TORC

ROBOTIC BUILDING BLOCKS

PinPoint™

Localization & Precision IMU

405 Partnership Drive, Blacksburg VA 24060 Ph: 540.443.9262 Fx: 540.443.3667 www.torcrobotics.com

Legal Notices

All information contained in this manual is believed to be accurate at the time of printing; however, TORC reserves the right to make modifications to the specifications and operation of this product without obligation to notify any person or entity of such revision.

Limited Warranty Terms and Conditions

Subject to the following terms and conditions, TORC warrants that for a period of one (1) year from date of purchase that this TORC product ("Product") will substantially conform to TORC's publicly available specifications for the Product and that the Product will be substantially free from defects in materials and workmanship. To obtain additional information on TORC's product limited warranty, please contact TORC.

Product firmware may incorporate portions of the lwIP networking stack, copyright ©2001-2004 Swedish Institute of Computer Science. The source code and full text of the associated license agreement may be downloaded from http://savannah.nongnu.org/projects/lwip.

Software Fixes

During the limited warranty period, the customer will be entitled to receive such Fixes to the Product software that TORC releases and makes commercially available and does not charge separately. TORC reserves the right to determine, in its sole discretion, what constitutes a software fix. To obtain additional information on TORC's product software fixes and firmware upgrades, please contact TORC.

How to Obtain Warranty Service

To obtain warranty service, please contact TORC and be prepared to include the following information:

- Your name, company, address, e-mail address, phone number
- Proof of purchase
- Product model and serial number
- Explanation of the problem

Limitation of Liability

DO NOT OPERATE UNTIL USER MANUAL IS REVIEWED AND UNDERSTOOD. PRODUCT USE IS SUBJECT TO STRICT TERMS AND CONDITIONS. SEE TERMS AND CONDITIONS DOCUMENT FOR ADDITIONAL USE RESTRICTIONS. OPERATING PRODUCT IN VIOLATION OF USER RESTRICTIONS COULD RESULT IN PRODUCT MALFUNCTION, PROPERTY DAMAGE, AND PERSONAL INJURY INCLUDING DEATH.

USER ASSUMES ALL RISKS ASSOCIATED WITH POSSESSION OR USE OF PRODUCT AND RELATED SYSTEMS. USER AGREES TO INDEMNIFY, DEFEND AND HOLD HARMLESS TORC ROBOTICS, LLC ("TORC") FROM ANY DAMAGES ARISING OUT OF POSSESSION OR USE OF PRODUCT AND RELATED SYSTEMS. TORC IS NOT LIABLE FOR ANY DAMAGES OF ANY KIND.

FCC Compliance

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation. The equipment listed generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications.

1. Table of Contents

2. Table of Figures

3. Table of Tables

4. Introduction

The PinPoint™ localization system is a continuous positioning system for ground vehicles. PinPoint™ provides multi-sensor fusion of dual-GPS receivers, inertial sensors, and wheel speed sensors to provide real time position, orientation, velocity, and time information. All outputs are continuously updated regardless of a GPS fix, allowing operation during GPS degradation or complete signal loss.

Figure 1: PinPoint™ Filter Output vs. Raw GPS Data

4.1 Hardware Architecture

PinPoint™ operates with either a low-cost internal IMU or a high-precision external IMU and supports both GPS and GLONASS reception with OmniSTAR or Terrastar corrections. A generic electrical wheel-speed interface connects with a variety of sensors. A CAN port is also available for PinPoint™ to interface with existing vehicle speed sensors.

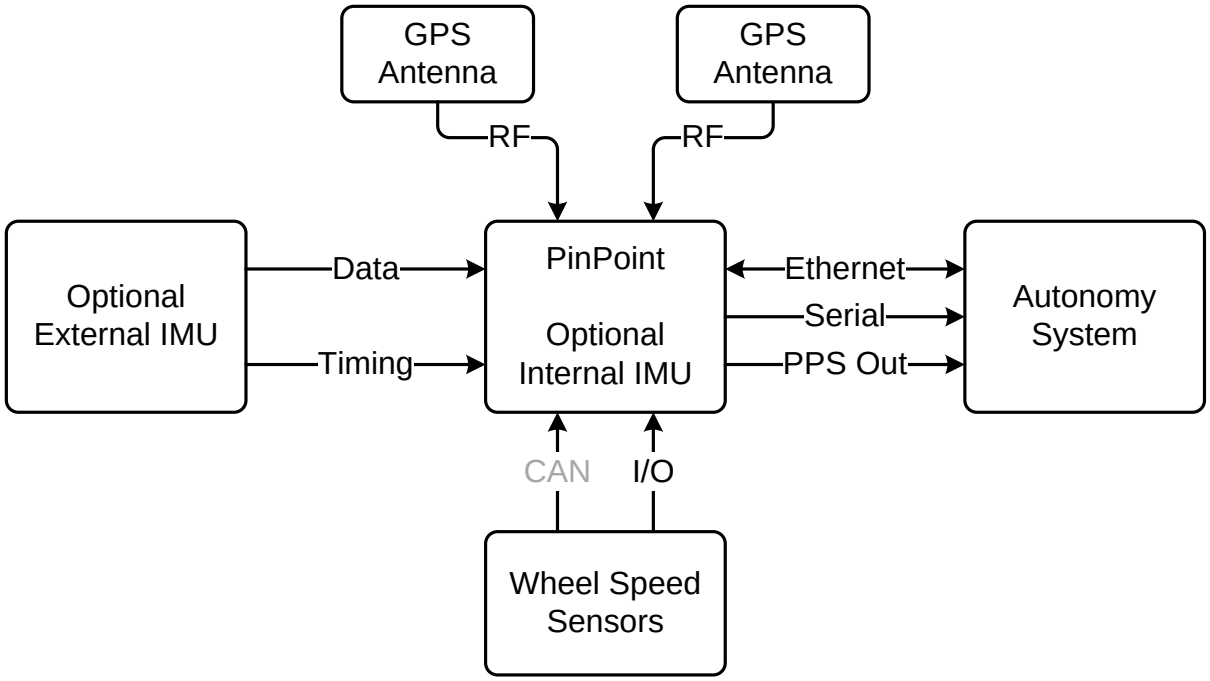


Figure 2: PinPoint™ Hardware Overview

4.2 Software Architecture

PinPoint is based on an error-state, multiplicative, extended Kalman filter that estimates the vehicles position, velocity, and attitude. The IMU measurements propagate these estimates as the vehicle moves, and the GPS and wheel speed sensors are used to correct the estimates. A simplified block diagram of the Localization filter is shown in Figure 3.

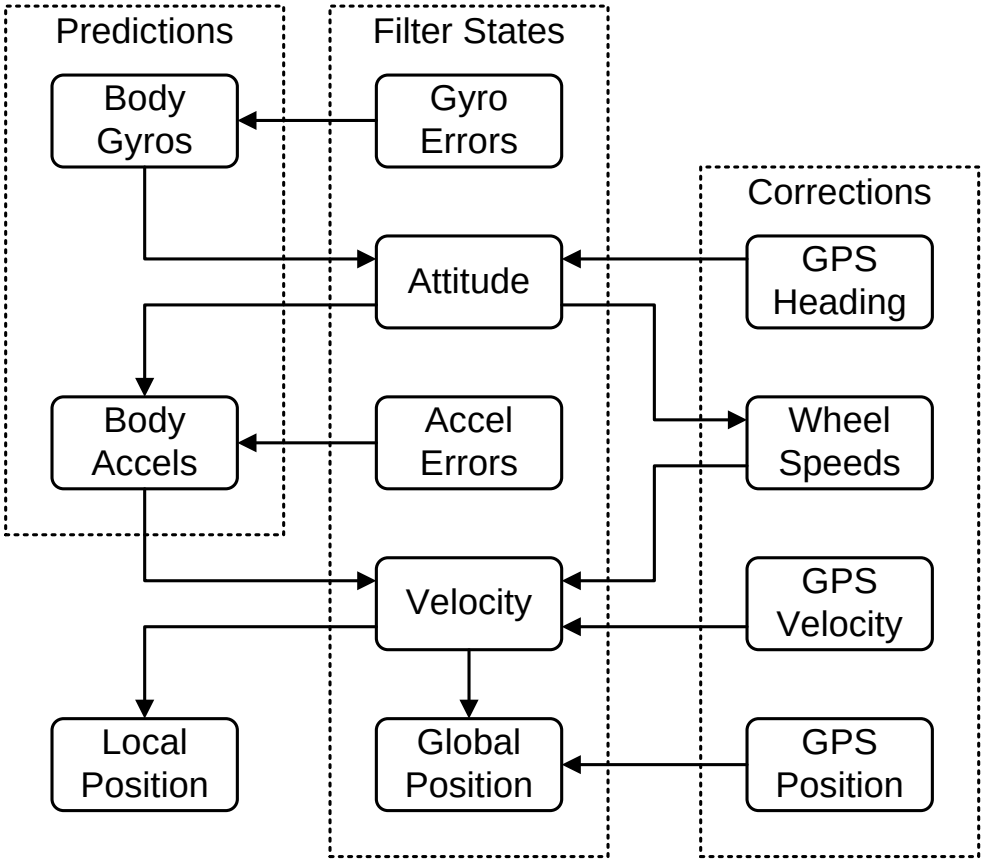


Figure 3: PinPoint™ Filter Overview

After a filter reset, an initialization and alignment routine is performed before the filter starts processing. This routine sets the initial states and covariance, and requires a valid GPS signal from one antenna for the initial global position. To determine the initial vehicle heading, the filter requires either vehicle motion, or a valid GPS signal from both antennas. Once alignment is complete, the filter will continue to run with or without actively receiving GPS updates.

4.3 Coordinate System Overview

PinPoint™ uses three different coordinate frames. A global frame is used for representing where in the world a vehicle is located, a local frame is used for representing where the vehicle is relative to nearby objects, and a vehicle frame is used for representing vehicle velocities.

The interaction between the local and global frames is analogous to rolling a plane on top of a sphere, where the contact point is the current vehicle position. Both the local position (the location of the contact point on the plane) and the global location (the location of the contact point on the sphere) change as the vehicle moves across the surface of the earth. The origin of the local plane is not fixed to any global coordinate, and any transforms between the two coordinate systems are done with respect to the current vehicle location. Refer to Section 12 for the mathematical relationship between local and global frames.

The vehicle frame provides a coordinate system that is referenced to the vehicle origin, unlike the local or global frames. The vehicle coordinate frame is used for reporting velocities and is typically aligned with the vehicle's primary direction of travel.

4.3.1 Global Position

PinPoint™ uses world geodetic (WGS-84) latitude, longitude, and height (LLH) above the ellipsoid to represent global position. This frame should be used for any globally referenced waypoints or object locations. Accuracy estimates of the global position are provided in meters.

4.3.2 Local Position

PinPoint™ uses a floating, north-aligned local frame in north, east, down (NED) coordinates to represent local position. The local position is initialized on filter reset, and is intended to provide a stable frame for local navigation, for instance, to represent the location of objects in the vicinity of the vehicle. The local position is updated as the vehicle moves but never corrected; allowing calculations performed in local frame to be immune to GPS "pops", where the solution quickly changes as satellites come into and go out of view.

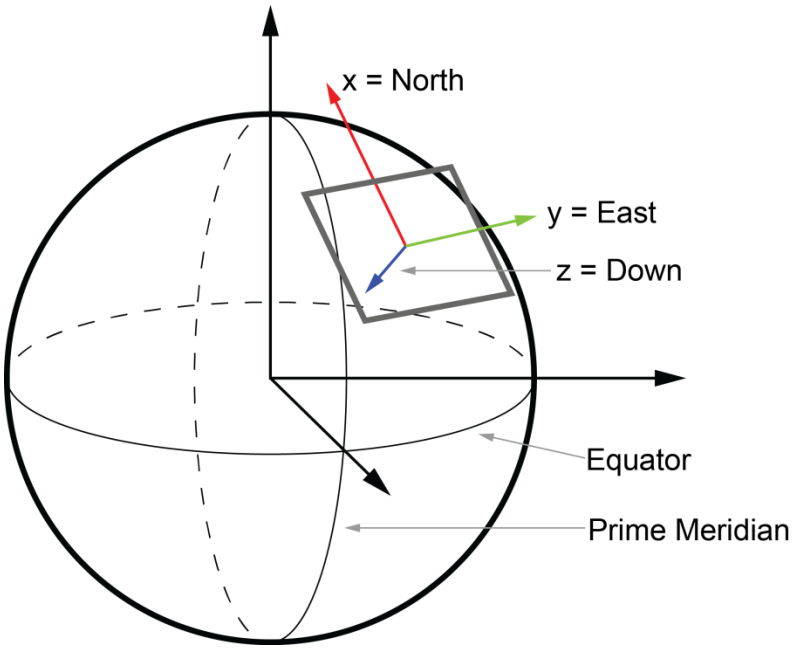


Figure 4: Local Coordinate Frame

It is important to note that any Cartesian projection will introduce some sort of distortion. Other projections such as UTM introduce an angular heading error (frame north is not equal to true north) and requires logic to be implemented for traversing different zones. PinPoint's™ local frame does not have discrete zones and is always aligned to true north; however it does introduce a distortion that may be observed when traversing extended distances or travelling near the poles. For example, if you were to drive due east, the local frame would show you driving in a straight line in the east direction. In actuality, you would have to be turning to the left in the northern hemisphere or to the right in the southern hemisphere to maintain your due east heading.

Because the local position may drift over time, no error estimates are output for the local frame. The velocity error estimates may be monitored if an estimate of how quickly the local position is drifting is desired. This drift may cause the origin of the local frame to float with respect to the global frame, so the current position in both frames must be used to understand their relationship.

4.3.3 Vehicle Body

The vehicle specific parameters and the velocity state output are represented in vehicle body frame. This frame is represented using forward, right, down (FRD) convention, with the associated rotations given as roll, pitch, yaw, respectively. The origin of the frame should be placed at the center of the rear wheels for Ackerman vehicles or center of mass for skid-steer vehicles. See Section 5 for setup and calibration of PinPoint™.

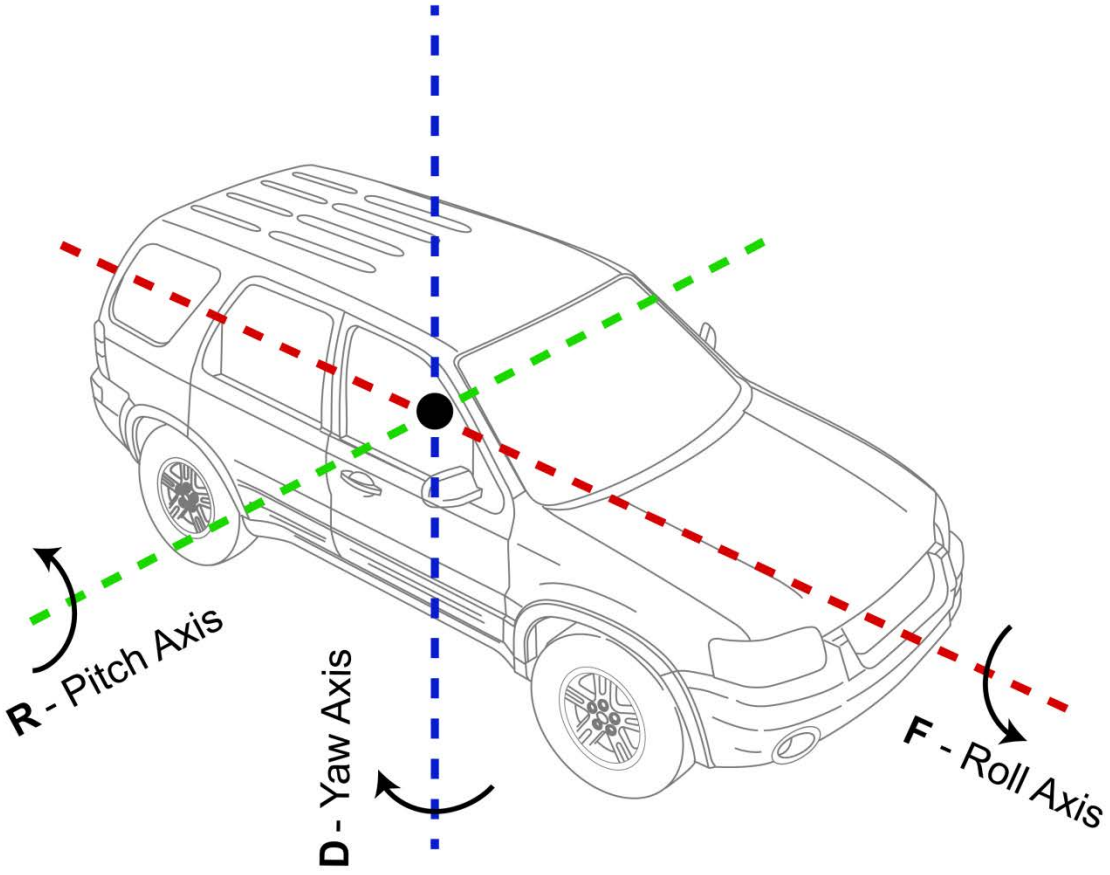


Figure 5: Vehicle Coordinate Frame

The orientation, given in Euler angles, represents the rotation from the local tangent plane to the vehicle body frame. The rotation order is roll, pitch, yaw if performed in the local NED frame, or yaw, pitch, roll if performed in the vehicle FRD frame.

4.3.4 IMU Frame

The IMU frame is only important for mounting, as the XYZ axes are rotated to align with the vehicle FRD axes for all calculations. The IMU orientation can be configured via the "vehicle" tab of the web interface, shown in Section 6. The IMU X-axis points out of the connectors, the Z-axis points down, and the Y-axis completes the right-handed coordinate frame. The origin and orientation of the IMU frame is shown in Sections 14 & 15.

5. Installation

⚠ TORC recommends that you read this section before installation of PinPoint™.

5.1 Unpacking and Inspecting Shipment

Visually inspect the shipping cartons for any signs of damage or mishandling before unpacking. Immediately report any damage to the shipping carrier. The shipment will include one or more shipping cartons depending on the quantity, model, and accessories ordered. Open the shipping cartons and make sure all of the components on the bill of lading are present. Report any problems discovered after you unpack to both TORC and the shipping carrier.

