

Pigeon 2.0 User's Guide

Revision 1.2



Cross The Road Electronics

www.ctr-electronics.com

Table of Contents

1. Device description	5
1.1. Kit Contents	5
1.2. Features.....	6
1.3. Electrical/Inertial Specifications	7
1.4. General/Mechanical Specifications	7
1.5. LED States	8
1.6. Functional Diagram	9
1.6.1. Mount Orientated Signals	9
1.6.2. Enclosure Orientated Signals	9
1.7. Changes between Pigeon 1 and Pigeon 2	10
2. IMU Error Sources.....	11
2.1. Location – Center of Rotation	11
2.2. Temperature	11
2.3. Gyroscope Sensitivity Error	12
2.4. Avoid Magnetic/Ferromagnetic materials	13
2.5. Vibration.....	14
2.6. Saturated inputs	14
3. Calibration	15
3.1. Temperature Calibration	15
3.2. Gyroscope Bias Calibration.....	15
3.3. Gyroscope Sensitivity Calibration.....	15
3.4. Accelerometer Calibration	15
3.5. Compass Calibration.....	15
4. Boot behavior	16
4.1. Boot behavior - Test Results	17
5. Wiring	19
6. Orientation Convention	20
6.1. World Frame Reference	20
6.2. Euler Angles.....	20
6.3. Gimble Lock.....	20
6.4. Does X/Y Axis placement matter?	21
6.5. Default Mounting Orientation	21
6.6. Custom Mounting Orientation	21
6.6.1. Custom Mounting Orientation – Explicit Values	22
6.6.2. Custom Mounting Orientation – Phoenix Tuner	23
7. FAQ	24

7.1. Is there a way to tell if the device is present/powered?	24
7.2. How do I change the Mount Orientation?	24
7.3. What changes do I need to make when upgrading from Pigeon (1) IMU to Pigeon 2.0?	24
7.4. What are the requirements of Pigeon 2.0 when booting up?	24
7.5. Does the device support CAN FD?	24
7.6. Trying to keep robot from tipping over. Any suggestions?	24
7.7. Closed-looping an Arm. Any suggestions?	24
7.8. When Pigeon IMU boots, starting Yaw is not zero?	24
8. Mechanical Drawings.....	25
9. Revision History	26

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your CTRE products. To this end, we will continue to improve our publications, examples, and support to better suit your needs.

If you have any questions or comments regarding this document, or any CTRE product, please contact support@crosstheroadelectronics.com

To obtain the most recent version of this document, please visit www.ctr-electronics.com.

1. Device description

The Pigeon 2.0 is an inertial measurement unit (IMU) that can sense acceleration, angular velocity, and magnetic fields. With this information, Pigeon 2.0 can be used to sense a mobile platform's pose, which then can be used for a variety of applications.

1.1. Kit Contents

Each kit contains a single Pigeon 2.0 IMU.

IMU is preassembled and **includes wire leads for power and CAN bus.**

The CAN bus wire leads are terminated with **3-pin pre-installed connectors (0.1" pitch).**

One lead pair has a **female** connector, and the other has a **male** connector. This way, several CTR-Electronics CAN bus devices **can be daisy chained together.**



1.2. Features

- **Nine Degrees of Freedom** (3 Axis **Accelerometer**, 3 Axis **Gyroscope**, 3 Axis **Magnetometer**)
- Full AHRS: **Yaw, Pitch, Roll**
- **Quaternion** Output
- **Gravity Vector** Output
- Boot-up **does not require stillness**. Get useful heading information as soon as it is powered on.
- **Kalman** Filter fusion algorithm
- **Gyro** automatically **re-biases** after **4 seconds of no-motion**
- **Temperature Compensation** for temperature sensitive components
- **Temperature factory-calibrated**
- **Accelerometer factory-calibrated**
- **Gyroscope factory-calibrated**
- **No user-calibration** required for accurate **6-axis fusion**.
- Mount IMU in **any orientation** (**not** limited to horizontal or vertical orientation) ^(Note 1,2)
- **Heading** and **Yaw** are **continuous**, ideal for robot heading servos and motion control.
- **Enclosure** protects against debris
- **Wide Input Voltage Range** 6V – 28V
- Protection **Reverse Input Power Protection**
- **Polycarbonate housing** prevents debris from entering inside device
- **One 22 AWG wire lead-pair** for **power**
- **Two 22 AWG wire lead-pairs** for **CAN Bus** (3-pin connector, one male, one female) for **daisy chaining** devices
- **Robust bootloader** and **reliable field-upgrade** (no physical button required, no “stuck states” that requires user intervention)
- **Wirelessly check, configure and field-upgrade** using **roboRIO Wi-Fi** and **Phoenix Tuner**.
- Users can **download software API** binaries on our **reliable Maven server (99.9% reliability)**
- **Hardware Simulation** Support
- Supports **CAN bus** and **CAN FD bus**.
- Supported by **CANivore** and **roboRIO use case**

Note 1: Pigeon 2.0 can be configured for any mounting orientation. Other IMUs provide a fixed number of orientations, but Phoenix Tuner/ Phoenix API can be used to select any orientation.

Note 2: When using Euler angles, user must avoid Gimble Lock conditions for valid Yaw, Pitch, Roll values. This is a limitation of expressing pose using Euler angles, and not a limitation of the Pigeon IMU.

1.3. Electrical/Inertial Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Unit
T _{amb}	Ambient temperature		-40		+85	°C
I _{supp}	Supply Current	DC supply 12.0V DC supply 28.0V		40 21	46 23	mA
V _{dd}	Supply voltage		6.0	12.0	28.0	V
ESD Rating						
	ESD Protection Contact Discharge				±30	kV
	ESD Protection Air-Gap Discharge				±30	kV
Yaw Drift Rates						
DR _{NO-MOTION}	No-Motion Drift			0.12		Deg / hour
DR _{IN-MOTION}	In-Motion Drift			0.4		Deg / min
DR _{BOOT}	Boot-Into-Motion Drift			1		Deg / min
Gyroscope Specifications						
Range	Angular Velocity Measurement Range ^(Note 1)		±125	±1000	±2000	dps
Resolution	Resolution of angular velocity measurement			16		bits
Accelerometer Specifications						
Range	Range Acceleration Measurement			±8		g
Resolution	Resolution Acceleration Measurement			16		bits
Magnetometer Specifications						
Range	Magnetometer Range			±1150		uT
Resolution	Magnetometer Resolution			0.3		uT
Accuracy	Compass Accuracy (with proper calibration and placement)			1		deg

Note 1: Device firmware currently selects ±1000 dps. Future software updates may allow configuration based on customer feedback.

