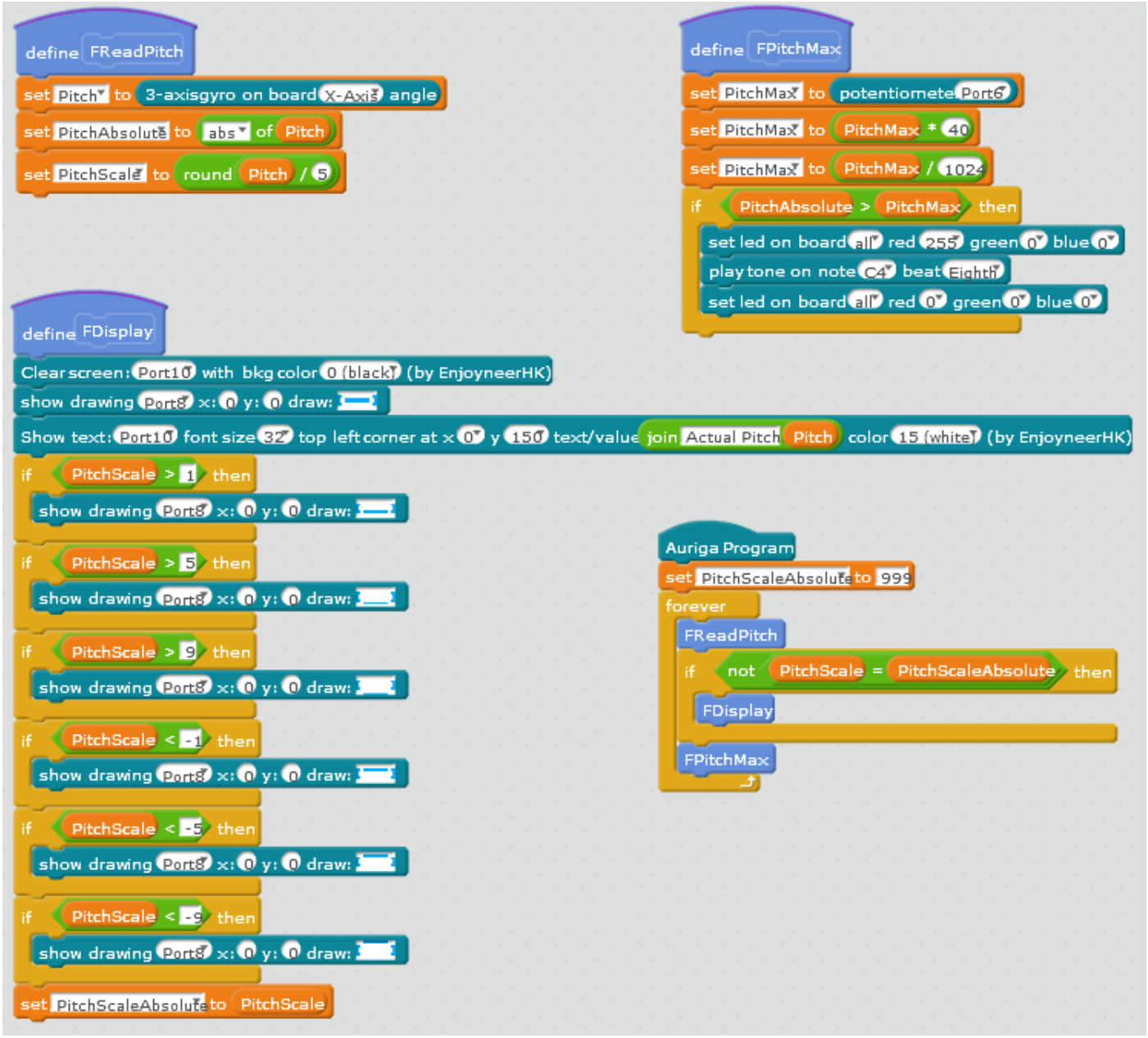
Y el valor de la variable "PitchScale" es el valor de la variable "Pitch" redondeado y dividido por 5..

