Part 06
-
IMU

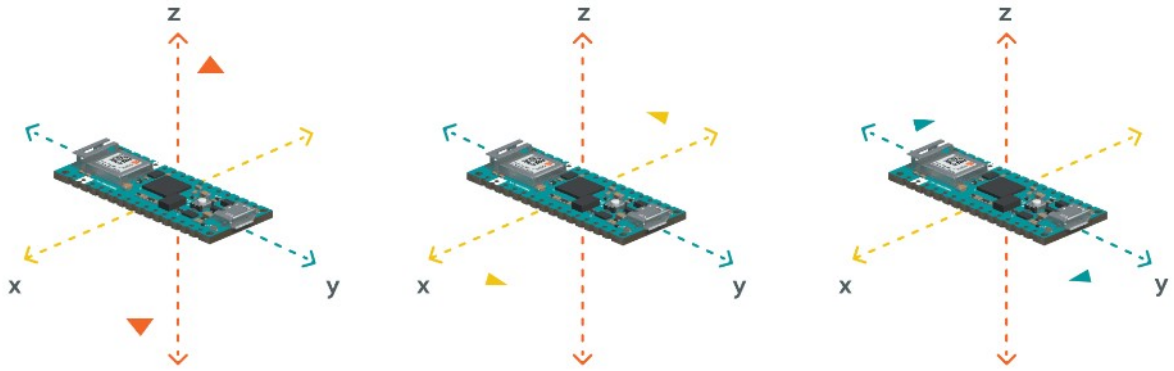
Version: 2021-06-01

Accessing IMU data

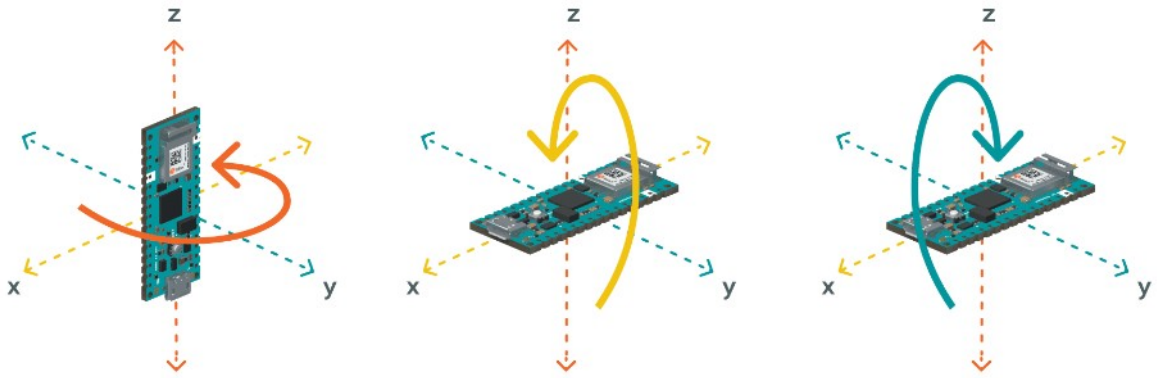
In this tutorial, we will learn how to access the gyroscope and accelerometer onboard the Nano RP2040 Connect. For this, we will be using the Arduino_LSMDS63 library.
Note: if you need help setting up your environment to use your Arduino Nano RP2040 board, please refer to this installation guide.

An IMU (Inertial Measurement Unit) is a component that exists of different sensors that records data such as specific force, angular rate, orientation. On the Nano RP2040 Connect, there is one gyroscope and one accelerometer**.** Let's take a look at how they work!

Accelerometer
An accelerometer is an electromechanical device used to measure acceleration forces. Such forces may be static, like the continuous force of gravity or, as is the case with many mobile devices, dynamic to sense movement or vibrations.
Acceleration.
In this example, we will use the accelerometer as a "level" that will provide information about the position of the board. With this application we will be able to read what the relative position of the board is, as well as the degrees by tilting the board up, down, left or right.



Gyroscope
A gyroscope sensor is a device that can measure and maintain the orientation and angular velocity of an object. Gyroscopes are more advanced than accelerometers, as they can measure the tilt and lateral orientation of an object, whereas an accelerometer can only measure its linear motion.



