UM2220
User manual

Getting started with MotionFX sensor fusion library in X-CUBE-MEMS1 expansion for STM32Cube

Introduction

The MotionFX is a middleware library component of the X-CUBE-MEMS1 software and runs on STM32. It provides real-time motion-sensor data fusion. It also performs gyroscope bias and magnetometer hard iron calibration.

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

It is built on top of STM32Cube software technology to ease portability across different STM32 microcontrollers.

The software comes with sample implementation running on X-NUCLEO-IKS01A1 (with optional STEVAL-MKI160V1) or X-NUCLEO-IKS01A2 expansion board on a NUCLEO-F401RE or NUCLEO-L476RG development board.

Contents

List of tables

List of figures

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 environments

2 MotionFX middleware library in X-CUBE-MEMS1 software expansion for STM32Cube

2.1 MotionFX overview

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

The library acquires data from the accelerometer, gyroscope (6-axis fusion) and magnetometer (9-axis fusion) and provides real-time motion-sensor data fusion.

The MotionFX filtering and predictive software uses advanced algorithms to intelligently integrate outputs from multiple MEMS sensors, regardless of environmental conditions, for an optimum performance.

A sample implementation is available on X-NUCLEO-IKS01A2 and X-NUCLEO-IKS01A1 (with optional STEVAL-MKI160V1) expansion boards, mounted on a NUCLEO-F401RE or NUCLEO-L476RG development board.

2.2 MotionFX library

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

2.2.1 MotionFX library description

The MotionFX sensor fusion library manages data acquired from accelerometer, gyroscope and magnetometer sensor; it features:

- real-time 9-axis motion-sensor data fusion (accelerometer, gyroscope, magnetometer)
- real-time 6-axis motion-sensor data fusion (accelerometer, gyroscope)
- computation of rotation, quaternions, gravity and linear acceleration data
- gyroscope bias calibration
- magnetometer hard iron calibration
- recommended sensor data sampling frequency of 100 Hz
- 45 kByte of code memory and 8 kByte of data memory

Real size might differ for different IDEs (toolchain)

- available for ARM Cortex-M4 architecture

2.2.2 MotionFX 6-axis and 9-axis sensor fusion modes

The MotionFX library implements a sensor fusion algorithm for the estimation of 3D orientation in space. It uses a digital filter based on the Kalman theory to fuse data from several sensors and compensate for limitations of single sensors. For instance, gyroscope data can drift and this impacts the orientation estimation; this issue can be fixed by using the magnetometer to provide absolute orientation information.

Similarly, the magnetometer does not have a very high bandwidth and suffers from magnetic disturbance, but these weaknesses can be compensated with a gyroscope.

9-axis sensor fusion uses data from the accelerometer, gyroscope and magnetometer and provides absolute orientation in 3D space including heading (i.e., the magnetic North direction).

6-axis sensor fusion uses the accelerometer and gyroscope data only. It has lower computational requirements, but does not provide information about the device absolute orientation.

6-axis sensor fusion is fit for fast movements (e.g., for gaming) and when absolute orientation is not necessary.

2.2.3 MotionFX library operation

The MotionFX library integrates 6- and 9-axis sensor fusion algorithms in one library; they can even run simultaneously.

The library implements the following critical internal functions associated with sensor fusion computation:

1. the `MotionFX_propagate` is a prediction function used to estimate the orientation in 3D space; gyroscope data is given more weight in this phase.
2. the `MotionFX_update` is the corrective function which adjusts the predicted value when necessary; accelerometer and magnetometer data are given more weight in this phase.

The `MotionFX_update` function can be called whenever the `MotionFX_propagate` is invoked, or less often in systems that have less computation power.

The `MotionFX_update` function takes approximately three times more MCU computation time than the `MotionFX_propagate` function.

2.2.4 MotionFX library parameters

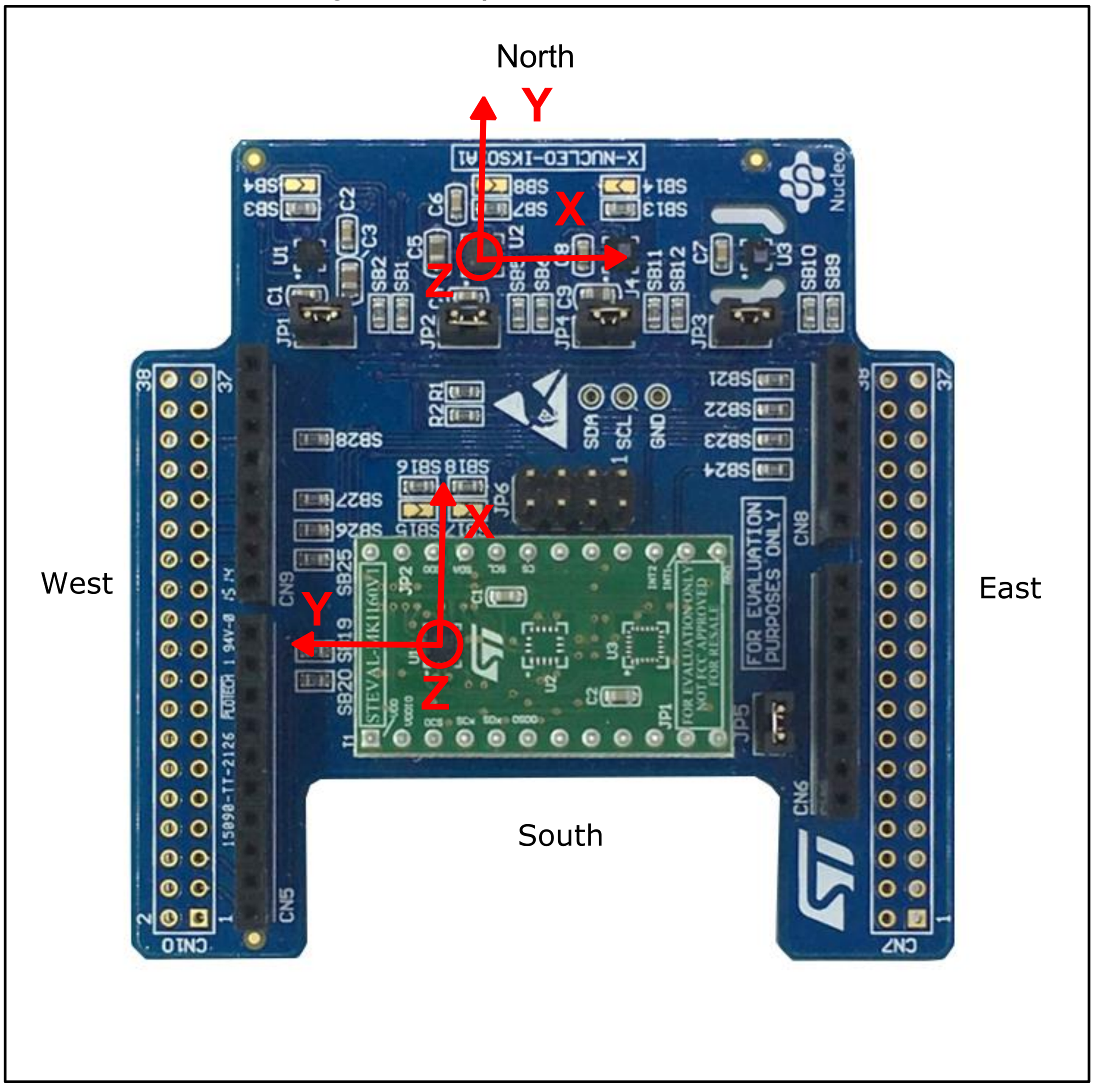
The MotionFX library is "parameterized" using an `MFX_knobs_t` structure.