Aquí, podemos ver cómo la variable "PitchMax" recoge el valor del potenciómetro y establece el valor máximo al que se puede inclinar el mBot Ranger.

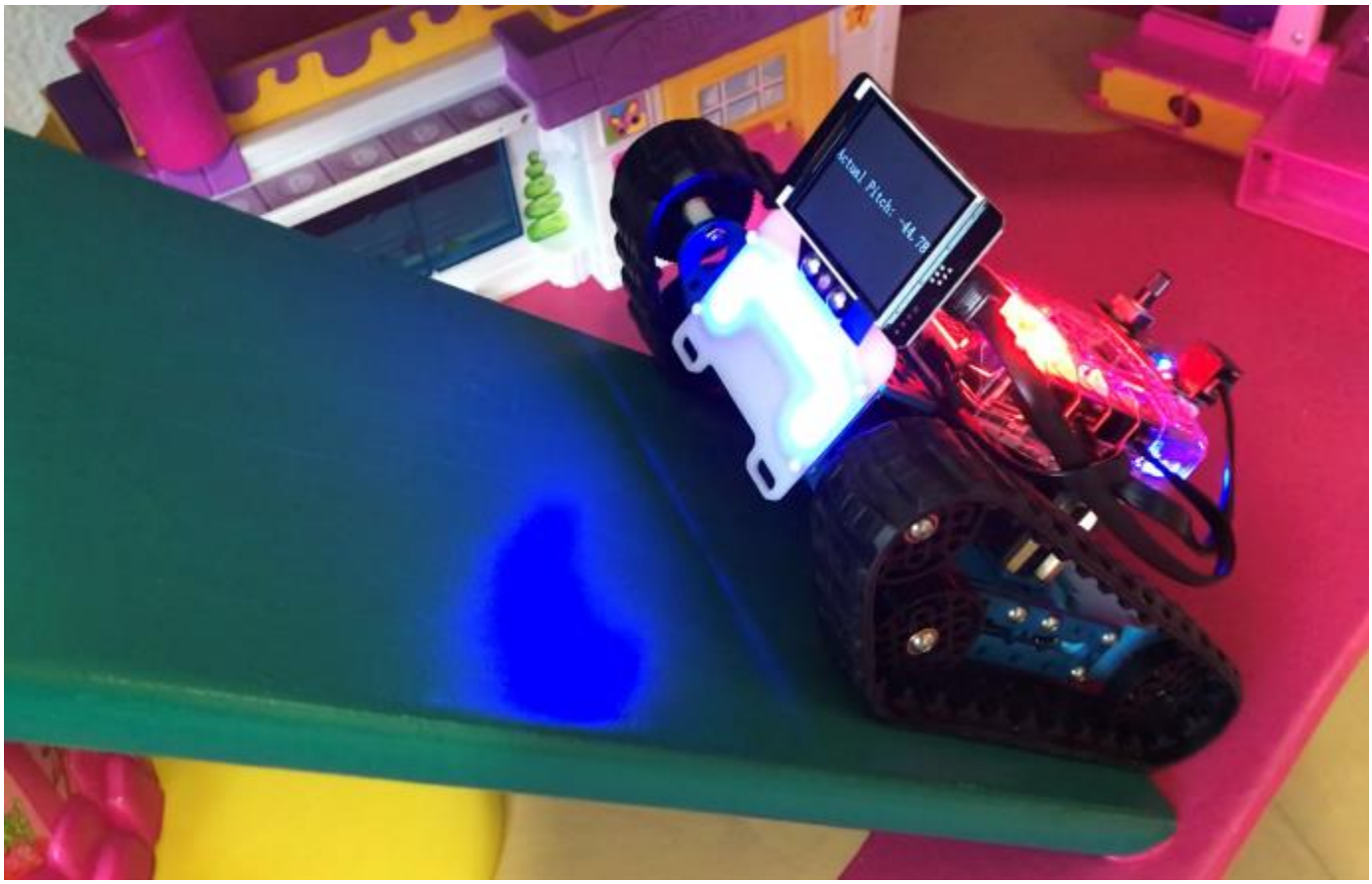
Si el mBot excede ese ángulo máximo, comenzará a emitir una señal acústica y luminosa.