1.4. General/Mechanical Specifications

Outside Dimensions	1.77" x 1.77" x 0.51"
Weight	1.07 ounces (30.39g) w/ enclosure & wires
Supported Communication Protocols	CAN 2.0 (1Mbps) CAN FD with CANivore
Boot Calibration ^(Note 1)	0 seconds
Gyroscope bias no-motion time ^(Note 2)	4 seconds
Mechanical Shock ^(Note 3)	10,000 g (duration 200 us) 2,000 g (duration 1 ms)
Maximum Free fall ^(Note 3)	5'

Note 1: Drift rate will start at DR_{BOOT}.

Note 2: Gyroscope is re-biased after 4 seconds of no motion, which reduces drift rate to DR_{NO-MOTION}

Note 3: Stresses above listed rating may cause permanent damage to the device.

1.5. LED States

The Pigeon 2.0 features 2 tri color LEDs that indicate CAN bus health. This feature can be used to confirm proper CAN bus wiring. The table below shows the possible color states.

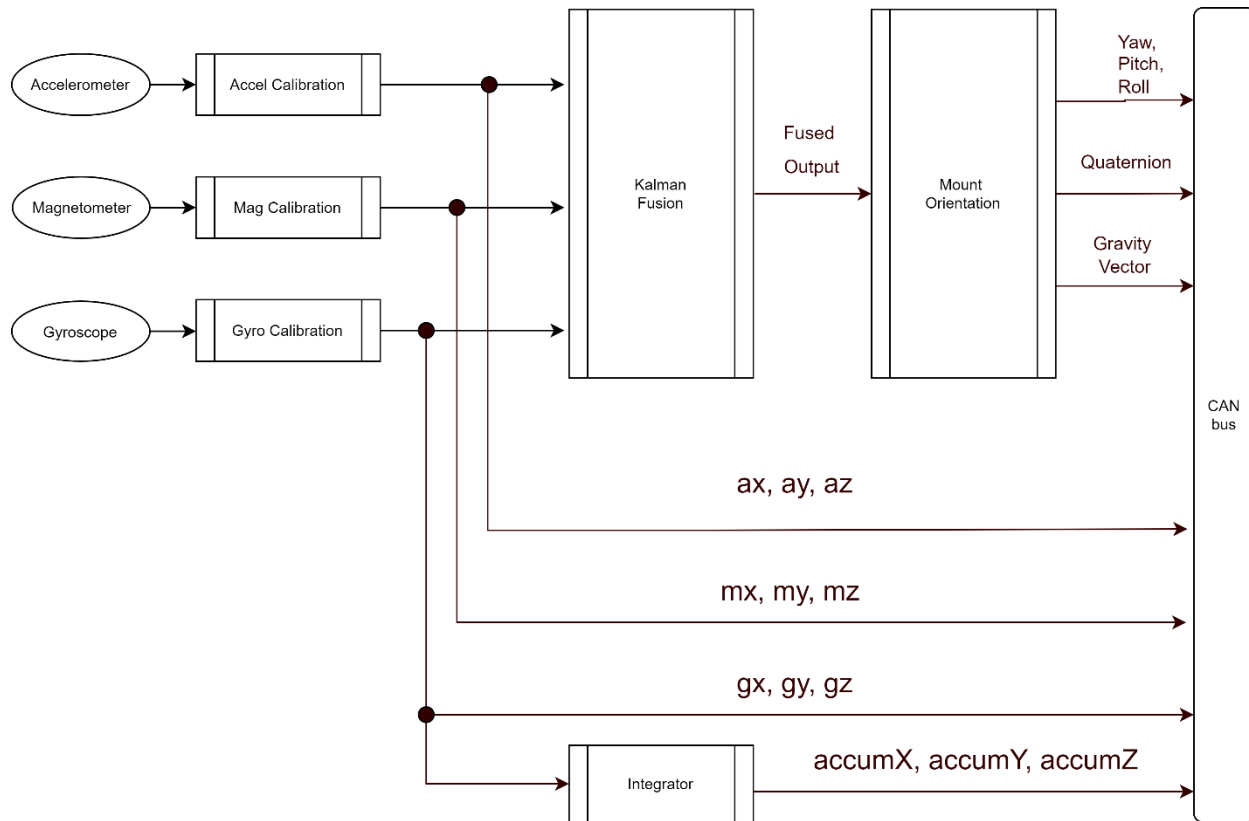
LED Color	Blink Pattern	Description
Off		Pigeon 2.0 is not powered/ plugged in. Check power cabling to the Pigeon 2.0.
Yellow/Green	Only one LED will blink this pattern.	Device is in bootloader, most likely because field-upgrade failed in middle of event. Inspect CAN bus wiring and re-field-upgrade using Phoenix Tuner. If device has valid firmware, turn device off, wait 10 seconds, and turn device on to boot strap it.
Red/Yellow	LEDs are never off – one of the two colors are always illuminated	Hardware is damaged
Red/Green	LEDs are never off – one of the two colors are always illuminated	Device has Phoenix Pro firmware and is unlicensed. Either connect this device to a Phoenix Pro licensed CANivore, apply a Phoenix Pro device license, or change firmware to use Phoenix v5.
Red Blink		Check CAN Bus health and connection to the Pigeon 2.0.
Yellow	Alternate Blinking ^(Note 1)	CAN bus detected. Robot controller is not present on CAN bus, or Pigeon2 software object not created in user application.
Yellow	Simultaneous Blinking ^(Note 2)	CAN bus detected. Robot is disabled.
Green Blink		CAN bus detected. Robot is enabled.

Note 1: Only one LED is on at any given moment.

Note 2: Both LEDs are on at the same time.

1.6. Functional Diagram

Signal paths are shown in the functional diagram below.



1.6.1. Mount Orientated Signals

Note that the **Yaw, Pitch, Roll, Quaternion, and Gravity Vector** are **compensated** for the **selected Mount Orientation**.

1.6.2. Enclosure Orientated Signals

The outputs of the **gyroscope, accelerometer and magnetometer** are **not compensated** for the **selected Mount Orientation**.

These signals are biased/calibrated to canonical units. ^(Note 1)

As a result, these signals are **referenced** to the **enclosure's orientation**.

If these *enclosure-oriented signals* are used, it is recommended to choose a Mount Orientation that is reasonably horizontal or vertical so that Yaw, Pitch, Roll, and Gravity Vector are also aligned to these signals.

Note 1: Magnetometer requires user calibration



1.7. Changes between Pigeon 1 and Pigeon 2

Pigeon 2.0 is the next evolution in Pigeon IMU. Listed below are the differences between the latest iteration and the original Pigeon IMU.

