Getting started with MotionGR real-time gesture recognition library in X-CUBE-MEMS1 expansion for STM32Cube

Introduction

The MotionGR is a middleware library part of X-CUBE-MEMS1 software and runs on STM32. It provides real-time information about the gesture just performed by the user with the device, such as a cell phone.

It is able to distinguish the following gestures: pick up, glance, wake up.

This library is intended to work with ST MEMS only.

The algorithm is provided in static library format and is designed to be used on STM32 microcontrollers based on the ARM® Cortex®-M3, ARM® Cortex®-M33, ARM® Cortex®-M4 or ARM® Cortex®-M7 architecture.

It is built on top of STM32Cube software technology that eases portability across different STM32 microcontrollers.

The software comes with sample implementation running on X-NUCLEO-IKS4A1 or X-NUCLEO-IKS01A3 expansion board on a NUCLEO-F401RE, NUCLEO-U575ZI-Q or NUCLEO-L152RE development board.

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
API	Application programming interface
BSP	Board support package
GUI	Graphical user interface
HAL	Hardware abstraction layer
IDE	Integrated development environment

2 MotionGR middleware libray in X-CUBE-MEMS1 software expansion for STM32Cube

2.1 MotionGR overview

The MotionGR library expands the functionality of the X-CUBE-MEMS1 software.

The library acquires data from the accelerometer and provides information about the gesture just performed by the user with the device.

The library is designed for ST MEMS only. Functionality and performance when using other MEMS sensors are not analyzed and can be significantly different from what described in the document.

Sample implementation is available for X-NUCLEO-IKS4A1 and X-NUCLEO-IKS01A3 expansion boards, mounted on a NUCLEO-F401RE, NUCLEO-U575ZI-Q or NUCLEO-L152RE development board.

2.2 MotionGR library

Technical information fully describing the functions and parameters of the MotionGR APIs can be found in the MotionGR_Package.chm compiled HTML file located in the Documentation folder.

2.2.1 MotionGR library description

The MotionGR gesture recognition library manages the data acquired from the accelerometer; it features:

- possibility to distinguish among the following activities: pick up, glance, wake up
- recognition based only on accelerometer data
- required accelerometer data sampling frequency is 50 Hz
- resources requirements:
  - Cortex-M3: 10.0 kB of code and 4.4 kB of data memory
  - Cortex-M33: 10.1 kB of code and 4.4 kB of data memory
  - Cortex-M4: 10.2 kB of code and 4.4 kB of data memory
  - Cortex-M7: 10.3 kB of code and 4.4 kB of data memory
- available for ARM® Cortex®-M3, ARM® Cortex®-M33, ARM® Cortex®-M4 and ARM® Cortex®-M7 architectures

2.2.2 MotionGR APIs

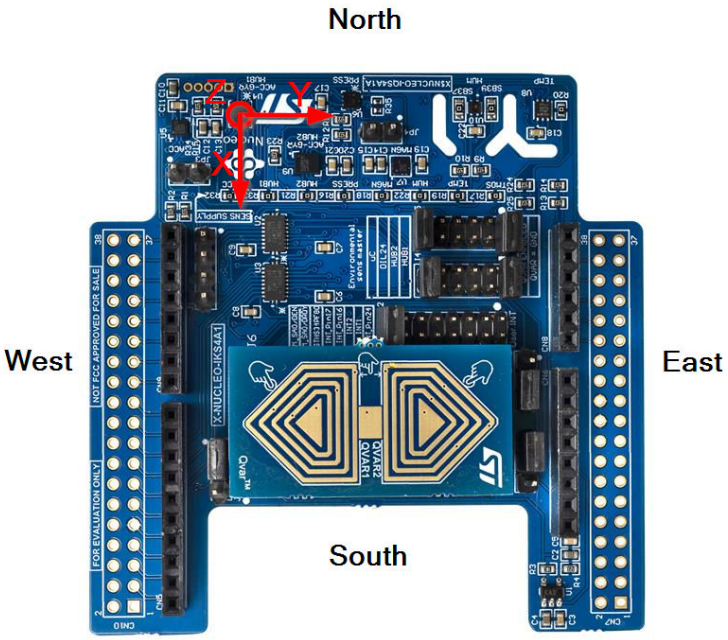
The MotionGR library APIs are:

- `uint8_t MotionGR_GetLibVersion(char *version)`
  - retrieves the library version
  - `*version` is a pointer to an array of 35 characters
  - returns the number of characters in the version string
- `void MotionGR_Initialize(void)`
  - performs MotionGR library initialization and setup of the internal mechanism

Note: This function must be called before using the gesture recognition library and the CRC module in the STM32 microcontroller (in RCC peripheral clock enable register) has to be enabled.

