navX2-Micro Robotics Navigation Sensor User Guide

Scott Libert

Table of Contents

Overview
navX2 Micro

navX2-Micro is a second-generation **9-axis inertial/magnetic sensor** and **motion processor**. Designed for **plug-n-play** installation onto robots such as those used in the FIRST Technology Challenge (FTC) and the FIRST Robotics Challenge (FRC), navX2-Micro helps **build better robots** by adding sophisticated navigation capabilities.

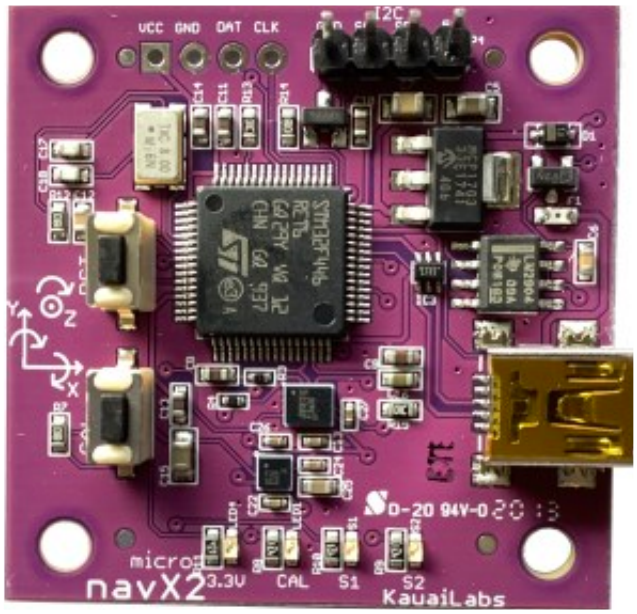
"Generation 2" navX2-Micro is a drop-in replacement for "Classic" navX-Micro. See the Frequently Asked Questions (FAQ) ***for more information about navX2-Micro improvements.***

navX2-Micro is a must-have add on to any robot control system, and includes free software libraries, example code and many more features.

navx2-Micro also works with the Kauai Labs Sensor Fusion Framework (SF2) to provide even more advanced capabilities.

Super-charge your robot:

- Field-Oriented Drive
- Auto-balance
- Auto-rotate to angle
- Motion Detection
- Collision Detection
- and more…



Features

Sophisticated Motion Processing

- High Accuracy, Low-latency Yaw, Pitch and Roll Angles
- Automatic Accelerometer/Gyroscope Calibration
- Quaternions and Gravity-corrected Linear Acceleration
- High-sensitivity Motion Detection
- Tilt-compensated Compass Heading
- 9-Axis absolute heading w/Magnetic disturbance detection

Inertial/Magnetic Measurement Unit

- Configurable Update Rate from 4 to 200 Hz
- Access to Raw Data from Gyroscopes, Accelerometers and Magnetometers

Easy to Use

- Plug-n-Play connection on FTC and FRC robots via I2C interface
- 4-pin I2C Connector cable included
- Connection to Windows PC via USB Cable
- FTC, FRC, Linux and Android Software libraries and sample code
- Support for Mounting in horizontal, vertical and even upside-down configurations via Omnimount
- Tools for Magnetometer Calibration

Enclosure

- A custom navX2-Micro enclosure can be created with a 3D printer using provided Enclosure design files
- Alternatively, the navX2-Micro enclosure can be purchased directly from the Kauai Labs store, or custom-printed by Shapeways.

Technical Specifications

The navX2-Micro circuit board and official firmware provide inertial and magnetic measurements, with a range, accuracy and update rate as described on this page.

Note that certain performance specifications are only valid after a start-up Gyroscope/Accelerometer Calibration *period, during which time the navX2-Micro circuit board must be held still.*

Additional details can be found in the navX2-Micro datasheet.

Electrical Specifications	
Voltage:	5V DC
Current Consumption:	50 millamps
Communications Interface:	USB, I2C
Power Connector:	USB, I2C
USB Connector:	USB Mini-B

Frequently Asked Questions

How does navX2-Micro improve upon the "Classic" navX-Micro sensor?

- navX2-Micro replaces the older MPU-9250 9-axis IMU with the newer 6-axis LSM6DSM IMU and LIS2MDL Magnetometer from ST Microelectronics; these modern sensors feature lower noise, higher accuracy and improved shock resistance than the Invensense MPU-9250 found in the "Classic" navX-Micro sensor.
- navX2-Micro features an upgraded onboard microcontroller which doubles the speed of that found on the "Classic" navX-Micro.
- navX2-Micro uses a new Kalman Filter-based algorithm with improved accuracy and running at a much higher 415Hz update rate.
- As a result of these enhancements, yaw drift is now much lower, startup time is reduced to only 5 seconds, and pitch/roll accuracy is also increased.

Is navX2-Micro compatible with the "Classic" navX-Micro sensor?

- navX2-Micro is a "drop-in replacement" for navX-Micro, featuring the exact same board footprint, electrical interface and software interface (via USB and I2C) as navX-Micro.

What interface/installation options are available for navX2-Micro?

- Connection to a Android-based FTC Robotics Control System via I2C.
- Connection to a RoboRIO-based FRC Robotics Control System via USB and I2C.
- Connection to a Windows-based PC via a USB Cable

Can navX2-Micro be used with the REV Expansion Hub or Control Hub for FTC?

Yes, navX2-Micro's I2C interface is compatible with the REV Expansion Hub and the REV Control Hub. Please see the instructions for installing the navX2-Micro on a FTC Control System .

Can navX2-Micro be used with the National Instruments RoboRIO™?

Yes, navX2-Micro USB and I2C interfaces are compatible with the navX-MXP, and can interface to the RoboRIO via the USB or I2C Ports. Please see the instructions for installing the navX2-Micro on a RoboRIO for more details.

Aren't the magnetometer (compass heading) readings unreliable when the navX2-Micro is used on a Robot with powerful motors?

Yes, this is correct. If the navX2-Micro is mounted nearby any energized motors, the magnetometer's ability to measure the (weak) earth's magnetic field is severely diminished. For this reason, most robotics teams avoid using navX2-Micro's magnetometer data. ***Magnetometer Calibration is not typically required for use in many FIRST FTC and FRC robot applications, including Field-oriented drive. Magnetometer Calibration is a manual process and is recommended for advanced users who need to calculate absolute heading.***

However, at the beginning of each match, the robot will be powered on for about a minute before the match begins. During this time period, the motors are not energized and thus do not add magnetic interference that would disturb the magnetometer readings. Once the magnetometer is calibrated, the navX2-Micro will return either an accurate magnetometer reading, or an indication that its measurement of the earth's magnetic field has been disturbed.

Magnetometer readings taken at the beginning of a match, when combined with the navX2-Micro yaw measurements, enable a robot's pose and absolute heading to be maintained throughout the match. This feature of the navX2-Micro is referred to as a "9-axis" heading.

Why do the Yaw angles provided by the navX2-Micro drift over time?

The short answer is that the yaw angle is calculated by integrating reading from a gyroscope which measures changes in rotation, rather than absolute angles. Over time, small errors in the rotation measurements build up over time. The navX2-Micro features sophisticated digital motion processing and calibration algorithms that limit this error in the yaw angle of less than ~1 degree per minute; and when the sensor is still, yaw drift is an amazingly low .1 degree per minute. For further details, please see the Yaw Drift page.

Can the navX2-Micro "Displacement" estimates be used for tracking a FTC or FRC robot's change in position (dead-reckoning) during autonomous?

Accelerometer data from the navX2-Micros's onboard sensors are double-integrated by the navX2-Micro firmware to estimate displacement, *and are accurate to approximately 1 meter of error during a 15 second period*.

To track a FTC or FRC robot's position during autonomous requires an accuracy of about 1 cm of error per 15 seconds. While the accuracy of the navX2-Micro displacement estimates might be good enough to track the position of an automobile on a road, it is too low for use in tracking a FRC or FTC robot's position during the 15 second autonomous period.

The root cause of the displacement estimate error rate is *accelerometer noise*. Estimating displacement requires first that each acceleration sample be multiple by itself twice (cubed), and then integrated over time. Practically, if a noisy signal is cubed, the result is very noisy, and when this very noisy value is integrated over time, the total amount of error grows very quickly.

The current noise levels (approximately 60 micro-g per square-root-hertz) would need to be reduced by a factor of 10 (one orders of magnitude) before displacement estimates with 1 cm of error per 15 seconds can be achieved by double-integration of accelerometers.

MEMS accelerometers which feature these low noise levels are beginning to emerge, but are currently very expensive. KauaiLabs is actively researching these technology developments. KauaiLabs plans to develop a product which can be used for accelerometer-based dead-reckoning at that time.

What's the difference between navX2-Micro and navX-MXP?

- navX-MXP was designed from the ground-up to interface easily with the RoboRIO,

making it simpler and less-expensive to connect to the robot controller. The navX2-Micro is a smaller version of the navX-MXP which is designed for use in robots such as those found in FIRST FTC robotics.
- navX-MXP and navX2-Micro both feature a powerful microcontroller, enabling sophisticated calibration algorithms, which together with the improved gyro noise characteristics mentioned above decreases the yaw drift by a factor of at least 2, enables the initial startup calibration to occur more quickly, adds support for soft-iron magnetometer calibration.

Installation

Installation

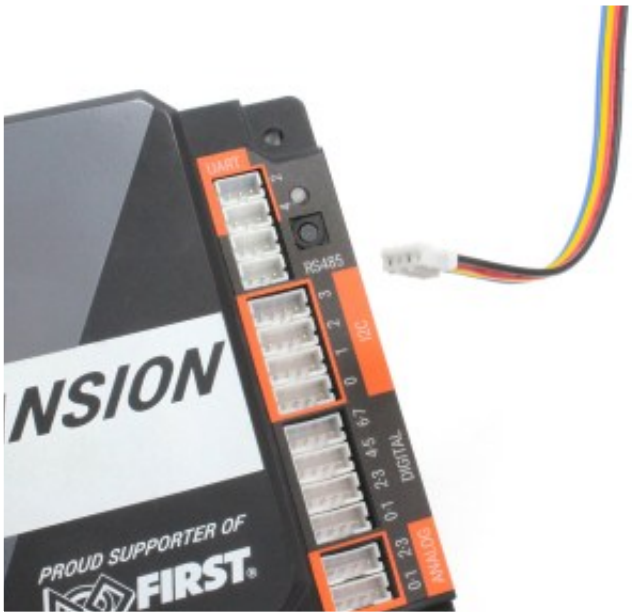
Plug-n-play: navX2-Micro is designed for rapid, plug-n-play installation on a robot, making it easy to install and integrate onto robots including FIRST FTC and FRC Robots.

Orientation: Tips and tricks for ensuring navX2-Micro measurements are aligned with your robot, including the Omnimount flexible mounting feature.

Enclosure: To protect an installed navX2-Micro, an enclosure is available – which can be either purchased, or printed on a 3D printer using open-source design files.

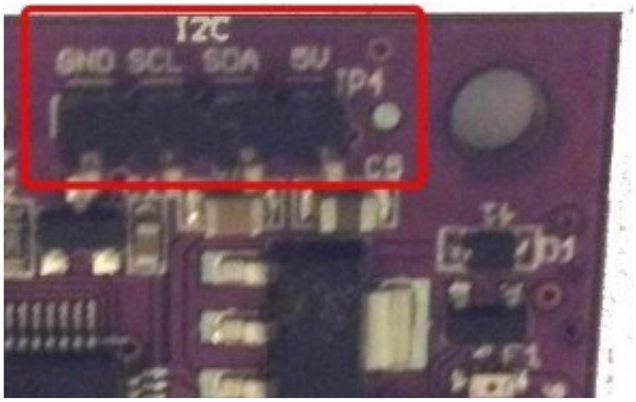
FTC Robot Installation

The navX2-Micro can be easily used with the FTC Android-Based Robot Control System. Both power to and signaling to/from the navX2-Micro occurs via the I2C interface by way of either a Expansion Hub or a Control Hub from REV Robotics.



Electrical Wiring Instructions

- Select one of the 4 I2C ports on the Hub, as shown below. Note that the ports are numbered from 0.
- Using a I2C to JST PH Cable, connect the +5V, Data (SDA), Clock (SCL) and GND pins on the selected DIM I2C port to the corresponding pins on the navX MXP External I2C Port Connector.



NOTE: Although the navX2-Micro indicates “5V” above the “power” pin on the circuit board, navX2-Micro is fully 3.3V compatible, and thus may be powered via the Expansion or Control Hub.