5.2 Installation Considerations based on Environment Conditions

Avoid installing PinPoint™ in locations with extreme environmental conditions including:

- Corrosive fluids and gases
- Heat greater than 71°C
- Cold less than -33°C
- High shock and vibration

Avoid installing PinPoint™ near sources of electrical and magnetic noise typically found in:

- Engines (spark plugs)
- Computer monitors
- Alternators and generators
- Electric motors
- Radio antennas
- Power converters and
- Switching power supplies

5.2.1 Installing and alignment of PinPoint™ internal and external precision IMUs

The PinPoint™ internal IMU and precision IMU (referred to as "IMU" in this section) should be mounted in a location that can be accurately measured with reference to the vehicle frame (FRD) from the vehicle origin. The vehicle origin is typically the center of the rear wheels as shown in Figure 5. The measurement should be made between the IMU origin (Section 13) and the vehicle origin. It is also important to accurately measure the rotation of the IMU's XYZ frame to the vehicle's FRD frame. These measurements will be saved in the web configuration utility as described in Section 6. While measurements of the linear offsets within 5cm will typically provide good results, accurately measuring the angular offsets is critical, as GPS-denied errors will be proportional to the sine of the angular error.

Vibration Considerations
Select a mounting location in area with minimal vibration. High angular vibration is treated as high velocity noise for the GPS and wheel speed measurements due to the moment arm between the IMU and these sensors. This noise will effectively down-weight the measurements, limiting the long-term drift performance of the overall system.

5.2.2 Mounting GPS Antenna(s)

Choosing the correct location for the GPS antennas is critical to performance. Follow the following guidelines to selecting the optimal GPS antenna placement:

- Choose an area that does not occlude the antenna from any portion of the sky
- Avoid mounting close to electrical cables, metal masts, and other antennas.
- Mount the antennas so that they are level when the vehicle is parked on level ground.
- Place the two GPS antennas as far apart as possible to improve the accuracy of heading corrections. TORC recommends a baseline between the two GPS antennas of at least 1 meter.
- Avoid areas with high vibration, excessive heat, electrical interference, and strong magnetic fields.

When configuring PinPoint™, it will be necessary to accurately measure the location of the antennas in vehicle frame (FRD) from the vehicle origin. Typically the FRD is referenced from the center of the rear wheel as shown in Figure 5. Measurements should use the antenna's phase center as the antenna location; please see Section 16. This location is then entered into the web configuration utility.

Installing GPS Antenna Cables
After mounting the GPS antennas, PinPoint™, and/or the Precision-IMU, route the GPS antenna cable. Avoid the following:

- Tight bends or kinks in the GPS antenna cable
- Sharp or abrasive surfaces
- Rotating or moving equipment
- Door or window jams
- Corrosive fluids or gases
- Hot surfaces such as exhaust manifolds or engines
- Areas in which people have routine access

The minimum bend radius for the GPS coaxial cable should be four-times the diameter of the cable. Secure the cable at several points. It is recommended that strain relief is provided for antenna cables.

5.2.3 Wheel Speed Sensors

PinPoint™ supports two pairs of quadrature wheel encoders used to sense the speed and direction of both left and right wheels. Wheel encoders are not supplied with the PinPoint™ product as most customers already have wheel encoders integrated into their platform or prefer a tightly integrated solution.

Each sensor pair should be configured in quadrature, meaning that one of the sensors lags the other electrically by 90 degrees, and the pulses should have a duty cycle between 40% and 60%. The wheel speed sensor pulses-per-meter (PPM) calibration factor must be calculated and entered on the vehicle tab of the web configuration. This value is calculated by dividing the number of counts per revolution by the wheel circumference, and can be fine-tuned using the included diagnostic utility. While higher pulse counts result in less error, care should be taken to select a pulse count that, combined with the wheel circumference and maximum expected vehicle speed, does not exceed the maximum input frequency of 30 kHz.

The 2-wire interface on PinPoint™ consists of a 7.5 volt current-limited supply, on which the output current is measured. The current is interpreted as follows:

- Less than 2mA - open-circuit fault
- 2~7 mA – logical 0
- 7~10 mA – hysteresis
- 10 ~ 18 mA – logical 1
- Greater than 18 mA – short-circuit fault

This interface is intended to be connected directly to 2-wire hall-effect sensors. However, several other common sensor types can be used with minimal interface circuitry. For 3-wire hall-effect sensors with open-collector outputs, the sensor power should be connected directly to PinPoint™ and the output should be connected to the power through a 750 ohm resistor.

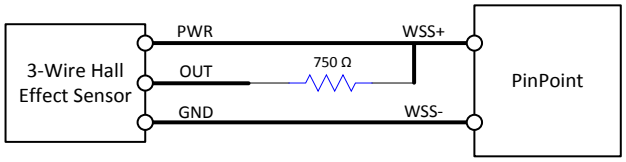


Figure 6: Hall Effect sensor configuration

For encoders with open collector outputs, the output should still be connected to PinPoint™ through a 750 ohm resistor; however, a 1.5k ohm resistor should be connected to the inputs near the sensor to allow PinPoint's open-circuit detection to function properly. The encoder should be powered from a separate supply.

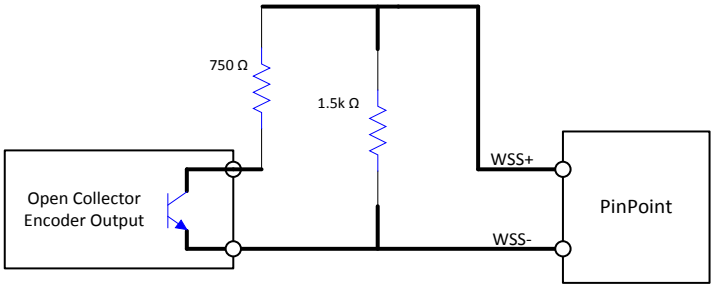


Figure 7: Encoder wheel speed configuration

5.3 Power

PinPoint™ will operate off voltages from 9 to 36 VDC, allowing it to be directly powered from a 14VDC or 28VDC vehicle bus. PinPoint™ contains active protection circuitry on the power supply input that protects from under-voltage, over-voltage, and reverse polarity. The pin-out of the PinPoint™ power input can be found in Section 11.

5.4 Ethernet

PinPoint™ provides two 10/100baseT Ethernet ports for primary communication over an IP-based network (Section 8). An embedded webserver allows for product configuration, firmware updates, and KML generation (Section 6). Also available over Ethernet is a built in NTP server allowing for system-wide time synchronization. The two Ethernet ports are connected to an internal switch, allowing customers to daisy-chain Ethernet devices. More information on the Ethernet connectors can be found in Section 11.

5.5 PPS outputs

Three pulse-per-second (PPS) signals are output whenever PinPoint™ is powered, even when a GPS fix is unavailable. These signals all have a 10% duty-cycle, but differ in polarity and voltage levels for connecting directly to a variety of 3rd-party components. These signals are shown in Figure 8.

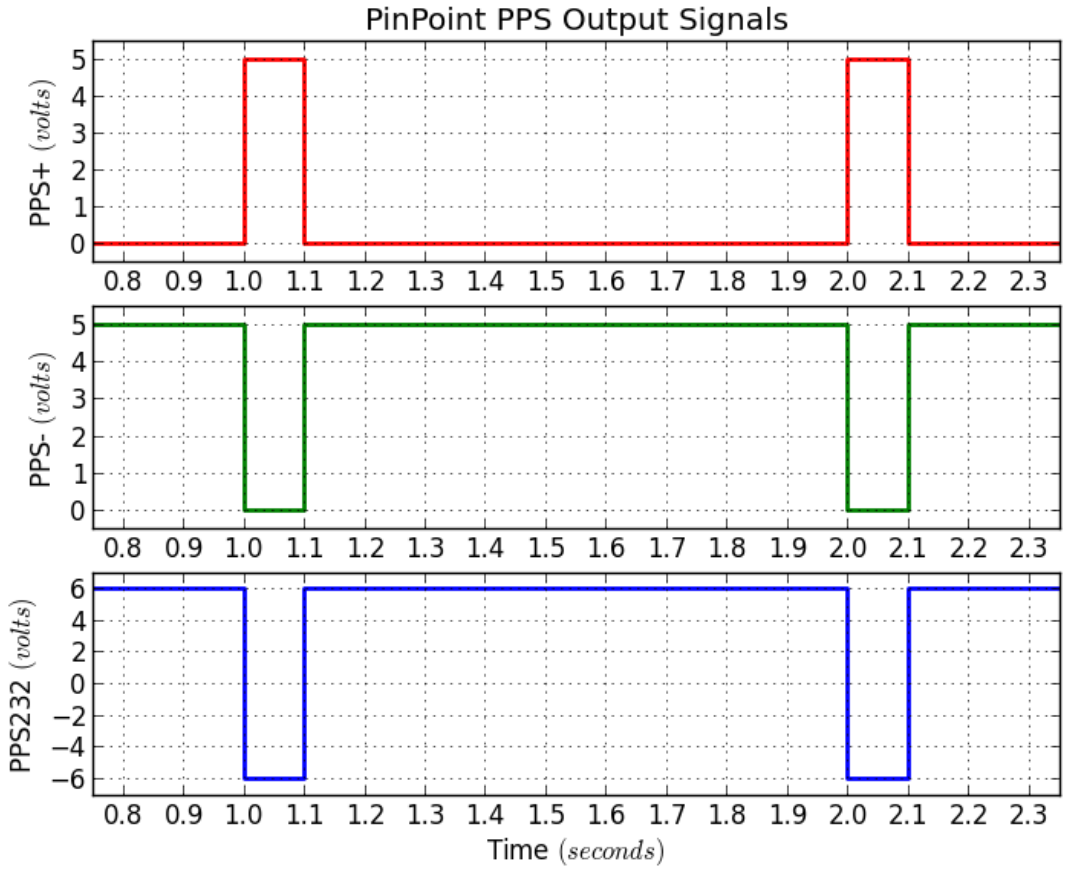


Figure 8: PPS output signal detail

5.5.1 NMEA serial stream for sensor time synchronization.

PinPoint™ includes an output-only serial port that sends a $GPRMC NMEA message once per second. The port is configured to 9600 baud, and is intended for direct connection with sensors for time synchronization, such as those made by Velodyne Lidar.

The format of the GPRMC message is as follows:

```
$GPRMC,204653,A,,,,,,080313,,,*05
          1   2          9      12

      1   204653     Time Stamp - HHMMSS
      2   A          Validity – A:ok, V:invalid
      9   080313     Date Stamp - DDMMYY
      12  *05        Checksum
```

5.5.2 Installation Validation

The installation can be validated with the included diagnostic utility, outlined in Section 9.

6. Web Configuration

PinPoint™ includes an embedded webserver to allow for basic status and configuration. Using a computer on the same local network as PinPoint™, navigate to http://172.24.0.29/, where 172.24.0.29 is the default IP address of PinPoint™. All modern browsers are compatible with the PinPoint™ web interface; however, if you are using an older browser, please ensure it has support for XSL transformations.

6.1 System Status Webpage

The status tab provides a snapshot of the localization filter outputs. The current global position, attitude, body velocity, system time, and odometer are all displayed. Additionally, filter error and warning messages are displayed at the bottom in red. Clicking the "Automatic Refresh" link at the bottom of the page will toggle between a single refresh of the web page and an automatic refresh every second.

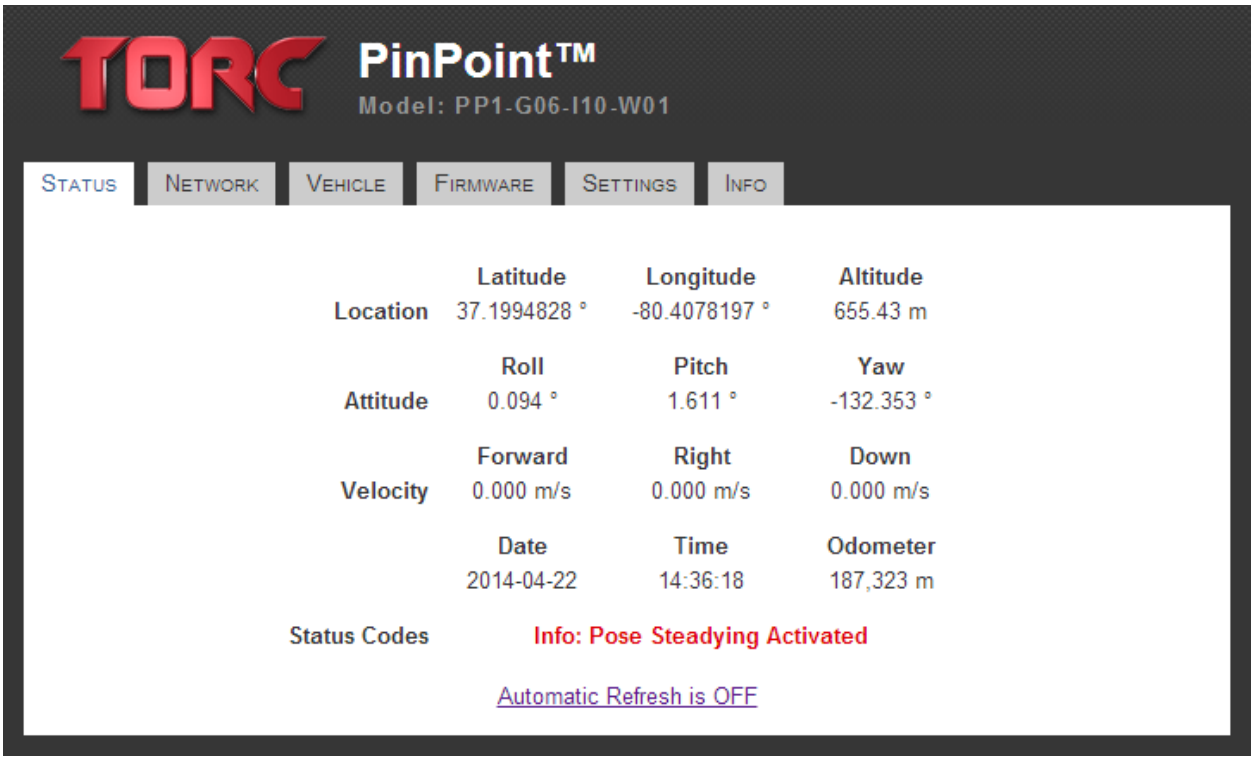


Figure 9: Status webpage screenshot

6.2 Network Parameters Webpage

The network tab allows configuration of standard networking parameters.

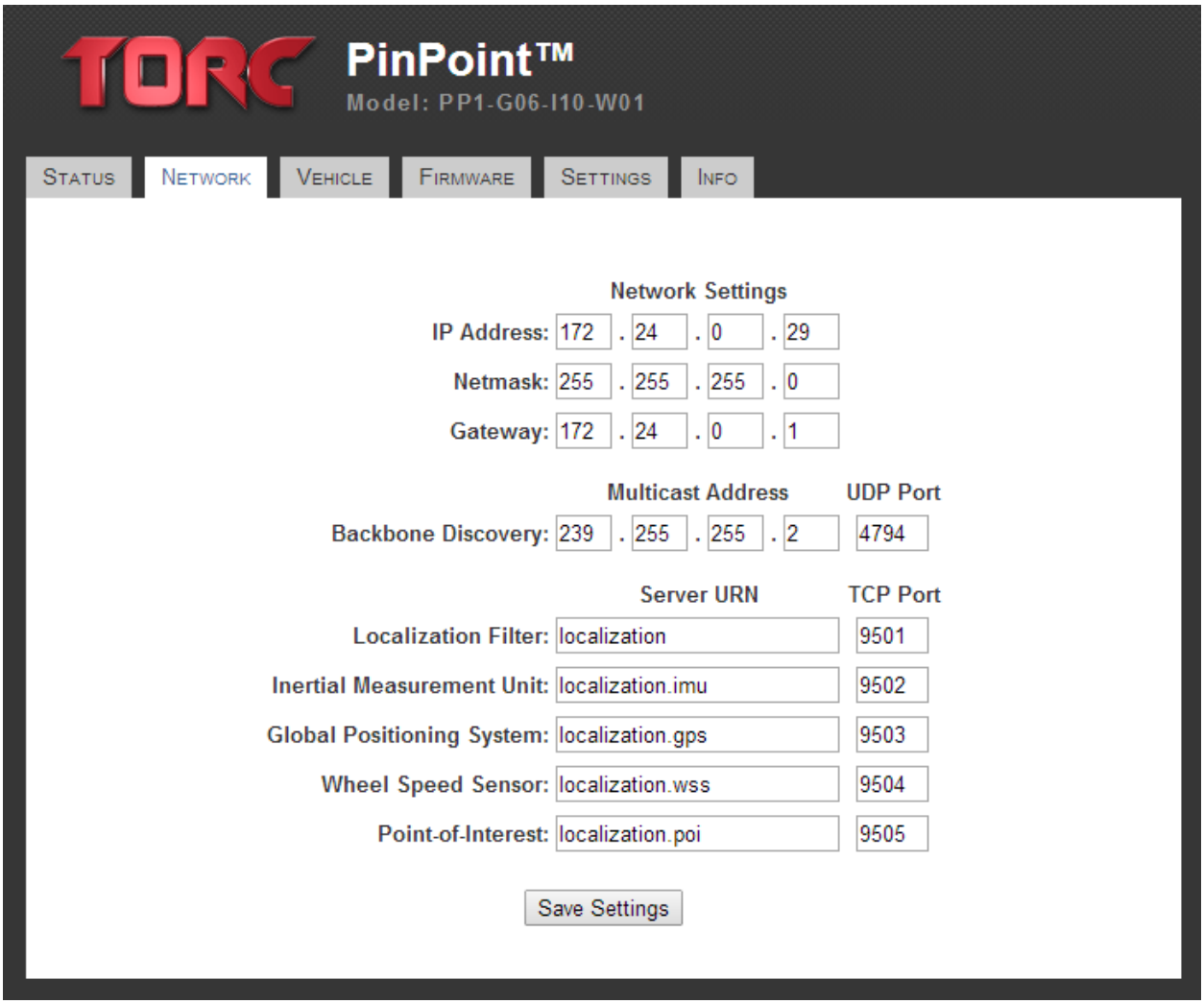


Figure 10: Network webpage screenshot

6.2.1 IP Address

The IP address should be set to a unique IPv4 address accessible to all computers that need to communicate with PinPoint™.