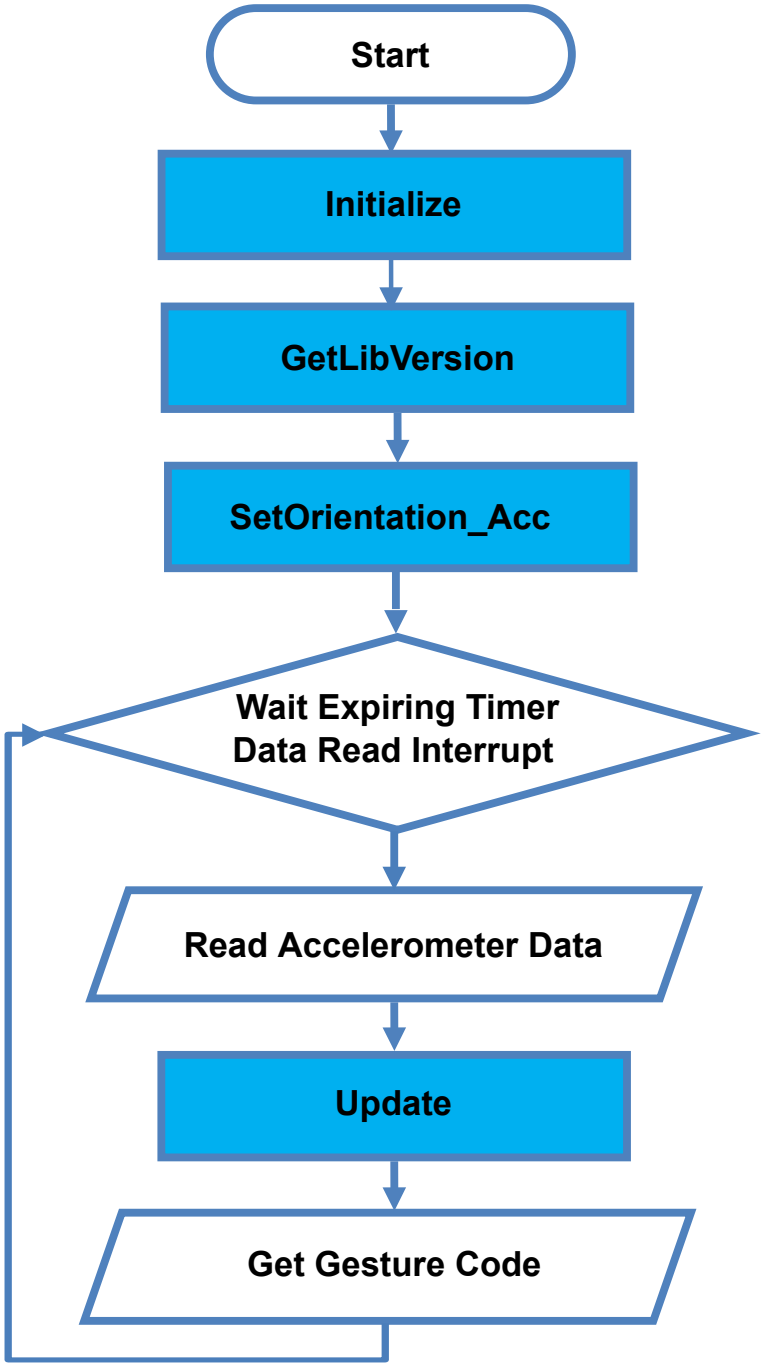
- `void MotionGR_Update(MGR_input_t *data_in, MGR_output_t *data_out)`
  - executes gesture recognition algorithm
  - `*data_in` parameter is a pointer to a structure with input data
  - the parameters for the structure type `MGR_input_t` are:
    - `AccX` is the accelerometer sensor value in X axis in g
    - `AccY` is the accelerometer sensor value in Y axis in g
    - `AccZ` is the accelerometer sensor value in Z axis in g
  - `*data_out` parameter is a pointer to an enum with the following items:
    - `MGR_NOGESTURE = 0`
    - `MGR_PICKUP = 1`
    - `MGR_GLANCE = 2`
    - `MGR_WAKEUP = 3`
- `void MotionGR_SetOrientation_Acc (const char *acc_orientation)`
  - this function is used to set the accelerometer data orientation
  - configuration is usually performed immediately after the `MotionGR_Initialize` function call
  - `*acc_orientation` parameter is a pointer to a string of three characters indicating the direction of each of the positive orientations of the reference frame used for accelerometer data output, in the sequence x, y, z. Valid values are: n (north) or s (south), w (west) or e (east), u (up) or d (down).
  - As shown in the figure below, the X-NUCLEO-IKS4A1 accelerometer sensor has an SEU (x - South, y - East, z - Up), so the string is: “seu”.