Electrical Wiring Verification

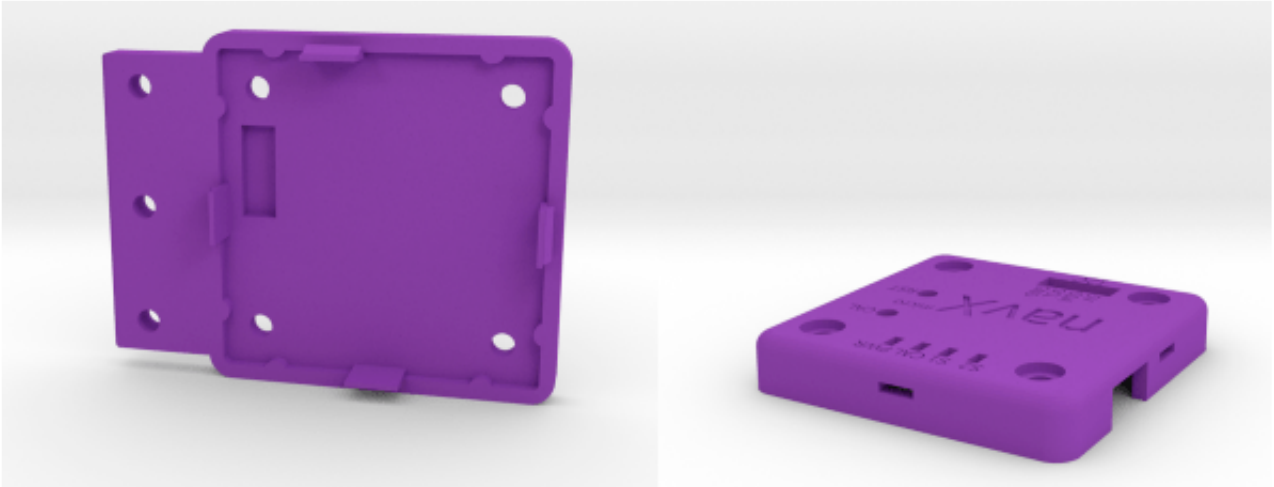
If properly wired, when power is applied to the Expansion or Control Hub, the Red 3.3V LED on the navX2 Micro should light up.

If trouble occurs communication with the navX2-Micro, double-check that the SDA and the SCL wires on the Expansion Control Hub match the corresponding pins on the navX MXP.

Physical Installation on the Robot

Enclosure

Kauai Labs highly recommends mounting the navX2-Micro circuit board to the robot chassis using the navX2-Micro enclosure.



Securing navX2-Micro to the robot chassis

navX-Micro should be mounted such that it is firmly attached to the robot chassis. ***The quality of this mounting will be directly reflected in the quality of navX2-Micro inertial measurements.*** To ensure quality, carefully follow these guidelines:

- navX2-Micro should be tightly mounted to the robot chassis; it should be a part of the chassis mass, and should move exactly as the chassis moves. Avoid mounting navX2-Micro in an area of the chassis that might be flexible, as this could introduce vibration to the inertial sensors that does not represent the chassis inertial properties.
- navX2-Micro should be mounted in the center of the chassis if at all possible, which ensures the origin of the yaw/pitch/roll axes truly represent the chassis center.
- Be sure to understand the orientation of the navX2-Micro circuit board, relative to the chassis, and decide whether OmniMount is needed.

- Housing the navX2-Micro circuit board in some form of protective enclosure is highly recommended, to protect it from damage. This should both protect the circuit board from damage.

One-wire Connect to a PC via USB cable

By using a USB Mini-B type (Male) to USB A type (Male) connector, the navX2-Micro will receive both power and also communicate with a Windows-based PC.

RoboRIO Installation

navX-Micro may be connected to the RoboRIO via several methods. Once you have selected your preferred communications interface, follow the appropriate installation instructions below.

One-wire Connect via USB cable

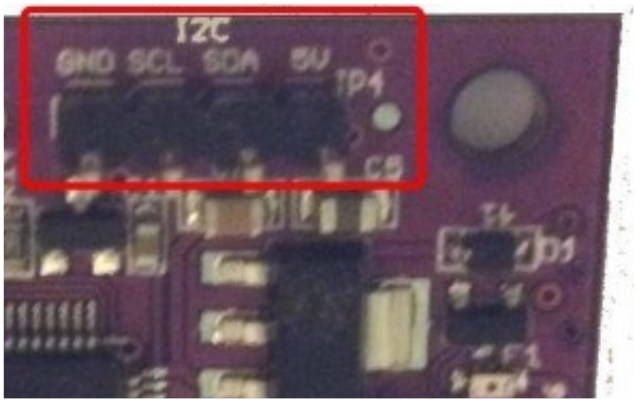
By using a USB Mini-B type (Male) to USB A type (Male) connector, navX-Micro can receive both power and also communicate with the RoboRIO.

This preferred installation method allows the navX-Micro circuit board to be placed up to 6 meters away from the RoboRIO.

IMPORTANT NOTE: To avoid having the navX-Micro reset due to a RoboRIO brownout (as discussed further in the navX-Micro Best Practices), connecting the navX-Micro to the RoboRIO via USB is highly recommended.

I2C

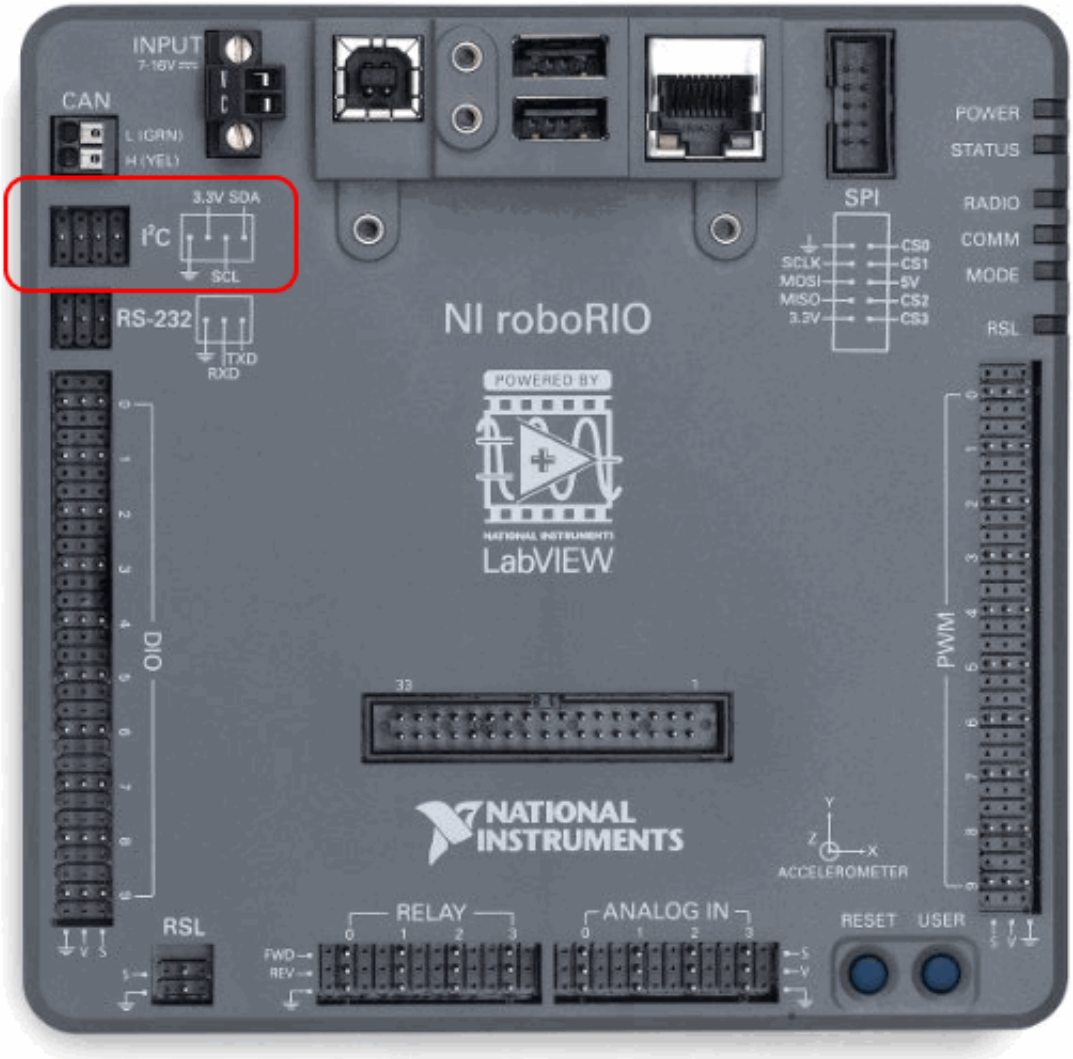
To use the I2C interface connect the navX-Micro 5VDC, GND, SCL and SDA pins to the corresponding pins on the RoboRIO.



The RoboRIO provides two separate I2C interfaces: the Onboard I2C interface, and the MXP Expansion Port I2C interface.

IMPORTANT NOTE: The left-to-right order of the RoboRIO Onboard I2C Interface Pins is different than the navX-Micro I2C pin left-to-right order. Pay careful attention to the I2C connector pin order when connecting navX-Micro to the RoboRIO.

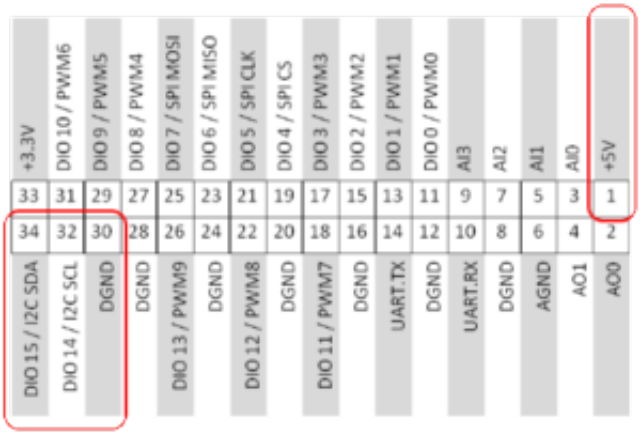
RoboRIO Onboard I2C Interface



IMPORTANT NOTE: The left-to-right order of the RoboRIO Onboard I2C Interface Pins is different than the navX-Micro I2C pin left-to-right order.

RoboRIO MXP I2C Interface

As shown in the diagram below, the RoboRIO MXP Connector’s I2C-related pins are:



Pin 34: I2C SDA

Pin 32: I2C SCL

Pin 30: Digital Ground (DGND)

Pin 1: +5V (or +3.3V)

Electrical Notes

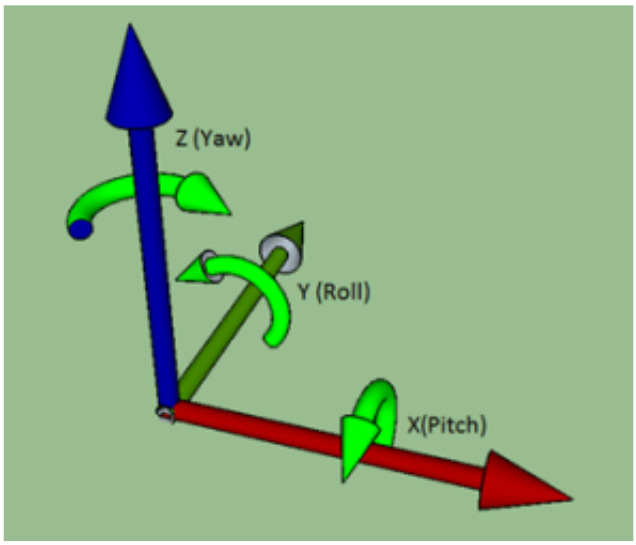
- The I2C bus standard requires that the SDA and SCL pins be pulled up with a pull-up resistor on each line. The RoboRIO internally pulls these lines high – but if connecting to a host computer without pullups, the SDA and SCL lines will need 1.5KOhm pullups.
- navX-Micro can be powered via it's I2C connector with either 3.3V or 5V DC. A minimum of 3.1V DC is required.
- The navX-Micro I2C signal pins are 5V tolerant, so the host computer can use either 5V or 3.3V DC levels on these pins.

Orientation

navX2-Micro measures a total of 9 sensor axes (3 gyroscope axes, 3 accelerometer axes and 3 magnetometer axes) and fuses them into a 3-D coordinate system. In order to effectively use the values reported by navX2-Micro, a few key concepts must be understood in order to correctly install navX2-Micro on a robot.

3-D Coordinate System

When controlling a robot in 3 dimensions a set of 3 axes are combined into a 3-D coordinate system, as depicted below:



In the diagram above, the green rounded arrows represent Rotational motion, and the remaining arrows represent Linear motion.