Symbol	Pigeon IMU (Version 1)	Pigeon 2.0
Boot Requirements	5 seconds of stillness	No requirements
6 Axis angle drift (no-motion)	15 deg per hour	0.12 degrees per hour
6 Axis angle drift (motion)	1 deg per minute	0.4 deg per minute (with 4 second no-motion event) 1 deg per minute (instantly after boot-up)
Axes conventions	+X points to right +Y points to forward +Z points to sky Pitch is about +X Roll is about +Y Yaw is about +Z	+X points forward. +Y points to the left. +Z points to the sky. Pitch is about +Y Roll is about +X Yaw is about +Z
Mount orientation	Must be mounted flat and level (Z axis pointed up)	Mount in any orientation. Orientation does not need to be purely vertical or horizontal. Pitch and Roll can be zeroed this way as well.
Enclosure	None	Polycarbonate Enclosure
Communication	CAN bus Gadgeteer Ribbon Cable	CAN bus CAN FD bus (CANivore)
User Requirements (Indoor 6-axis applications)	Temperature Calibration for best performance.	None for best performance. Select Mount Orientation if using non-default orientation.
General Software API Changes	PigeonIMU class WPI_PigeonIMU class	Pigeon2 class WPI_Pigeon2 class getState () is not available – there is no need as the IMU is always ready. getFusedHeading() is not available – there is no need as IMU Yaw is the fused heading in all circumstances. Use getYaw() instead. Full Software documentation available at: https://ctr-electronics.com/documentation

Note 1: Both Pigeon 1 and Pigeon 2 use a right-handed orientation, and Pigeon 2 defaults to Z-axis up (similar to Pigeon 1). However, Pigeon 2 uses an X forward orientation to better match the academic research papers used in the development of the product.

2. IMU Error Sources

All IMUs are susceptible to accrued error due to sources of error listed below. However, this can be avoided by following a few basic principles and careful IMU placement. The suggestions below are applicable to many IMUs, including those available in FRC Robotics.

Below is information and instructions for how to root-cause each error source using features of Pigeon 2.0

2.1. Location – Center of Rotation

Due to the fusion between the accelerometer and the gyroscope, excessive and sustained g-force caused by centripetal acceleration can impact the Yaw. This can be resolved by moving the IMU closer to the COR (center of rotation).

The root cause can be confirmed by printing/plotting the accelerometer magnitude while spinning the robot at maximum velocity. Ideally, the magnitude should be as close to 1G as possible.

Alternatively, the accumulated gyro Z value can be printed/plotted while performing a sustained high-speed rotation. If the accumulated gyro Z value is more correct than the Yaw, it sufficiently confirms the root cause.

Procedure:

1. First drive the robot to immovable flat obstacle. A wall works best against the robot's flattest surface
2. Zero the yaw and accumulated gyro Z
3. Drive the robot in a zero turn at max speed for ~30 seconds
4. Drive the robot in the opposite direction at max speed until it approaches 0 heading.
5. Drive back up against the obstacle and read Yaw and accumulated gyro Z.
6. If root-cause is confirmed, move IMU closer to COR and repeat procedure.

2.2. Temperature

IMUs (in general) also drift due to subtle changes in temperature. This is primarily due to the temperature's impact on the zero-bias for the gyroscope. However, it can also affect accelerometer and magnetometer. One strategy to reduce this is to continually re-bias the gyroscope whenever a no-motion event is detected (Pigeon 2.0 does this along with other market IMUs). This combined with an environment where temperature does not change rapidly can be adequate depending on the application.

Another strategy is on-the-fly compensation for changes in temperature. Pigeon 2.0 supports this as well and comes out of the box with a temperature calibration. If the temperature calibration is called into question, the user can disable it through a configuration parameter.

2.3. Gyroscope Sensitivity Error

Gyroscopes may have a range in the reported angular rate of their measurements (despite being advertised as being constant). As a result, a gyroscope will typically have **sensitivity error**. This can cause a consistent under/over-reporting of measured yaw compared to actual rotation.

Pigeon 2.0 gyroscope sensitivities are factory calibrated. However different applications may require end-user to provide additional trimming. The procedure below can be used to measure and trim the sensitivity error correction.

This can be measured by the following procedure:

1. First drive the robot to immovable flat obstacle. A wall works best against the robot's flattest surface.
2. Zero the yaw and accumulated gyro Z
3. Drive the robot in a zero turn at max speed for 10 rotations
4. Drive back up against the obstacle and read Yaw. It should be 3600 (\pm expected drift for the duration of the test) degrees.
5. Drive the robot in a zero turn at max speed in the opposite direction for 10 rotations
6. Drive back up against the obstacle and read Yaw. It should be 0 (\pm expected drift for the duration of the test) degrees

Using table below, determine if sensitivity error is likely.

	Step 4 Error is low	Step 4 Error is high
Step 6 Error is low	No sensitivity error	Possible sensitivity error
Step 6 Error is high	Inconclusive	Inconclusive

2.4. Avoid Magnetic/Ferromagnetic materials

Most IMUs have some type of compass calibration procedure, including Pigeon 2.0. However, if the IMU is placed in an area with severe magnetic disturbance, then minimally the benefit of compass is reduced, and maximally the robot heading may be severely incorrect. Although many IMUs (including Pigeon 2.0) will automatically detect invalid magnitude field strengths, depending on the type and direction of the distortion, the compass can still be affected. For this reason, compass-features are often utilized in an outdoor setting in a chassis designed to accommodate compass features.



The common disturbances are due to proximity to ferromagnetic materials such as iron/steel and proximity to magnetic materials (motors). Additionally, if an IMU is placed within an extreme magnetic field (e.g., proximity to neodymium, rare-earth magnets), the hard-iron offsets within the magnetometer may permanently change, requiring a new calibration. Care should be taken to ensure such magnets are not placed near the IMU PCB (e.g., CTRE Mag Encoder/CANcoder magnets).

This impact can be root-caused by waving a simple mechanical compass in the presence of where the IMU is intended to be placed. Look for any major changes in the

needle position as the compass is waved near the critical areas of the robot, while maintaining consistent orientation between compass and Earth.

Next place the mechanical compass in the location where the IMU is meant to be placed. While the robot is placed on blocks, drive all motors, chains, and articulators while watching the compass. If the compass needle reacts to robot actuation, then IMU should be relocated, or simply avoid relying on compass features if the application will allow it.

Phone compass applications (software) can be used to a lesser degree; however, a mechanical compass is best, and are widely available for <\$5.

Another means of confirming this source of error is to print/plot the compass magnitude as reported by the IMU while moving the robot into various poses. This can be compared with the expected magnetic field strength which can be determined with a Phone compass application, or with an online search such as...

<http://www.ngdc.noaa.gov/geomag-web/>

Any massive dip or rise in the magnetic norm may indicate magnetic disturbance.

However, it should be noted that there can be magnetic disturbance despite the norm not changing by much. For this reason, the mechanical compass testing above is recommended.

2.5. Vibration

Vibration of an IMU can cause errors by contributing noise to the accelerometer, gyro, and magnetometer. Additionally, if the vibration is severe enough, a sensor input may see a saturated value. A common example of this is vibration in the airframe of a UAV/drone (for example, and unbiased propeller).