Figure 1. Example of sensor orientations



2.2.3 API flow chart

Figure 2. MotionGR API logic sequence



2.2.4 Demo code

The following demonstration code reads data from accelerometer sensor and gets the gesture code.

```
[…]
#define VERSION_STR_LENG 35
[…]

/* Initialization */
char lib_version[VERSION_STR_LENG];
char acc_orientation[3];

/* Gesture recognition API initialization function */
MotionGR_Initialize();

/* Optional: Get version */
MotionGR_GetLibVersion(lib_version);

/* Set accelerometer orientation */
acc_orientation[0] ='n';
acc_orientation[1] ='w';
acc_orientation[2] ='u';
MotionGR_SetOrientation_Acc(acc_orientation);

[…]

/* Using Gesture Recognition algorithm */
Timer_OR_DataRate_Interrupt_Handler()
{
  MGR_input_t data_in;
  MGR_output_t data_out;

  /* Get acceleration X/Y/Z in g */
  MEMS_Read_AccValue(&data_in.AccX, &data_in.AccY, &data_in.AccZ);

  /* Gesture recognition algorithm update */
  MotionGR_Update(&data_in, &data_out);
}
```

2.2.5 Algorithm performance

The gesture recognition algorithm only uses data from the accelerometer and runs at a low frequency (50 Hz) to reduce power consumption.

It detects and provides real-time information on the following user gestures:

- pick up: raising/lifting the board from a table;
- glance: approximately 30° rotation of the board, similar to the gesture of rotating a phone to glance at it;
- wake up: shaking action.

Table 2. Algorithm elapse time (µs) Cortex-M4, Cortex-M3

Cortex-M4 STM32F401RE at 84 MHz			Cortex-M3 STM32L152RE at 32 MHz		
Min	Avg	Max	Min	Avg	Max
136	240	341	473	870	1235

Table 3. Algorithm elapse time (µs) Cortex-M33 and Cortex-M7

Cortex- M33 STM32U575ZI-Q at 160 MHz			Cortex- M7 STM32F767ZI at 96 MHz		
Min	Avg	Max	Min	Avg	Max
87	140	191	459	504	808