6.2.2 Netmask

The Netmask defines the scope of IP addresses directly accessible to PinPoint™, and any outgoing packets destined for IP addresses outside this local subnet will be sent to the gateway address.

6.2.3 Gateway

The gateway address is the IP address of the router configured to forward packets to other subnets.

6.2.4 Backbone Discovery

This multicast address and port is used in conjunction with the URNs by other TORC products and software to automatically discover PinPoint on the network. The defaults should not need to be modified unless otherwise instructed by TORC.

6.2.5 Server URNs

These names should be unique on the network and are used in conjunction with the multicast address and port by other TORC products and software to automatically discover PinPoint on the network. The defaults should not need to be modified unless otherwise instructed by TORC.

6.2.6 TCP Ports

These ports, in conjunction with the IP address, are used to connect to the various servers on PinPoint. Valid port number settings range from 1024 to 49151, but under most circumstances the default values should be left as is.

6.3 Vehicle Parameters Webpage

This tab is used to configure all of the vehicle specific lever arms and calibration parameters used by the localization processor.

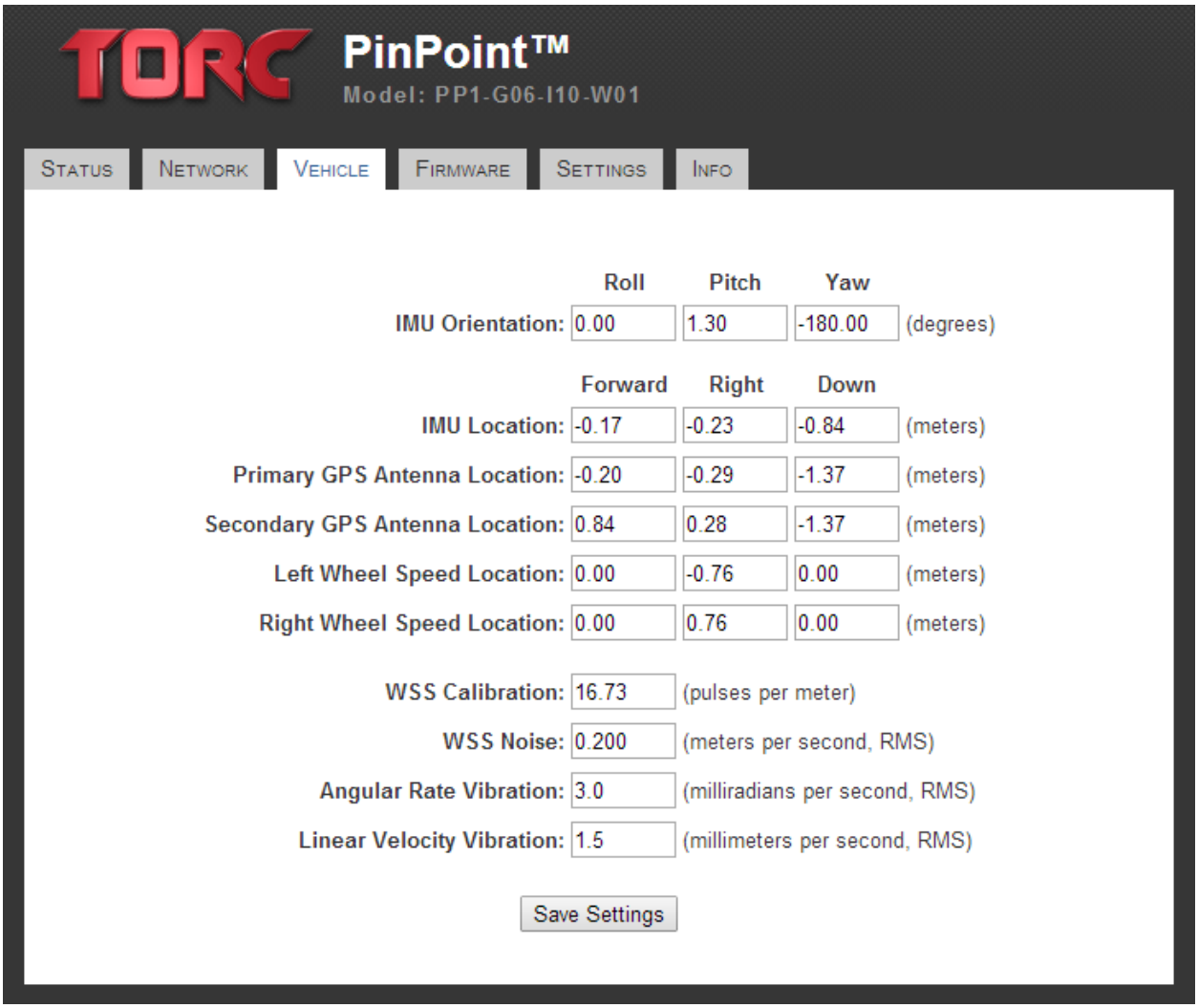


Figure 11: Vehicle webpage screenshot

Depending on factory configured options, not all of the parameters may be present.

6.3.1 IMU Orientation

The right handed rotations from the vehicle frame (FRD) to the IMU frame (XYZ). The order of rotations is roll-pitch-yaw when applied in vehicle frame, or yaw-pitch-roll when applied in IMU frame.

6.3.2 IMU Location

The offset from the vehicle origin to the IMU origin. Since firmware release 12, it is no longer necessary to place the origin at the center of the rear wheels, and any arbitrary point may be used for the origin.

6.3.3 Primary Antenna Location

The offset from the vehicle origin to the phase center of the primary GPS antenna. The primary antenna is used to determine the position of the vehicle. The primary antenna is connected to J1.

6.3.4 Secondary Antenna Location

The offset from the vehicle origin to the phase center of the secondary GPS antenna. The secondary antenna is used to determine the heading of the vehicle. The secondary antenna is connected to J2.

6.3.5 Left Wheel Speed Location

The offset from the vehicle origin to the measurement point of the left speed sensor. This point is typically the center of the contact patch for an Ackerman steered vehicle, or outside the tracks or wheels on a skid-steer vehicle. If a single speed sensor pickup is located before the rear differential, or if the vehicle has a locked differential or solid rear axle, this should be centered (left-to-right) on the vehicle.

6.3.6 Right Wheel Speed Location

The offset from the vehicle origin to the measurement point of the left speed sensor. This point is typically the center of the contact patch for an Ackerman steered vehicle, or outside the tracks or wheels on a skid-steer vehicle. If a single speed sensor pickup is located before the rear differential, or if the vehicle has a locked differential or solid rear axle, this should be centered (left-to-right) on the vehicle.

6.3.7 WSS Calibration

The distance the vehicle travels between wheel speed sensor pulses. For example, if the sensor pickup uses 24 holes per wheel rotation and the wheel has a circumference of 2 meters, there are 12 pulses per meter of distance travelled. This value can be fine-tuned using the included diagnostic utility.

6.3.8 WSS Noise

An estimate of the wheel speed measurement noise. Vertical suspension travel and vehicle sideslip should be accounted for in addition to raw sensor noise. The localization processor assumes the vehicle is stationary for speeds below this threshold. A value of 0.1 to 0.3 is appropriate for most vehicle types.

6.3.9 Angular Vibration

The localization processor assumes ridged body constraints between the GPS antennas and IMU. This parameter is used to account for any rotational velocity vibration that violates this constraint. Typical values with the IMU mounted to a vehicle with a gas engine range from 3 to 50 milliradians per second. This value can be measured using the included diagnostic utility.

6.3.10 Linear Vibration

The localization processor assumes ridged body constraints between the GPS antennas and IMU. This parameter is used to account for any linear velocity vibration that violates this constraint. Typical values with the IMU mounted to a vehicle with a gas engine range from 1 to 10 millimeters per second. This value can be measured using the included diagnostic utility.

6.4 Firmware Updates Webpage

The firmware tab allows customers to take advantage of the latest PinPoint™ features and bug-fixes by updating the firmware. PinPoint™ firmware will have a filename similar to “localization_S10.bin,” where

S10 is the version number. Browse to this file by clicking the "Browse…" or "Choose File" button in your browser, select the new firmware file, and click the "Update Firmware" button. PinPoint™ will load the firmware into flash memory, and prompt to reset the board. On the next boot, PinPoint™ will take an additional minute to copy the file into program memory before running the updated firmware.

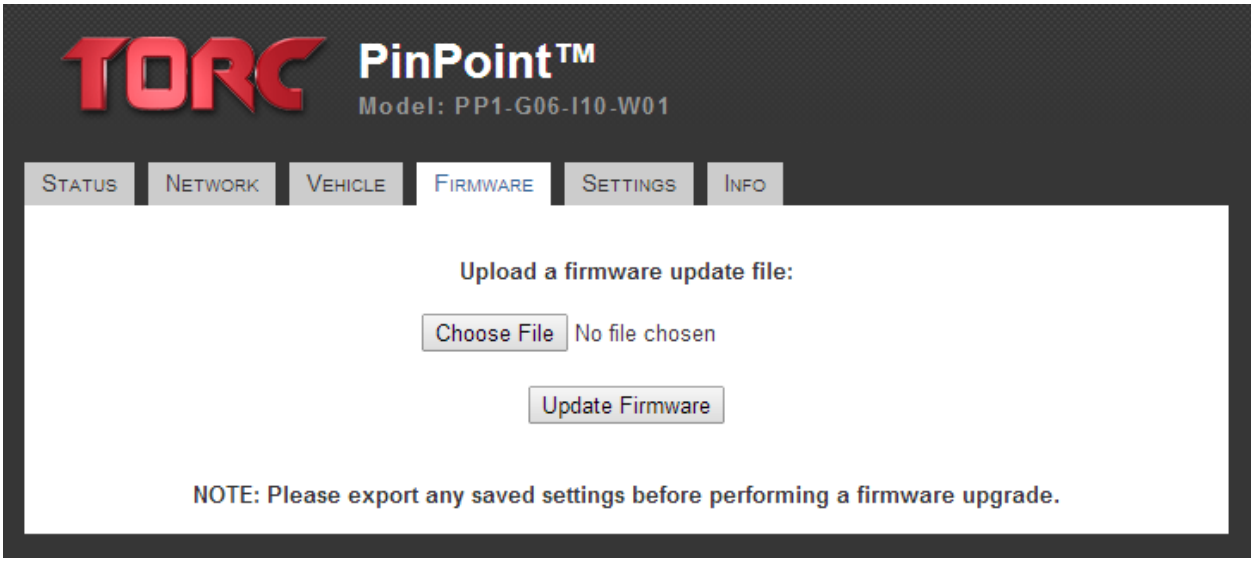


Figure 12: Firmware webpage screenshot

6.5 Settings Webpage

PinPoint™ settings are stored in a human-readable XML file that can be backed up and restored via the settings tab of the web interface. To back up the settings, click the "Download Settings" button on the web interface and save the file to a local directory on your computer. To restore a backup, browse to this file by clicking the "Browse…" or "Choose File" button in your browser, select the configuration file, and click the "Import Settings File" button. PinPoint™ will load the configuration into nonvolatile memory, and prompt to reset the board. Additionally, factory default settings can be restored by clicking the "Reset Settings" button.

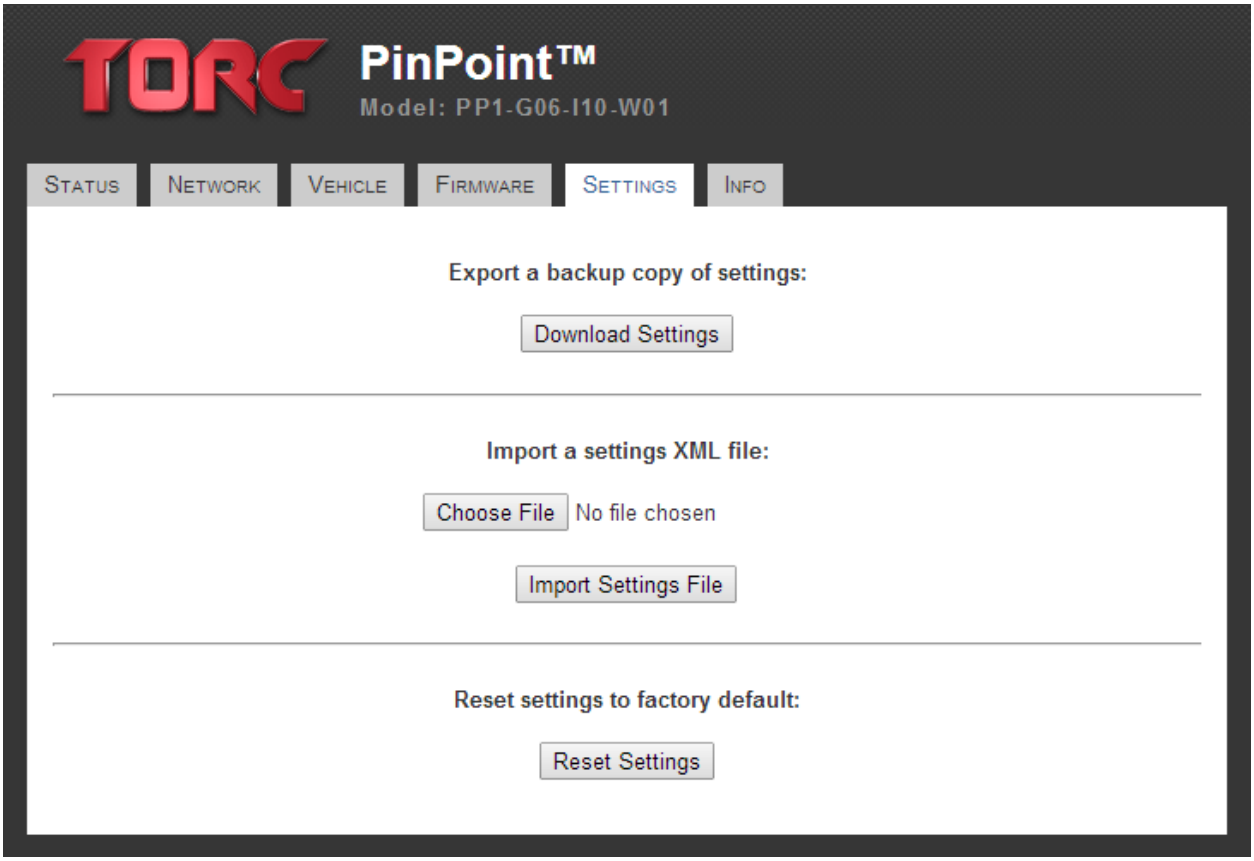


Figure 13: Settings webpage screenshot

6.6 Info Webpage

The info tab displays information about the hardware and software version that PinPoint™ is running, as well as diagnostic information such as board temperature and wheel speed sensor input currents.

Figure 14: Info webpage screenshot

6.7 KML Generation

PinPoint™ also includes built-in generation of real-time KML data suitable for use with Google Earth. Google Earth is freely available from http://earth.google.com/ and provides worldwide mapping and aerial imagery coverage. Opening http://172.24.0.29/localization.kml, with 172.24.0.29 replaced with PinPoint's IP address, will cause Google Earth to display an arrow indicating the current location and heading, and a blue line indicating the past track of the vehicle. This will refresh automatically every second. Double-clicking the arrow will bring up a slider showing the time history, and will allow the user to replay the data. Clicking on the blue track will show the velocity at that point in time.

7. Restore Network and System Defaults

In addition to the "Reset Settings" button on the web interface, factory defaults can also be restored by connecting pins 13 and 14 on connector J3, and applying power for 5 seconds. Make sure to disconnect pins 13 and 14 and cycle power before attempting to connect to PinPoint's default IP address of http://172.24.0.29/.

Resetting the factory defaults will not modify any of the factory configured options nor clear the odometer.

8. Software Interface

8.1 Interface Protocol Overview

PinPoint™ is designed to provide data for a variety of applications. The following section is a complete list of the entire software interface. A single application may not utilize all of the available functionality.

There are three approaches for obtaining information from PinPoint™. The first approach is by directly requesting the data through the available methods. This approach requires a method call every time data is requested. To receive periodic data, the second approach, a method can be scheduled with the scheduleMessage call. This will allow the scheduled method to be called at a configurable periodic rate. The final approach utilizes signals. By subscribing to available signals, via the *connectSignal* protocol message, PinPoint™ will send data as soon as new data is available. Subscribing to signals is recommended to minimize latency and avoid the aliasing possible when scheduling a message at a different rate than the data is available.

Connecting to PinPoint™ requires a specific initialization routing that includes sending a specific set of protocol message commands as outlined in Figure 15. These protocol message commands must be sent over TCP and should be the first messages sent. Each message server (Localization Filter, GPS, IMU, WSS, and POI) requires a separate connection to be opened by user client software. A maximum of 32 connections are supported.