Finalmente, la función "FDisplay" muestra en tiempo real la inclinación de la matriz de LED y el valor del ángulo de inclinación en la pantalla LCD TFT.

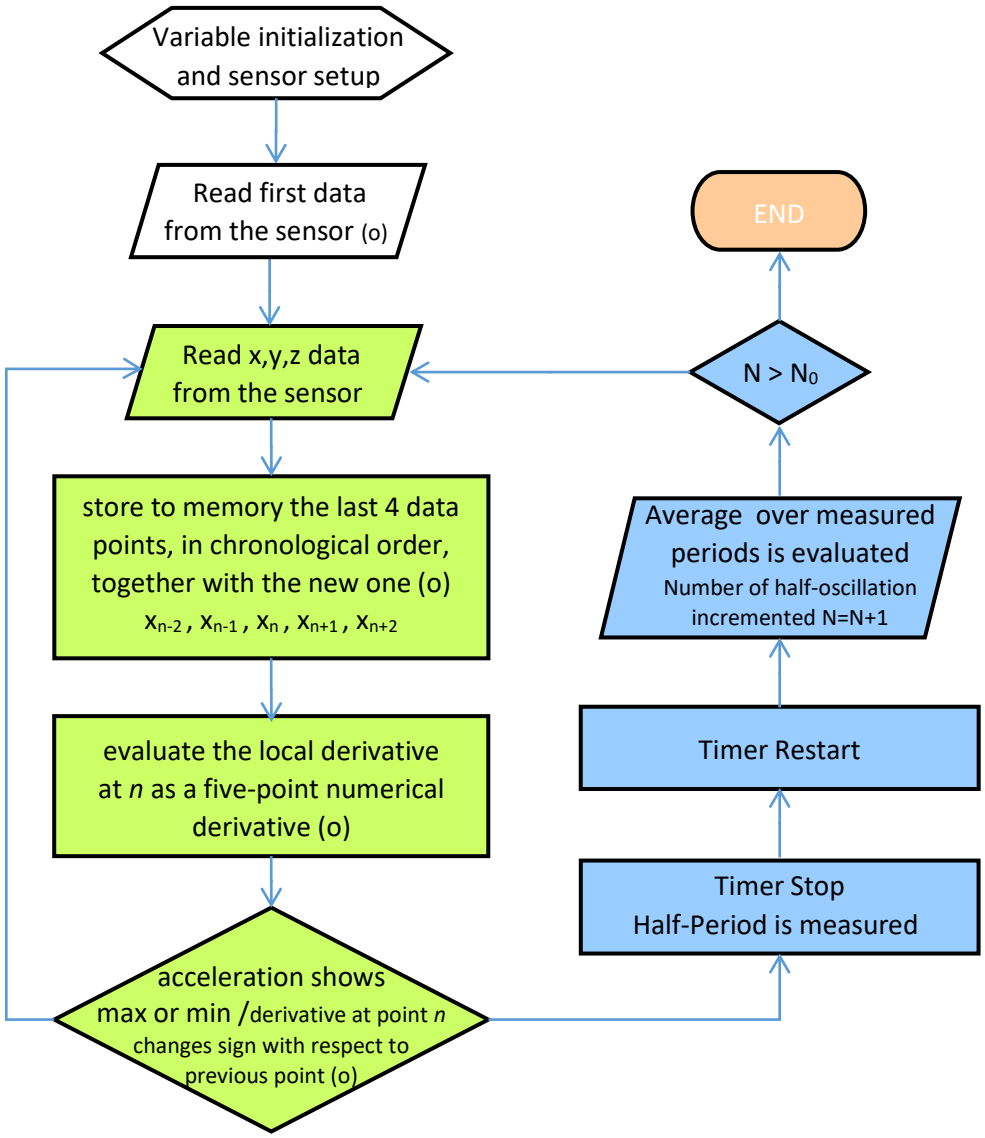
La versión completa del programa se puede encontrar a continuación y en las siguientes páginas se comparten algunas fotos de la actividad.