Axis	Orientation	Linear motion	Rotational Motion
X (Pitch)	Left/Right	– Left / + Right	+ Tilt Backwards
Y (Roll)	Forward/Backward	+ Forward / – Backward	+ Roll Left
Z (Yaw)	Up/Down	+ Up / – Down	+ Clockwise/ – Counter-wise

More discussion on the 3-D Coordinate system is on the Terminology page.

Reference Frames

Note that the 3-axis coordinate system describes relative motion and orientation; it doesn't specify the orientation with respect to any other reference. For instance, what does "left" mean once a robot has rotated 180 degrees?

To address this, the concept of a reference frame was invented. There are three separate three-axis "reference frames" that should be understood:

Coordinate System	Technical Term	X Axis	Y Axis
Field	World Frame	Side of Field	Front (Head) of Field
Robot	Body Frame	Side of Robot	Front (Head) of Robot
navX- Micro	Board Frame	See diagram Below	See diagram below

Joysticks and Reference Frames

Since a three-axis joystick is typically used to control a robot, the robot designer must select upon which Reference Frame the driver joystick is based. This selection of Reference Frame typically depends upon the drive mode used:

Drive mode	Reference Frame	Coordinate Orientation
Standard	Body Frame	Forward always points to the

Drive	front (head) of the robot
Field-orientedWorld Frame Drive	Forward always points to the front (head) of the field

navX2-Micro Board Orientation (Board Frame)

Aligning Board Frame and Body Frame

In order for the navX2-Micro sensor readings to be easily usable by a robot control application, the navX2-Micro Coordinate System (Board Frame) must be aligned with the Robot Coordinate system (Body Frame).

Aligning the Yaw (Z) axis and Gravity

The navX2-Micro motion processor takes advantage of the fact that gravity can be measured with its onboard accelerometers, fusing this information with the onboard gyroscopes to yield a very accurate yaw reading with a low rate of drift. In order to accomplish this, ***the yaw (Z) axis must be aligned with the "gravity axis" (the axis that points directly up and down with respect to the earth).***

When installing navX2-Micro on a robot, the navX2-Micro yaw (Z) axis and the gravity axis must be aligned.

Default navX2-Micro Board Orientation

The default navX2-Micro circuit board orientation is with the navX2-Micro logo on the Front *Right, with the top of the circuit board pointing up (with respect to the earth).*

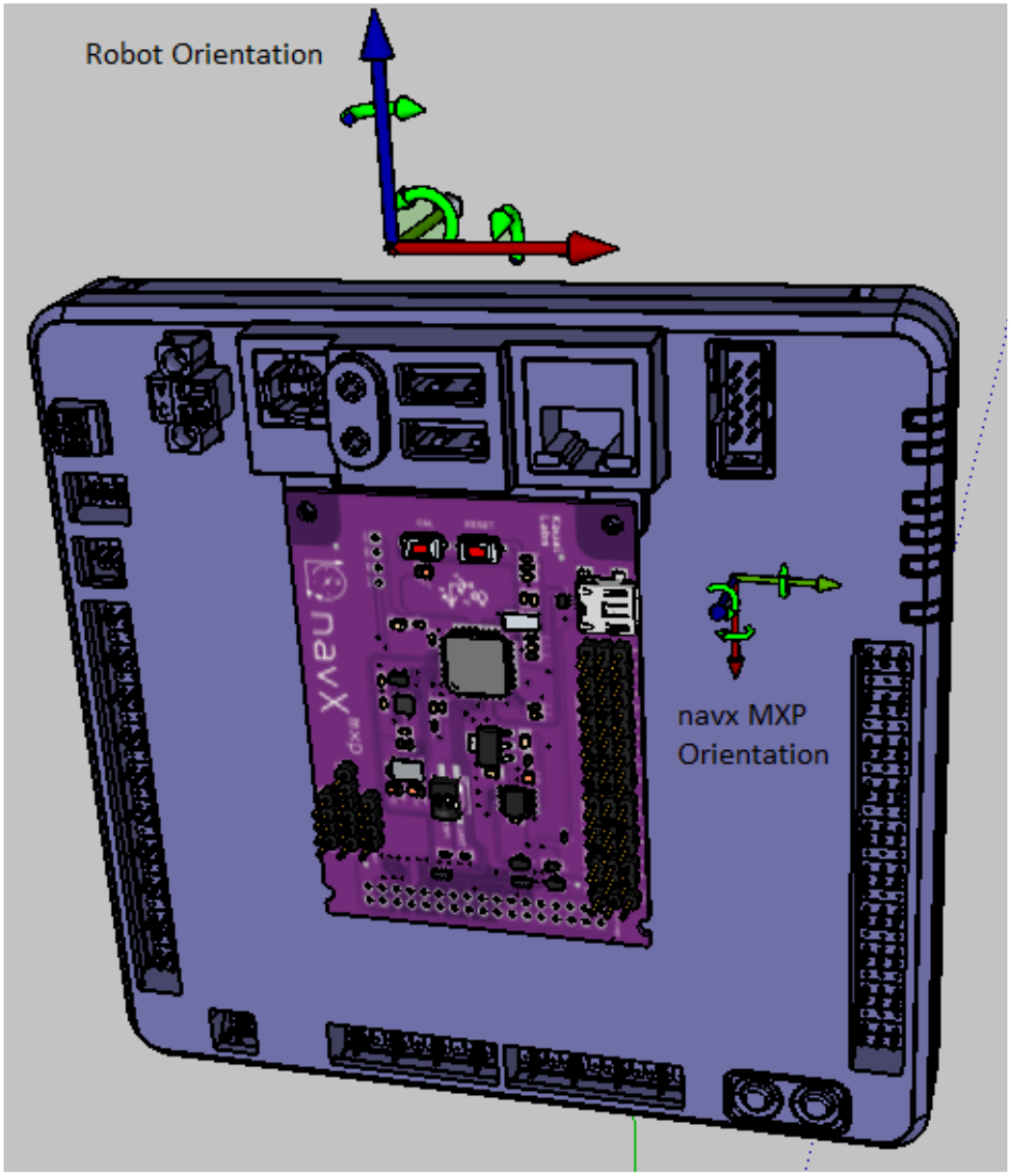
Since Body Frame and Board Frame coordinates should be aligned, and because the Yaw axis must be aligned with gravity, by default you must orient the navX2-Micro with the top of the board facing up, and with the Y axis (on the circuit board) pointing to the front of the robot.

If you need to mount the navX2-Micro circuit board in a different orientation (vertically, horizontally, or upside down), you can use the OmniMount ***feature to transform the orientation.***

OmniMount

If the navX2-Micro default yaw axis orientation isn't sufficient for your needs, you can use the **OmniMount** feature to re-configure the navX2-Micro yaw axis, allowing high-accuracy yaw axis readings when navX2-Micro is mounted horizontally, vertically, or even upside down.

In certain cases, the navX2-Micro axes (Board Frame) may not be oriented exactly as that of the Robot (Body Frame). For instance, if the navX2-Micro circuit board is mounted sideways, the navX2-Micro axes will not be oriented identically to the Robot. This configuration is shown in the diagram below:



Transforming navX2-Micro Board Frame to Body Frame with OmniMount

The navX2-Micro's "OmniMount" feature can transform the navX2-Micro X, Y and Z axis sensor data (Board Frame) into Robot Orientation (Body Frame) by detecting which of its three axes is perpendicular to the earth's surface.

This is similar to how a modern smart phone will rotate the display based upon the phone's orientation. However unlike a smart phone, the OmniMount detection of orientation does not happen all the time – since the orientation should not change while the robot is moving. Rather, each time OmniMount configuration occurs, the navX2-Micro records this transformation in persistent flash memory, and will continue to perform this transformation until the transform is reconfigured.

To configure OmniMount, follow these simple steps:

- Install navX2-Micro onto your robot. ENSURE that one of the navX2-Micro axes (as shown on the navX2-Micro circuit board) is perpendicular to the earth's surface. This axis will become the yaw (Z) axis. Note that this axis can either be pointing away from the earth's surface, or towards the earth's surface.
- Press the 'CAL' button on the navX2-Micro Circuit board AND HOLD THE BUTTON DOWN FOR AT LEAST 5 SECONDS.
- Release the 'CAL' button, and verify that the orange 'CAL' light flashes for 1 second and then turns off.
- Press the 'RESET' button on the navX2-Micro circuit board, causing it to restart.
- The nav2-Micro circuit board will now begin OmniMount auto-calibration. During this auto-calibration period, the orange 'CAL' LED will flash repeatedly. This process takes approximately 15 seconds, and requires two things:
    - 1. During auto-calibration, ***one of the navX2-Micro axes MUST be perpendicular to the earth's surface***.
    - 2. During auto-calibration, ***the navX2-Micro must be held still***.
    - If either of the above conditions is not true, the 'CAL' LED will be flashing quickly, indicating an error. To resolve this error, you must ensure that conditions 1 and 2 are met, at which point the 'CAL' LED will begin flashing slowly, indicating calibration is underway.
- Once the navX2-Micro auto-calibration is complete, the Board Frame to Body Frame Transform will be stored persistently into navX2-Micro flash memory and used until auto-calibration is run once again.

Creating an Enclosure

The navX2-Micro circuit board contains sensitive circuitry, and should be handled carefully.

An enclosure is recommended to protect the navX2-Micro circuit board from excessive handling, "swarf" , electro-static discharge (ESD) and other elements that could potentially damage the navX-Micro circuitry.

An enclosure for the navX2-Micro is available, and designed to support the 16mm mounting grid on a Matrix or Tetrix robotics system as used in FTC robotics. An alternate enclosure version designed by a FRC mentor is also available.

Build vs. Buy

Those who prefer to print the enclosure using their personal 3D printer, enclosure design files (in STL format) is available in the "enclosure" directory of the .

Those who prefer to purchase an injection-molded version of this enclosure can order it the Kauai Labs store.

Design Files

The enclosure design files include:

- navx-micro.skp: Sketchup 3D Design File for the navX-Micro circuit board
- navx-micro-lid.skp: Sketchup 3D Design File for a lid-style enclosure for the navX-Micro circuit board.
- navx-micro-lid.stl: STL Format File for 3d printing the lid-style enclosure for the navX-Micro circuit board.
- navx-micro-base.skp: Sketchup 3D Design File for a base of the enclosure for the navX-Micro circuit board.
- navx-micro-base.stl: STL Format File for 3d printing the base of the enclosure for the navX-Micro circuit board.

Printing and Customizing the Enclosure

The Sketchup (.skp) files can be edited using . Then, the files can be exported to a STL format using the Sketchup STL Import/Export extension. Finally, these exported STL format files can be opened and 3d-printed using applications including

Securing the Enclosure

The Lid Enclosure can be secured to the robot by four screws. This will secure not only the Lid, but will also secure the navX2-Micro circuit board.

Software

Software

navX2-Micro includes software which makes navX2-Micro easier to understand, integrate and use with a robot than other navigation technologies and products available today. This software includes the following components:

- Libraries for accessing navX2-Micro from:
  - Android-based FIRST FTC robot control systems
  - RoboRIO-based FIRST FRC robot control systems
  - Linux
  - Arduino
- The navXUI, which demonstrates navX2-Micro capabilities on a PC.

For advanced users, several calibration/configuration tools are also available.

Libraries

Libraries for accessing navX2-Micro from the following environments are available:

- Android-based FTC robots
- RoboRIO-based FRC robots
- Linux-based computers
- Arduino platforms

Android (2019 FTC Version)

NOTE: The 2019 Version of the navX-Micro Android Library for FTC has currently been tested with the FTC "ftc_app" library for the 2019 seson.

The navx_ftc Android software library supports access to navX-Model devices via the I2C communication interface. Several example programs are provided, demonstrating how to use a navX-Model device in a FTC-based robot control application.

To use the library, you can download the of the libraries, or you can the source code with Git. To learn more about the library, online help **is available.**

Getting Started

Before getting started, ensure you have installed Android Studio and the latest FTC Robot Controller Application ("ftc_app") package.

Several sample Java Robot Applications are provided. After running the setup program included in the latest build, the libraries and samples will be installed to the following location:

\navx-micro\android

For example, if your user name is Robot, the directory name will be C:\Users\Robot\navx-micro\android.

Within this directory, the "examples" sub-directory contains several example programs. Select the example you wish to start with and copy it into your project as follows:

- Copy one or more of the example navX-Model "op modes" files from the \navx-micro\android\examples directory into your project's "TeamCode" top-level directory. (i.e., org.firstinspires.ftc.teamcode).