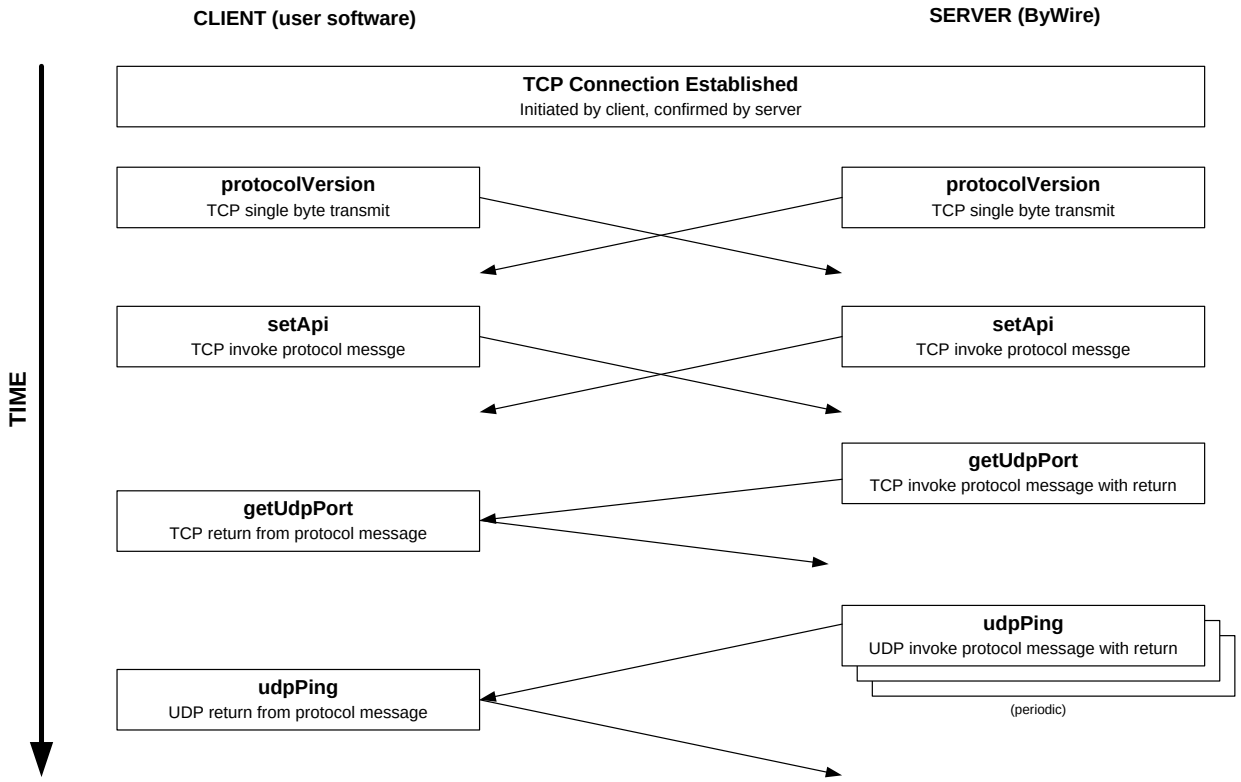


Figure 15: Initialization Flow Diagram

The packet structure for all messages is shown below in Table 1. All variables are stored in little endian format.

Table 1: Packet Structure

Message Header			Message Body
Control Byte 8 bits	ID 8 bits	Payload Length Defined by Control Byte	Payload

The *ControlByte* describes the type of message and contains a protocol flag, message type, and length of the data size parameter as shown in Table 2.

Table 2: Control Byte Format

Bits	Name	Control Byte Interpretation
[7]	ProtocolFlag	0: service specific message 1: protocol message
[6:3]	MessageType	0: Return Value 1: Invoke 2: Invoke With Return 3: Signal Emitted 4: Schedule Return 5: Error 6: Invoke With Return Code 7: Return Value With Code 8: Exception 9: Schedule Exception 10: Exception With Code
[2]	Reserved	
[1:0]	SizeLength	0: No size field (also no data) 1: Single byte size field (uint8) 2: Two byte size field (uint16) 3: Four byte size field (uint32)

The *ProtocolFlag* is used to distinguish protocol methods from specific device messages. The *MessageType* is an enumeration describing the contents of the message. Not all message types are necessarily supported for every message ID. Finally, the *SizeLength* is used to determine the representation of the *PayloadLength* parameter.

The next component of the message header, *ID,* is a unique identifier for the specific method type included in the interface. For *ReturnValue, Invoke, InvokeWithReturn,* and *Exception* message types, the *ID* field is interpreted as a Method ID, for *SignalEmitted* a Signal ID, for *ScheduleReturn* and *Schedule Exception*, a Schedule ID, and for *Error* the ID is interpreted as an error code.

The *SizeLength* parameter is used to indicate the number of bytes used by the *Size* parameter. This *Size* parameter is sent as the first part of the payload, and is followed by *Size* number of bytes of the message data.

Messages that are malformed or contain data which is invalid or out or range will be discarded and the system will return a protocol error message. For protocol error messages, the ID byte is used to indicate the type of error.

Protocol error code enumeration	
Error Code	Meaning
0	No Error
1	Unspecified Error
2	Unsupported Message Type
3	Invalid Serialization
4	Invalid Method ID
5	Invalid Signal ID
6	Invalid Schedule ID

8.2 Protocol Messages

Protocol messages are used for initialization and configuration of PinPoint™. The protocol messages described in this section include both the message header and message body, and allow you to subscribe to signals, schedule methods, and establish connections.

The following data types are used throughout this section. All data is packed little-endian (least significant byte first):

Type	Meaning
U8	1-byte unsigned integer
U16	2-byte unsigned integer
U32	4-byte unsigned integer
U64	8-byte unsigned integer
I16	2-byte signed integer (two's complement)
I24	3-byte signed integer (two's complement)
I32	4-byte signed integer (two's complement)
F32	4-byte float (IEEE 754, single precision)
F64	8-byte float (IEEE 754, double precision)

8.2.1 protocolVersion

This single-byte message must be sent over a TCP connection and should be the first message sent to PinPoint™. After a TCP connection is established, PinPoint™ also immediately sends its protocol version. This message is unique in that it only is one byte long and does not include a Message Header.

protocolVersion				
Offset	Name	Type	Data	Interpretation
0x00	ProtocolVersion	U8	0x42	The version of the protocol that is used by PinPoint

8.2.2 setAPI

After confirming the protocol version, the PinPoint™ will send a *setAPI* message over the open TCP connection. PinPoint™ also expects to receive a *setAPI* message after receiving the *ProtocolVersion*.

Because PinPoint™ is not configured to call external functions, this API may be empty as shown below. PinPoint will send an API as well, but this can be discarded as it is the information contained in this manual. To discard the message, parse the control byte and the message size and ignore the payload bytes.

Message ID: 0x01 – setAPI				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x89	Invoke protocol message, one-byte size field
0x01	Message ID	U8	0x01	SetApi's message id.
0x02	Size	U8	0x03	Payload length
0x03	URN Length	U16	0x0000	Empty URN
0x05	Number of Interfaces	U8	0x00	No Interfaces

8.2.3 getUdpPort

This message is sent from PinPoint™ via TCP, requesting which UDP Port (if any) to send UDP messages to and is sent immediately after PinPoint™ sends its *setAPI* message.

Message ID: 0x05 – getUdpPort				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x90	Invoke protocol message with return, zero-byte size field
0x01	Message ID	U8	0x05	getUdpPort's message id.

8.2.4 getUDPPort - Return

This message is returned over the TCP connection and is the expected response to the *getUDPPort* message. PinPoint™ will also respond with this message if it receives a *getUDPPort* message.

Message ID: 0x05 - getUDPPort – Return				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x81	Return from protocol message, one-byte size field
0x01	Message ID	U8	0x05	getUDPPort's message id.
0x02	Size	U8	0x02	Payload length
0x03	UDP Port	U16		UDP Port number

8.2.5 UDPPing

This message is sent by PinPoint™ over UDP, to confirm an opened port. When this message is received, an identical response should be sent (echoing the Ping Data and changing the control byte to indicate a response message type) to indicate a valid UDP port. If PinPoint™ does not receive a valid ping response for a period of 30 seconds, PinPoint™ will assume the connection was lost and clean up associated resources, making them available for a new connection.

Message ID: 0x06 - UDPPing				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x91	0x91: ping - Invoke protocol message with return, one-byte size field 0x81: ping response - Return from protocol message, one-byte size field
0x01	Message ID	U8	0x06	ping's message id.

0x02	Size	U8	0x08	Payload length
0x03	Ping Data	U64		Random data that should be included in the responding message to verify a UDP port

8.2.6 scheduleMessage

The scheduleMessage message configures PinPoint™ to call a specific method at a periodic rate. The Schedule ID must be unique and is used to change the interval of the schedule timer by using the *setScheduleInterval* message. The scheduled data will be returned using this Schedule ID.

Message ID: 0x03 – scheduleMessage				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x91	Invoke protocol message with return, one-byte size field
0x01	Message ID	U8	0x03	ScheduleMessage's message id.
0x02	Size	U8	0x0F	Payload length – 15 bytes
0x03	Schedule Id	U8		Unique identifier for the schedule
0x04	ScheduleInterval	U32		Interval between method calls in milliseconds
0x08	ScheduleOffset	U32		Timer offset in milliseconds
0x0C	MessageID	U8		Message ID of method to schedule
0x0D	ParameterSize	U32		Method parameters size
0x11	SchReturnMethod	U8		0x00 = UDP return method (low latency) 0x01 = TCP return method (guaranteed)
0x12	Parameters	defined by ParameterSize		parameter list for scheduled method call not included for methods without parameters

8.2.7 setScheduleInterval

This message modifies the rate at which a given scheduled event occurs. It is also used to cancel a scheduled event by setting the new interval to zero.

Message ID: 0x04 - setScheduleInterval				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x91	Invoke protocol message with return, one-byte size field
0x01	Message ID	U8	0x04	setScheduleInterval's message id.
0x02	Size	U8	0x09	Payload length
0x03	Schedule ID	U8		Schedule ID of schedule to modify
0x04	ScheduleInterval	U32		New Interval in ms (interval of 0 cancels scheduled message)
0x08	ScheduleOffset	U32		Offset in ms

8.2.8 connectSignal

This message configures PinPoint™ to send select messages every time the message data is updated. These messages can be configured to be sent over TCP or UDP (if setup by responding to the *getUDPPort* message from PinPoint™). This message is also used to disconnect from the signal by re-sending this message with the *ConnectionType* byte set to 0x00. Signals will be emitted with their Signal ID as the ID byte in the protocol frame.

Message ID: 0x02 – connectSignal				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x91	Invoke protocol message with return, one-byte size field
0x01	Message ID	U8	0x02	connectSignal()'s Function id.
0x02	Size	U8	0x02	Payload length
0x03	Signal ID	U8		Unique id of the signal
0x04	ConnectionType	U8		0x00: disconnect 0x01: TCP 0x02: UDP

8.2.9 Exception

When a method cannot properly return its data, the server will respond with an *Exception* message type followed by the typical *ReturnValue* with invalid data. The ID of the exception is the method ID of the method that forced the exception. For PinPoint™, exceptions are used to indicate that data is temporarily unavailable, likely due to not yet being initialized by the hardware or due to a lower level communication error.

Message ID: Exception Return message				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x31	Indicates an exception was thrown.
0x01	Message ID	U8	0x01	ID of the method throwing the exception
0x02	Size	U8	0x04	Payload length
0x03	Error Code	U32		Error code corresponding to the exception that was thrown.

8.2.10 Schedule Exception

The schedule exception is identical to the standard Exception except for the ID is interpreted as a schedule ID opposed to a method ID.

Message ID: Schedule Exception Return message				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x39	Exception was thrown for a scheduled method
0x01	Schedule ID	U8	0x01	ID of the scheduled event throwing the exception
0x02	Size	U8	0x04	Payload length
0x03	Error Code	U32		Error code corresponding to the exception that was thrown.

8.2.11 Error Response

This message is used to indicate protocol errors and is sent back by PinPoint™ every time it receives one of the following messages as an invoke with return: *connectSignal*, *scheduleMessage*, and *setScheduleInterval*.

Message ID: Error Return message				
Offset	Name	Type	Data	Interpretation
0x00	Control Byte	U8	0x81	protocol error response
0x01	Message ID	U8		Message ID returning the error
0x02	Size	U8		Payload length
0x03	Protocol Error Code	I16		Error code, 0 = no error
0x05	Message Size	U16		Length of the message string
0x07 + N	ASCII Message	U8		Characters of the ASCII message

8.3 Localization Message Server

The localization message server contains a "StatusReporter" interface in addition to the "localization::Filter" interface described in the following section.

8.3.1 getBoardInfo: Method ID = 4

This method returns ASCII key => value pairs of general board information.

Input Parameters, Size = 0:
N/A

Return Value, Size = Variable:

Offset	Type	Description
H + 0	U8	Number of key => value pairs
H + 1 + 2*N + L	U8	Key size
H + 2 + 2*N + L	U8	ASCII key (L += size)
H + 2 + 2*N + L	U8	Value size
H + 3 + 2*N + L	U8	ASCII value (L += size)

For example, the pairs "FIRSTKEY" => "FIRSTVALUE", "SECONDKEY" => SECONDVALUE" would be packed as follows, for a total payload size of 43 bytes:

0x02,
0x08,
'F', 'I', 'R', 'S', 'T', 'K', 'E', 'Y',
0x0A,
'F', 'I', 'R', 'S', 'T', 'V', 'A', 'L', 'U', 'E',
0x09,
'S', 'E', 'C', 'O', 'N', 'D', 'K', 'E', 'Y',
0x0B,
'S', 'E', 'C', 'O', 'N', 'D', 'V', 'A', 'L', 'U', 'E'

PinPoint™ supports the following keys:

Key Name	Description
MODEL	PinPoint™ model number
HWVER	PinPoint™ hardware version
SERIAL	PinPoint™ serial number
FWVER	Tagged firmware release
FWREV	Internal firmware revision
FWDATE	Date the firmware was built
GCCVER	Compiler used to build the firmware

8.3.2 resetFilter: Method ID = 5

This method will reinitialize the localization filter, which will reset the local frame and require the filter to be realigned. A nonzero error code is returned if this method fails.

Input Parameters, Size = 0:
N/A

Return Value, Size = Variable:

Offset	Type	Description
H + 0x00	U16	Error Code
H + 0x02	U16	Message Size
H + 0x04 + N	U8	ASCII Message

8.3.3 getFilterAccuracy: Method ID = 6

This method returns the standard deviation of the global position, velocity, and attitude estimates. All error estimates are provided in the local NED frame.

Input Parameters, Size 0x00:
N/A

Return Value, Size = 44:

Offset	Type	Description
H + 0x00	U64	Time since 1970 (microseconds)
H + 0x08	F32	North position accuracy (m)
H + 0x0C	F32	East position accuracy (m)
H + 0x10	F32	Down position accuracy (m)
H + 0x14	F32	North velocity accuracy (m/s)
H + 0x18	F32	East velocity accuracy (m/s)
H + 0x1C	F32	Down velocity accuracy (m/s)
H + 0x20	F32	North rotational accuracy (rad)
H + 0x24	F32	East rotational accuracy (rad)
H + 0x28	F32	Down rotational accuracy (rad)

8.3.4 getGlobalPose: Method ID = 7

This method returns time-stamped position and attitude estimates in global frame.

Input Parameters, Size = 0:

N/A

Return Value, Size = 26:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	I32	Latitude (180 deg / 2^31)
H + 0x0C	I32	Longitude (180 deg / 2^31)
H + 0x10	I32	HAE Altitude (mm)
H + 0x14	I16	Roll (180 deg / 2^15)
H + 0x16	I16	Pitch (180 deg / 2^15)
H + 0x18	I16	Yaw (180 deg / 2^15)

8.3.5 getLocalPose: Method ID = 9

This method returns time-stamped position and attitude estimates in local frame.

Input Parameters, Size = 0:
N/A

Return Value, Size = 26:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	I32	North (mm)
H + 0x0C	I32	East (mm)
H + 0x10	I32	Down (mm)
H + 0x14	I16	Roll (180 deg / 2^15)
H + 0x16	I16	Pitch (180 deg / 2^15)
H + 0x18	I16	Yaw (180 deg / 2^15)

8.3.6 getVelocityState: Method ID = 11

This method returns time-stamped linear and rotational velocities in vehicle frame.

Input Parameters, Size = 0:
N/A

Return Value, Size = 26

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	I24	Forward Velocity (mm/s)
H + 0x0B	I24	Right Velocity (mm/s)
H + 0x0E	I24	Down Velocity (mm/s)
H + 0x11	I24	Roll Rate (mrad/s)
H + 0x14	I24	Pitch Rate (mrad/s)
H + 0x17	I24	Yaw Rate (mrad/s)

8.3.7 getQuaternionCovariance: Method ID = 13

This method returns a time-stamped quaternion representing the local NED to vehicle FRD rotation and a 3x3 covariance matrix representing the accuracy of the attitude estimate.

Input Parameters, Size = 0:
N/A

Return Value, Size = 60:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32[4]	NED -> FRD Quaternion, [qw, qx, qy, qz]
H + 0x18	F32[3][3]	NED Attitude Covariance, rad^2

8.3.8 globalPoseChanged: Signal ID = 6

This signal contains time-stamped position and attitude estimates in global frame, and is emitted at approximately 100 Hz. This signal is not emitted until the filter has been initialized with a valid GPS location and time.

Output Parameters, Size = 26:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	I32	Latitude (180 deg / 2^31)
H + 0x0C	I32	Longitude (180 deg / 2^31)
H + 0x10	I32	HAE Altitude (mm)
H + 0x14	I16	Roll (180 deg / 2^15)
H + 0x16	I16	Pitch (180 deg / 2^15)
H + 0x18	I16	Yaw (180 deg / 2^15)

8.3.9 localPoseChanged: Signal ID = 7

This signal contains time-stamped position and attitude estimates in local frame, and is emitted at approximately 100 Hz. This signal is not emitted until the filter has been initialized with a valid GPS location and time.

Output Parameters, Size = 26:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	I32	North (mm)
H + 0x0C	I32	East (mm)
H + 0x10	I32	Down (mm)
H + 0x14	I16	Roll (180 deg / 2^15)
H + 0x16	I16	Pitch (180 deg / 2^15)
H + 0x18	I16	Yaw (180 deg / 2^15)

8.3.10 velocityStateChanged: Signal ID = 8

This signal contains time-stamped linear and rotational velocities in vehicle frame, and is emitted at approximately 100 Hz. This signal is not emitted until the filter has been initialized with a valid GPS location and time.