The parameters for the structure are:

- `ATime`, `MTime`, `FrTime` represent the weighting stability of sensors for prediction (trust factor), from 0 to 1. Default values are recommended.
- `LMode` represents the gyroscope bias learning mode; the library automatically tracks and calibrates gyro zero-rate bias drift.
  This possible parameter values are:
  – LMode = 0 – learning off; use this mode if the gyro is already calibrated
  – LMode = 1 – static learning; learning is performed only when the system is not moving
  – LMode = 2 – dynamic learning; learning is also performed when the system is moving
- `gbias_acc/gyro/mag_th_sc_6X, gbias_acc/gyro/mag_th_sc_9X` represent the thresholds below which the gbias algorithm automatically starts. These values should be established through testing (they are different for different part numbers). The values in the example project are usually correct:
  – `modx` represents the decimation of `MotionFX_update` call frequency
  – `output_type` represents the sensor fusion library output orientation: 0 = NED, 1 = ENU
  – `start_automatic_gbias_calculation` represents a flag that restarts gyroscope bias calibration when set to 1
  – `acc/gyro/mag_orientation` is the acc/gyro/mag data orientation string of three characters indicating the direction of each positive orientation of the

reference frame used for the 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-IKS01A1 accelerometer sensor has an ENU orientation (x - East, y - North, z - Up), so the string is: “enu”, while the accelerometer sensor in STEVAL-MKI160V1 is NWU (x-North, y-West, z-Up): “nwu”.

Figure 1: Example of sensor orientations



The `MotionFX_propagate` and the `MotionFX_update` functions receive input from sensors in the `MFX_input_t` structure:

- `mag` represents magnetometer data after calibration in µT/50
- `acc` represents accelerometer data in g
- `gyro` represents gyroscope data in dps

The `MotionFX_propagate` and the `MotionFX_update` functions provide the sensor fusion output in the `MFX_output_t` structure:

- `rotation_6/9X` represents the system orientation in three-angle format: yaw, pitch and roll

- `quaternion_6/9X` represents the system orientation in four-number format; this format gives the same information as `rotation_6/9X` but it has advantages for computation and is therefore usually used by other algorithms (based on the sensor fusion)
- `gravity_6/9X` represents the static acceleration (i.e., Earth gravity) vector extracted from the acceleration data
- `linear_acceleration_6/9X` represents the dynamic acceleration (i.e., movement) vector extracted from the acceleration data.

2.2.5 MotionFX library output data rate

It is important to set up the sensor fusion library output data rate properly; 100 Hz is recommended.

The output data rate for:

- the gyroscope and the accelerometer should be equal to or greater than 100 Hz;
- the magnetometer can be lower - 20/40 Hz is typically good for a magnetic field sensor.

It is possible to scale the library system requirements in terms of MCU/MPU load. As the `MotionFX_update` function requires approximately three times more computation power than the `MotionFX_propagate` function, it can be called at a lower frequency than the library output data rate if the system resources are limited (e.g., in embedded systems).

Use the `modx` parameter in `MFX_knobs_t` structure to decrease the frequency of `MotionFX_update` function calls. For example, setting `modx` to 2 calls the `MotionFX_update` function once every two `MotionFX_propagate` function calls.

The recommended settings are:

- `modx` = 1, for tablets or other systems with MCU/MPU and for STM32F4;
- `modx` = 2, for STM32F1.

2.2.6 Sensor calibration in the MotionFX library

Gyroscope and accelerometer calibration

Accelerometer calibration is not necessary for sensor fusion except for applications demanding very high orientation precision; it involves aligning the system in several positions according to the gravity direction.

Gyroscope calibration is handled automatically by the MotionFX library by continuously compensating the zero-rate offset effect.

Magnometer calibration

The MotionFX library contains routines for magnetometer hard iron calibration.

The magnetometer is affected by the hard-iron and soft-iron phenomena described below.

Hard-iron distortion

Hard-iron distortion is normally generated by ferromagnetic material with permanent magnetic fields that are part of the object (e.g., a tablet) in use. These materials can be permanent magnets or magnetized iron or steel. They are time invariant and deform the local geomagnetic field with different offsets in different directions.

As each board can be magnetized differently, the hard iron effect is specific to the individual board.