Next, several configuration changes must be made in order that the Android Studio ftc_app-based project can locate the navx_ftc library:

- Modify the op mode example file to change the following line near the top of the file to match the I2C port on the Core Device Interface Module to which you have connected the navX-Model device:

```
private final int NAVX_DIM_I2C_PORT = 0; /* See the installation
page for details on port numbering. */
```

- Modify your robot application's (the "TeamCode" project) build.release.gradle file repository list to add a reference the directory where the navx_ftc library is installed:

```
repositories {
    flatDir {
        dirs 'libs', 'C:\\Users\\Robot\\navx-
micro\\android\\libs'
    }
}
```

- Again in the same build.release.gradle file, add the navx_ftc library to the list of libraries the ftc_app will link to – by adding this line near the bottom of the gradle build file, in the dependencies section:

```
dependencies {
  ...
     compile (name:'navx_ftc-release', ext:'aar')
}
```

Linux/MacOS distribution

For developers on Linux and Mac OS platforms, the latest non-Windows build is also available. Please note that this build does not contain any of the navX2-Micro tools, but does contain the RoboRIO and Android libraries.

This distribution is packaged as a .zip file; unzip the file and follow the instructions in the readme.txt file in the top-level directory.

Once you have installed the Android libraries onto your computer, use the instructions in the Getting Started section above to use the library. However, you will need to replace "C:\Users\Robot\navx-micro" in several places shown above with the corresponding path on Linux/MacOS where you installed the Android libraries.

RoboRIO Libraries

[“Generation 2” navX2-Micro Update: 6/15/2020 – navX2-Micro Firmware Version 4.0.400 is the latest version of firmware for the navX2-Micro model, and has been verified compatible with the RoboRIO.]

[“Classic” navX-Micro Update: 2/2/2019 – navX-Micro Firmware Version 3.1.366 has been released; this update is recommended if you using the navX-Micro USB Interface with the RoboRIO. To download this latest firmware, please follow the firmware update instructions***.]***

When using navX2-Micro on a FRC RoboRIO-based system, the USB and I2C interface options available in the *navX-MXP RoboRIO libraries* should be used. navX2-Micro is 100% compatible with the FRC navX2-MXP libraries when using USB and I2C.

Note: the link below leads to the navX-MXP website.

RoboRIO Libraries

Linux Library

Note: the link below leads to the navX-MXP website.

Linux Library

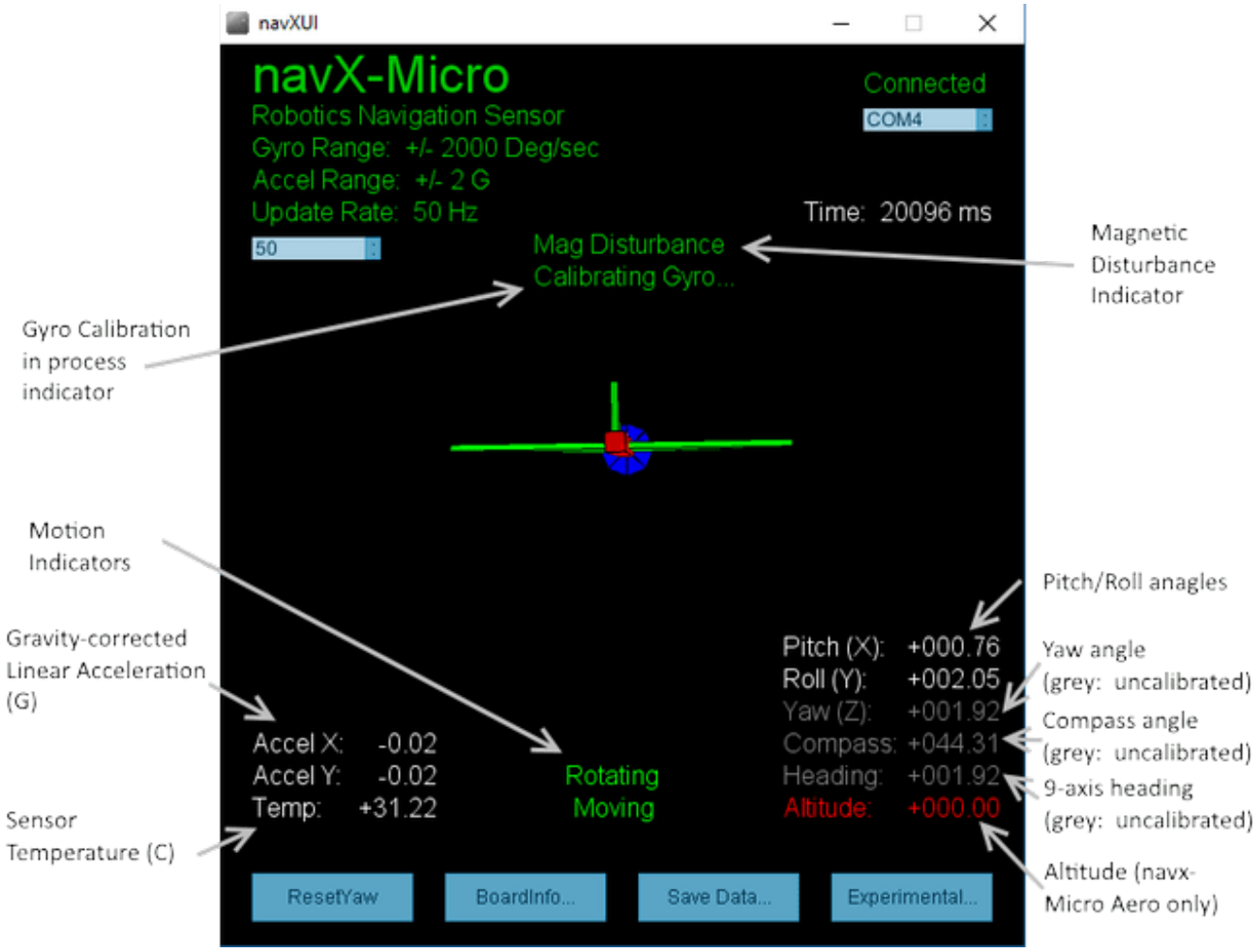
Arduino Library

Note: the link below leads to the navX-MXP website.

Arduino Library

navXUI

The navXUI user interface application provides a simple way to visualize the data provided by navX2-Micro.



Gyro Calibration in Progress Indicator

The Gyro Calibration in Progress Indicator is displayed during initial gyroscope calibration, which occurs immediately after power is applied to navX2-Micro. If the gyroscope calibration does not complete, navX2-Micro yaw accuracy will be adversely impacted. For more information on Gyro Calibration, please see the Gyro/Accelerometer Calibration page.

Motion Indicators

navX2-Micro provides dynamic motion indicators: (a) the “Moving” indicator and (b) the “Rotating” indicator.

The Moving indicator is present whenever the current Gravity-corrected Linear Acceleration exceeds the

“Motion Threshold”.

The Rotating indicator is present whenever the change in yaw value within the last second exceeds the “Rotating Threshold”. Note that navX2-Micro Gyroscope Calibration only occurs when navX-Micro is not Rotating for a few seconds.

Gravity-corrected Linear Acceleration (G)

navX2-Micro automatically subtracts acceleration due to gravity from accelerometer data, and displays the resulting linear acceleration. These measures are in units of instantaneous G, and are in World Reference Frame.

Sensor Temperature

The Sensor Temperature indicates the die temperature of the onboard sensor circuitry. Since shifts in gyro temperature can impact yaw accuracy, navX2-Micro will automatically perform Gyroscope calibration whenever navX2-Micro is still. See the Gyro/Accelerometer Calibration page for more details.

Magnetic Disturbance Indicator

Once the navX2-Micro Magnetometer has been calibrated (see the Magnetometer Calibration page), whenever the current magnetic field diverges from the calibrated value for the earth’s magnetic field, a magnetic disturbance is indicated.

Yaw Angle

The Yaw Angle is displayed in grey text if Gyro Calibration has not yet been completed. Once Gyro Calibration is complete, the Yaw Angle text color will change to white.

Pitch/Roll Angles

The Pitch/Roll Angles are always displayed in white text, since Accelerometer calibration occurs at the Kauai Labs factory.

Compass Angle

The Compass Angle displays the tilt-compensated compass heading calculated from navX2-Micro’s Magnetometer combined with the tip/tilt measure from the Accelerometers.

The Compass Angle is displayed in grey text if Magnetometer Calibration has not yet been completed. Once Magnetometer Calibration is complete, the Compass Angle text color will change to white.

9-axis (“Fused”) Heading

The 9-axis heading is displayed in grey text if Magnetometer Calibration has not yet been completed and/or if no undisturbed magnetic readings have occurred.

Altitude

The Altitude displays the navX2-Micro-calculated current altitude, based upon the reading from the pressure sensor, the current temperature and the sea-level pressure.

The Altitude is displayed in red text if a Pressure Sensor is not installed. Pressure Sensors are only installed on navX2-Micro Aero. Valid altitude readings are displayed in white text.

Installing/Running navXUI

- To run the navXUI, navX2-Micro must be connected to a PC running Windows via USB.
- Make sure Java 7 (version 1.7) or Java 8 (version 1.8) is installed on your computer. The 64-bit version of Java is recommended. To tell which version of java is currently “Active”, open up a command window, and type this command:

```
java -version
```

- Download the and unzip the contents to your local computer.
- Run the setup.exe program, which will install navXUI, as well as all necessary device drivers for communicating over USB with navX2-Micro, as well as some additional tools.
- Start navXUI:

From your Start Menu, select “Kauai Labs” and then “navX2-Micro” and click on the “navXUI” icon to start navXUI.

If your computer has more than one serial port, you can select which serial port to use by clicking on the up/down arrows in the COM port selection control in the UI.

Tools

The navX2-Micro software includes several tools for magnetometer calibration and advanced configuration. These tools run on a Windows PC and communicate with navX-Micro via USB.

NOTE: These tools are provided for use by advanced users; please carefully read the tool descriptions before using them.

Magnetometer Calibration Tool

The navXMagCalibrator tool is used to calibrate the navX MXP magnetometer. Please visit navX-Sensor Support for Magnetometer Calibration Tool usage instructions.

Advanced Configuration Tool

The Advanced Configuration Tool (navXConfig) is used to modify certain navX-Micro settings which impact the behavior of certain navX-Micro algorithms.

Please visit navX-Sensor Support for navXConfig Tool usage instructions.

Examples

Examples

Example source code for various navX2-Micro capabilities are available for both for FTC and FRC Robotics Control Systems.

FTC Examples

This section provides example code for several common navX2-Micro applications used by FIRST FTC teams on their robots to add sophisticated navigation capabilities. These examples are in Java.

NOTE: When you run the setup program contained in the latest navX2-Micro build, the navX2-Micro examples will be installed to subdirectories underneath <**home_directory**>\navx-micro\java\examples (e.g., **C:\Users\Robot**\navx-micro\java\examples).

FRC Examples

If you are looking for FRC examples, please see the navX-MXP Examples.

Field-Oriented Drive (FTC)

An easy-to-use, highly-maneuverable drive system is at the heart of a successful robot. Omnidirectional drive systems provide motion in the Y axis (forward-backward), X-axis (strafe), and Z axis (rotating about it's center axis). Each "degree of freedom" is independent, meaning that the overall robot motion is comprised of a "mix" of motion in each of the X, Y and Z axes, control of which is easily provided with a 3-degree of freedom joystick. This resulting maneuverability is quite useful during FRC competitions to avoid other robots, pick up and place game pieces, line up for shooting to a target, etc.

Yet the driver who remains in a fixed position is now presented a new challenge: *when the driving joystick is pushed forward, the robot does not necessarily move forward with respect to the field – rather it moves forward with respect to the robot*. This forces the driver to develop the skill of "placing their head in the robot" and performing the angular transformation mentally. This skill can take quite awhile to develop meaning that rookie drivers face an uphill climb before they can be productive team contributors. Additionally, the mental energy involved in field-to-robot rotational transformations reduces the driver's cognitive ability to focus other game-related tactical tasks, as evidenced by drivers who are so intently focused on driving that their response to their teammates is diminished. Moreover, when the driver does not have a clear line of sight to the robot, the "head in the robot" becomes even more challenging.

Solving this challenge is conceptually straightforward. First, the current angle (?) of rotation between the head of the field, and the head of the robot must be measured; secondly, the joystick X/Y coordinates are transformed by ?, as shown in following pseudo-code:

```
double rcw = pJoystick->GetTwist();
double forwrd = pJoystick->GetY() * -1; /* Invert stick Y axis */
double strafe = pJoystick->GetX();

float pi = 3.1415926;

/* Adjust Joystick X/Y inputs by navX MXP yaw angle */

double gyro_degrees = ahrs->GetYaw();
float gyro_radians = gyro_degrees * pi/180;
float temp = forwrd * cos(gyro_radians) +
 strafe * sin(gyro_radians);
strafe = -forwrd * sin(gyro_radians) +
 strafe * cos(gyro_radians);
fwd = temp;

/* At this point, Joystick X/Y (strafe/forwrd) vectors have been
*/
/* rotated by the gyro angle, and can be sent to drive system */
```

Rotate to Angle (FTC)

navX2-Micro can be used to rotate a robot quickly and accurately to a specified angle. This can occur not only with holonomic drive systems that provide independent Z-axis rotation (the capability to “spin on a dime” while also moving in a linear direction), but also on robots with only two drive motors This same technique can be used to help a robot drive in a straight line.