Output Parameters, Size = 26:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970

Offset	Type	Description
H + 0x08	I24	Forward Velocity (mm/s)
H + 0x0B	I24	Right Velocity (mm/s)
H + 0x0E	I24	Down Velocity (mm/s)
H + 0x11	I24	Roll Rate (mrad/s)
H + 0x14	I24	Pitch Rate (mrad/s)
H + 0x17	I24	Yaw Rate (mrad/s)

8.3.11 quaternionCovarianceChanged: Signal ID = 9

This signal contains a time-stamped quaternion representing the local NED to vehicle FRD rotation and a 3x3 covariance matrix representing the accuracy of the attitude estimate. It is emitted at approximately 100 Hz. This signal is not emitted until the filter has been initialized with a valid GPS location and time.

Output Parameters, Size = 60:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32[4]	NED -> FRD Quaternion, [qw, qx, qy, qz]
H + 0x18	F32[3][3]	NED Attitude Covariance, rad^2

8.3.12 filterAccuracyChanged: Signal ID = 10

This signal contains the standard deviation of the global position, velocity, and attitude estimates. All error estimates are provided in the local NED frame.

Output Parameters, Size = 44:

Offset	Type	Description
H + 0x00	U64	Time since 1970 (microseconds)
H + 0x08	F32	North position accuracy (m)
H + 0x0C	F32	East position accuracy (m)
H + 0x10	F32	Down position accuracy (m)
H + 0x14	F32	North velocity accuracy (m/s)
H + 0x18	F32	East velocity accuracy (m/s)
H + 0x1C	F32	Down velocity accuracy (m/s)
H + 0x20	F32	North rotational accuracy (rad)
H + 0x24	F32	East rotational accuracy (rad)
H + 0x28	F32	Down rotational accuracy (rad)

8.3.13 Localization Status Messages

Aligning
Status Code = 1
The localization filter has not been aligned, and the output data should not be used. This will clear after 5 seconds of GPS heading updates (if equipped) or 5 seconds of vehicle motion above 5 mph.

NoImuData
Status Code = 2
IMU data is not available, the localization filter will not be propagated, and none of the output signals will be emitted.

NoGpsUpdates
Status Code = 3
GPS corrections are not available. If both WSS and GPS corrections are not available, the filter is running uncorrected and the output data should not be trusted. This will be reflected in the estimated filter accuracy.

NoLeftWssUpdates
Status Code = 4
The left wheel speed corrections are not available. If both WSS and GPS corrections are not available, the filter is running uncorrected and the output data should not be trusted. This will be reflected in the estimated filter accuracy.

NoRightWssUpdates
Status Code = 5
The right wheel speed corrections are not available. If both WSS and GPS corrections are not available, the filter is running uncorrected and the output data should not be trusted. This will be reflected in the estimated filter accuracy.

BadGpsPosAgreement
Status Code = 6
A discrepancy has been detected between the localization filter and the GPS position measurements. This is most likely caused by incorrect configuration parameters.

BadGpsVelAgreement
Status Code = 7
A discrepancy has been detected between the localization filter and the GPS velocity measurements. This is most likely caused by incorrect configuration parameters.

BadWssVelAgreement
Status Code = 8
A discrepancy has been detected between the localization filter and the WSS velocity measurements. This is most likely caused by incorrect configuration parameters.

BadGyroBiasEstimate
Status Code = 9
The gyroscope sensor bias estimates are outside of the acceptable range for this IMU. This is most likely caused by incorrect configuration parameters.

BadAccelBiasEstimate
Status Code = 10
The accelerometer sensor bias estimates are outside of the acceptable range for this IMU. This is most likely caused by incorrect configuration parameters.

PoseSteadying
Status Code = 11
The localization filter has detected the vehicle to be stationary and is applying additional corrections to minimize drift.

NoHeadingUpdates
Status Code = 12

Heading corrections from the dual GPS receivers are unavailable.

BadHeadingAgreement
Status Code = 13
A discrepancy has been detected between the localization filter and the GPS heading measurement. This is most likely caused by incorrect configuration parameters.

BadMeasurementTime
Status Code = 14
The measurement is too old and has been delayed more than the filter can account for (approximately 150 milliseconds).

8.4 IMU Message Server

The IMU message server contains a "StatusReporter" interface in addition to the "localization::Imu" interface described in the following section.

8.4.1 newRawImuData: Signal ID = 6

This signal contains time stamped delta angles and delta velocities from the IMU, and is emitted at approximately 100 Hz. To convert to angular rates and accelerations, divide the delta angle and velocity measurements by the elapsed time since the last signal emission.

Output Parameters, Size = 32:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32	X Delta Angle (rad)
H + 0x0C	F32	Y Delta Angle (rad)
H + 0x10	F32	Z Delta Angle (rad)
H + 0x14	F32	X Delta Velocity (m/s)
H + 0x18	F32	Y Delta Velocity (m/s)
H + 0x1C	F32	Z Delta Velocity (m/s)

8.4.2 IMU Status Codes

CommsFailure
Status Code = 1
PinPoint™ was unable to connect to the inertial measurement unit. If using an external IMU, check all cabling and connections. If using an internal IMU, the unit should be returned to TORC for evaluation and repair.

ProcessorFailure
Status Code = 2
An error has occurred with the onboard IMU processor. The unit should be returned to TORC for evaluation and repair.

GyroFailure
Status Code = 3
One or more of the gyroscopes have failed. The unit should be returned to TORC for evaluation and repair.

AccelFailure
Status Code = 4
One or more of the accelerometers have failed. The unit should be returned to TORC for evaluation and repair.

8.5 GPS Message Server

The GPS message server contains a "StatusReporter" interface in addition to the "localization::Gps" interface described in the following section.

8.5.1 getGpsFixInfo: Method ID = 4

This method returns information about the GPS solution and the number of satellites.

Input Parameters, Size = 0:
N/A

Return Value, Size = 5:

Offset	Type	Description
H + 0x00	U8	GPS Fix Type
H + 0x01	U8	Number of satellites visible on primary antenna
H + 0x02	U8	Number of satellites visible on secondary antenna
H + 0x03	U8	Number of satellites used in position calculation
H + 0x04	U8	Number of satellites used in heading calculation

The GPS fix type will be one of the following:

GPS Fix Type	Description
0	Unknown
1	None
2	2D
3	3D
4	SBAS
5	OmniSTAR VBS
6	OmniSTAR XP
7	OmniSTAR HP
8	Terrastar

8.5.2 getLbandInfo: Method ID = 5

This method returns information about the active Terrastar or OmniSTAR subscription. The activation string needs to be given to the corrections provider and is either the OmniSTAR Serial Number (OSN) or the Terrastar Product Activation Code (PAC) depending on the corrections being used.

Input Parameters, Size = 0:
N/A

Return Value, Size = 13:

Offset	Type	Description
H + 0x00	U8	Lband subscription type
H + 0x01	U64	Lband expiration date (microseconds since 1970)

H + 0x09	U8	Lband Activation Length (# of characters)
H + 0x0A + N	U8	Lband Activation String (ASCII characters)

Lband Subscription Type	Description
0	Unknown
1	Expired
2	OmniSTAR VBS
3	OmniSTAR XP
4	OmniSTAR HP
5	Terrastar M
6	Terrastar D

8.5.3 setLbandSat: Method ID = 6

This method is used to enable Lband corrections and force the receiver to listen to a particular satellite. It is required to send this string when activating a new subscription, but once activated corrections will be enabled and the satellite will be automatically chosen based on receiver location. For OmniSTAR, the satellite frequency in kHz should be sent using ASCII characters, e.g. 0x07"1557845". For Terrastar, the beam name should be sent using ASCII characters, e.g. 0x04"aorw". Sending a zero length string (0x00"") will disable lband corrections and prevent them from being enabled on future power cycles.

Input Parameters, Size = 4:

Offset	Type	Description
H + 0x00	U8	String Length
H + 0x01 + N	U8	ASCII String

Return Value, Size = Variable:

Offset	Type	Description
H + 0x00	U16	Error Code
H + 0x02	U16	Message Size
H + 0x04 + N	U8	ASCII Message

8.5.4 newRawGpsData: Signal ID = 6

This signal contains time-stamped GPS position and velocity, and is emitted at 20 Hz whenever measurements are available.

Output Parameters, Size = 52:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F64	Latitude (rad)
H + 0x10	F64	Longitude (rad)
H + 0x18	F64	HAE Altitude (m)
H + 0x20	F32	3D Position Accuracy (m)
H + 0x24	F32	North Velocity (m/s)
H + 0x28	F32	East Velocity (m/s)
H + 0x2C	F32	Down Velocity (m/s)
H + 0x30	F32	Velocity Accuracy (m/s)

8.5.5 newRawGpsHeading: Signal ID = 7

This signal contains time-stamped GPS heading, and is emitted at 20 Hz whenever measurements are available.

Output Parameters, Size = 24:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32	Antenna Heading (rad)
H + 0x0C	F32	Antenna Pitch (rad)
H + 0x10	F32	Heading Accuracy (rad)
H + 0x14	F32	Pitch Accuracy (rad)

8.5.6 newGpsFixInfo: Signal ID = 8

This signal contains information about the GPS solution and the number of satellites, and is emitted whenever any of the values change.

Output Parameters, Size = 5:

Offset	Type	Description
H + 0x00	U8	GPS Fix Type
H + 0x01	U8	Number of satellites visible on primary antenna
H + 0x02	U8	Number of satellites visible on secondary antenna
H + 0x03	U8	Number of satellites used in position calculation
H + 0x04	U8	Number of satellites used in heading calculation

8.5.7 GPS Status Messages

CommsFailure
Status Code = 1
PinPoint™ was unable to connect to the GPS receiver. The unit should be returned to TORC for evaluation and repair.

ReceiverFailure
Status Code = 2
The GPS receiver has failed. The unit should be returned to TORC for evaluation and repair.

BadTemperature
Status Code = 3
The GPS receiver is operating outside the recommended temperature range. Further use should be discontinued until the ambient temperature can be reduced.

BadVoltage
Status Code = 4
The GPS receiver is operating outside the recommended voltage range. The unit should be returned to TORC for evaluation and repair.

AntennaOpen
Status Code = 5
The GPS antenna is not consuming any current. Check all cables and connectors.

AntennaShort
Status Code = 6
The GPS antenna is consuming too much current. Check all cables and connectors.

CpuOverload
Status Code = 7
An error has occurred with the onboard GPS processor. The unit should be returned to TORC for evaluation and repair.

InvalidAlmanac
Status Code = 8
The almanac is not valid. Move the vehicle to an area with a better view of the sky or wait for the almanac to be received.

InvalidClock
Status Code = 9
The clock is not valid. Move the vehicle to an area with a better view of the sky or wait for it to be downloaded.

InvalidPosition
Status Code = 10
The position is not valid. Move the vehicle to an area with a better view of the sky or wait for it to be downloaded.

8.6 Wheel Speed Message Server

The IMU message server contains a "StatusReporter" interface in addition to the "localization::Wss" interface described in the following section.

8.6.1 getAvgWheelSpeed: Method ID = 4

This method returns the best estimate of the vehicle speed from the wheel speed sensors. Additional error handling logic is applied to this value as follows:

- If there are no errors, this is the average of the left and right wheel speed measurements
- If the direction is unknown on one of the wheels, the magnitudes of the two sensors are averaged and the sign is taken from the wheel that has a known direction
- If the direction is unknown on both of the wheels, the magnitudes are averaged and the sign is positive.
- If measurements are not available from one of the wheels, this is the speed and direction of the working wheel.

Input Parameters, Size = 0:
N/A

Return Value, Size = 3:

Offset	Type	Description
H + 0x00	I24	Speed in mm/s

8.6.2 getLeftWheelSpeed: Method ID = 5

This method returns the signed speed of the left wheel. If there are errors with either the in-phase or quadrature sensor, this method returns 0.

Input Parameters, Size = 0:
N/A

Return Value, Size = 3:

Offset	Type	Description
H + 0x00	I24	Speed in mm/s

8.6.3 getRightWheelSpeed: Method ID = 6

This method returns the signed speed of the right wheel. If there are errors with either the in-phase or quadrature sensor, this method returns 0.

Input Parameters, Size = 0:
N/A

Return Value, Size = 3:

Offset	Type	Description
H + 0x00	I24	Speed in mm/s

8.6.4 getOdometer: Method ID = 7

This method returns the current odometer value in meters.

Input Parameters, Size = 0:
N/A

Return Value, Size = 4:

Offset	Type	Description
H + 0x00	U32	Distance in meters

8.6.5 newRawWssData: Signal ID = 6

This signal contains time stamped left and right wheel speeds, and is emitted at 20 Hz.

Output Parameters, Size = 16:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32	Left Wheel Speed (m/s)
H + 0x0C	F32	Right Wheel Speed (m/s)

8.6.6 odometerChanged: Signal ID = 7

This signal contains a time stamp and the current odometer reading, and is emitted after every meter of travel.

Output Parameters, Size = 12:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	U32	Odometer (m)

8.6.7 newLeftWssVector: Signal ID = 8

This signal contains time stamped left wheel speed, direction, and errors, and is emitted at 20 Hz.

Output Parameters, Size = 21:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32	Left WSS Velocity (m/s)
H + 0x0C	F32	Left WSS Pitch (rad)
H + 0x10	F32	Left WSS Yaw (rad)
H + 0x0C	U8	Left WSS Errors

The bits in the errors byte should be interpreted as follows:

GPS Fix Type	Description
0	Inphase sensor error
1	Quadrature sensor error
2	(unused)
3	(unused)
4	Sensor timeout error
5	(unused)
6	Pitch sensor error
7	Yaw sensor error

8.6.8 newRightWssVector: Signal ID = 9

This signal contains time stamped right wheel speed, direction, and errors, and is emitted at 20 Hz.

Output Parameters, Size = 21:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	F32	Right WSS Velocity (m/s)
H + 0x0C	F32	Right WSS Pitch (rad)
H + 0x10	F32	Right WSS Yaw (rad)
H + 0x0C	U8	Right WSS Errors

The bits in the errors byte are the same as defined for the left sensor.

8.6.9 Wheel Speed Status Codes

LISensorFailure
Status Code = 1
The hardware has detected a failure of the left in-phase wheel speed sensor.

LQSensorFailure
Status Code = 2

The hardware has detected a failure of the left quadrature wheel speed sensor.

RISensorFailure
Status Code = 3
The hardware has detected a failure of the right in-phase wheel speed sensor.

RQSensorFailure
Status Code = 4
The hardware has detected a failure of the right quadrature wheel speed sensor.

UsingOnlyOneWheel
Status Code = 5
Both sensors on one of the wheels have failed, and the average wheel speed is being calculated from the working wheel only.

DirectionUnknown
Status Code = 6
One sensor on each wheel has failed, and the average wheel speed will be positive regardless of actual vehicle direction of travel. If this average wheel speed is being used for vehicle speed control, it is suggested to implement additional logic based on the transmission gear if this error is set.

SensorTimeout
Status Code = 7
No speed measurements are currently available. If using quadrature speed sensors, this error will be set on startup until vehicle motion produces a single pulse on the speed sensors.

8.7 Point-of-Interest Message Server

The point-of-interest message server contains a "StatusReporter" interface in addition to the "localization::PoiProducer" interface described in the following section.

8.7.1 markVehicleFrd: Method ID = 5

Calling this method causes a global frame POI to be emitted corresponding to a point given in vehicle frame.

Input Parameters, Size = 14:

Offset	Type	Description
H + 0x00	U16	POI Source ID
H + 0x02	I32	POI Forward Location (mm)
H + 0x06	I32	POI Right Location (mm)
H + 0x0A	I32	POI Down Location (mm)

Return Value, Size = Variable:

Offset	Type	Description
H + 0x00	U16	Error Code
H + 0x02	U16	Message Size
H + 0x04 + N	U8	ASCII Message

8.7.2 markLocalNed: Method ID = 6

Calling this method causes a global frame POI to be emitted corresponding to a point given in local frame.

Input Parameters, Size = 14:

Offset	Type	Description
H + 0x00	U16	POI Source ID
H + 0x02	I32	POI North Location (mm)
H + 0x06	I32	POI East Location (mm)
H + 0x0A	I32	POI Down Location (mm)

Return Value, Size = Variable:

Offset	Type	Description
H + 0x00	U16	Error Code
H + 0x02	U16	Message Size
H + 0x04 + N	U8	ASCII Message

8.7.3 newPoi: Signal ID = 6

This signal contains the current time, vehicle global location, point-of-interest global location, point-of-interest source identification, and point-of-interest count. The global location of the POI is calculated using the vehicle frame or local frame position provided in one of the above methods.

Output Parameters, Size = 36:

Offset	Type	Description
H + 0x00	U64	Microseconds since 1970
H + 0x08	I32	Vehicle Latitude (180 deg / 2^31)
H + 0x0C	I32	Vehicle Longitude (180 deg / 2^31)
H + 0x10	I32	Vehicle Altitude (mm)
H + 0x14	I32	POI Latitude (180 deg / 2^31)
H + 0x18	I32	POI Longitude (180 deg / 2^31)
H + 0x1C	I32	POI Altitude (mm)
H + 0x20	U16	POI Source ID
H + 0x22	U16	POI Count

8.7.4 Point-of-Interest Status Codes

The POI interface does not have any status codes.

8.8 Status Reporter Interface

The status reporter is implemented on each of the four component servers, and is used to convey errors and warnings about the state of that component. The method and signal IDs are the same on all of the message servers. Status codes are defined separately for each component, and conditions are represented as follows:

Status Condition	Description
0	Clear – No error or warning
1	Info – Information about the state of the system

2	Warning – The system is operating in a degraded state
3	Error – The system cannot function properly

8.8.1 getStatus: Method ID = 1

This method returns the current status of a given status code.

Input Parameters, Size = 1:

Offset	Type	Description
H + 0x00	U16	Status Code

Return Value, Size = 3:

Offset	Type	Description
H + 0x00	U8	Status Condition
H + 0x01	U16	Status Code

8.8.2 getStatusWithCondition: Method ID = 2

This method returns a list of statuses that are equal to or greater than a given condition.

Input Parameters, Size = 1:

Offset	Type	Description
H + 0x00	U8	Status Condition

Return Value, Size = Variable:

Offset	Type	Description
H + 0x00	U16	Number of statuses
H + 0x02 + N*3	U8	Status Condition
H + 0x03 + N*3	U16	Status Code

8.8.3 statusChanged: Signal ID = 0

This signal is emitted every time a status condition changes.