If you move the board around a space approximating (as much as possible) a 3D sphere in an ideal environment (no hard-iron/soft-iron distortion) and plot the collected magnetic sensor raw data, the result is a perfect sphere with no offset.

The hard-iron distortion effect offsets the sphere along the x, y and z axes; in the x-y plane, the hard-iron distortion is identified by an offset of the origin of the ideal circle from (0, 0), scatter plots for XY and XZ axes are sufficient to determine if there is an offset.

Soft-iron distortion

Soft-iron distortion is generated by magnetically soft materials or current carrying PCB traces. While hard-iron distortion is constant regardless of the orientation, the soft-iron distortion changes with the orientation of the object in the Earth's field (soft magnetic materials change their magnetization direction).

The local geomagnetic field is deformed with different gain on different directions. The effect of the soft-iron to distort the ideal full round sphere into a tilted ellipsoid; in the x-y plane, the soft-iron distortion is identified by a tilted ellipse with the origin at (0,0) for the XY axis (XZ).

The soft iron effect is the same across all boards of the same design, which is why a good PCB design which takes magnetometer placement (high current traces/other component clearance) into account can generally avoid any soft iron effects (valid for X-NUCLEO-IKS01A1 and X-NUCLEO-IKS01A2 expansion boards).

Calibration procedure

The MotionFX library magnetometer calibration library compensates for hard-iron distortions.

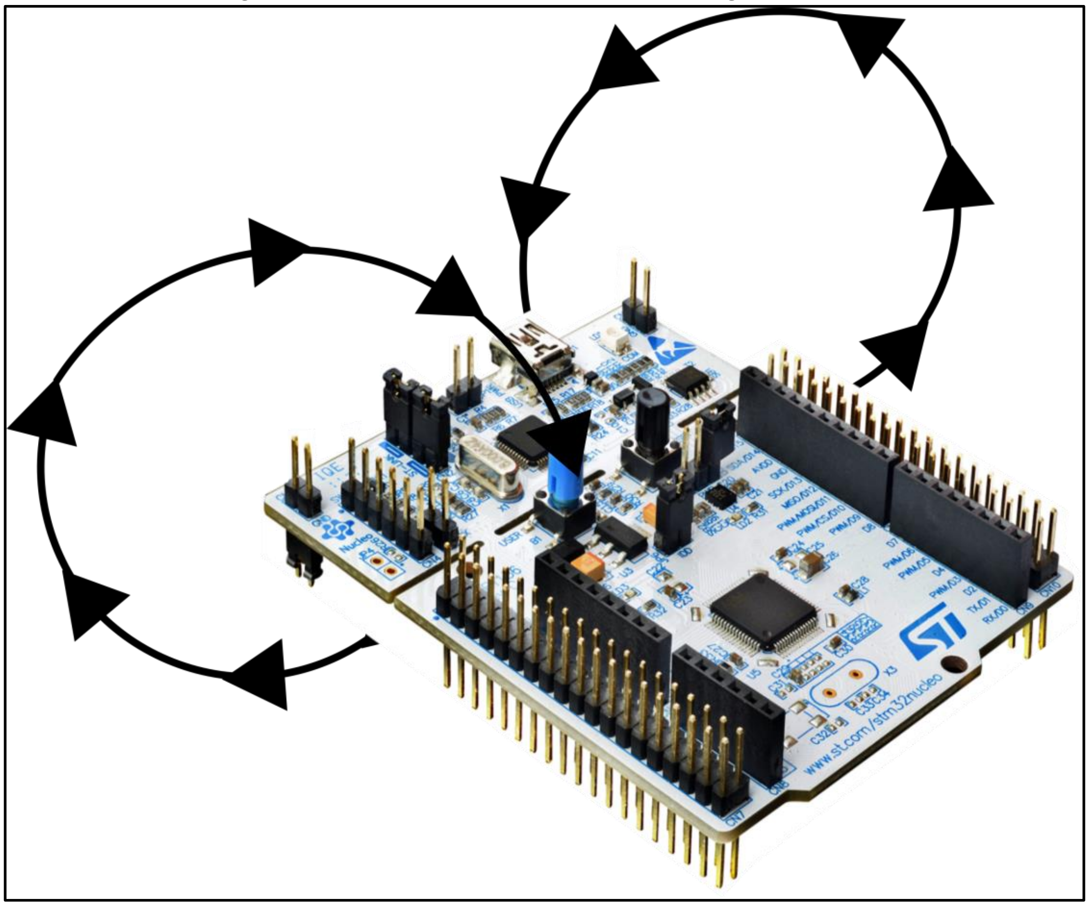
The magnetometer calibration can be performed at a slower frequency than the sensor fusion output data rate (e.g., 25 Hz).

To run the calibration:

1. initialize magnetometer calibration library (`MotionFX_MagCal_init`)
2. call periodically calibration function (`MotionFX_MagCal_run`) until the calibration is successfully performed
3. check if calibration was successful (`MotionFX_MagCal_getParams`); if the function returns `mag_data_out.cal_quality = MFX_MAGCALGOOD`, the calibration was successfully performed
4. apply calibration results:
   – − MAG_Calibrated.AXIS_X = MAG_Value.AXIS_X - MAG_Offset.AXIS_X
   – − MAG_Calibrated.AXIS_Y = MAG_Value.AXIS_Y - MAG_Offset.AXIS_Y
   – − MAG_Calibrated.AXIS_Z = MAG_Value.AXIS_Z - MAG_Offset.AXIS_Z

After calibration routine initialization, slowly rotate the device in a figure 8 pattern in space. While performing this movement, keep the device clear of other magnetic objects such as cell phones, computers and other steel objects.

Figure 2: STM32 Nucleo board rotation during calibration



To check that the calibration was performed correctly, check magnetometer data (after applying calibration results) using the scatter plot.

2.2.7 MotionFX APIs

The MotionFX APIs are:

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

This function must be called before using the sensor fusion library.