2.6. Saturated inputs

Pigeon 2.0 will mark a sticky-fault if a sensor measurement is saturated. This is useful in determining if sensitivity configurations are too high, or if the Pigeon physical mount needs dampening. Typically, a rigid mount to the robot chassis is recommended, unless the IMU sensors are saturating, causing loss of information in the pose estimation.

3. Calibration

3.1. Temperature Calibration

Pigeon 2.0 is factory temperature calibrated. Additional user calibration is not necessary.

3.2. Gyroscope Bias Calibration

Gyroscope is factory calibrated. Additional user calibration is not necessary.

Additionally, the gyroscope's bias calibration is automatically adjusted by firmware during runtime.

3.3. Gyroscope Sensitivity Calibration

Gyroscope sensitivity is factory calibrated. However, the required sensitivity precision may be application dependent. As such there will be a future software update to allow for additional trimming.

If sensitivity error correction needs to be trimmed, calculate how many degrees your test rotation overshoot/undershot the expected result per rotation. Then configure the device with the overshoot/undershot per-rotation value.

Note that sensitivity error is positive if measurement overshoot or produced a magnitude above expected value. Similarly, sensitivity error is negative if measurement undershot or produced a magnitude below expected value.

Example below:

1. Perform test procedure in [Section 2.3](#).
2. IMU reports Yaw travel from 0 deg to -3602 deg.
3. However mechanically IMU traveled -3600 deg.
4. Therefore, measurement **overshot** by 2 deg.
5. Set corrective error amount to +0.2 deg per rotation (See Phoenix API or Phoenix Tuner)
6. Retest using above procedure to confirm improved result.

3.4. Accelerometer Calibration

Accelerometer is factory calibrated. Additional user calibration is not necessary.

3.5. Compass Calibration

2022 Pigeon 2.0 release firmware supports a compass calibration that requires rotating about the three principal axis.

A GitHub example will be made available demonstrating how to leverage this.

Compass can then be enabled via software configuration. Note that the 9-axis compass is recommend for **outdoor robotics applications** where stray magnetic fields caused by the environment are minimized.

We recommend leaving the compass disabled for **indoor robotics applications** as this means customers can use the Pigeon 2 as is, with no supplemental calibration procedures.

4. Boot behavior

Most IMUs requires no-motion during boot-up to tare the gyroscope, thereby reducing yaw drift. This includes the original Pigeon IMU (version 1.0), and several competitor IMUs.

Pigeon 2.0, however, **does not require no-motion during boot-up**. This is because Pigeon 2.0 can **predict the gyroscope bias during boot-up**. This allows it to produce reasonable heading information immediately on boot, even when in motion.

This is a particularly important feature to resolve complications due to:

- IMU is booting up in environment with vibration sources (fans, compressors, user-handling)
- IMU is booting up with extremely loud music in near proximity (also a vibration source)
- IMU is booting up while users are moving platform
- Cannot guarantee that users will ensure IMU is “ready” before executing robot action.

For more information on the drift rates in various configurations, see [Section 1.3. Electrical Specifications](#).

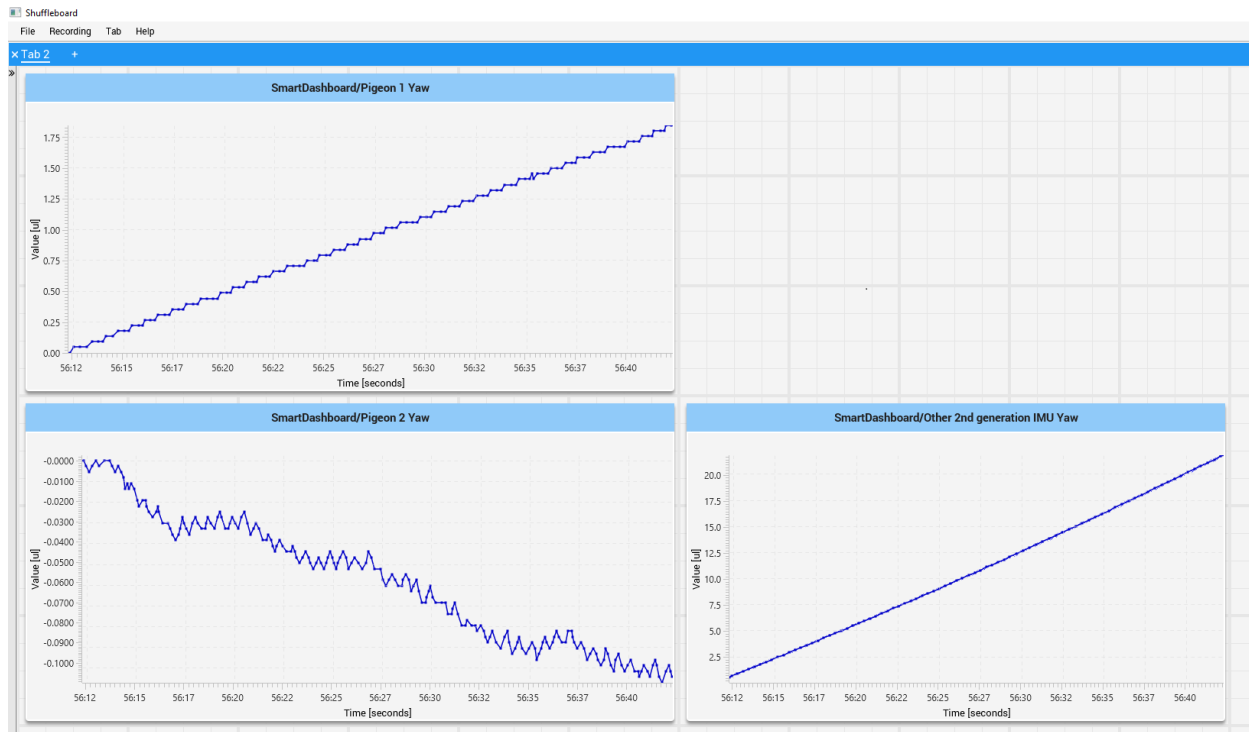
4.1. Boot behavior - Test Results

The test results demonstrating this feature is shown below. Test setup involved:

- **Pigeon 1.0** (documentation suggests five seconds of stillness required)
- **Pigeon 2.0** (stillness **not** required)
- **Competitor IMU** on roboRIO MXP port (documentation suggests five seconds of stillness required)

All three IMUs were mounted on a fixture that vibrates at approximately 7 Hz. IMUs were all power booted during vibration and drift was tracked over time. The ongoing vibration prevents all three IMUs from performing no-motion calibration.