This example code below will automatically rotate the robot to a specified angle. This rotation can occur not only when the robot is still as in this example, but also when the robot is driving. When using field-oriented control on a holonomic drive system, this will cause the robot to drive in a straight line, in whatever direction is selected.

The PID Controller coefficients defined in the example code will need to be tuned for your drive system.

For more details on this approach, please visit Chief Delphi, including this .

FTC Android Example

Automatic Balancing (FTC)

The Automatic Balancing example demonstrates how to implement a self-balancing robot, which can be useful to help avoid a robot tipping over when driving. As an example, FRC team 263 demonstrated the auto-balance feature effectively during the 2018 FRC Championships.

The basic principle used in the example is based upon measurement of the navX-Micro Pitch (rotation about the X axis) and Roll (rotation about the Y axis) angles. When these angles exceed the “off balance” threshold and until these angles fall below the “on balance” threshold, the drive system is automatically driven in the opposite direction at a magnitude proportional to the Pitch or Roll angle.

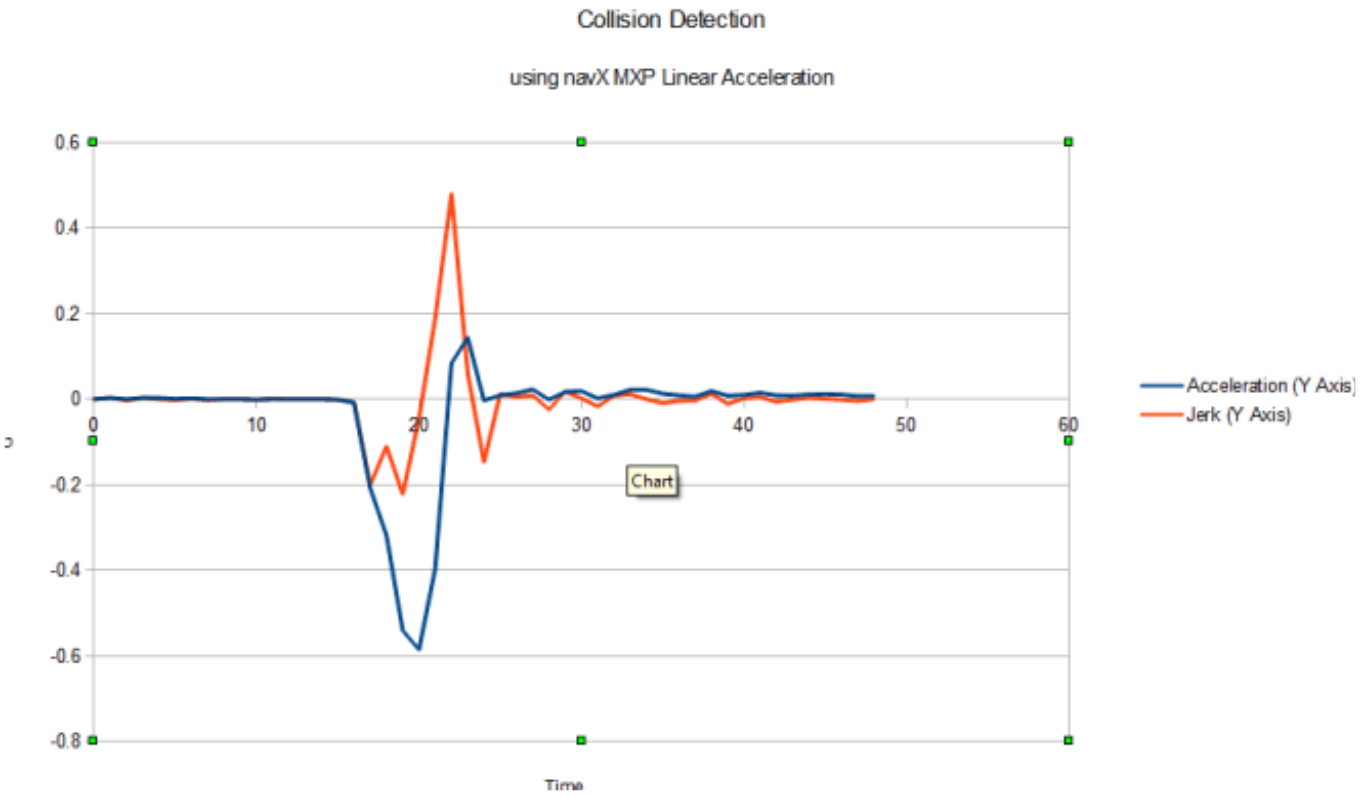
Note that this is just a starting point for automatic balancing, and will likely require a reasonable amount of tuning in order to work well with your robot. The selection of the magnitude of correction to apply to the drive motors in response to pitch/roll angle changes could be replaced by a PID controller in order to provide a tuning mechanism appropriate to the robot.

FTC Android Example

Collision Detection (FTC)

Collision Detection is commonly used in automobiles to trigger airbag deployment, which can reduce the force of an impact and save lives during an accident. A similar technique can be used on a robot to detect when it has collided with another object.

The principle used within the Collision Detection example is the calculation of Jerk (which is defined as the change in acceleration). As shown in the graph below (taken from navX2-Micro data of a small collision), whenever the jerk (in units of G) exceeds a threshold, a collision has occurred.



In the sample code shown below, both the X axis and the Y axis jerk are calculated, and if either exceeds a threshold, then a collision has occurred.

The “collision threshold” used in these samples will likely need to be tuned for your robot, since the amount of jerk which constitutes a collision will be dependent upon the robot mass and expected maximum velocity of either the robot, or any object which may strike the robot.

FTC Android Example

Motion Detection (FTC)

Detecting motion/no-motion can be simply detected by determining if a body’s linear acceleration exceeds a small threshold.

Using the data directly from accelerometers, this is not as easy as it seems, *since raw accelerometer readings contain both acceleration due to gravity as well as acceleration due to a body's motion*. One method for detecting motion with raw acceleration data is to use a high-pass filter, which lets quickly-changing information through but blocks information that doesn't change frequently.

However, a more comprehensive and reliable approach is to subtract the acceleration due to gravity from the raw acceleration values. The result value is known as "world linear acceleration", representing the actual amount of acceleration due to motion, and is calculated automatically by navX-Micro's motion processor. Whenever the sum of the world linear acceleration in both the X and Y axes exceeds a "motion threshold", motion is occurring.

FTC Android Example

Data Monitor (FTC)

The Data Monitor example code demonstrates how to perform navX2-Micro initialization and display all sensor values on a FIRST FTC robotics dashboard. The output data values include:

- Yaw, Pitch and Roll angles
- Compass Heading and 9-Axis Fused Heading (requires Magnetometer calibration)
- Linear Acceleration Data
- Motion Indicators
- Estimated Velocity and Displacement
- Quaternion Data
- Raw Gyro, Accelerometer and Magnetometer Data

As well, Board Information is also retrieved; this can be useful for debugging connectivity issues after initial installation of the navX-Micro sensor.

FTC Android Example

Guidance
Guidance

To ensure the highest possible accuracy and reliability, several recommendations regarding installation and use are provided.

This includes a set of best practices which summarize key factors which help ensure successful operation.

navX-Micro also features several calibration processes to ensure easy-to-use, high-reliability operation. This includes automatic gyroscope/accelerometer calibration, as well as manual magnetometer calibration . This section discusses these calibration processes and the theory behind them, including how to ensure minimal yaw drift.

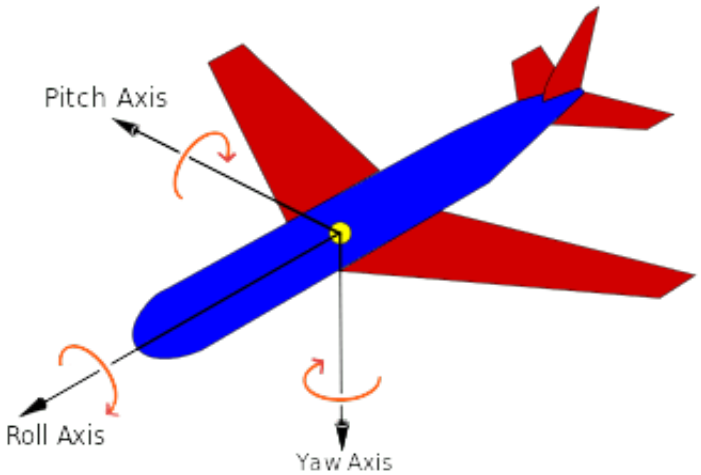
Terminology

Several terms used throughout the navX-Micro libraries and documentation may not be commonly understood and are defined herein.

Basic Terminology

A working knowledge of the following Basic Terminology is highly recommended when working with navX-Micro or any other Inertial Measurement Unit (IMU).

Pitch, Roll and Yaw



Pitch, Roll and Yaw are measures of angular rotation about an object’s center of mass, and together

provide a measure of “orientation” of that object with respect to an “at rest” position. When units of degrees are used, their range is from -180 to 180 degrees, where 0 degrees represents the “at rest” position of each axis.

Axis	Orientation relative to object’s center of mass	Rotational Motion
X (Pitch)	Left/Right	+ Tilt Backwards
Y (Roll)	Forward/Backward	+ Roll Left
Z (Yaw)	Up/Down	+ Clockwise/ – Counter-wise

Important Note: Pitch, Roll and Yaw angles represent rotation from the “origin” (0,0,0) of a 3-axis coordinate system. navX-Micro Pitch and Roll angles are referenced to earth’s gravity – so when navX-Micro is flat, Pitch and Roll angles should be 0.

The Yaw angle is different – Yaw is not referenced to anything external. When navX-Micro startup calibration completes, the Yaw angle is automatically set to 0 – thus at this point, 0 degrees represents where the “head” of the navX-Micro circuit board is pointing. The Yaw angle can be reset at any time after calibration completes if a new reference direction is desired.

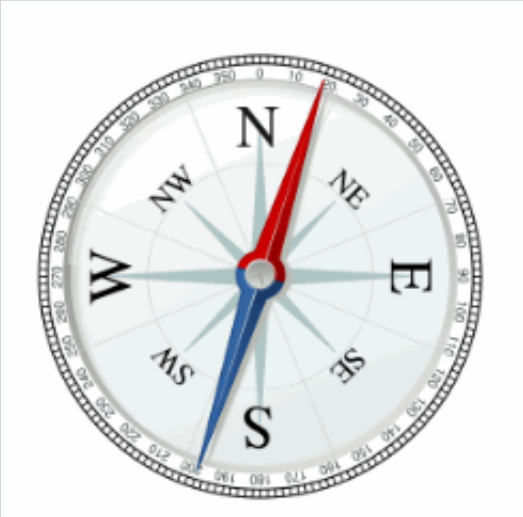
Linear Acceleration

Linear Acceleration is a measure of the change in velocity in a specific direction. For example, when a car starts from a standstill (zero relative velocity) and travels in a straight line at increasing speeds, it is accelerating in the direction of travel.

Axis	Orientation	Linear motion
X	Left/Right	– Left / + Right
Y	Forward/Backward	+ Forward / – Backward
Z	Up/Down	+ Up / – Down

Because the gyroscope and accelerometer axes are aligned, navX-Micro measures linear acceleration in the same set of 3 axes used to measure Pitch, Roll and Yaw. However unlike Pitch, Roll and Yaw, acceleration measures linear motion rather than rotation, and is measured in units of G, with a range of +/- 2.0.

Compass Heading



A compass measures the earth's magnetic field and indicates the current direction (heading) relative to magnetic north (N). Compass Heading is measured in degrees and is similar to Yaw, but has a few key differences:

- Compass Heading has a range of 0-360 (where magnetic north is 0).
- Compass Heading is absolute – it is referenced to magnetic north, and thus Compass Heading does not drift over time

Important Note 1: Compass Heading relies upon being able to measure the earth's magnetic field. Since the earth's magnetic field is weak, Compass Heading may not be able to measure earth's magnetic field when the compass is near a strong magnetic field such as that generated by a motor.