- `voidMotionFX_setKnobs(MFX_knobs_t *knobs)`
  - sets the internal knobs
  - `*knobs` parameter is a pointer to a structure with knobs

- `void MotionFX_getKnobs(MFX_knobs_t *knobs)`
  - gets the current internal knobs
  - `*knobs` parameter is a pointer to a structure with knobs

- `MFX_engine_state_t MotionFX_getStatus_6X(void)`
  - gets the 6 axes library status
  - returns 1 if enabled, 0 if disabled

- `MFX_engine_state_t MotionFX_getStatus_9X(void)`
  - gets the 9 axes library status
  - returns 1 if enabled, 0 if disabled

- `void MotionFX_enable_6X(MFX_engine_state_t enable)`
  - enables or disables the 6 axes function (ACC + GYRO)
  - `enable` parameter is 1 to enable, 0 to disable

- `void MotionFX_enable_9X(MFX_engine_state_t enable)`
  - enables or disables the 9 axes function (ACC + GYRO+MAG)
  - `enable` parameter is 1 to enable, 0 to disable

- `void MotionFX_setGbias(float *gbias)`
  - sets the initial gbias
  - `*gbias` parameter is a pointer to a float array containing the gyro bias value for each axis

- `void MotionFX_getGbias(float *gbias)`
  - gets the initial gbias
  - `*gbias` parameter is a pointer to a float array containing the gyro bias value for each axis

- `void MotionFX_update(MFX_output_t *data_out, MFX_input_t *data_in, float eml_deltatime, float *eml_q_update)`
  - runs the Kalman filter update
  - `*data_out` parameter is a pointer to output data structure
  - `*data_in` parameter is a pointer to input data structure
  - `eml_deltatime` parameter is a delta time between two propagate calls [sec]
  - `*eml_q_update` parameter is a pointer set to NULL

- `void MotionFX_propagate(MFX_output_t *data_out, MFX_input_t *data_in, float eml_deltatime)`
  - runs the Kalman filter propagate
  - `*data_out` parameter is a pointer to output data structure
  - `*data_in` parameter is a pointer to input data structure
  - `eml_deltatime` parameter is a delta time between two propagate calls [sec]

- `void MotionFX_MagCal_init(int sampletime, unsigned short int enable)`
  - initializes the magnetometer calibration library
  - `sampletime` parameter is a period in milliseconds [ms] between the update function calls
  - `enable` parameter is to enable (1) or disable (0) library

- `void MotionFX_MagCal_run(MFX_MagCal_input_t *data_in)`
  – runs the magnetometer calibration algorithm
  – `*data_in` parameter is a pointer to input data structure
  – the parameters for the structure type `MFX_MagCal_input_t` are:
    – `mag` is uncalibrated magnetometer data [µT]/50
    – `time_stamp` is the timestamp [ms]

- `void MotionFX_MagCal_getParams(MFX_MagCal_output_t *data_out)`
  – gets magnetometer calibration parameters
  – `*data_out` parameter is a pointer to output data structure
  – the parameters for the structure type `MFX_MagCal_output_t` are:
    – `hi_bias` is the hard iron offset array [µT]/50
    – `cal_quality` is the calibration quality factor:
      – `MFX_MAGCALUNKNOWN` = 0; accuracy of the calibration parameters is unknown
      – `MFX_MAGCALPOOR` = 1; accuracy of the calibration parameters is poor, cannot be trusted
      – `MFX_MAGCALOK` = 2; accuracy of the calibration parameters is OK
      – `MFX_MAGCALGOOD` = 3; accuracy of the calibration parameters is good

Storing and loading magnetometer calibration parameters

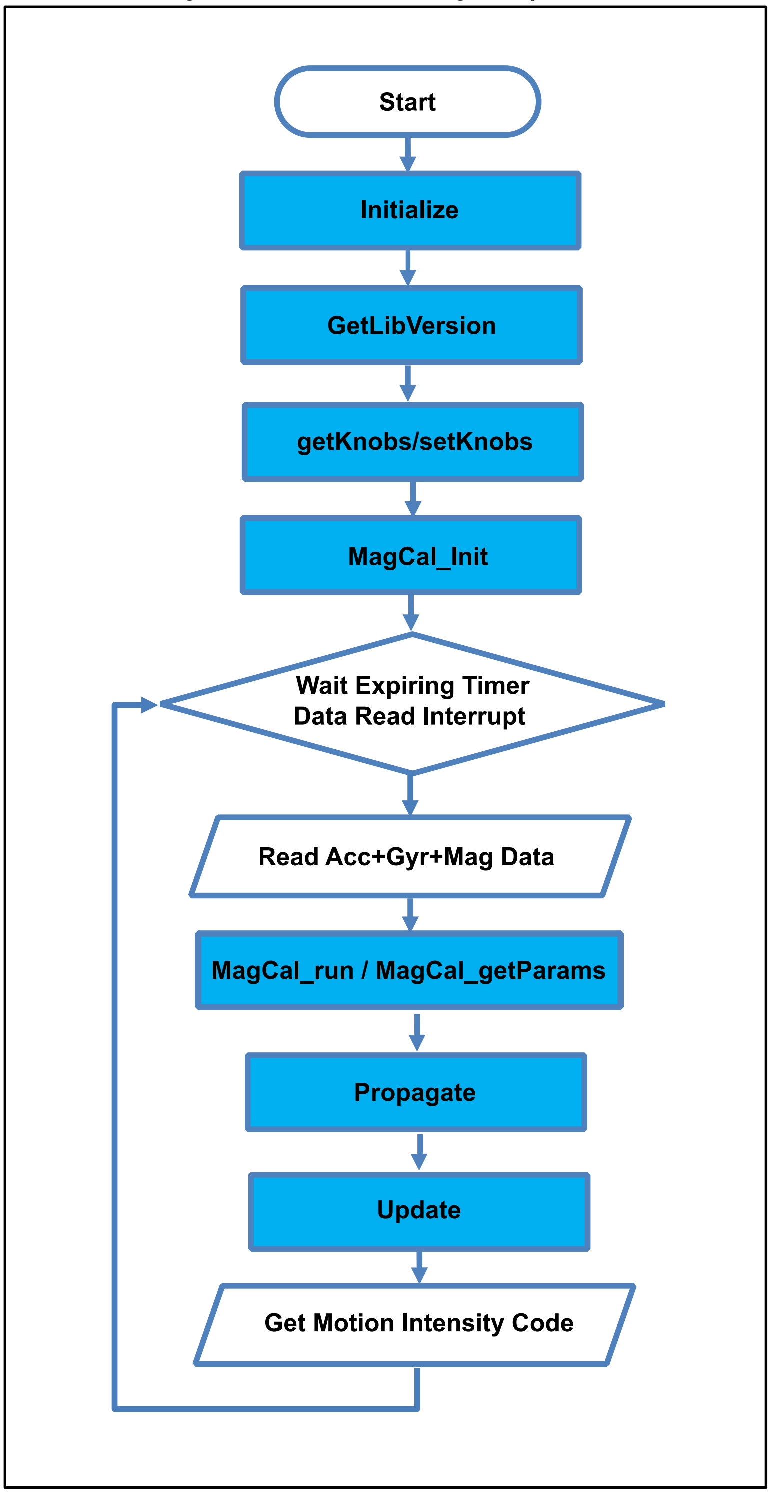
The following functions have to be implemented specifically for each target platform:

- `char MotionFX_LoadMagCalFromNVM(unsigned short int dataSize, unsigned int *data)`
  – the function is used to retrieve the calibration parameters from storage, the function is called when magnetometer calibration library is enabled
  – `dataSize` is the size of the data in bytes
  – `*data` is the data location pointer
  – returns 0 for correct loading, 1 otherwise

- `char MotionFX_SaveMagCalInNVM(unsigned short int dataSize, unsigned int *data)`
  – the function is used to store the calibration parameters and is called when the magnetometer calibration library is disabled
  – `dataSize` is the size of the data in bytes
  – `*data` is the data location pointer
  – returns 0 for correct saving, 1 otherwise

2.2.8 API flow chart

Figure 3: MotionFX API logic sequence



2.2.9 Demo code

The following demonstration code reads data from the accelerometer, gyroscope and magnetometer sensors and gets the rotation, quaternions, gravity and linear acceleration.

```
[…]
#define VERSION_STR_LENG 35
#define MFX_DELTATIME 10
[…]

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

/* Sensor Fusion API initialization function */
MotionFX_initialize();

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

MotionFX_getKnobs(&iKnobs);
/* Modify knobs settings */
MotionFX_setKnobs(&iKnobs);

/* Enable 9-axis sensor fusion */
MotionFX_enable_9X(MFX_ENGINE_ENABLE);

[…]

/*** Using Sensor Fusion algorithm ***/
Timer_OR_DataRate_Interrupt_Handler()
{
MFX_input_t data_in;
MFX_output_t data_out;

/* Get acceleration X/Y/Z in g */
MEMS_Read_AccValue(data_in.acc[0], data_in.acc[1], data_in.acc[2]);

/* Get angular rate X/Y/Z in dps */
MEMS_Read_GyroValue(data_in.gyro[0], data_in.gyro[1], data_in.gyro[2]);

/* Get magnetic field X/Y/Z in uT/50 */
MEMS_Read_MagValue(data_in.mag[0], data_in.mag[1], &data_in.mag[2]);

/* Run Sensor Fusion algorithm */
MotionFX_propagate(&data_in, &data_out, MFX_DELTATIME);
MotionFX_update(&data_in, &data_out, MFX_DELTATIME, NULL);
}
```

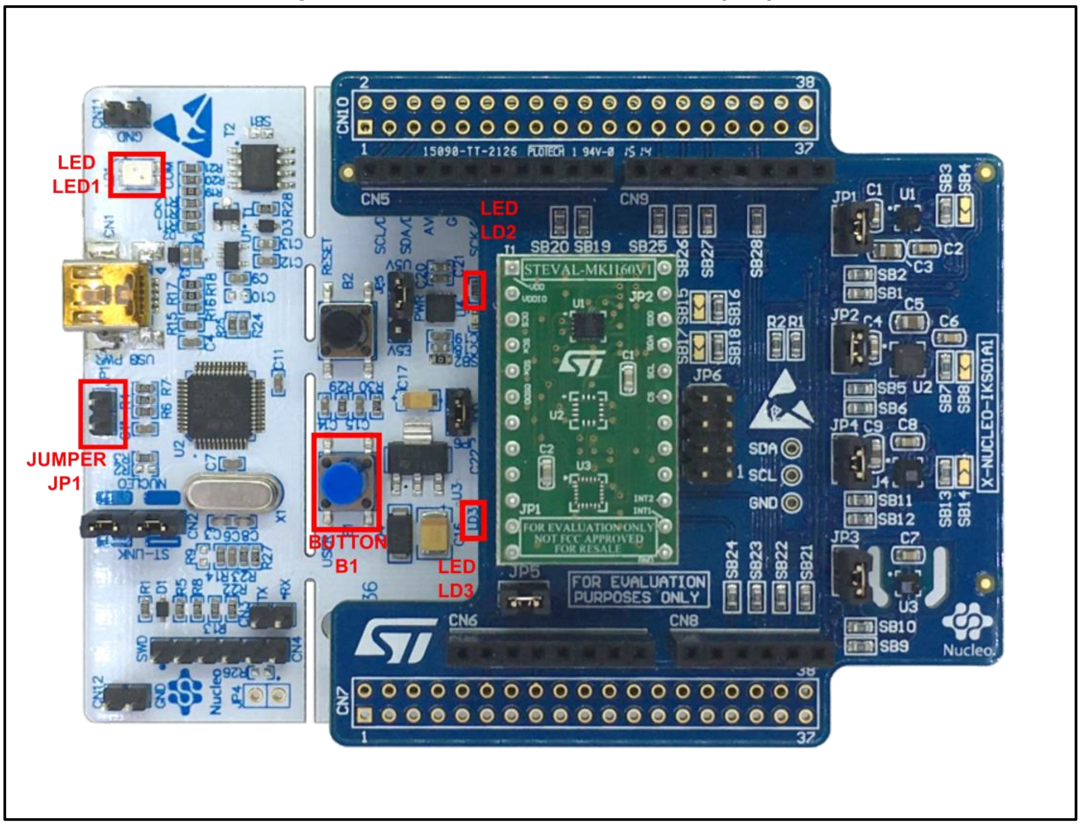
3 Sample application

The MotionFX 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 or a NUCLEO-L476RG development board connected to an X-NUCLEO-IKS01A1 (based on LSM6DS0) or an X-NUCLEO-IKS01A2 (based on LSM6DSL) expansion board, with optional STEVAL-MKI160V1 board (based on LSM6DS3).