4.1.1. Boot behavior – Drift Rates

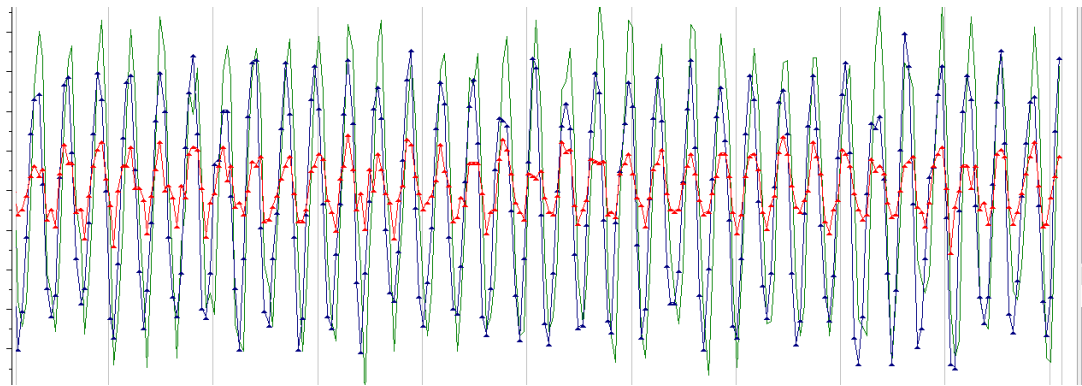


Device	Change in Yaw over 30 sec	Drift
Pigeon 1.0 (Note 1)	+ 1.8 deg	3.6 deg per min
Other 2 nd Generation Competitor IMU (Note 2)	+ 20 deg	40 deg per min
Pigeon 2.0	- 0.10 deg	0.2 deg per min

Note 1: Pigeon 1.0 User's Guide clearly states that IMU requires no-motion on bootup. We do not recommend deviating from the product documentation during use. The gyro bias prediction is only a feature with **Pigeon 2.0**

Note 2: Competitor IMU Documentation clearly states that IMU requires no-motion on bootup. We do not recommend deviating from the product documentation during use.

4.1.2. Boot behavior – Accelerometer



Note 1: Source of vibration is in the Y and Z axis, therefore X-axis measurements are reduced in comparison.

Legend	
ax	Pigeon 2 Accelerometer: X Axis
ay	Pigeon 2 Accelerometer: Y Axis
az	Pigeon 2 Accelerometer: Z Axis
X axis	0.5 seconds per gradient
Y axis	2.5 mg per gradient

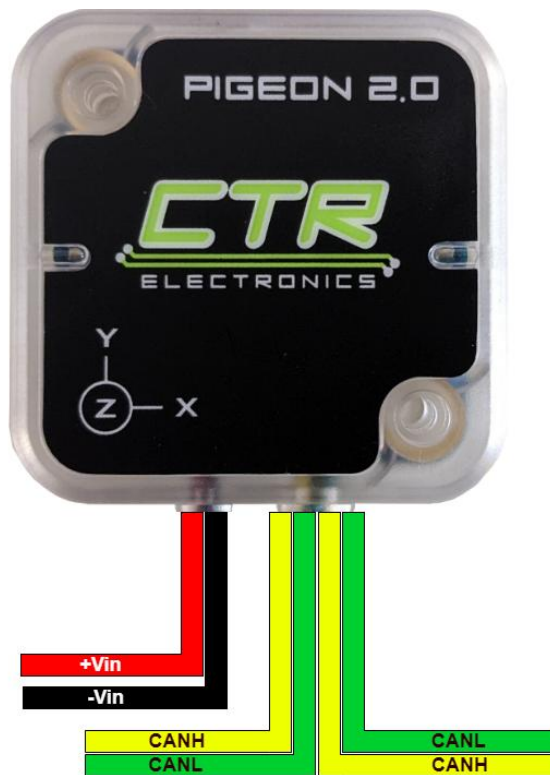
5. Wiring

Pigeon 2.0 comes with wires pre-installed on it. The leads should be wired following the table below.

Color	Signal
Red	Vdd
Black	Ground
Yellow (Note 1, 2)	CANH
Green (Note 1, 2)	CANL

Note 1: The two yellow wires are electrically common.

Note 2: The two green wires are electrically common.

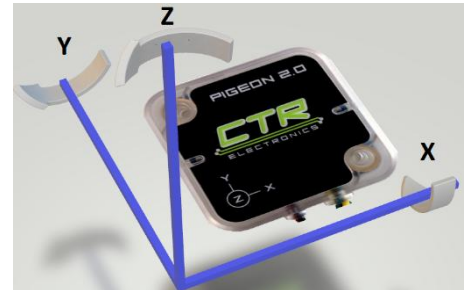


6. Orientation Convention

6.1. World Frame Reference

Pigeon2 treats +X as the forward axis, +Y as the left axis, and +Z towards the sky (right-hand orientation). In researching common axis orientations, this is a common way to define the world frame reference for ground-based vehicles.

When the Pigeon is using the **Default Mount Orientation**, these axes match the XYZ logo on the enclosure.

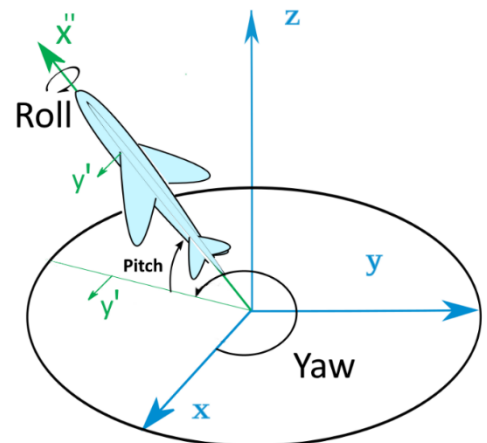


6.2. Euler Angles

For applications that require **Euler Angles (Yaw, Pitch, Roll)** to represent the pose within the world frame reference, Pigeon2 performs the following:

- First calculate **Yaw** by rotating about **+Z** in the **world frame reference**. (Turning to the left is positive).
- From this new body frame, calculate **Pitch** about the **IMU's local reference Y'** (Pitching nose down is positive, diagram demonstrates a negative pitch).
- Calculate a final **Roll** about the **IMU's local reference X''**.

Note that because **Yaw** is calculated first, it is done so in the **world frame reference (defined by gravity)**. Because of this, Yaw is defined as travel about the plane orthogonal to gravity. Therefore, the Pigeon 2.0 **does not require being aligned to gravity for reliable Yaw**.



6.3. Gimble Lock

Given the angle definitions used by Pigeon 2.0, **Gimble Lock** occurs when the **Pitch reaches near ± 90 degrees**.

At this location the tip of the airplane would be aligned with **World Frame Reference Z Axis**. At this point **Roll and Yaw cannot be distinguished** from each other. As a result, the Euler Angles are not reliable near this orientation.