define FReadPitch
set Pitch to 3-axisgyro on board X-Axis angle
set PitchAbsolute to abs of Pitch
set PitchScale to round Pitch / 5
define FPitchMax
set PitchMax to potentiometer Port6
set PitchMax to PitchMax * 40
set PitchMax to PitchMax / 1024
if PitchAbsolute > PitchMax then
set led on board all red 255 green 0 blue 0
play tone on note C4 beat Eighth
set led on board all red 0 green 0 blue 0
define FDisplay
Clear screen: Port10 with bkg color 0 (black) (by EnjoyneerHK)
show drawing Port8 x: 0 y: 0 draw:
Show text: Port10 font size 32 top left corner at x 0 y 150 text/value join Actual Pitch Pitch color 15 (white) (by EnjoyneerHK)
if PitchScale > 1 then
show drawing Port8 x: 0 y: 0 draw:
if PitchScale > 5 then
show drawing Port8 x: 0 y: 0 draw:
if PitchScale > 9 then
show drawing Port8 x: 0 y: 0 draw:
if PitchScale < -1 then
show drawing Port8 x: 0 y: 0 draw:
if PitchScale < -5 then
show drawing Port8 x: 0 y: 0 draw:
if PitchScale < -9 then
show drawing Port8 x: 0 y: 0 draw:
set PitchScaleAbsolute to PitchScale
Auriga Program
set PitchScaleAbsolute to 999
forever
FReadPitch
if not PitchScale = PitchScaleAbsolute then
FDisplay
FPitchMax