The application provides real-time motion-sensor data fusion and returns rotation, quaternions, gravity and linear acceleration.

Figure 4: 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.

After powering the board, LED LD2 blinks once indicating the application is ready.

Initially, the magnetometer calibration data are loaded from the MCU internal flash memory and checked for data validation and good calibration quality.

If the data are valid and the calibration quality is good the LED2 is switched ON, if not the magnetometer needs calibration and LED2 is turned OFF; in this case the calibration routine is initialized.

To calibrate the magnetometer, perform the figure 8 calibration movement.

You can calibrate the magnetometer only when the sensor fusion is activated.

When user button B1 is pressed, the system clears old magnetometer calibration data stored in the flash memory and starts the calibration routine again.

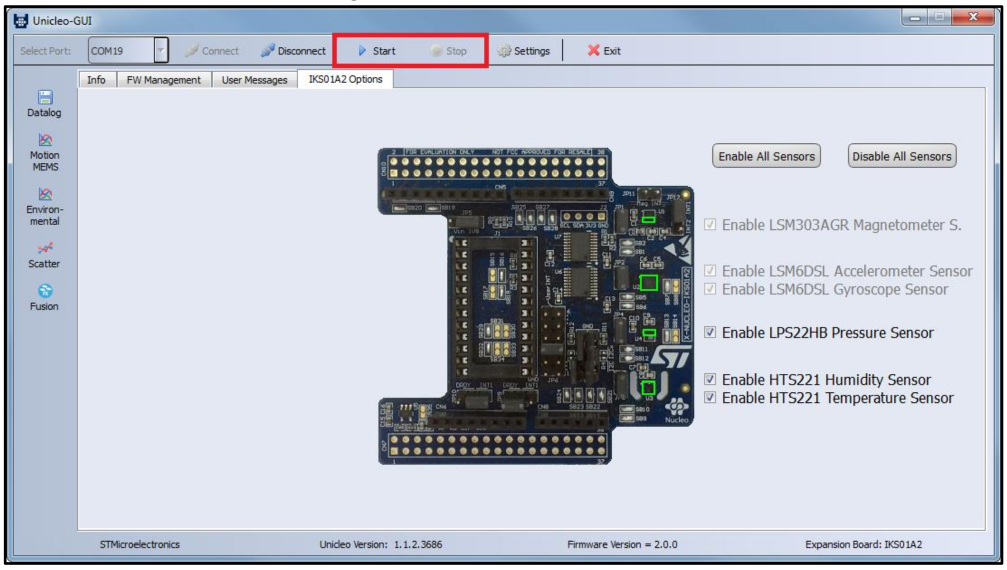
As soon as the magnetometer calibration finishes, after acquiring enough data, LED2 turns ON indicating that the calibration quality is good and calibration data are stored in the flash memory.

3.1 Unicleo-GUI application

The sample application uses the Windows Unicleo-GUI utility, which can be downloaded from *www.st.com*.

1. Ensure that the necessary drivers are installed and the STM32 Nucleo board with appropriate expansion board is connected to the PC.
2. Launch the Unicleo-GUI application to open the main application window.
   If an STM32 Nucleo board with supported firmware is connected to the PC, it is automatically detected and the appropriate COM port is opened.

Figure 5: Unicleo main window



Select Port: COM19

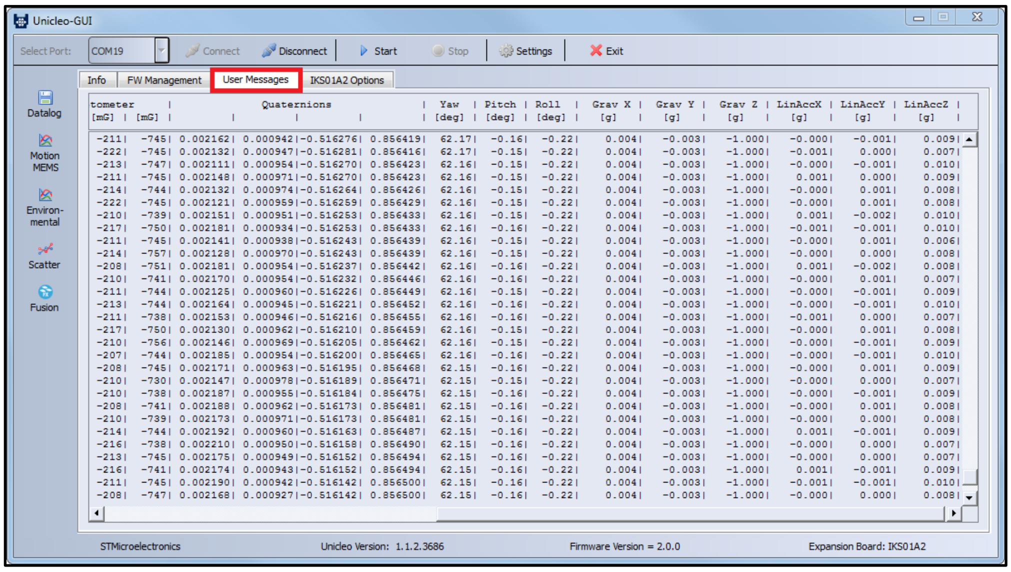
☑ Enable LSM303AGR Magnetometer S.
☑ Enable LSM6DSL Accelerometer Sensor
☑ Enable LSM6DSL Gyroscope Sensor
☑ Enable LPS22HB Pressure Sensor
☑ Enable HTS221 Humidity Sensor
☑ Enable HTS221 Temperature Sensor

STMicroelectronics Unicleo Version: 1.1.2.3686 Firmware Version = 2.0.0 Expansion Board: IKS01A2

3 Start and stop data streaming by using the appropriate buttons on the vertical tool bar.

The data coming from the connected sensor can be viewed in the User Messages tab.