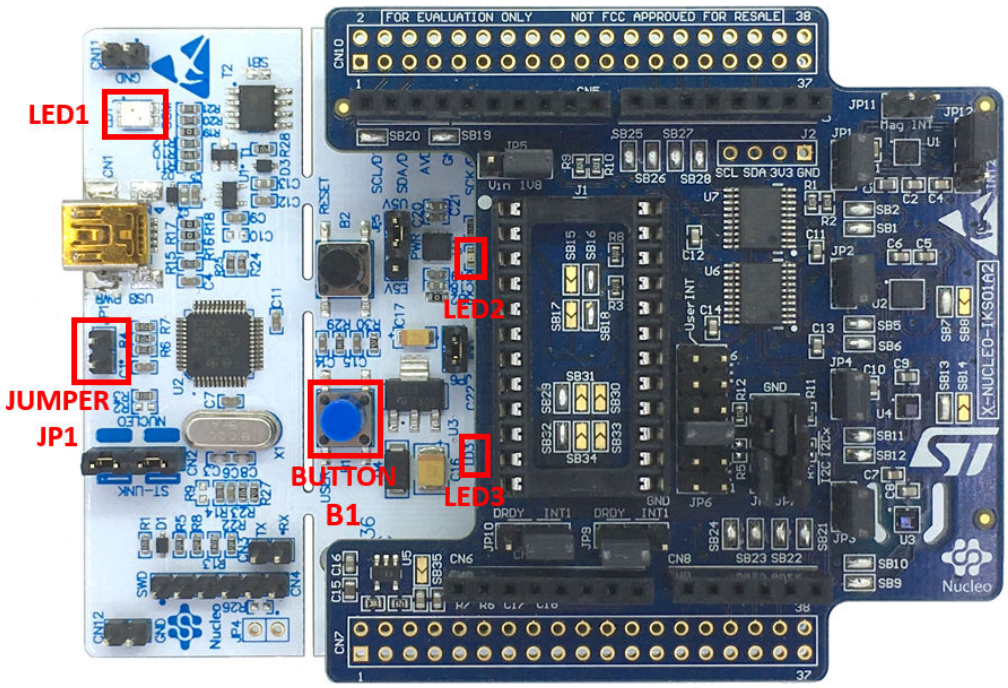
2.3 Sample application

The MotionGR middleware can be easily manipulated to build user applications; a sample application is provided in the Application folder.

It is designed to run on a NUCLEO-F401RE, NUCLEO-U575ZI-Q or NUCLEO-L152RE development board connected to an X-NUCLEO-IKS4A1 or X-NUCLEO-IKS01A3 expansion board.

The application recognizes performed gestures in real-time.

Figure 3. STM32 Nucleo: LEDs, button, jumper



The above figure shows the user button B1 and the three LEDs of the NUCLEO-F401RE board. Once the board is powered, LED LD3 (PWR) turns ON.

A USB cable connection is required to monitor real-time data. The board is powered by the PC via USB connection. This working mode allows the user to display detected gesture, accelerometer data, time stamp and eventually other sensor data, in real-time, using the MEMS-Studio.

2.4 MEMS Studio application

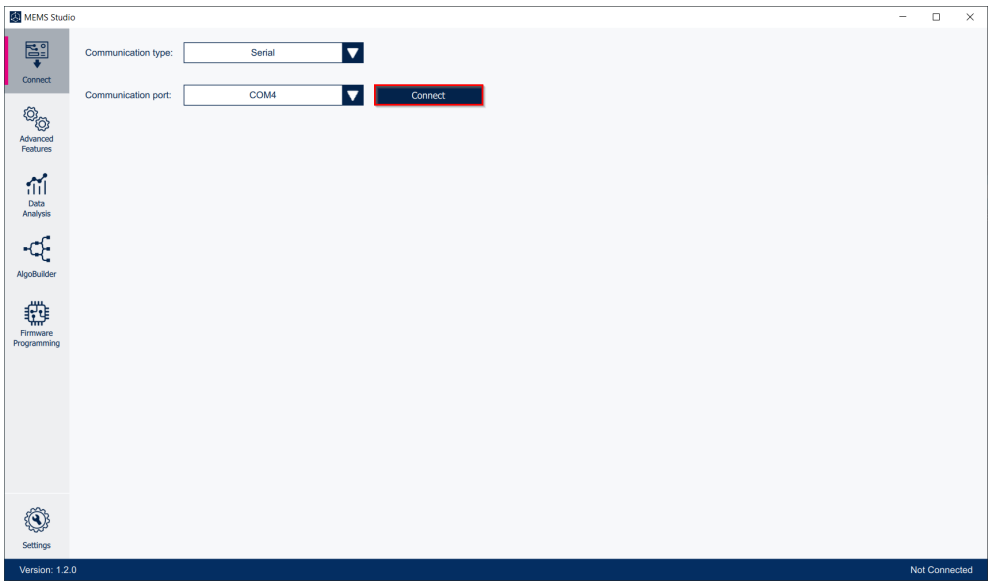
The sample application uses MEMS-Studio application, which can be downloaded from www.st.com.

Step 1. Ensure that the necessary drivers are installed and the STM32 Nucleo board with appropriate expansion board is connected to the PC.

Step 2. Launch the MEMS-Studio application to open the main application window.

If an STM32 Nucleo board with supported firmware is connected to the PC, it is automatically detected. Press the [**Connect**] button to establish connection to the evaluation board.