Output Parameters, Size = 3:

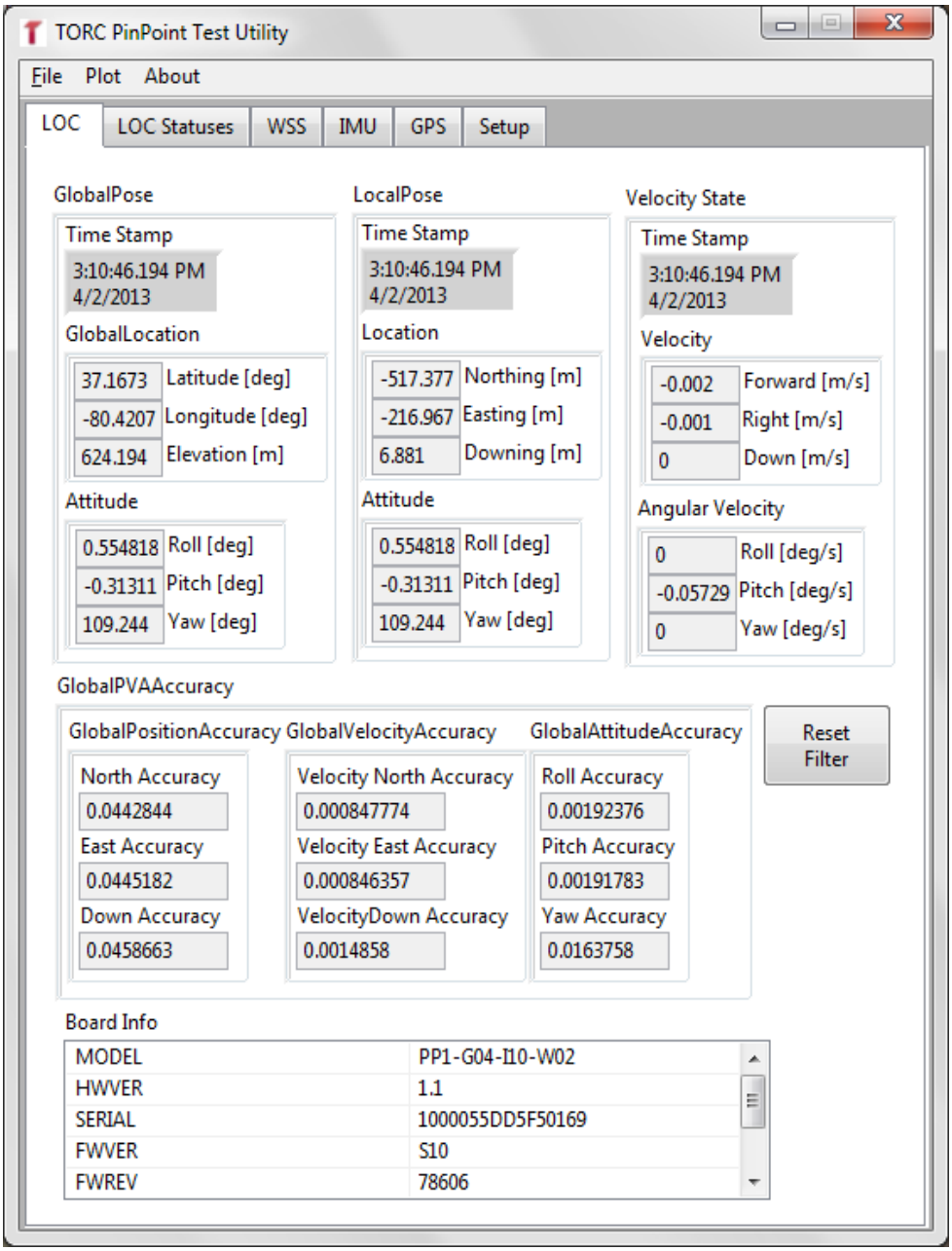
Offset	Type	Description
H + 0x00	U8	Status Condition
H + 0x01	U16	Status Coode

9. Diagnostic Utility

A diagnostic utility is included on the CD accompanying the PinPoint™ hardware. It allows the user to view all of the available outputs and statuses, as well as plot some of the outputs commonly used to verify the system has been installed and configured correctly.

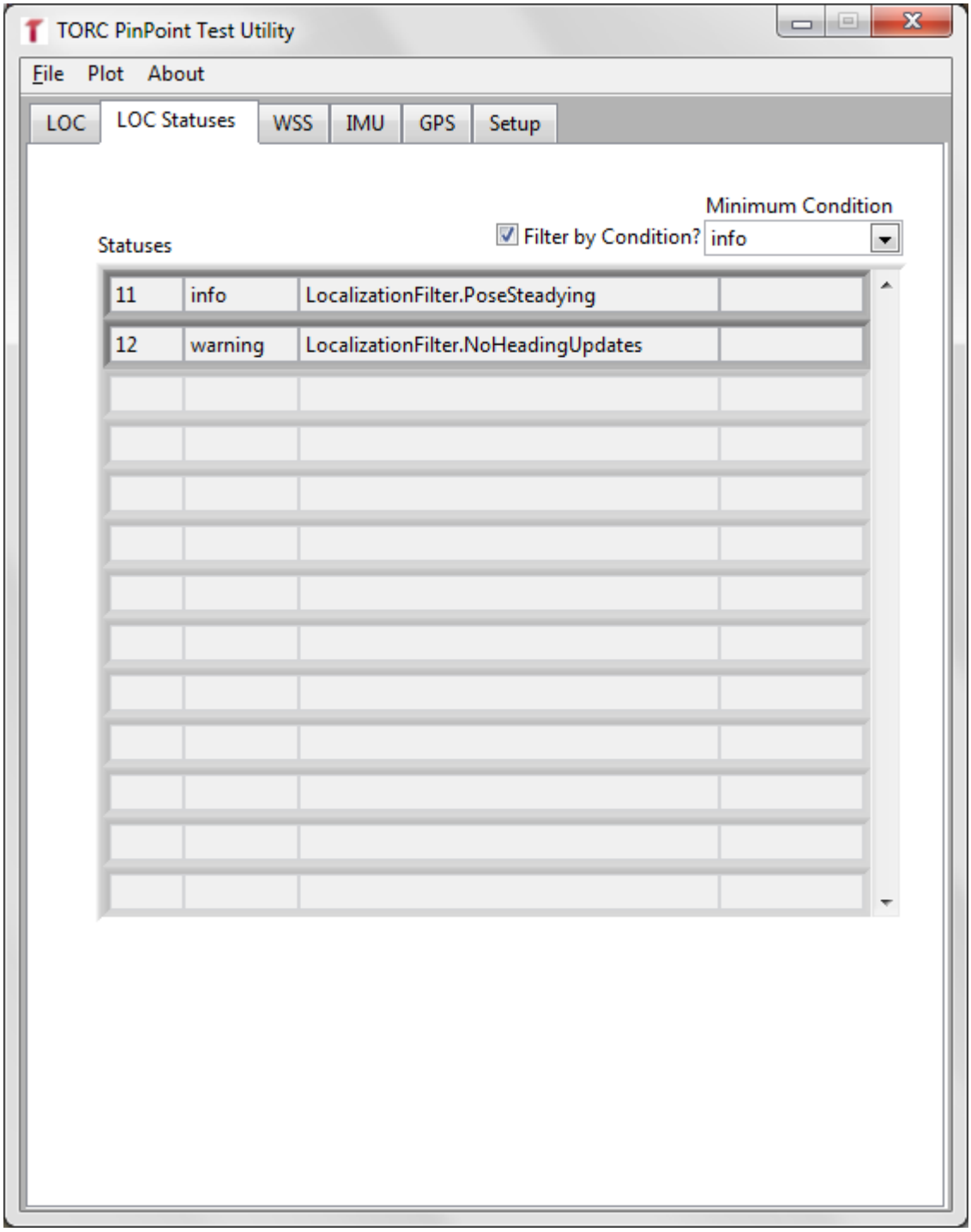
9.1 Localization Outputs Tab

This tab displays the return values and parameters for the methods and signals exposed by the localization server (see Software Interface section). Clicking the "Reset Filter" button calls the resetFilter() method as defined in section 8.3.2.



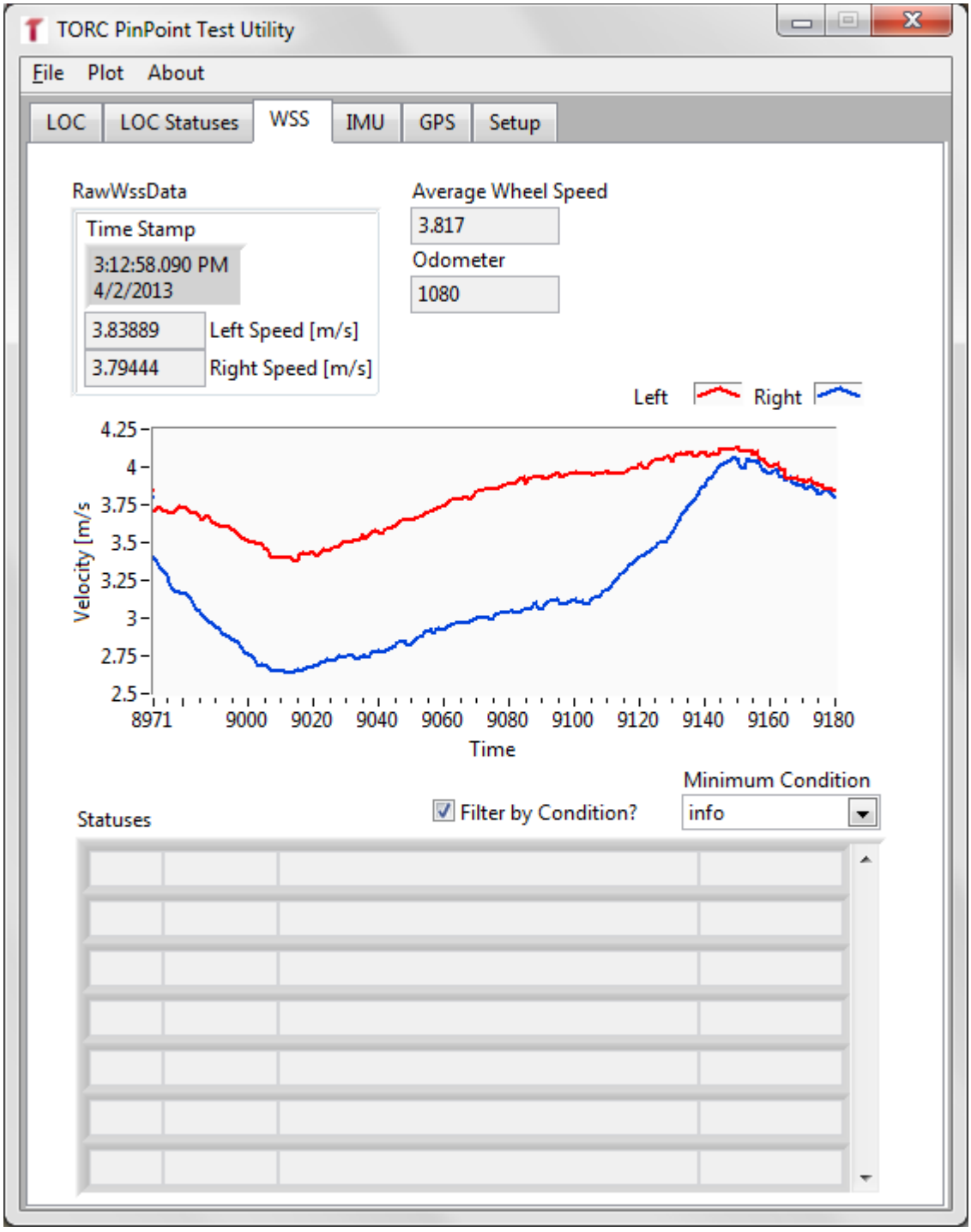
9.2 Localization Status Reporter Tab

This displays the status codes exposed by the 8.3Localization Message Server. Checking the "filter by condition" box will cause the program only to display statuses higher than the selected level as defined in section 8,8.



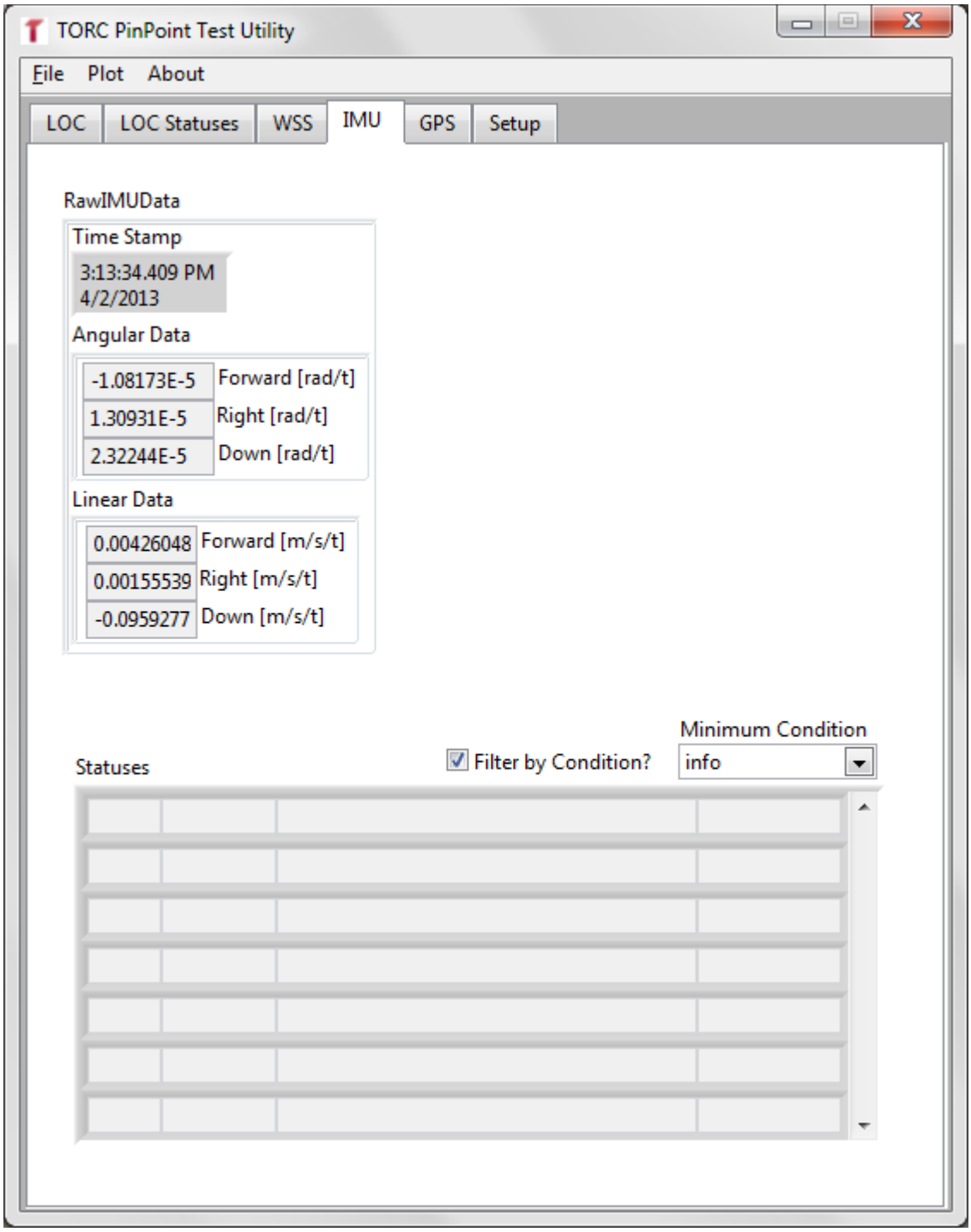
9.3 Wheel Speed Sensor Tab

This tab displays the return values and parameters for the methods and signals exposed by the 8.6 Wheel Speed Message Server. Tuning the wheel speed sensors can be performed using the wheel speed calibration window as described in section 9.11 Wheel Speed Calibration Plot.