Actual Pitch: -22.48

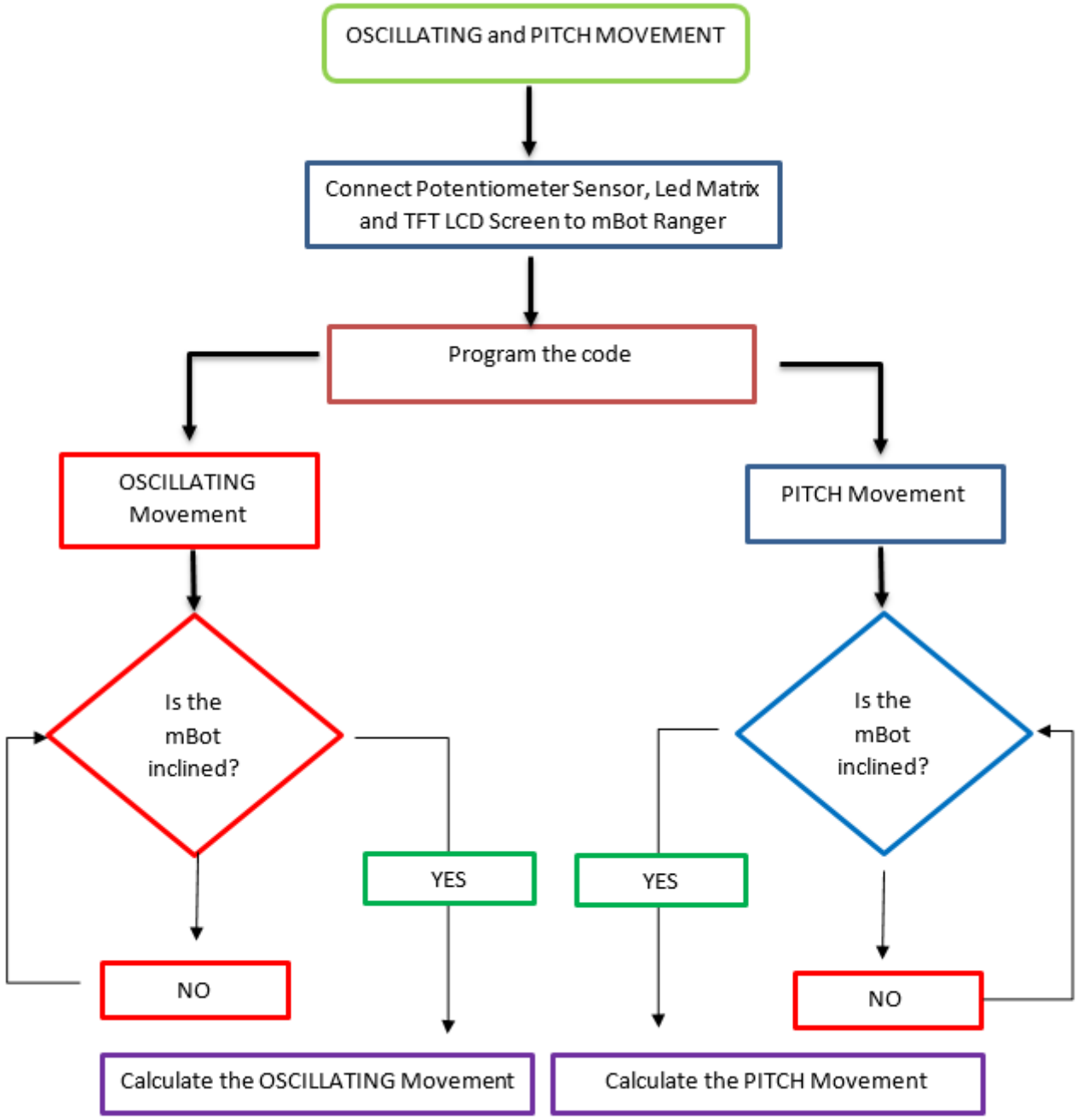

Actual Pitch: -44.78

Primera versión



(o) solo para la variable con la mayor oscilación

Segunda versión



EVALUACIÓN DEL ESTUDIANTE

Los indicadores para la evaluación de los estudiantes pueden incluir:

- Física: Ella / Él realiza mediciones de laboratorio con cuidado y precisión.
- Física: Ella / Él es capaz de comparar resultados experimentales con modelos teóricos.
- Física: Ella / Él aplica correctamente las relaciones entre los diferentes parámetros en ejercicios simples sobre resortes.
- Física: Ella / Él reconoce la física y las matemáticas de las oscilaciones armónicas en otro contexto además de la mecánica.

BIBLIOGRAFÍA

[1] Descripción del sensor http://learn.makeblock.com/en/me-3-axis-accelerometer-and-gyro-sensor/

[2] Definición de clase del sensor Gyro (para Arduino/C++)
http://learn.makeblock.com/en/Makeblock-library-for-Arduino/class_me_gyro.html

[3] En derivadas numéricas de cinco puntos.
L.Demeio, Course in numerical analysis, Università Politecnica delle Marche
https://dipmat.univpm.it/~demeio/public/Analisi_Numerica/Lezioni/Derivate.pdf
https://en.wikipedia.org/wiki/Finite_difference_coefficient

ESCALABILIDAD

La actividad es adecuada para estudiantes mayores de 12 años.

Con estudiantes mayores (14-15) se pueden incluir detalles cada vez mayores y se puede ejecutar la simulación desde Arduino para registrar y trazar resultados.

En futuros desarrollos podría ser interesante realizar más "movimientos de la física" con el mBot.