Figure 4. MEMS-Studio - Connect

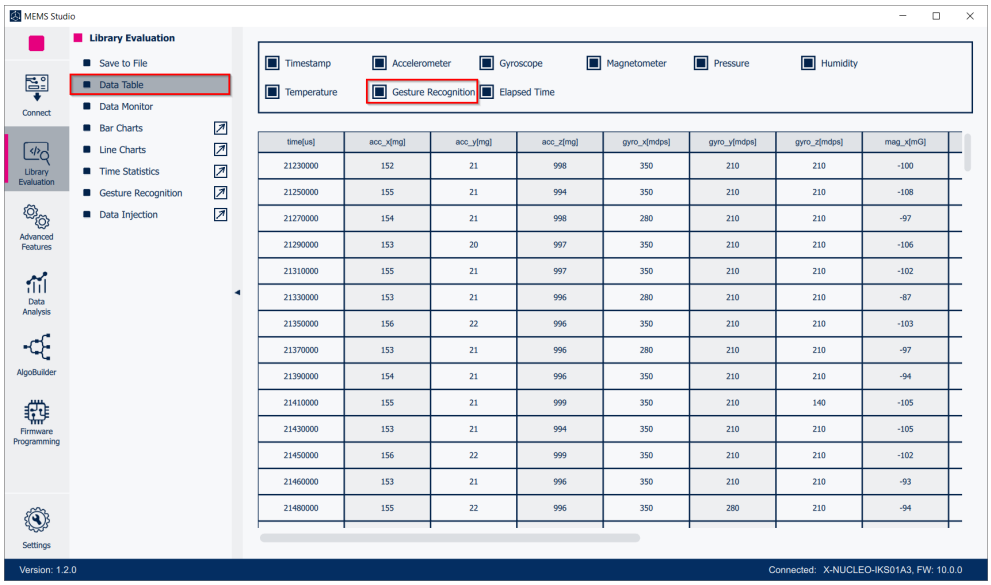


Step 3. When connected to a STM32 Nucleo board with supported firmware [**Library Evaluation**] tab is opened.

To start and stop data streaming, toggle the appropriate [**Start**] or [**Stop**] button on the outer vertical tool bar.

The data coming from the connected sensor can be viewed selecting the [**Data Table**] tab on the inner vertical tool bar.

Figure 5. MEMS-Studio - Library Evaluation - Data Table



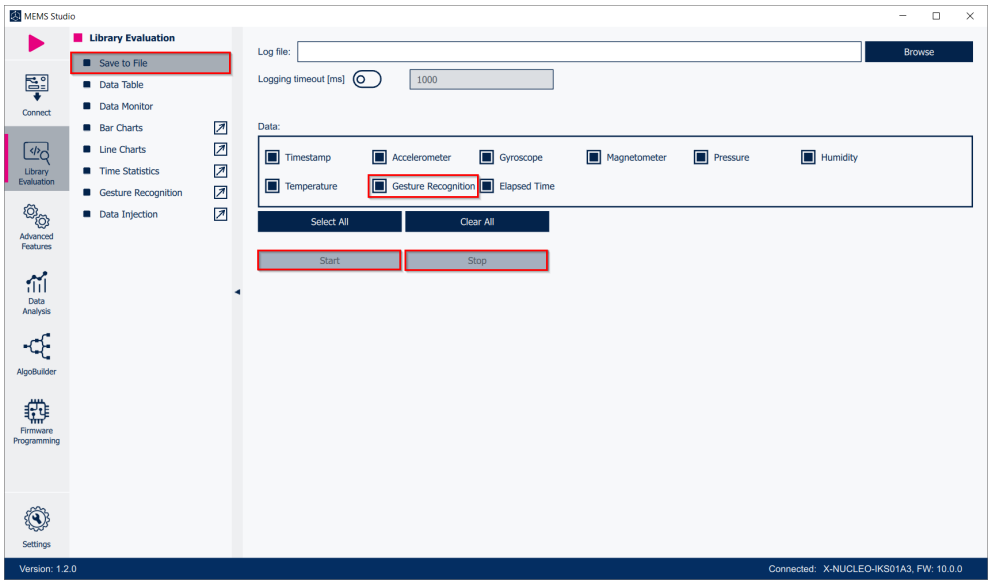
time[us]	acc_x[mg]	acc_y[mg]	acc_z[mg]	gyro_x[mdps]	gyro_y[mdps]	gyro_z[mdps]	mag_x[mG]
21230000	152	21	998	350	210	210	-100
21250000	155	21	994	350	210	210	-108
21270000	154	21	998	280	210	210	-97
21290000	153	20	997	350	210	210	-106
21310000	155	21	997	350	210	210	-102
21330000	153	21	996	280	210	210	-87
21350000	156	22	996	350	210	210	-103
21370000	153	21	996	280	210	210	-97
21390000	154	21	996	350	210	210	-94
21410000	155	21	999	350	210	140	-105
21430000	153	21	994	350	210	210	-105
21450000	156	22	999	350	210	210	-102
21460000	153	21	996	350	210	210	-93
21480000	155	22	996	350	280	210	-94

Step 4. Click on the [**Gesture Recognition**] to open the dedicated application window.