Figure 6: User Messages tab



```
tometer     |              Quaternions                    |  Yaw  | Pitch | Roll  |  Grav X |  Grav Y |  Grav Z | LinAccX | LinAccY | LinAccZ |
 [mG] | [mG] |          |          |          |          | [deg] | [deg] | [deg] |   [g]   |   [g]   |   [g]   |   [g]   |   [g]   |   [g]   |
  -211|  -745| 0.002162| 0.000942|-0.516276| 0.856419|  62.17|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.009|
  -222|  -745| 0.002132| 0.000947|-0.516281| 0.856416|  62.17|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.001|    0.000|    0.007|
  -213|  -747| 0.002111| 0.000954|-0.516270| 0.856423|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.010|
  -211|  -745| 0.002148| 0.000971|-0.516270| 0.856423|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|    0.001|    0.000|    0.009|
  -214|  -744| 0.002132| 0.000974|-0.516264| 0.856426|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.008|
  -222|  -745| 0.002121| 0.000959|-0.516259| 0.856429|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.008|
  -210|  -739| 0.002151| 0.000951|-0.516253| 0.856433|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|    0.001|   -0.002|    0.010|
  -217|  -750| 0.002181| 0.000934|-0.516253| 0.856433|  62.16|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.001|   -0.001|    0.010|
  -211|  -745| 0.002141| 0.000938|-0.516243| 0.856439|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.006|
  -214|  -757| 0.002128| 0.000970|-0.516243| 0.856439|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.000|    0.008|
  -208|  -751| 0.002181| 0.000954|-0.516237| 0.856442|  62.16|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|    0.001|   -0.002|    0.008|
  -210|  -741| 0.002170| 0.000954|-0.516232| 0.856446|  62.16|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.007|
  -211|  -744| 0.002125| 0.000960|-0.516226| 0.856449|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.009|
  -213|  -744| 0.002164| 0.000945|-0.516221| 0.856452|  62.16|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.010|
  -211|  -738| 0.002153| 0.000946|-0.516216| 0.856455|  62.16|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.001|    0.000|    0.007|
  -217|  -750| 0.002130| 0.000962|-0.516210| 0.856459|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.008|
  -210|  -756| 0.002146| 0.000969|-0.516205| 0.856462|  62.16|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.009|
  -207|  -744| 0.002185| 0.000954|-0.516200| 0.856465|  62.16|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.010|
  -208|  -745| 0.002171| 0.000963|-0.516195| 0.856468|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.009|
  -210|  -730| 0.002147| 0.000978|-0.516189| 0.856471|  62.15|  -0.15|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.000|    0.007|
  -210|  -738| 0.002187| 0.000955|-0.516184| 0.856475|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|   -0.001|    0.009|
  -208|  -741| 0.002188| 0.000962|-0.516173| 0.856481|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.001|    0.008|
  -210|  -739| 0.002173| 0.000971|-0.516173| 0.856481|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.000|    0.008|
  -214|  -744| 0.002192| 0.000960|-0.516163| 0.856487|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|    0.001|   -0.001|    0.009|
  -216|  -738| 0.002210| 0.000950|-0.516158| 0.856490|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.000|    0.007|
  -213|  -745| 0.002175| 0.000949|-0.516152| 0.856494|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.000|    0.007|
  -216|  -741| 0.002174| 0.000943|-0.516152| 0.856494|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|    0.001|   -0.001|    0.009|
  -211|  -745| 0.002190| 0.000942|-0.516142| 0.856500|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.001|   -0.001|    0.010|
  -208|  -747| 0.002168| 0.000927|-0.516142| 0.856500|  62.15|  -0.16|  -0.22|    0.004|   -0.003|   -1.000|   -0.000|    0.000|    0.008|
```

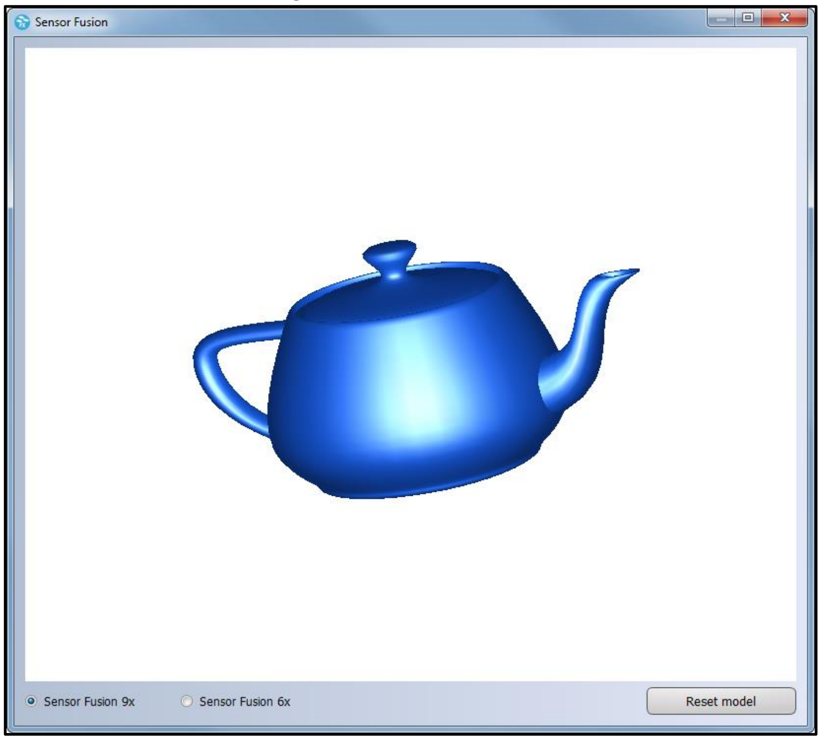
STMicroelectronics Unicleo Version: 1.1.2.3686 Firmware Version = 2.0.0 Expansion Board: IKS01A2

4 Click on the **Fusion** icon in the vertical toolbar to open the dedicated application window.

To switch between 9-axes and 6-axes sensor fusion click on the appropriate button.