Gyroscope sensors are also called "Angular Rate Sensors" or "Angular Velocity Sensors". Measured in degrees per second, angular velocity is the change in the rotational angle of the object per unit of time.

In this example, we will use the gyroscope as an indicator for the direction of the force that is applied to the board. This will be achieved by swiftly moving the board for an instant in four directions: forward, backward, to the left and to the right. The results will be visible through the Serial Monitor.

Programming the board

We will now get to the programming part of this tutorial.

1. First, let's make sure we have the drivers installed. If we are using the Web Editor, we do not need to install anything. If we are using an offline editor, we need to install it manually. This can be done by navigating to `Tools > Board > Board Manager....` Here we need to look for the `Arduino Mbed OS Nano Boards` and install it.

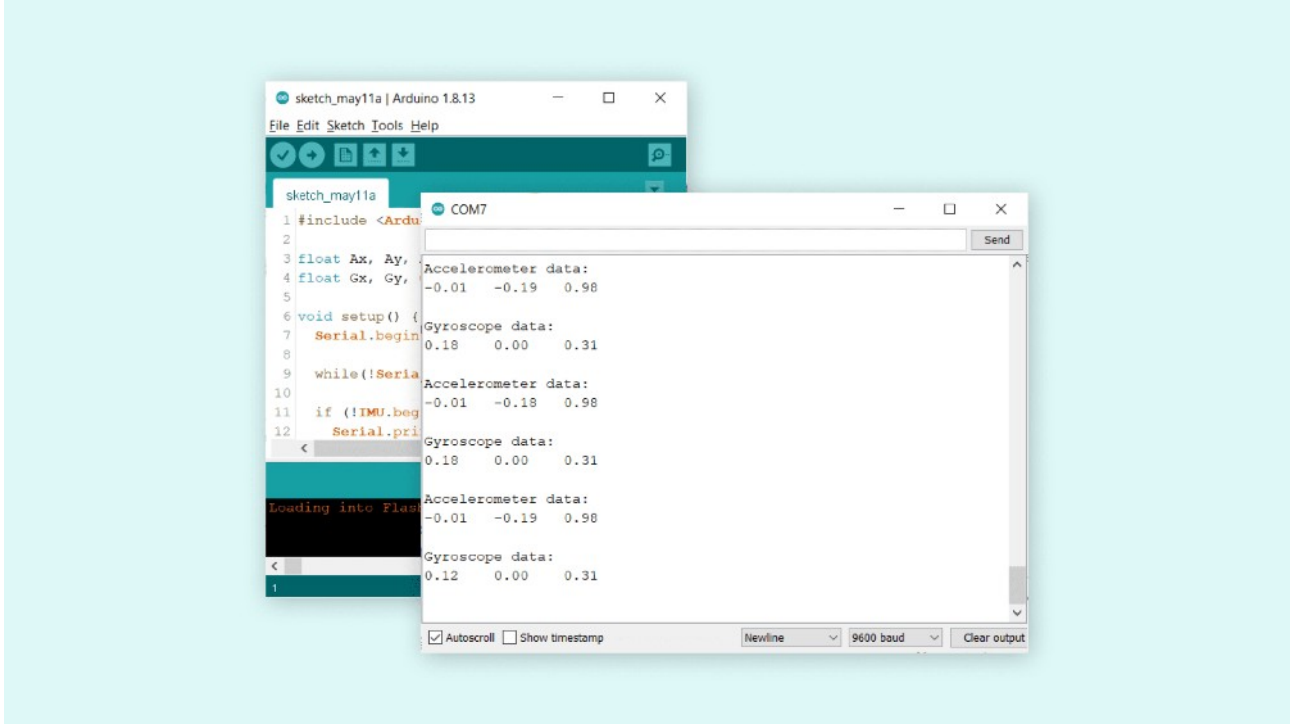
2. Now, we need to install the libraries needed. If we are using the Web Editor, there is no need to install anything. If we are using an offline editor, simply go to `Tools > Manage libraries..`, and search for `Arduino_LSM6DS3` and install it.