Figure 6. MEMS-Studio - Library Evaluation - Gesture Recognition

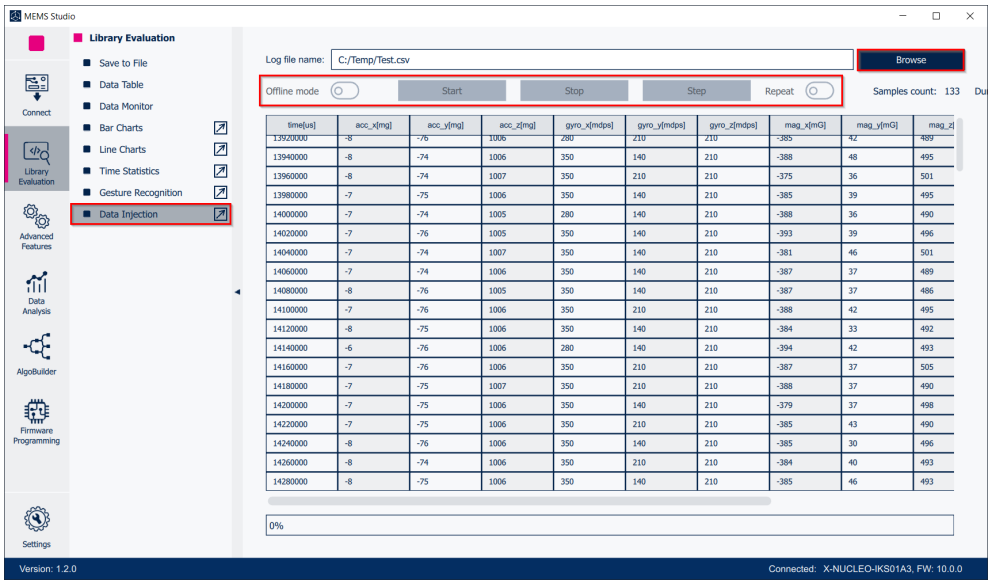
Step 5. Click on the [**Save To File**] to open the dataloging configuration window. Select the sensor and gesture recognition data to be saved in the file. You can start or stop saving by clicking on the corresponding button.

Figure 7. MEMS-Studio - Library Evaluation - Save To File



Step 6. Data Injection mode can be used to send the previously acquired data to the library and receive the result. Select the [**Data Injection**] tab on the vertical tool bar to open the dedicated view for this functionality.

Figure 8. MEMS-Studio - Library Evaluation - Data Injection



Step 7. Click on the [**Browse**] button to select the file with the previously captured data in CSV format.

The data will be loaded into the table in the current view.
Other buttons will become active. You can click on:

– [**Offline Mode**] button to switch the firmware offline mode on/off (mode utilizing the previously captured data).

– [**Start**]/[**Stop**]/[**Step**]/[**Repeat**] buttons to control the data feed from MEMS-Studio to the library.

3 References

All of the following resources are freely available on www.st.com.

1. UM1859: Getting started with the X-CUBE-MEMS1 motion MEMS and environmental sensor software expansion for STM32Cube
2. UM1724: STM32 Nucleo-64 boards (MB1136)
3. UM3233: Getting started with MEMS-Studio

Revision history

Table 4. **Document revision history**

Date	Version	Changes
06-Jun-2017	1	Initial release.
26-Jan-2018	2	Added references to NUCLEO-L152RE development board and Table 2. Elapsed time (µs) algorithm.
20-Mar-2018	3	Updated Introduction and Section 2.1 MotionGR overview.
21-Feb-2019	4	Updated Figure 1. Example of sensor orientations, Table 2. Elapsed time (µs) algorithm and Figure 3. STM32 Nucleo: LEDs, button, jumper. Added X-NUCLEO-IKS01A3 expansion board compatibility information.
24-Mar-2020	5	Updated Introduction, *Section 2.2.1 MotionGR library description* and *Section 2.2.5 Algorithm performance*. Added ARM Cortex-M7 architecture compatibility information.
17-Sep-2024	6	Updated Section Introduction, Section 2.1: MotionGR overview, Section 2.2.1: MotionGR library description, Section 2.2.2: MotionGR APIs, Section 2.2.5: Algorithm performance, Section 2.3: Sample application, Section 2.4: MEMS Studio application

Contents

List of tables

List of figures