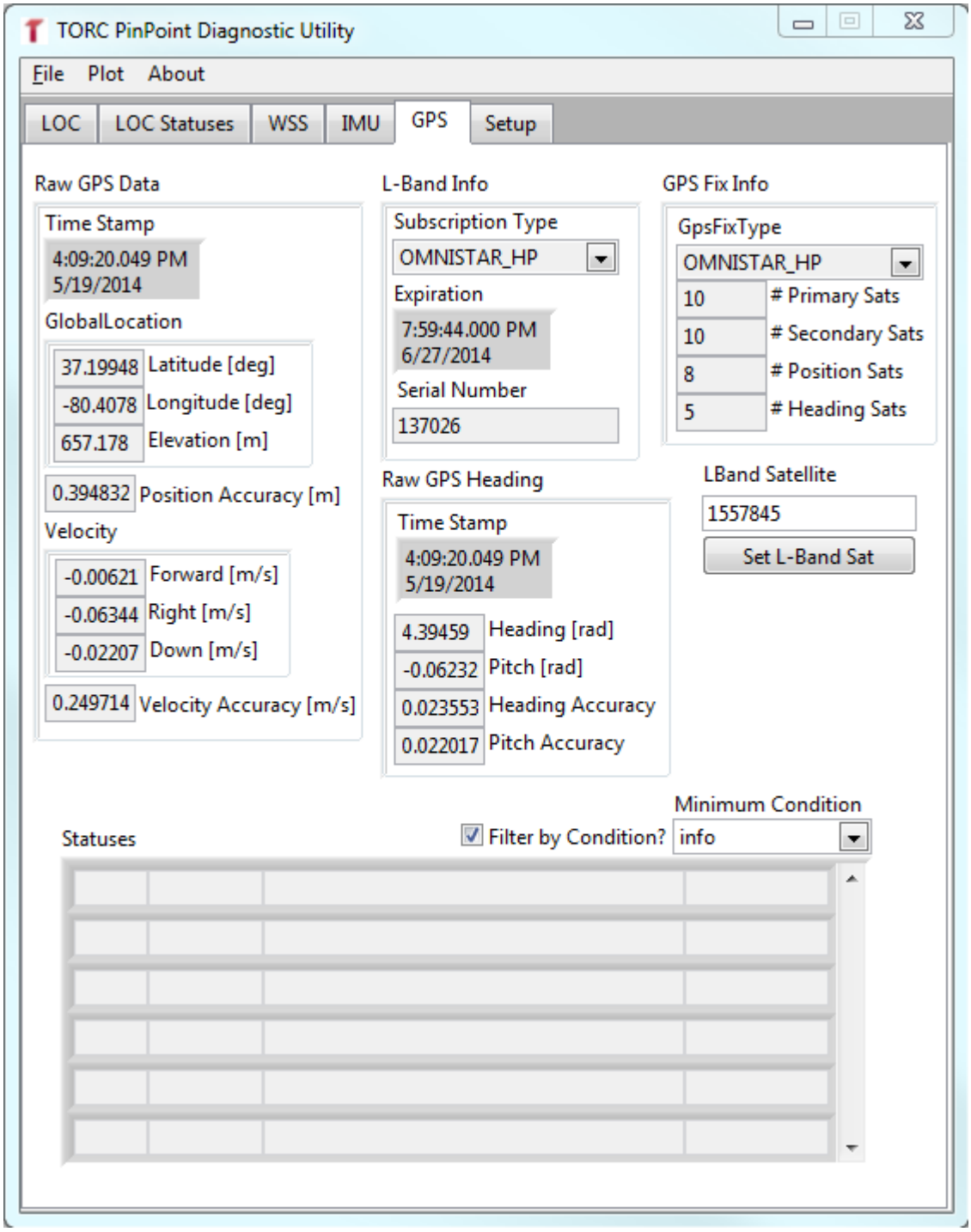
9.4 Inertial Measurement Unit Tab

This tab displays the return values and parameters for the methods and signals exposed by the 8.4 IMU Message Server.



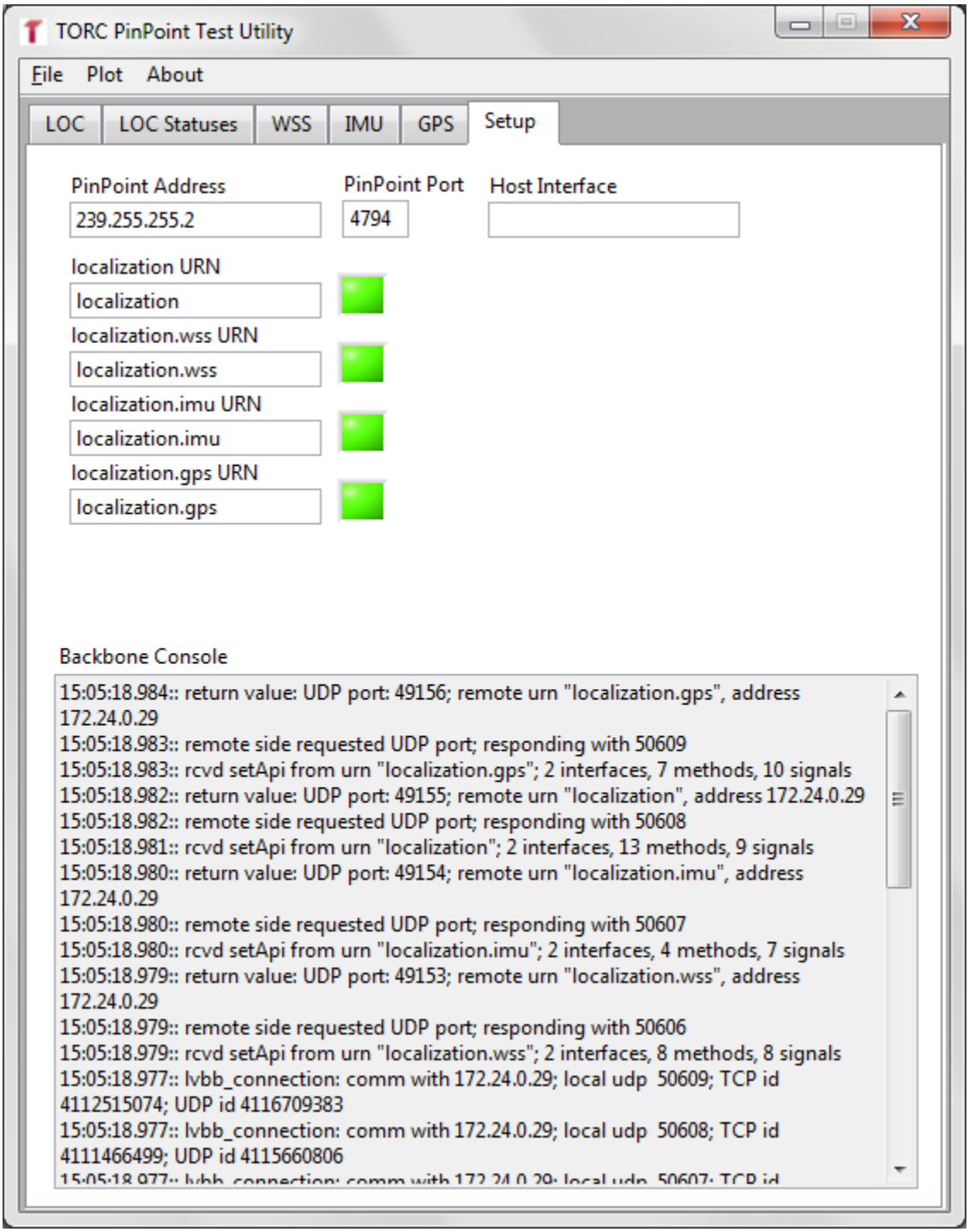
9.5 GPS Receiver Tab

This tab displays the return values and parameters for the methods and signals exposed by the GPS Message Server. It also allows calling the setLbandSat() method with an arbitrary OmniSTAR frequency (in kHz) or Terrastar beam name. Send a null string (empty input field) to disable corrections.



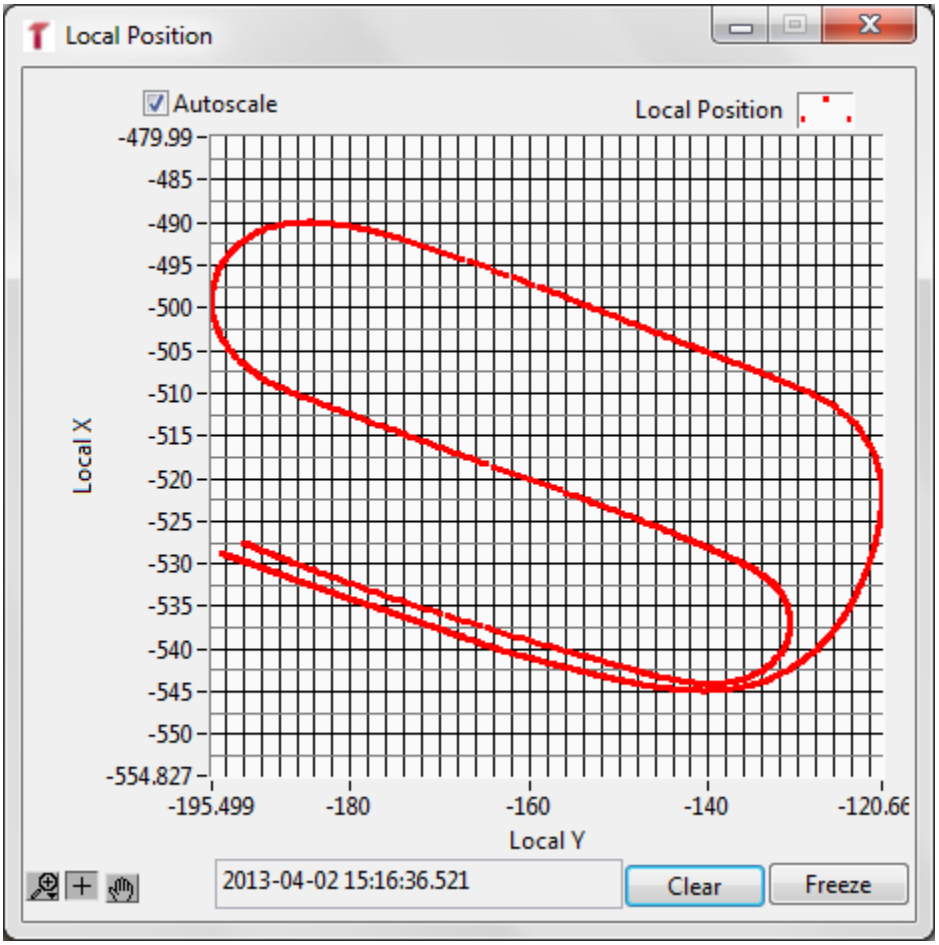
9.6 Communication Setup Tab

The setup tab should match the “Backbone Settings” tab of PinPoint’s™ web interface. Under normal circumstances, it should not be necessary to change any of these parameters. If trying to use the utility over a network that does not support multicast, the IP address of PinPoint™ can be used instead of the backbone multicast address.



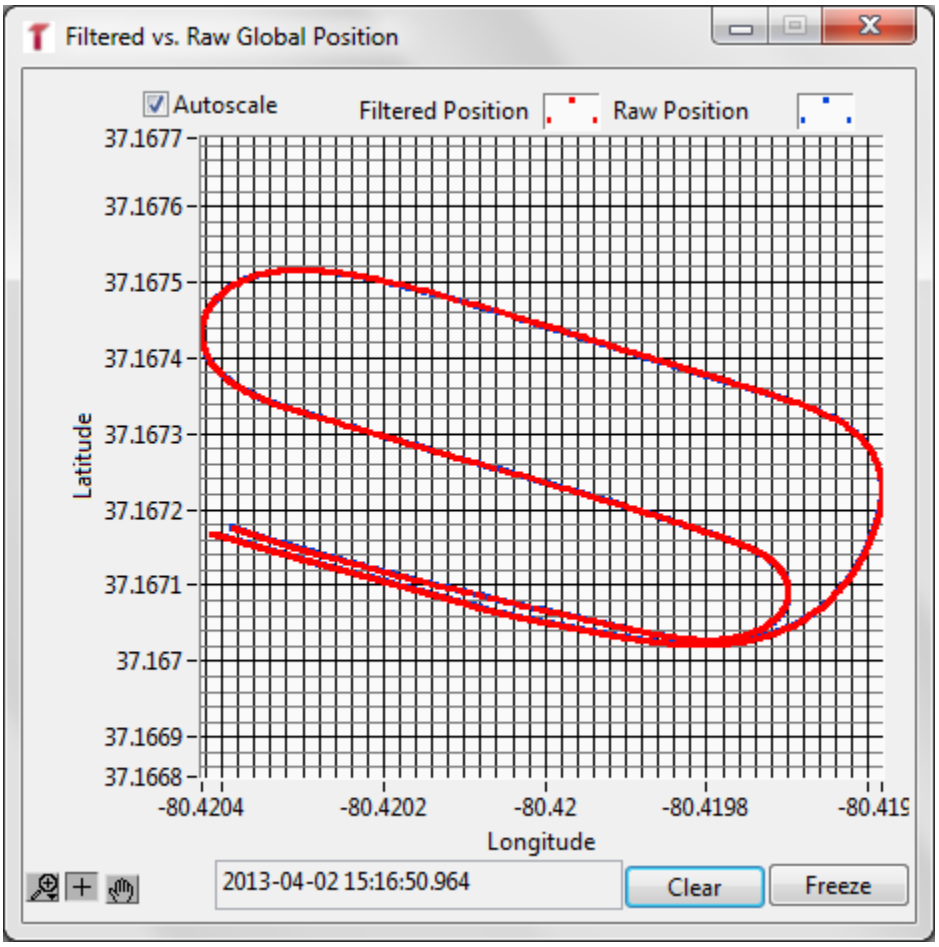
9.7 Local Position Plot

This allows the user to see PinPoint's™ local position. Note that while the global position will trend towards GPS position over time, the local position is only affected by vehicle movement. Press the clear button to clear the plot and the freeze button to pause the updates.



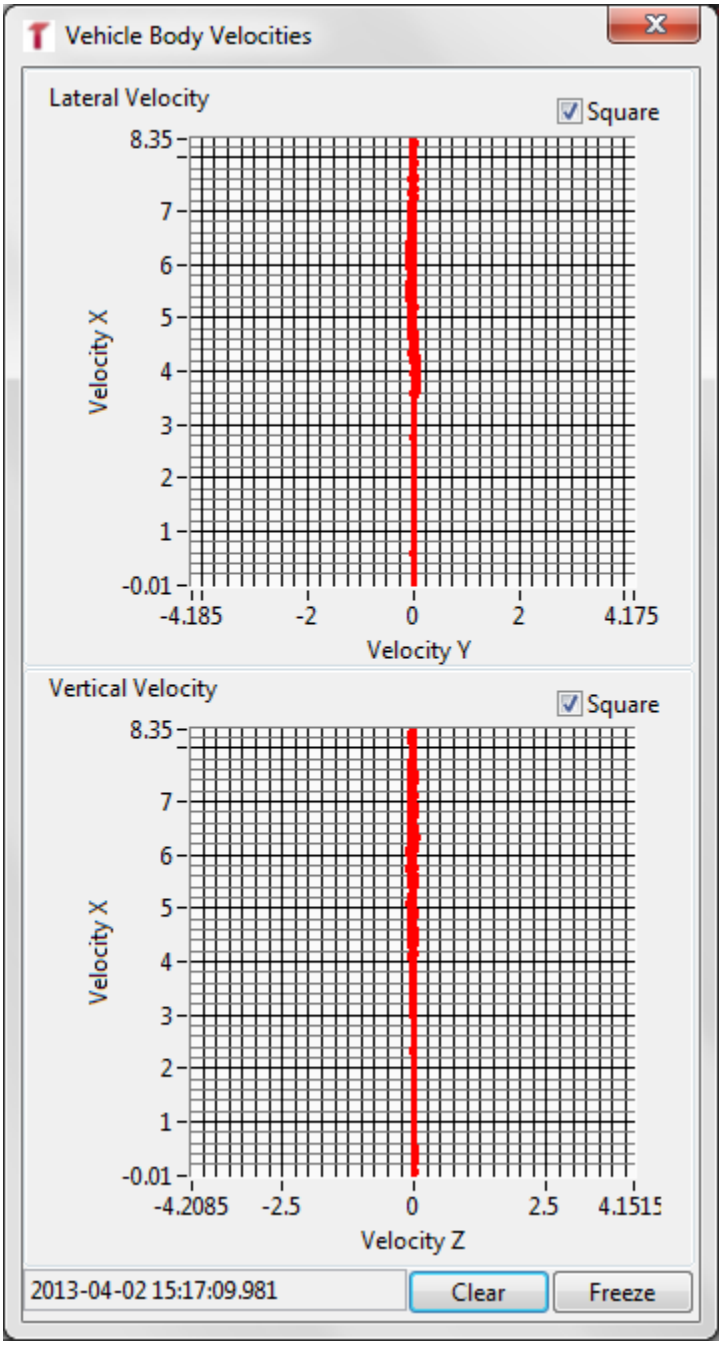
9.8 Global Position Plot

This plot allows the user to compare PinPoint's™ filtered global position against raw GPS position. Press the clear button to clear the plot and the freeze button to pause the updates. Keep in mind that the global position is the location of the vehicle origin, and the GPS position is the location of the primary antenna, so they may be slightly different if there is any horizontal moment arm between the two.



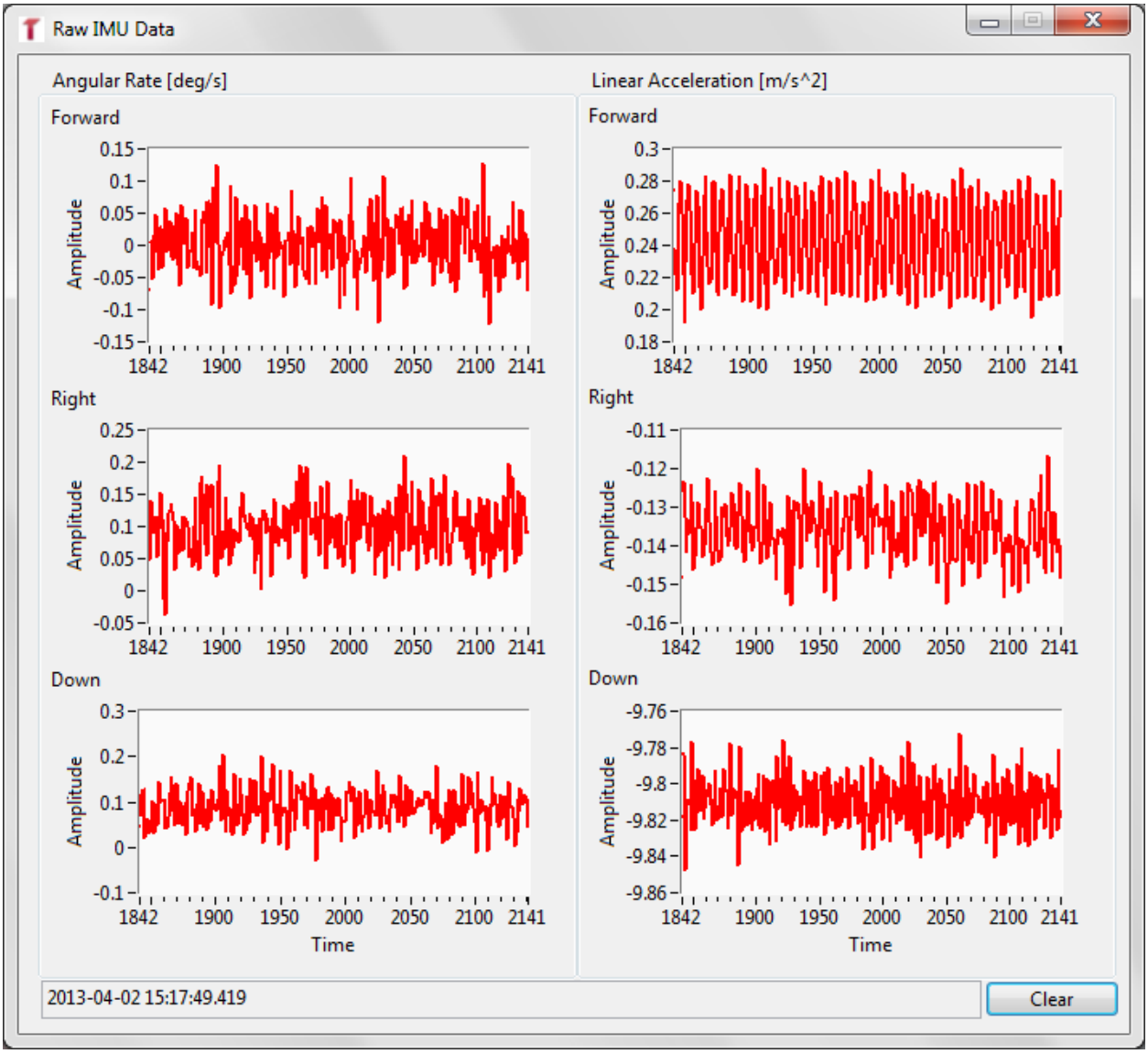
9.9 Body Velocity Plot

These plots display the forward body velocity with respect to the right and down velocities. Press the clear button to clear the plot and the freeze button to pause the updates. Under normal operation, both these plots should be close to a straight vertical line. A positive Y velocity indicates the IMU is yawed to the left. A positive Z velocity indicates the IMU is pitched up. This information can then be used to fine-tune the IMU orientation on the web configuration. Any sort of repeating or circular pattern indicates something is grossly misconfigured. Examples of things to check include the IMU installation (rotated 90° or 180°) and the wheel speed sensors (swapped in-phase and quadrature signals, invalid calibration factor).