Important Note 2: Magnetic North is not exactly the same as True North*. Your robot can calculate True North given a Magnetic North reading, as long as the current declination is known. Declination is a measure of the difference in angle between Magnetic North and True North, and changes depending upon your location on earth, and also changes over time at that same location. An* online calculator *is provided allowing one to calculate declination for a given earth location and date.*

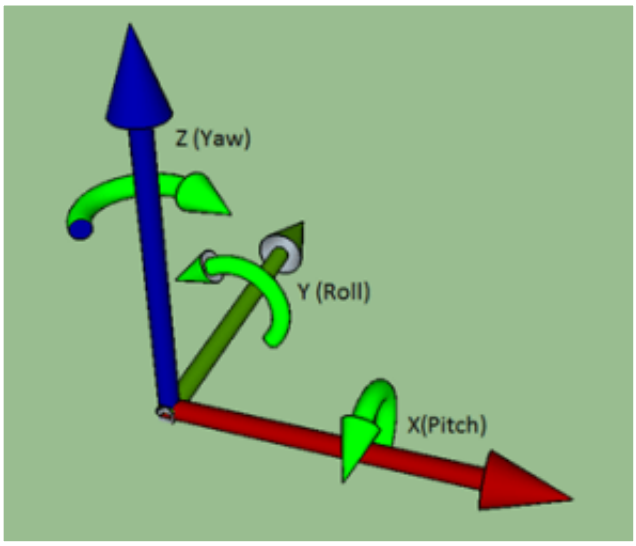
Altitude

Altitude is a measure of distance in the "up" direction from a reference; navX-Micro (Aero Edition) calculates altitude above sea-level using a pressure sensor.

navX-Micro (Aero Edition) altitude has a range of 0 to 25,000 meters.

Important Note: Altitude is calculated based upon barometric pressure. In order to accurately estimate altitude above the earth, navX-Micro should be configured with the sea-level barometric pressure in the surrounding area. This setting can be configured via the navX-Micro Advanced Configuration Tool*.*

3-D Coordinate System



navX-Micro 3-D Coordinate System

A 3-D Coordinate System uses one or more numbers (coordinates), often used to uniquely determine the position of a point within a space measured by that system. The origin of a 3-D coordinate system has a value of (0, 0, 0).

navX-Micro features gyroscopes, accelerometers and magnetometers which are all aligned with each other in a 3-D coordinate system. Each sensor type measures values with respect to that coordinate system, as follows:

Gyroscopes: measure rotation (as shown in the green arrows) about each axis. The coordinate system origin represents the center of the navX-Micro circuit board.

Accelerometers: measure acceleration, where the origin represents the position in space at which the previous acceleration sample was acquired.

Magnetometers: measure earth's magnetic field, where the origin represents the center of the navX-Micro circuit board.

Important Note: Because the navX-Micro Gyroscopes, Accelerometers and Magnetometers are all aligned to this 3-D Coordinate System, navX-Micro's motion processor can also use Sensor Fusion to provide additional information and processing including Tilt Correction, "Fused Heading", a Gravity Vector, World Reference Frame-based Linear Acceleration and Quaternions, as discussed in the Motion Processing section below.

Motion Processing

Users should also have a working knowledge of the terms defined in the Motion Processing Terminology.

Tilt Correction

Without correction, the compass heading calculated by a 3-axis magnetometer will only be accurate if the magnetometers are held “flat” with respect to the earth. To ensure the compass heading is valid even in cases when the sensor is “pitched” (Pitch angle != 0) or “rolled” (Roll angle != 0), navX-Micro performs “tilt correction” fusing the reading from the magnetometers with the pitch and roll angles from the accelerometers. Once corrected, the compass heading is aligned with the navX-Micro Z axis, which ensures that the Yaw angle and the Compass Heading measure rotation similarly.

“Fused” Heading

Given the gravity-referenced orientation provided by the Yaw angle, as well as the absolute compass heading angle which has been aligned to the navX-Micro 3-D coordinate system, both angles can be fused together. As shown in the table below, over a period of several minutes this can minimize the drift inherent in the Yaw angle, as well as provide an absolute reference for the Yaw angle – as long as the magnetometer is calibrated and a valid magnetometer reading is available every minute or so.

Value	Accuracy	Update Rate	Drift
Yaw	.01 degrees	Up to 66 Hz	~1 degree/minute
Compass Heading	2 degrees	1 Hz (if not magnetically disturbed)	None
Fused Heading	2 degrees (as long as a valid magnetometer reading is received in the last minute or so)	Up to 66Hz	None (~1 degree/minute, during periods of magnetic disturbance)

Like the Compass Heading, the Fused Heading has a range from 0-360 degrees.

Important Note: If the Compass Heading is not valid, the Fused Heading origin is the same as the Yaw angle. When valid (magnetically undisturbed) compass readings are received, the Fused Heading’s origin shifts to magnetic north (0 degrees on the Compass).

Gravity Vector

Accelerometers measure both acceleration due to gravity, as well as acceleration due to linear acceleration. This fact makes using raw accelerometer data difficult. navX-Micro’s automatic accelerometer calibration determines the component of measured acceleration which corresponds to gravity, and uses this information together with gyroscope readings to calculate a gravity vector, which represents acceleration due to gravity. Pitch and Roll angles are derived from this gravity vector.

Once the gravity vector is understood, this value is then subtracted from the raw accelerometer data to yield the acceleration due to linear motion.

Velocity and Displacement

Acceleration is defined as the change in Velocity. Therefore, linear velocity can be calculated by integrating linear acceleration over time.

Velocity is defined as the change in Position, otherwise known as Displacement. Therefore, linear displacement can be calculated by integrating linear velocity over time.

Important Note: Using currently-available MEMS-based accelerometers to calculate linear velocity and displacement is subject to large amounts of error primarily due to accelerometer "noise" (a difference between the actual acceleration and the measured acceleration inherent with MEMS sensors). This noise not only accumulates, but is also squared in the case of velocity, and is cubed in the case of displacement. Therefore, the resulting estimated velocity and displacement values are not typically useful for robotic navigation. The amount of error in displacement estimation can be several feet per second. As MEMS sensors improve in the coming years and accelerometer noise is reduced by approximately 100 times its currently value, this technique will become more useful for robotics navigation.

If you would like to experiment with using the navX-Micro to calculate displacement and velocity, you can use the navXUI's "Experimental" button to bring up a dialog which displays the integrated velocity and displacement values calculated in real-time by the navX-Micro.

World Reference Frame

Raw acceleration data measures acceleration along the corresponding sensor axis. This measurement occurs in a reference frame known as "Body Reference Frame". This works well as long as the navX-Micro circuit board is in it's original orientation. However as the navX-Micro circuit board rotates, the X and Y accelerometer axes no longer point "forward/back" and "left/right" with respect to the original orientation. To understand this more clearly, consider how the meaning of the term "left" changes once a robot has rotated 180 degrees? Introducing a World Reference Frame solves this issue by providing a reference upon which to measure "leftness".

To account for this, navX-Micro's motion processing adjusts each linear acceleration value by rotating it in the opposition direction of the current yaw angle. The result is an acceleration value that represents acceleration with respect to the area in which navX-Micro operates, which is known as "World Reference Frame". This world-frame linear acceleration value is much simpler to use for tracking motion of an object, like a robot, which might rotate while it moves.

Important Note: navX-Micro Linear Acceleration values are in World Reference Frame.

Advanced

Advanced users may require knowledge of the following terminology.

Quaternions

A quaternion is a four-element vector that can be used to encode any rotation in a 3D coordinate system. This single 4-element vector value can describe not only rotation about a reference frame's origin (Pitch,

Roll and Yaw) but also the rotation of that entire reference frame with respect to another. Furthermore, when Pitch, Roll and Yaw measures to perform certain calculations, it is not possible to clearly ascertain orientation when two axes are aligned with each other; this condition is referred to as “Gimbal Lock”. For robotics applications, Pitch, Roll and Yaw are sufficient, however for certain aerospace applications, Quaternions may be required to handle all possible orientations.

navX-Micro uses Quaternions internally, and also provides the 4 quaternion values for use by those who might need them.

Best Practices

Following best practices will help ensure high reliability and consistent operation. Please visit navX-Sensor Support for a summary of best practices.

Gyro/Accelerometer Calibration

navX2-Micro onboard sensors require calibration in order to yield optimal results. We highly recommend taking the time to understand this calibration process – successful calibration is vital to ensure optimal performance.

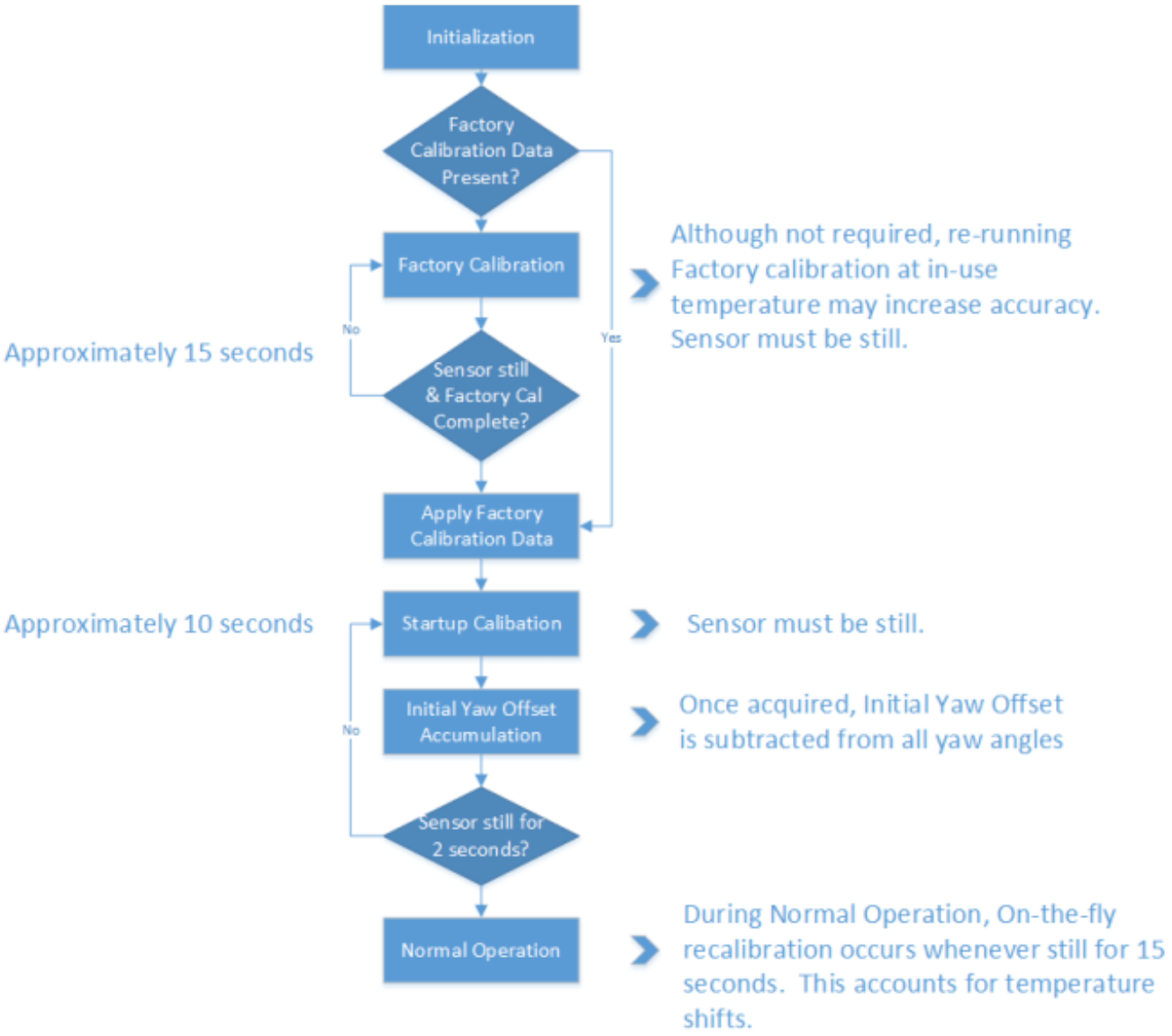
Accurate Gyroscope Calibration is crucial in order to yield valid yaw angles. Although this process occurs automatically, understanding how it works is required to obtain the best results.

If you are tempted to ignore this information, please read the section entitled “The Importance of Stillness” at the end of this page.

Calibration Process

The navX2-Micro Calibration Process is comprised of three calibration phases:

- Factory Calibration
- Startup Calibration
- On-the-fly Calibration



Factory Calibration

Before navX2-Micro units are shipped, the accelerometers and gyroscopes are initially calibrated at the factory; this calibration data is stored in flash memory and applied automatically to the accelerometer and gyroscope data each time the navX2-Micro circuit board is powered on.