For applications that **require 360 degrees of vertical travel**, it is strongly recommended to use **Roll** instead of **Pitch**. **Roll** allows for complete 360 rotation with no risk of ambiguity. This is because **Roll** is the **final calculated angle** and cannot introduce ambiguity if **Pitch** is sufficiently away from Gimble Lock.

Note that Gimble Lock is a **limitation** of using **Euler Angles**. In other words, there is always a location that produces this erroneous condition. More advanced applications may avoid this by using:

- Gravity Vector
- Quaternion – which can provide the Line of Rotation (LOR), and amount of rotation over LOR
- Combinations of the above with or without Euler Angles

But for relatively simple applications (ground vehicle, single axis arms, etc.), simply choosing an IMU orientation that will not encounter Gimble Lock is sufficient.

6.4. Does X/Y Axis placement matter?

Many users simply want to measure Yaw for ground-based vehicles. In which case the purpose of ensuring X or Y axis points to the front/rear/or side of the platform may not be critically necessary. This choice determines which tilt axis is considered Pitch and Roll, and signages (positive Roll vs negative Roll), which some applications may not require.

If deploying multiple platforms, the best strategy is to be consistent with the mounting orientation across the entire fleet of vehicles, or individually tune configuration so Yaw, Pitch, Roll behave the same on each vehicle.

6.5. Default Mounting Orientation

The default orientation for calculating Euler Angles, Gravity Vector, and Quaternion matches the XYZ logo on the enclosure. In this configuration, **Gimble Lock** will occur if **enclosure X axis is parallel to gravity (points to sky or ground)**.

If the orientation ensures **enclosure X axis** is not near parallel to gravity, then the **default Mount Orientation settings are acceptable**. This includes common orientations where the Pigeon 2.0 is **Z-up, Z-down, Y-up, and Y-down**.

Alternative orientations that risk **enclosure X axis** reaching **near parallel to gravity** should use a **Custom Mounting Orientation** configuration to avoid Gimble Lock.

6.6. Custom Mounting Orientation

An improvement over the original Pigeon is that the Pigeon 2.0 can **apply an internal rotation** to adjust the reported **Gravity Vector, Euler Angles, and Quaternion**.

This is important if the mounted orientation of the Pigeon risks placing the IMU into Gimble Lock. In fact, this method essentially moves where Gimble Lock occurs, ideally to a position your mechanism will never reach.

Custom mount orientation may also be used to effectively **zero Pitch and Roll**. This can be useful for balance applications (arm for example).

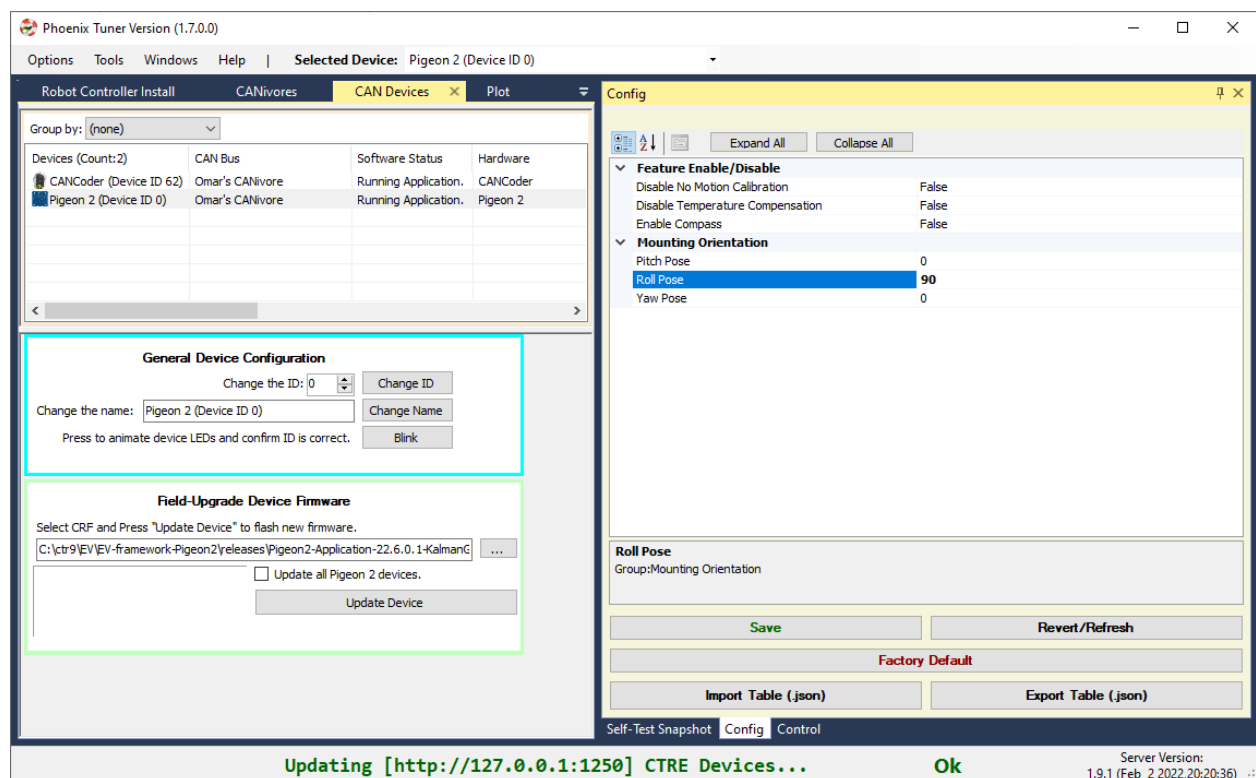
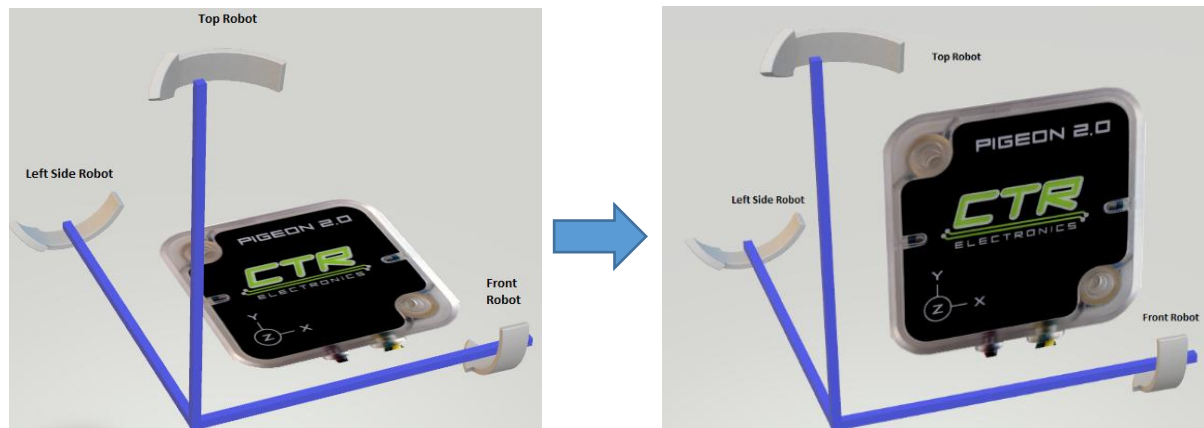
There are two methods for configuring the Mount Orientation:

- Specify them explicitly in Tuner Configuration Tab or Phoenix API routines.
- Use Phoenix Tuner to calculate these values

6.6.1. Custom Mounting Orientation – Explicit Values

As an example, suppose the mount orientation is modified so that Pigeon is upright (vertical) inside the robot platform. This would be expressed as rotation about the IMU's **X Axis** or a **+90 degree Roll**.