3. We can now take a look at some of the core functions of this sketch:
   - `IMU.begin()` initializes the library.
   - `IMU.accelerationSampleRate()` reads the sampling rate in Hz.
   - `IMU.accelerationAvailable()` checks if there's data available from the IMU.
   - `IMU.readAcceleration(Ax, Ay, Az)` reads the accelerometer, and returns the value of the x y and z axis.
   - `IMU.gyroscopeSampleRate()` reads the sampling rate in Hz.
   - `IMU.gyroscopeAvailable()` checks if there's data available from the IMU.
   - `IMU.readGyroscope(Gx, Gy, Gz)` reads the accelerometer, and returns the value of the x y and z axis.

The sketch can be found in the snippet below. Upload the sketch to the board.

```
#include <Arduino_LSM6DSOX.h>

float Ax, Ay, Az;
float Gx, Gy, Gz;

void setup()
{
  Serial.begin(9600);

  while(!Serial);

  if (!IMU.begin())
  {
    Serial.println("Failed to initialize IMU!");
    while (1);
  }

  Serial.print("Accelerometer sample rate = ");
  Serial.print(IMU.accelerationSampleRate());
  Serial.println("Hz");
  Serial.println();

  Serial.print("Gyroscope sample rate = ");
  Serial.print(IMU.gyroscopeSampleRate());
  Serial.println("Hz");
  Serial.println();

}

void loop()
{

  if (IMU.accelerationAvailable())
  {
    IMU.readAcceleration(Ax, Ay, Az);

    Serial.println("Accelerometer data: ");
    Serial.print(Ax);
    Serial.print('\t');
    Serial.print(Ay);
    Serial.print('\t');
    Serial.println(Az);
    Serial.println();
  }

  if (IMU.gyroscopeAvailable())
  {
    IMU.readGyroscope(Gx, Gy, Gz);

    Serial.println("Gyroscope data: ");
    Serial.print(Gx);
    Serial.print('\t');
    Serial.print(Gy);
    Serial.print('\t');
    Serial.println(Gz);
    Serial.println();
  }

delay(500);

}
```

Testing it out
After successfully uploading the code to the board, we will need to open the Serial Monitor to initialize the program. Once we open it, data will start printing.



The data is being printed with an interval of 500 milliseconds. This can be adjusted by changing the line `delay(500)` at the bottom of the sketch.

Using the IMU Machine Learning Core Features

In addtion to measuring the raw movement daata, the IMU mounted on the Nano RP2040 Connect has a machine learning core. This allows to do advanced movement detection on the IMU itself. This frees the microcontroller from this task so it can do other things like sending data to a network, interacting with other sensors or operating actuators. By using the machine learning core specific activity patterns such as walking, jogging or biking can be recognized. The patterns that can be recoginzed by the IMU depend on the machine learning model that is used. In this tutorial we will focus on the pre-trained model that comes with the library. Let's take a look at how the IMU's smart features work!

Programming the Board

We will now get to the programming part of this tutorial.

1. First, let's make sure the drivers are installed. If you are using the Web Editor, you don't need to install anything. If you are using an offline editor, you need to install it manually. This can be done by navigating to `Tools > Board > Board Manager....` Here you need to search for `Arduino Mbed OS Nano Boards` and install the package.

2. Now, you need to install the `STM32duino X-NUCLEO-IKS01A3` library. If you are using an offline editor, simply go to `Tools > Manage libraries..`, and search for `STM32duino X-NUCLEO-IKS01A3` and install it. If a dialog appears asking whether the dependent libraries should be installed comfirm this.

3. Once the library is installed open the example sketch `X_NUCLEO_IKS01A3_LSM6DSOX_MLC` that comes with the library from the examples menu. You may need to check in the `INCOMPATIBLE` group.

   We can now take a look at some of the core functions of this sketch:
   - `DEV_I2C.begin()` initializes the I2C communication with the IMU module.
   - `AccGyr.begin()` initialized the accelerometer and the gyroscope.
   - `AccGyr.Enable_X()` enables the accelerometer.
   - `AccGyr.Enable_G()` enables the gyroscope sensor.
   - `attachInterrupt(INT_1, INT1Event_cb, RISING)` registers an interrupt callback for when the IMU interrupt pin is pulled high. This indicates that a motion type was detected.
   - `AccGyr.Get_MLC_Output(mlc_out)` reads the machine learning core output and saves the decision tree into an array variable.
   - `printMLCStatus(status)` translates the number retrieved from the machine learning core into a human readable description of the detected activity type.