Note that the onboard gyroscopes are sensitive to temperature changes. Therefore, since the average ambient temperature at the factory (on the island of Kauai in Hawaii) may be different than in your environment, you can optionally choose to re-calibrate the gyroscope by pressing and holding the “CAL” button for at least 10 seconds. When you release the “CAL” button, ensure that the “CAL” Led flashes briefly, and then press the “RESET” button to restart navX2-Micro. When navX2-Micro is re-started, it will perform the Initial Gyro Calibration – the same process that occurs at our factory. *NOTE: It is very important to hold navX2-Micro still, and parallel to the earth’s surface, during this Initial Gyro Calibration period.* You might consider performing this process before using your robot the first time it is

used within a new environment (e.g., when you arrive at a FTC competition event).

The value of re-running Factory Calibration at the same temperature navX2-Micro will be operated at is potentially increased yaw accuracy as well as faster Startup Calibration. If a significant temperature shift has occurred since the last Factory Calibration, the Startup Calibration time may take longer than normal, and it's possible that yaw accuracy will be diminished until the next On-the-fly Gyro Calibration completes.

Startup Calibration

Startup Calibration occurs each time the navX2-Micro is powered on, and requires that the sensor be held still in order to complete successfully. Using the Factory Calibration as a starting point, the sensor calibrates the accelerometers and adjusts the gyroscope calibration data as well based upon current temperature conditions.

If the sensor continues to move during startup calibration, Startup Calibration will eventually timeout – and as a result, the navX2-Micro yaw angle may not be as accurate as expected.

Initial Yaw Offset Calibration

Immediately after Startup Calibration, an **Initial Yaw Offset** is automatically calculated. The purpose of the Initial Yaw Offset is to ensure that whatever direction the "front" of the navX2-Micro circuit board is pointed to at startup (after initial calibration is applied) will be considered "0 degrees".

Yaw Offset Calibration requires that navX2-Micro be still for approximately 2 seconds after Startup Calibration completes. After approximately 2 seconds of no motion, navX2-Micro will acquire the current yaw angle, and will subtract it from future yaw measurements automatically. The navX2-Micro protocol and libraries provide a way to determine the yaw offset value it is currently using.

NOTE: If navX2-Micro is moving during startup, this Yaw Offset Calibration may take longer than 2 seconds, and may not be calculated at all if the sensor continues moving long enough. Therefore it is important to keep navX2-Micro still until initial calibration and Initial Yaw Offset calibration completes.

On-the-fly Gyro Calibration

In addition to Startup Calibration, during normal operation navX2-Micro will automatically re-calibrate the gyroscope (e.g., to account for ongoing temperature changes) during operation, whenever it detects several seconds of no motion. This process occurs automatically and is completely transparent to the user. Therefore each time navX2-Micro is still for several seconds, the gyroscopes are re-calibrated "on-the-fly". The purpose of On-the-fly Gyro re-calibration is to help maintain yaw accuracy when shifts in ambient temperature occur during operation.

This On-the-fly Gyro Calibration can help deal with cases where the sensor was moving during Startup Calibration, but note that the yaw is not zeroed at the completion of On-the-fly Calibration. So once again, it's important to keep the sensor still during Startup Calibration.

Runtime Yaw Zeroing

Your robot software can optionally provide the robot operator a way to reset the yaw angle to Zero at any time. Please see the documentation for the navX2-Micro libraries for more details.

The importance of stillness

This is the most important takeaway from this discussion: It is very important that navX2-Micro be held still during the above Initial Gyro and Initial Yaw Offset calibration periods. In support of this, navX2-Micro indicates when it is calibrating; we recommend you incorporate this information into your software. Please see the discussion of the navXUI, and the navX-Micro libraries for more details on this indication.

Magnetometer Calibration

navX2-Micro onboard sensors require calibration in order to yield optimal results. We highly recommend taking the time to understand this calibration process – successful calibration is vital to ensure optimal performance.

Careful and accurate Magnetometer Calibration is crucial in order to yield valid compass heading, 9-axis heading and magnetic disturbance detection.

Magnetometer Calibration is not typically required for use in many FIRST FTC and FRC robot applications, including Field-oriented drive. Magnetometer Calibration is a manual process and is recommended for advanced users who need to calculate absolute heading.

Calibration Process

The magnetometer calibration encompasses three areas: (a) hard-iron calibration, (b) soft-iron calibration and (c) magnetic disturbance calibration.

Hard and soft-iron calibration allows the following equation to be used, and corrects for hard and soft-iron effects due to nearby ferrous metals and magnetic fields. This calibration is necessary in order to achieve valid compass heading readings:

Image not found

In addition, using the same calibration data the strength of the Earth’s Magnetic Field is determined. Whenever the data from the magnetometer indicates the current magnetic field differs from the calibrated Earth’s Magnetic Field strength by more than the “Magnetic Disturbance Ratio”, a Magnetic Anomaly is declared.

Therefore, careful and accurate Magnetometer Calibration is crucial in order to yield valid compass heading, 9-axis heading and magnetic disturbance detection.

Magnetometer Calibration can be accomplished with a single, simple calibration process through the use of the Magnetometer Calibration tool. This tool is designed to run on a Windows computer, and communicate to the navX2-Micro circuit board via a USB cable.

Selecting an Interface

navX2-Micro provides two methods for communicating with robotics control applications:

- I2C
- USB 2.0

Streaming vs. Register-based Communication

The navX2-Micro interfaces fall into two types: Streaming and Register-based.

Streaming: data is sent at regular intervals by navX2-Micro, and the host is notified when new data arrives. The streaming data is sent in two different formats: Processed Data and Raw data. Streaming is used over the USB interface. More details on the communication detail are available in the Serial Protocol Definition.

Register-based: communication is initiated by the host whenever new data is desired, and the host can request any data required. Register-based communication is used over the I2C interface. More details on the communication detail are available in the Register Protocol Definition.

Comparing the navX2-Micro Communication Interfaces

Interface Type	Speed	Latency	Type	Cable distance	Max Update Rate
USB	12 mbps	<1ms	Streaming	6 meters	200
I2C	400 kbps	~10ms	Register-based	1 meter	200

Recommendations

The USB interface is the recommended method for interfacing to navX2-Micro, based on for the higher communications bandwidth, lower latency and longer maximum cable distance.

There are however a few cases where using the I2C interface is a reasonable option:

- When interfacing to a robot control systems which don’t support USB-connected sensors (e.g.,

the FTC Android-based REV Expansion or Control Hub).
- When already using the USB interface connection to a secondary processor (e.g., a video processor)

Yaw Drift

A gyroscope measures the amount of angular rotation about a single axis. Since the gyroscope measures changes in angular rotation, rather than an absolute angle, calculation of the actual current angle of that axis is estimated via rather than an exact measurement.

Any Inertial Measurement Unit (IMU), including navX2-Micro, that integrates a signal from a gyroscope will also accumulate error over time. Accumulated error is due to several factors, including:

- (which occurs when an analog-to-digital converter (ADC) converts a continuous analog value to a discrete integral value)
- Scale factor error (which occurs due to manufacturing errors causing a specified scale factor [e.g., 256 bits per unit G] to be incorrect)
- Temperature instability (which occurs when a sensor is more or less sensitive to an input as temperature changes)
- Bias error (which occurs because the value the sensor reports at ‘zero’ is not known well enough to ‘subtract’ that value out during signal processing)

Over time, these errors accumulate leading to greater and greater amounts of error.

With the navX2-Micro, Quantization error is minimized due to the onboard sensor internal signal conditioning, high-resolution 16-bit Analog-to-Digital Converters (ADC), and extremely fast internal sampling (416Hz). Scale factor error is easily corrected for by factory calibration, which the navX2-Micro provides. So these two noise sources are not significant in the navX2-Micro.

The remaining sources of error – temperature instability and bias error – are more challenging to overcome:

- Gyro bias error is a major contributor to yaw drift error, but is inherent in modern MEMS-based gyroscope.
- Temperature instability can cause major amounts of error, and should be managed to get the best result. To address this, navX2-Micro automatically re-calibrates the gyro biases whenever it is still for several seconds, which helps manages temperature instability.

navX2-Micro Pitch and Roll values are extremely accurate over time since gyroscope values in the pitch/roll axes can be compared to the corresponding values from the accelerometer. This is because when navX2-Micro is still, the accelerometer data reflects only the linear acceleration due to gravity.

Correcting for integration error in the Yaw axis is more complicated, since the accelerometer values in this axis are the same no matter how much yaw rotation exists.

To deal with this, several different “data fusion” algorithms have been developed, including:

- Complementary filter
- Extended Kalman filter (EKF)
- Direction Cosine Matrix filter (DCM)

Note: See the References page for links to more information on these algorithms.

These algorithms combine the acceleromter and gyroscope data together to reduce errors.

The Complementary and EKF filter algorithms are designed to process 3-axis accelerometer and 3-axis gyroscope values and yield yaw/pitch/roll values. The Complementary filter is a simple approach, and works rather well, however the response time is somewhat slower than the EKF, and the accuracy is somewhat lower.

The DCM filtering approach is similarly accurate and responsive as the EKF, however it requires information from a 3-axis magnetometer as well to work correctly. Since the magnetometer on a FIRST FRC robot typically experiences significant amounts of magnetic disturbance, the DCM algorithm is not well suited for use in a Robotics Navigation Sensor.

For these reasons, the EKF is the preferred filtering algorithm to provide the highest performance IMU on a FIRST FRC robot. However, the EKF algorithm is complex and difficult to understand, making it typically beyond the capabilities of many robotics engineers. The navX2-Micro circuit board uses a sophisticated Extended Kalman Filter-based algorithm to ensure the highest possible accuracy.

With this processing, navX-Micro exhibits yaw drift on the order of ~1 degree per minute; yaw drift is much lower when navX-Micro is still.

Tips

What follows are some tips on how to deal with the yaw drift within the context of a FIRST FTC competition.

In general, the yaw will not drift significantly during a FTC match, based upon the following calculation:

> yaw drift(degrees) at end of match = yaw drift (~1 degree/minute) x match length (2.5 minute) = **~2.5 degrees**

However, during long practice matches the drift may become noticeable, and can be dealt with using the following approaches:

1) The simplest approach which is supported by the navX2-Micro Libraries is to periodically “re-zero”

navX2-Micro by applying an offset to the navX2-Micro yaw angle. To use this approach, when the robot is in the correct orientation, a driver can press a button which causes an offset to be added so that the reported angle at that orientation is 0.

2) Even though the navX2-Micro magnetometer will likely give erroneous readings once the robot motors are energized, a calibrated magnetometer can potentially provide a stable reading during the moments before a FTC competition round. The navX2-Micro provides a 9-axis “fused heading” which is combined with the ~1 degree per minute of drift in the yaw angles. Using the “fused heading”, it is possible to calculate the robots absolute orientation and maintain it. With the “fused heading”, that drift will be updated w/the absolute heading from the compass whenever a compass reading which is free from magnetic disturbance is detected. Note that to be effective this requires the magnetometer to be calibrated. Once calibrated, an initial magnetometer reading undisturbed by magnetic disturbances can be acquired at the beginning of a match, before the motors are energized. If the sensor is placed far enough away from motors, it may be possible to also get an undisturbed magnetometer during a match.

Support

Support

Please visit navX-Sensor Support if you are experiencing difficulty or trouble.

Factory Test Procedure

The Factory Test Procedure verifies correct operation of the circuit board and it’s key components. Please visit navX-Sensor Support for Factory Test Procedure instructions.

Updating Firmware

Please visit navX-Sensor support for sensor firmware update instructions.

Advanced

Advanced

This section provides instructions and pointers to additional reference materials for those who are adventurous, and wish to modify and/or more fully understand the technologies used in navX-Micro.

Serial Protocol

In order to communicate sensor data to a client (e.g., a RoboRio robot controller) the navX-Micro software uses a custom protocol. This protocol defines messages sent between the navX-Micro and the client over a serial interface, and includes an error detection capability to ensure corrupted data is not used by the client.

The navX-Micro Serial protocol uses two message types, the legacy ASCII messages initially introduced in the nav6 sensor, and the modern binary messages introduced in the navX-MXP.

Source code that implements the navX-Micro ASCII and binary protocols in C/C++ and Java are available online to simplify adding support for the navX-Micro protocol to a software project.