If the Mount Orientation Configuration Values (Yaw, Pitch, Roll) are set to (0 deg, 0 deg, +90 deg) ^{Note (1,2)}, the Pigeon 2 will re-orient the Euler Angles (and Gravity Vector / Quaternion) so that Pitch and Roll will report 0 deg. Additionally, rotation about the top-robot axis will be measured as Yaw (ideal for ground vehicles).



Note 1: The Mount Orientation Angles can be specified with Phoenix Tuner or with Phoenix API.

Note 2: The order of the signals in Tuner may be alphabetically ordered, and not Yaw/Pitch/Roll. Take care when entering values manually.

6.6.2. Custom Mounting Orientation – Phoenix Tuner

When using Phoenix Tuner, The **Pigeon 2 Mount Calibration** tab may be used to configure a custom mount orientation.

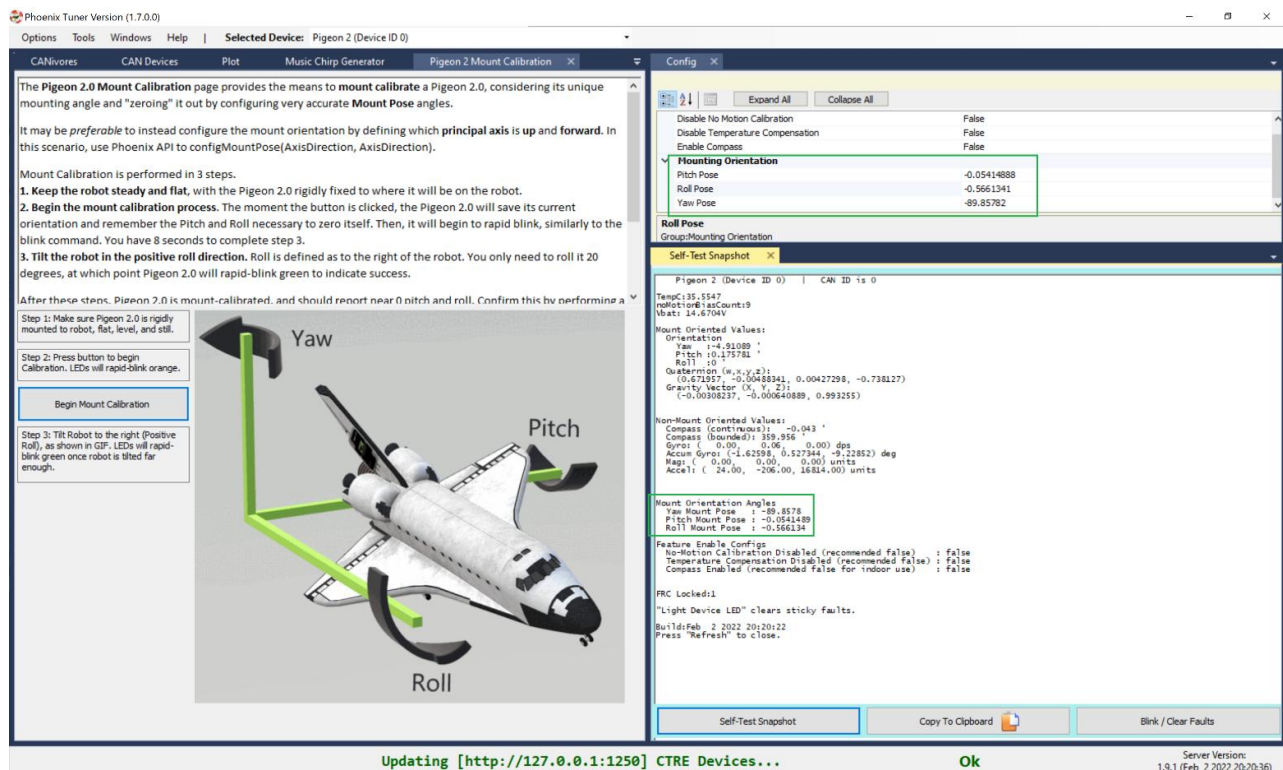
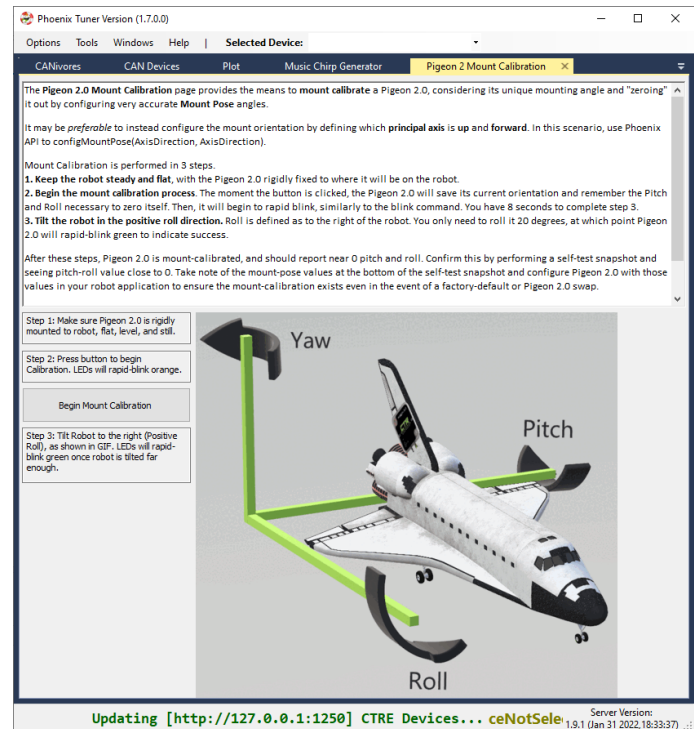
Once Tuner has discovered the Pigeon 2, user may press the “Begin Mount Calibration” button.

Once pressed, Pigeon LEDs will rapid-blink orange (same LED pattern as “blink” command in Tuner).