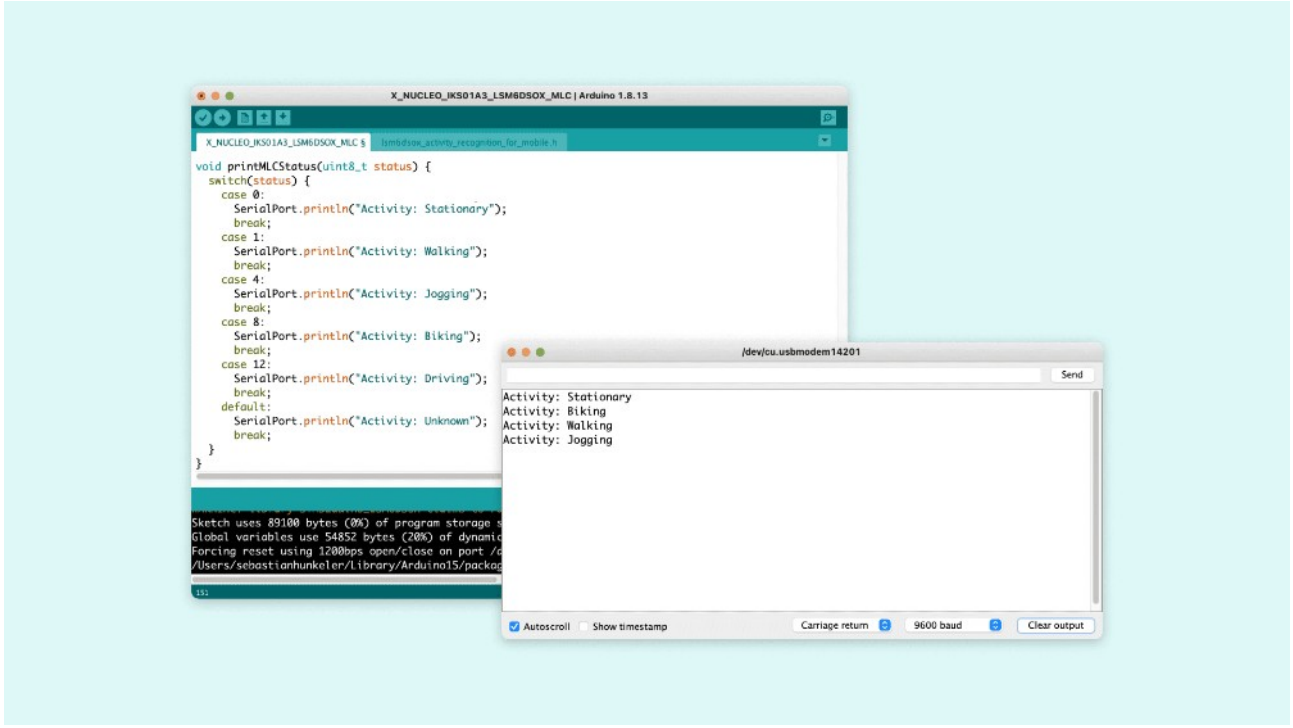
The machine learning model is stored in the `lsm6dsox_activity_recognition_for_mobile.h` file. The only thing that needs to be changed in the sketch is the pin mapping for the IMU interrupt pin at the beginning of the sketch. On the Nano RP2040 Connect it is on the GPIO pin 21 which can be referred to as `INT_IMU`:

```
#define INT_1 INT_IMU
```

Once you changed that, upload the sketch to the board.

Testing It Out

After successfully uploading the code to the board, open the Serial Monitor. It will print `Activity: Stationary` after a few seconds. When you now move the Nano RP2040 continuously up and down it will pick up this activity and detect it most likely as walking. Try moving it faster and check if it now detects this as jogging. If you have a power bank you can connect your Nano RP2040 Connect to it and go for a run or a walk and check if it detects those activities successfully.



If the code is not working, there are some common issues we can troubleshoot:

- You have not installed the STM32duino X-NUCLEO-IKS01A3 library. Install it through the IDE's library manager.
- An incorrect pin number was assigned to INT_1.
- You have a faulty USB cable. Try with a different Micro USB cable e.g. one from a mobile phone.