Message Structure

ASCII Protocol Messages

Each navX-Micro Serial ASCII protocol message has the following structure:

Start of Message	Message ID	Message Body	Message Termination
1 byte	1 byte	length is message-type dependent	4 bytes

Binary Protocol Messages

Each navX-Micro Serial Binary protocol message has the following structure:

Start of Message	Binary Message Indicator	Binary Message Length	Message ID	Message Body	Message Termination
1 byte	1 byte	1 byte	1 byte	length is message-type dependent	4 bytes

Data Type Encoding (ASCII)

Base16 encoding is used for ASCII message elements, as follows:

Data Type	Encoding	Example
Float	(Sign)(100s)(10s)(1s).(10ths)(100ths)	‘-132.96’. ‘ 257.38’
8-bit Integer	(HighNibble)(LowNibble)	‘E9’
16-bit Integer	(HighByte,HighNibble)(HighByte,LowNibble)(LowByte,HighNibble)(LowByte,LowNibble)	‘1A0F’

Data Type Encoding (Binary)

Binary encoding is used for all Binary message elements. All Binary-formatted data types that are signed are encoded as 2’s complement. All multi-byte data types are in little-endian format. Certain non-standard ‘packed’ data types are used to increase storage efficiency.

Data Type	Range	Byte Count
Unsigned Byte	0 to 255	1
Unsigned Short	0 to 65535	2
Signed Short	-32768 to 32768	2
Signed Hundredths	-327.68 to 327.67	2
Unsigned Hundredths	0.0 to 655.35	2
Signed Thousandths	-32.768 to 32.767	2
Signed Pi Radians	-2 to 2	2
Q16.16	-32768.9999 to 32767.9999	4
Unsigned Long	0 to 4294967295	4

*Unsigned Hundredths: original value * 100 rounded to nearest integer
*Signed Hundredths: original value * 100 rounded to nearest integer
*Signed Thousandths: original value * 1000 rounded to nearest integer
*Signed Pi Radians: original value * 16384 rounded to nearest integer

Start of Message

Each message begins with “start of message” indicator (a ‘!’ character), which indicates that the following bytes contain a message.

Binary Message Indicator

Each binary message includes a “binary message” indicator (a ‘#’ character), which indicates that the following bytes contain a binary message.

Binary Message Length

Each Binary message contains a length value (a value from 0-255), which indicates that the number of bytes which follow in the Message Body and Message Termination.

Message ID

The Message ID indicates the type of message, which may be one of the following:

ID	Message Type	Encoding
‘y’	Yaw/Pitch/Roll/Compass Heading Update	ASCII
‘g’	Raw Data Update	ASCII
‘p’	AHRS + Position Data Update	Binary
‘S’	Stream Configuration Command	ASCII
‘s’	Stream Configuration Response	ASCII
‘I’	Integration Control Command	Binary
‘j’	Integration Control Response	Binary

Message Body

The message body differs depending upon the Message Type; the various Message Body specifications are listed below.

Message Termination

The final four bytes of each Serial protocol message contain a Base16 unsigned 8-bit checksum (encoded in 2 bytes as an ASCII 8-bit integer) followed by a carriage return and then a line feed character.

Checksum

The checksum is calculated by adding each byte of the message except the bytes within the Message Termination itself. The checksum is accumulated within an 8-bit unsigned byte.

New Line

The carriage return (0x10) and newline characters (0x13) are present at the end of the message so that when the message is displayed in a console window, a new line will be inserted in the console at the end of the message.

Message Body Definitions

Yaw/Pitch/Roll/Compass Heading Update Message

The Yaw/Pitch/Roll/Compass Heading Update message indicates the navX-Micro current orientation and

heading, in units of degrees, as follows:

Byte Offset	Element	Data Type	Unit
0	Yaw	Float	Degrees (-180 to 180)
7	Pitch	Float	Degrees (-180 to 180)
14	Roll	Float	Degrees (-180 to 180)
21	Compass Heading	Float	Degrees (0 to 360)

Raw Data Update Message

The Raw Data update message communicates the raw gyro, accelerometer, magnetometer and temperature data. This data bypasses the Digital Motion Processor, and allows the individual sensors to be used directly without any intervening processing. This can allow the following types of use:

- Access to instantaneous measures of angular velocity in each of the X, Y and Z axes, provided by the tri-axial gyroscopes. Note that the accelerometer and gyroscope data has already had bias calibration applied.

- Additionally, raw magnetometer data is provided. Note that the raw magnetometer data may have already had soft/hard iron calibration applied, if the navX-Micro magnetometer calibration procedure has already been completed.

Byte Offset	Element	Data Type
0	Gyro X (15-bits, signed)	16-bit Integer
4	Gyro Y (15-bits, signed)	16-bit Integer
8	Gyro Z (15-bits, signed)	16-bit Integer
12	Acceleration X (16-bits, signed)	16-bit Integer
16	Acceleration Y (16-bits, signed)	16-bit Integer
20	Acceleration Z (16-bits, signed)	16-bit Integer
24	Magnetometer X (12 bits, signed)	16-bit Integer
28	Magnetometer Y (12 bits, signed)	16-bit Integer
32	Magnetometer Z (12 bits, signed)	16-bit Integer
36	Temperature (Centigrade degrees)	Float

Gyro Device Units: value in deg/sec * gyro full scale range

Accelerometer Device Units: value in G * accelerometer full scale range

Magnetometer Device Units: value in uTesla * .15

AHRS / Position Data Update

Byte Offset	Element	Data Type	Unit
0	Yaw	Signed Hundredths	Degrees
2	Pitch	Signed Hundredths	Degrees
4	Roll	Signed Hundredths	Degrees

6	Compass Heading	Unsigned Hundredths	Degrees
8	Altitude	Signed 16:16	Meters
12	Fused Heading	Unsigned Hundredths	Degrees
14	Linear Accel X	Signed Thousandths	G
16	Linear Accel Y	Signed Thousandths	G
18	Linear Accel Z	Signed Thousandths	G
20	Velocity X	Signed 16:16	Meters/Sec
24	Velocity Y	Signed 16:16	Meters/Sec
28	Velocity Z	Signed 16:16	Meters/Sec
32	Displacement X	Signed 16:16	Meters
36	Displacement Y	Signed 16:16	Meters
40	Displacement Z	Signed 16:16	Meters
44	Quaternion W	Signed Pi Radians	Pi Radians
46	Quaternion X	Signed Pi Radians	Pi Radians
48	Quaternion Y	Signed Pi Radians	Pi Radians
50	Quaternion Z	Signed Pi Radians	Pi Radians
52	MPU Temp	Signed Hundredths	Centigrade
54	Op. Status	Uint8	NAVX_OP_STATUS
55	Sensor Status	Uint8	NAVX_SENSOR_STATUS
56	Cal. Status	Uint8	NAVX_CAL_STATUS
57	Selftest Status	Uint8	NAVX_SELFTEST_STATUS

Stream Configuration Command

By default, navX-Micro begins transmitting YPR Updates upon power up. The Stream Configuration Command is sent in order to change the type of navX-Micro Streaming Update transmitted to the client.

Byte Offset	Element	Data Type
0	Stream Type	8-bit ASCII Character
1	Update Rate (Hz) – Valid range: 4-60	8-bit Integer

Stream Type	Description
‘y’	Yaw, Pitch, Roll & Compass Heading Update
‘g’	Gyro (Raw) Data Update
‘p’	AHRS + Position Data Update

Stream Configuration Response

Whenever a Stream Configuration Command is received, navX-Micro responds by sending a Stream Configuration Response message, which is formatted as follows:

Byte Offset	Element	Data Type
0	Stream Type	8-bit ASCII Character
1	Gyroscope Full Scale Range (Degrees/sec)	16-bit Integer
5	Accelerometer Full Scale Range	16-bit Integer

	(G)	
9	Update Rate (Hz)	16-bit Integer
13	Calibrated Yaw Offset (Degrees)	Float
20	Reserved	16-bit Integer
24	Reserved	16-bit Integer
28	Reserved	16-bit Integer
32	Reserved	16-bit Integer
36	Flags	16-bit Integer

Flag value	Desription
0, 1	Startup Gyro Calibration in progress
2	Startup Gyro Calibration complete

Integration Control Command

The Integration Control Command is sent in order to cause certain values being integrated on the navX-Micro to be reset to 0.

Byte Offset	Element	Data Type
0	Action	Uint8 (NAVX_INTEGRATION_CTL)
1	Parameter	Uint32

Integration Control Response

The Integration Control Response is sent in response to an Integration Control Command, confiming that certain values being integrated on the navX-Micro have been reset to 0.

Byte Offset	Element	Data Type
0	Action	Uint8 (NAVX_INTEGRATION_CTL)
1	Parameter	Uint32

Register Protocol

In addition to the streaming Serial protocol over USB, navX-Micro may be accessed over the I2C bus, using a register-based protocol.

Register-based protocol overview

Unlike the streaming Serial protocol, which periodically sends out updates messages whenever new data is available, the register based protocol is a “polled” interface, in that the consumer of the navX-Micro data (in this case referred to as a “bus master”) can request data from the navX-Micro at any time. At the same time, when using the register-based protocol the bus master does not know when new data is available.

To help this situation, a timestamp – which is updated whenever new data is available – is made available. Therefore, the general approach to ensure each new data sample is retrieved is to regularly (at the navX-Micro update rate) retrieve both the timestamp and the data of interest), and if the timestamp differs from the previous timestamp by the update rate as expressed in milliseconds, then the data sample just retrieved is current, and no data has been missed.

I2C Overview

The navX-Micro responds to 7-bit address 50 (0x32) on the I2C bus. If accessing the navX-Micro via the I2C bus, ensure that no other device at that address is on the same bus.

The navX-Micro I2C bus operates at a speed up to 400Khz.

When accessing the navX-Micro via the I2C bus, this following pattern is used:

– The I2C bus master sends the navX-Micro I2C address. The highest bit is set to indicate the bus master intends to write to the navX-MXP. If the highest bit is clear, this indicates the bus master intends to read from the navX-MXP.

– The I2C bus master next sends the starting register address it intends to write to or read from.

– The I2C bus master next initiates I2C an bus transaction. The navX-MXP supports I2C burst mode for read operations, therefore the navX-MXP will respond with register values as long as the I2C bus master continues the transaction, and as long as the last register address has not yet been reached.

If instead the I2C bus master intends to write data to a writable navX-MXP register, the bus master should transmit the new register value immediately after sending the register address.

navX-Micro Register Data Types

All multi-byte registers are in little-endian format.
All registers with ‘signed’ data are 2’s-complement.

Data Type	Range	Byte Count
Unsigned Byte	0 to 255	1
Unsigned Short	0 to 65535	2
Signed Short	-32768 to 32768	2
signed hundredths	-327.68 to 327.67	2
Unsigned Hundredths	0.0 to 655.35	2
Signed Thousandths	-32.768 to 32.767	2
Signed Pi Radians	-2 to 2	2
Q16.16	-32768.9999 to 32767.9999	4
Unsigned Long	0 to 4294967295	4

*Unsigned Hundredths: original value * 100 to rounded to nearest integer

*Signed Hundredths: original value * 100 rounded to nearest integer
*Signed Thousandths: original value * 1000 rounted to nearest integer
*Signed Pi Radians: original value * 16384 rounded to nearest integer

navX-Micro Register Map

Address (Hex)	Name	Access	Range/Data Type
0x00	WhoAmI	Read-only	50 (0x32): navX-MXP
0x01	Board Revision	Read-only	Unsigned byte
0x02	Firmware Major Version	Read-only	Unsigned byte
0x03	Firmware Minor Version	Read-only	Unsigned byte
0x04	Update Rate	Read/write	Unsigned byte (Hz)
0x05	Accel FSR	Read-only	Unsigned byte (Degrees/Sec)
0x06-0x07	Gyro FSR	Read-only	Unsigned short(G)
0x08	Operational Status	Read-only	See NAVX_OP_STATUS
0x09	Calibration Status	Read-only	See NAVX_CAL_STATUS
0x0A	Self-test Status	Read-only	See NAVX_SELFTEST_STATUS
0x0B	Capability Flags (low)	Read-only	See NAVX_CAPABILITY
0x0C	Capability Flags (high)	Read-only	“”
0x0D-0x0F	n/a	Read-only	

navXUI Customization

The navXUI Source Code is Open-Source and can be customized using the following instructions:

- Download and install the free . NOTE: the current navXUI code is compatible with version 3.0beta5 of the Processing development environment.
- Checkout the navX-Micro source code.
- Copy the contents of the navX source code’s ‘processing’ directory to <User Directory>\Processing directory.
- Open the Processing IDE and then open the navXUI sketch via the File->Sketchbook menu.
- Compile/Run the navXUI by selecting the Sketch->Run menu.

If your computer has more than one serial port, you will need to select the appropriate serial port (corresponding to the USB serial port the navX-Micro is connected to) from the COM port selection drop-down list in the top-right of the navXUI display.

Technical References

The references on this page are provided to help students gain a deeper understanding of the algorithms, technologies and tools used within the navX-Micro and other Inertial Measurement Unit (IMU) and Attitude/Heading Reference System (AHRS) products. Additionally, links to other notable open-source works which could perhaps be adapted to work on the navX-Micro are included.