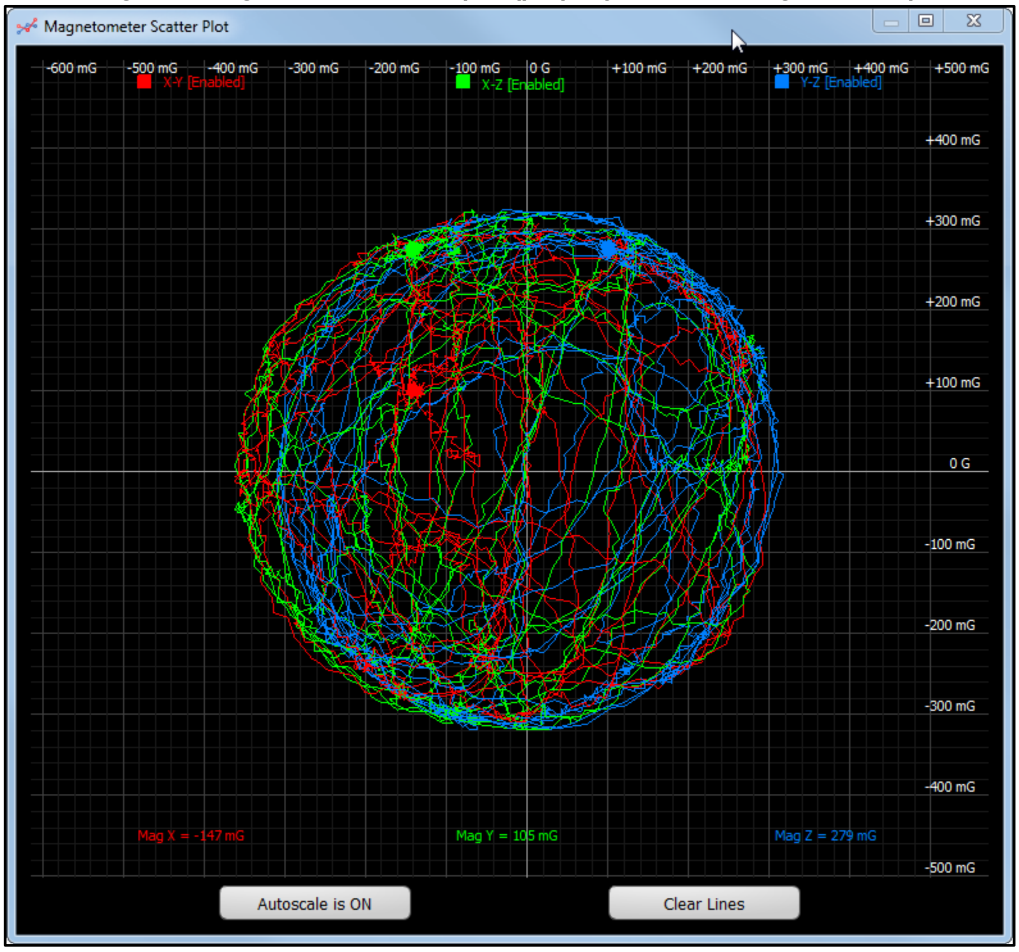
To align the tea pot position point the Nucleo board towards the screen and press **Reset** model button.

Figure 7: Fusion window



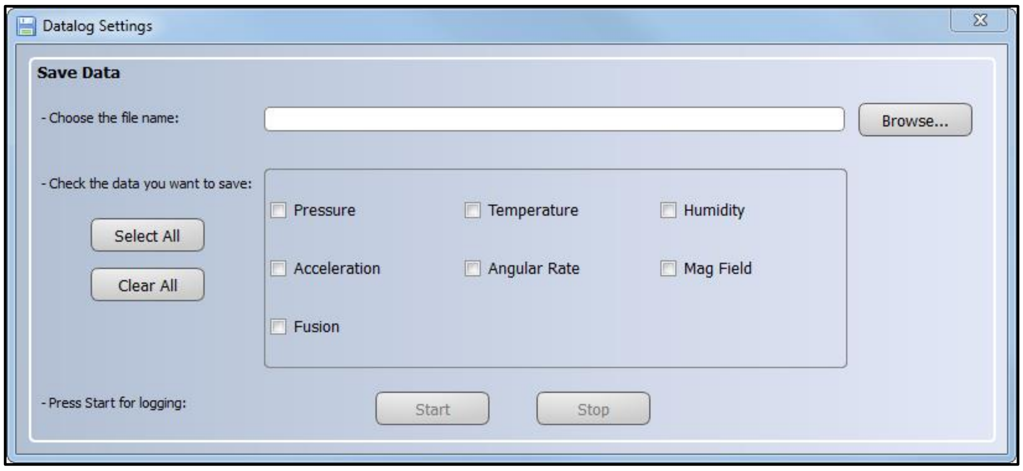
5 Click on the **Scatter Plot** icon to check the magnetometer calibration quality.

Figure 8: Magnetometer scatter plot (properly calibrated magnetometer)



6 Click on the **Datalog** icon in the vertical toolbar to open the datalog configuration window: you can select the sensor and fusion data to be saved in the files. You can start or stop saving by clicking on the corresponding button.

Figure 9: Datalog window



4 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 board
3. UM2128: Getting started with Unicleo-GUI for motion MEMS and environmental sensor software expansion for STM32Cube

5 Revision history

Table 2: Document revision history

Date	Version	Changes
12-May-2017	1	Initial release.