Then tilt the robot. Once robot is sufficiently tilt, Pigeon will rapid-blink green briefly to confirm success. The axis the robot was tilted about will become the **Roll axis**.

Confirm near-zero Pitch and Roll using Self-Test. Confirm basic Yaw by rotating robot and checking Self-Test.

The calculated configuration values can be read in the Configuration Page or in Self-Test.



Note 1: The order of the signals in Tuner may be alphabetically ordered, and not Yaw/Pitch/Roll. Take care when reading values.

7. FAQ

7.1. Is there a way to tell if the device is present/powered?

To determine visually if the device is powered and functioning, check the built-in LED, see [Section 1.5](#).

7.2. How do I change the Mount Orientation?

CTRE provides API that allows explicit selection of Mount Yaw, Pitch, and Roll.

Additionally, Phoenix Tuner allows for an automatic configuration that also sets these configuration values.

Software will be released and available for download at <https://ctr-electronics.com/software>.

7.3. What changes do I need to make when upgrading from Pigeon (1) IMU to Pigeon 2.0?

The bottom of the table in [Section 1.7](#) has general software guidance on this.

7.4. What are the requirements of Pigeon 2.0 when booting up?

Pigeon 2.0 has no hard requirements when booting. This means Pigeon 2.0 will start providing useful orientation information as soon as power is applied to it. For more information on the drift rates of Pigeon 2.0, look at [Section 1.3. Electrical Specifications](#).

7.5. Does the device support CAN FD?

Yes, Pigeon 2.0 supports CAN FD with a CTRE CANivore.

7.6. Trying to keep robot from tipping over. Any suggestions?

Generally, robots are tipped due to a combination of a high center of gravity, and excessive acceleration by the drivetrain.

In advanced cases, software can be developed to prevent the robot from tipping by monitoring the Gravity Vector. If the robot is upright, the Z component will contain nearly the entire magnitude of gravity (1.0 g). If the robot is tipping, the X and Y component will reflect this. Pitch and Roll may also be used for this purpose.

7.7. Closed-looping an Arm. Any suggestions?

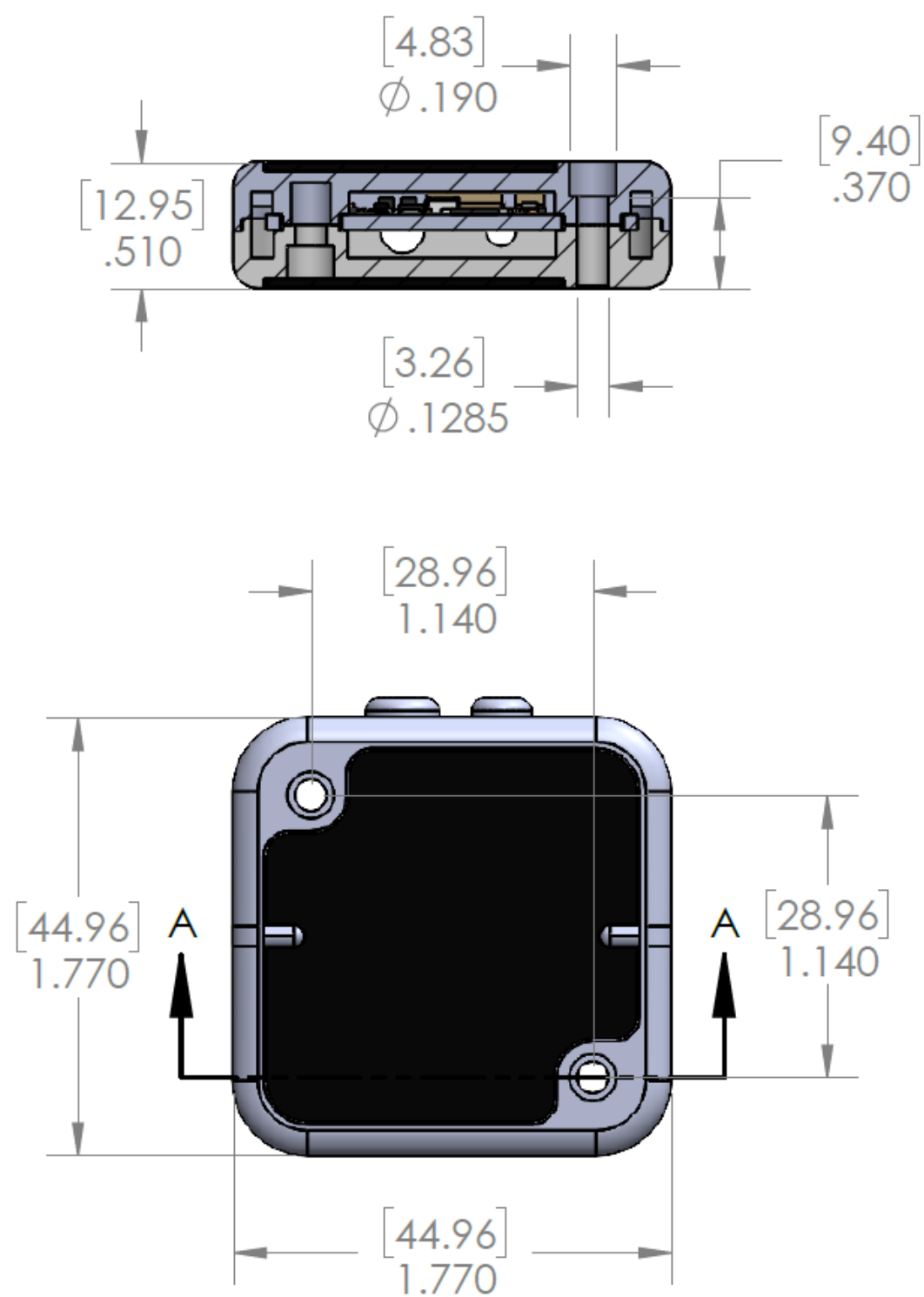
If the Pigeon 2.0 is placed on a vertically rotating platform, choose a Mount Orientation so that the rotation causes a **Roll**. This ensures 360 degrees range in measurement without risk of Gimble Lock.

Alternatively using the Gravity Vector may also prove useful as this cleanly provides the individual X, Y, Z components of the arm orientation, and may be used as a direct feed-forward for closed-loops.

7.8. When Pigeon IMU boots, starting Yaw is not zero?

Depending on the ship firmware flashed into the device, the initial Yaw angle may be nonzero after bootup. Phoenix API allows setting/taring the Yaw angle to accommodate the needs of the application.

8. Mechanical Drawings



9. Revision History

Revision	Date	Description
1.0	31-Jan-2022	Initial Release
1.1	9-Nov-2022	Fixed reliability number
1.2	13-Feb-2023	Added Phoenix Pro unlicensed blink code