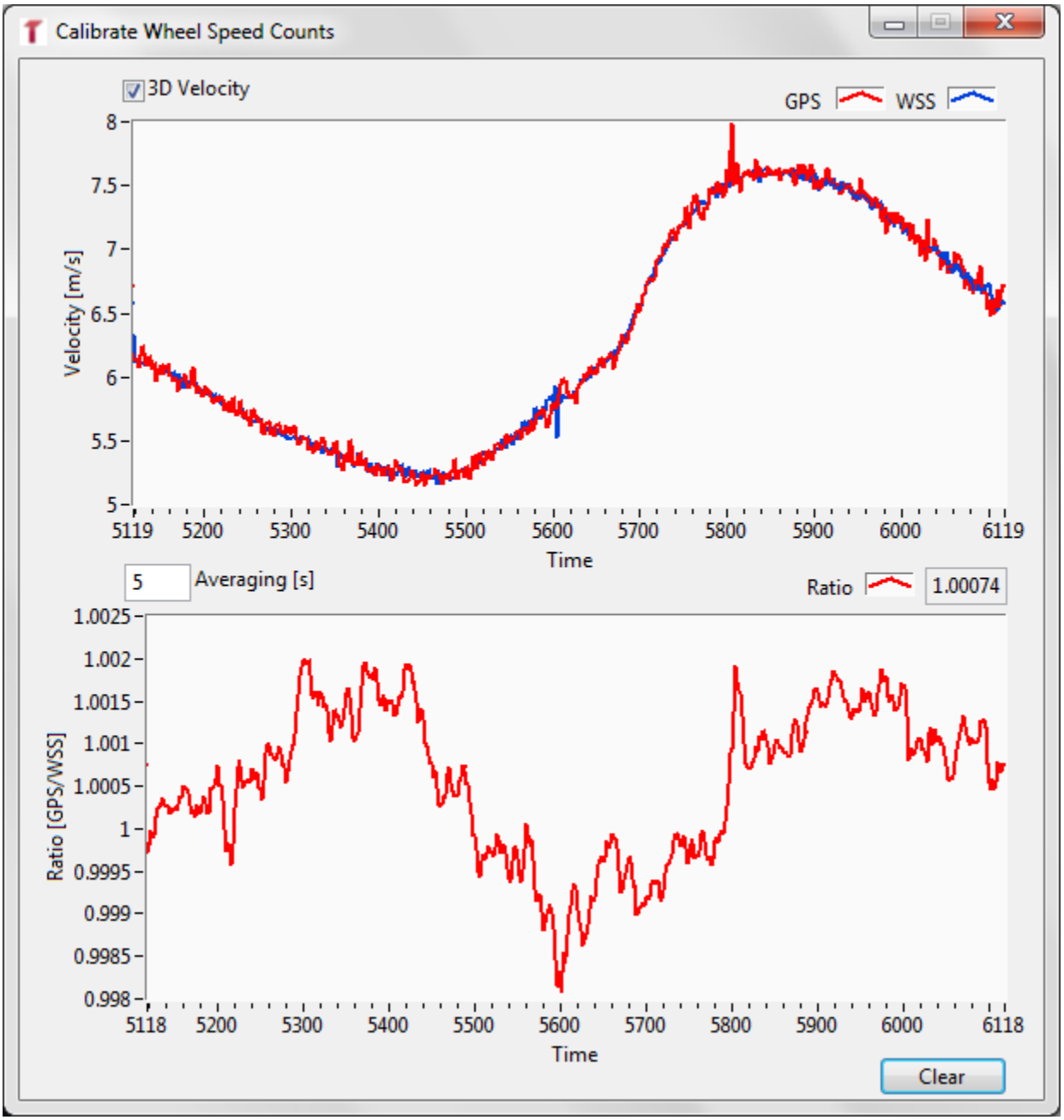
9.10 Raw IMU Data Plot

These plots display the raw data from the inertial measurement unit. It can be used to verify the connection to the precision IMU, or proper functionality of the internal IMU. Being raw data, the measurements are not corrected for rotation, scale factor, or bias.



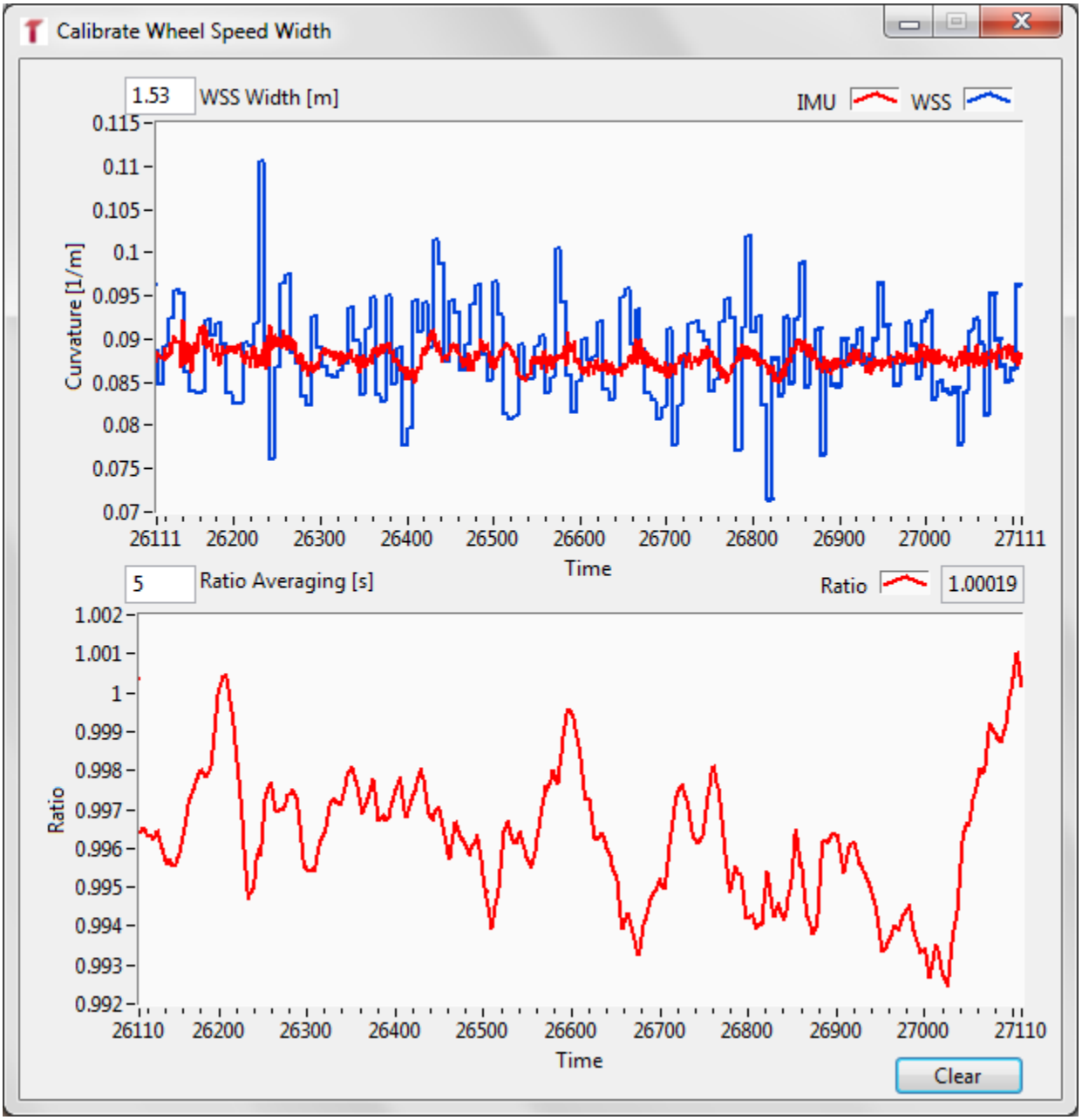
9.11 Wheel Speed Calibration Plot

This plot displays the average of left and right wheel speeds and the magnitude of the GPS speed, as well as the ratio between the two measurements. It can be used to fine-tune the pulses-per-meter calibration factor. Driving forward at moderate speeds over smooth terrain will produce the best results. A properly calibrated system should have a ratio that is close to 1 as shown in the image below.



9.12 Wheel Speed Track Plot

This plot displays the vehicle curvature derived from wheel speeds (left and right wheels), the vehicle curvature derived from velocity state (angular and linear velocities), and the ratio between the two. The WSS width (track) should be entered for the calculation to be valid, and the ratio can be used to fine tune this value. Driving the vehicle forward around a continuous circle over smooth terrain will produce the best results. After tuning the WSS track using these plots, this value should be compared to the left and right WSS locations on the vehicle tab of PinPoint's webpage.



9.13 Vibration Measurement Plot

This plot displays the frequency content of the rotational and linear velocities, and allows calculation of stationary vibration. The configurable buffer length sets the number of samples that an FFT is performed on, and the plots will not be valid until all samples are acquired. To measure the vibration parameters, make sure the vehicle is stationary, and click the corresponding button at the bottom of the plot to start collecting data. Once finished, click the button again and the vibration parameters will be calculated. These numbers can be used as a starting point, however there are often other vibrations associated with a moving vehicle that are not accounted for in a stationary measurement. Therefore, actual values entered in the web configuration should typically be 50% higher than those calculated in the plots.

10. Product Specifications

Table 3. Positioning Performance[1]

Signals	Dual Frequency L1, L2 Dual System GPS, GLONASS OmniSTAR Corrections	
Horizontal Position Accuracy (RMS)		
Single Point L1	1.5 m	
Single Point L1/L2	1.2 m	
SBAS[2]	0.6 m	
OminSTAR		
VBS	0.6 m	
XP	0.15 m	
HP	0.1 m	
Time to First Fix	50s (Cold Start) 35s (Hot Start)	
IMU Alignment (Internal or Precision)		
Single GPS Core	Kinematic align after GPS fix	
Dual GPS Core	Static align after GPS fix	
IMU Performance	Internal IMU	Precision IMU
Update Rate	102.5 Hz	100 Hz
Gyro operating range (deg/sec)	450	1000
Bias Stability (deg/hr)	6.25	1.0
Angular Random Walk (deg/rt.hr.)	0.3	0.06
Acceleration Range (g)	18	30

Table 4. Physical, Electrical and Environmental

Weight	3.2 lb – Localization 3.6 lb – Precision IMU 1.1 lb – GPS Antenna (each)
Input Voltage	9 – 36 VDC
Power Consumption	8 W (Single GPS & Internal IMU) 12 W (Dual GPS & Precision IMU)
Operating Temperature	-33 to +71 °C
Ingress Protection	IP67, Dust and Watertight (IEC 60529)
User Data Interface	10/100baseT Ethernet EIA-232 Serial
Wheel Speed Sensor Support	7.5 volt source, threshold at 10 mA

[1] Typical values. Performance specifications subject to GPS system characteristics, US DOD operational degradation, ionospheric and tropospheric conditions, satellite geometry, baseline length, multipath effects and the presence of intentional or unintentional interference sources.
[2] Only available for GPS and CONUS.

11. Hardware Interface

11.1 PinPoint™ Localization Module

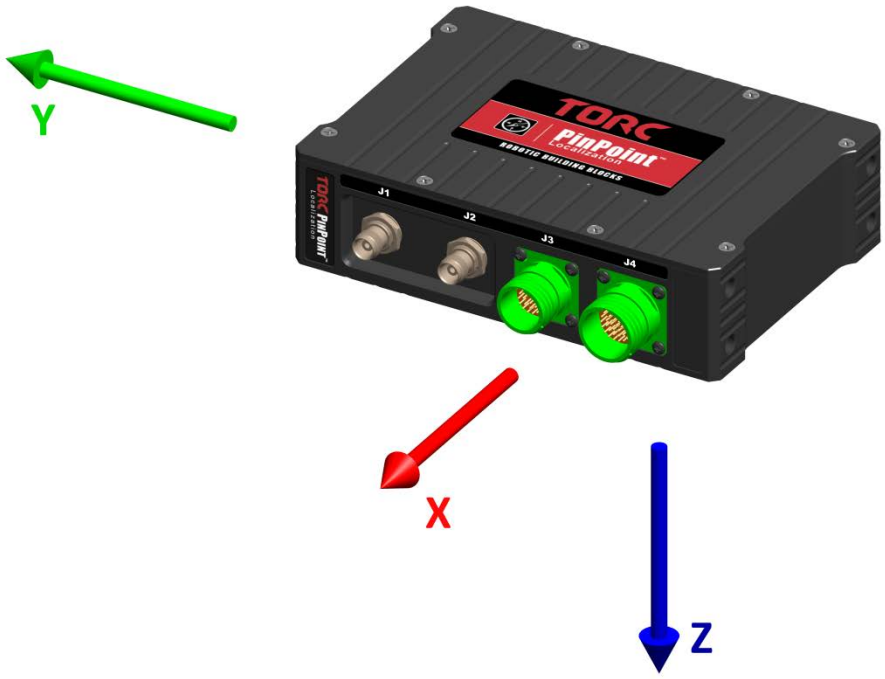


Figure 16: PinPoint™ Connector Detail and Coordinate Frame

11.1.1 J1 – Primary GPS Antenna Connector

The GPS connector is a standard polarity female TNC connector. If a using a single GPS configuration, the antenna needs to be connected to this connector. Measurements made to this antenna must be entered in the vehicle tab of the web interface.

11.1.2 J2 – Secondary GPS Antenna Connector

The GPS connector is a standard polarity female TNC connector and is used for the secondary GPS antenna connection (if equipped). Measurements made to this antenna must be entered in the vehicle tab of the web interface webpage.

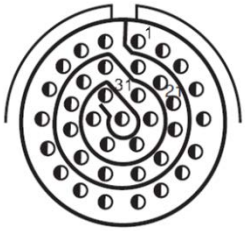
11.1.3 J3 – External IMU Connector

The external IMU connector is used for connecting a precision IMU to PinPoint™.

Pin	Name	Description	I/O	Contact
1	RX_DATA_IN+	Connection to External IMU	Input	#22D
2	RX_DATA_IN-	Connection to External IMU	Input	#22D
3	TX_DATA_OUT+	Connection to External IMU	Output	#22D
4	TX_DATA_OUT-	Connection to External IMU	Output	#22D
5	SYNC_IN+	Connection to External IMU	Input	#22D
6	SYNC_IN-	Connection to External IMU	Input	#22D
7	CLOCK_OUT+	Connection to External IMU	Output	#22D
8	CLOCK_OUT-	Connection to External IMU	Output	#22D
9	IMU_RESET_OUT	Connection to External IMU	Output	#22D
10	IMU_MODE_OUT	Connection to External IMU	Output	#22D
11	POWER_OUT	Connection to External IMU	Power	#22D
12	POWER_GND	Connection to External IMU	Ground	#22D
13	PC_RX	Factory Use Only	Output	#22D
14	PC_TX	Factory Use Only	Input	#22D
15	PC_DTR	Factory Use Only	Input	#22D
16	PC_RTS	Factory Use Only	Input	#22D
17	PC_GND	Factory Use Only	Ground	#22D
18-21	(reserved)	Not Used		#22D
22	SHIELD	Chassis Ground	Ground	#22D

11.1.4 J4 – Power and Data Connector

The power and data connector is used for powering PinPoint™, as well as all the external data signals as described in the table below.



Pin	Name	Description	I/O	Contact
1	POWER_IN	Input Power	Power	#22D
2	POWER_GND	Power Ground	Ground	#22D
3	SHIELD	Chassis Ground	Ground	#22D
4	CAN1_H	CAN High	Bidirectional	#22D
5	CAN1_L	CAN Low	Bidirectional	#22D
6	GND	Signal Ground	Ground	#22D
7	CAN2_H	CAN High	Bidirectional	#22D
8	CAN2_L	CAN Low	Bidirectional	#22D
9	GND	Signal Ground	Ground	#22D
10	ETH1_TX+	Ethernet Transmit	Output	#22D
11	ETH1_TX-	Ethernet Transmit	Output	#22D
12	ETH1_RX+	Ethernet Receive	Input	#22D
13	ETH1_RX-	Ethernet Receive	Input	#22D
14	SHIELD	Signal Ground	Ground	#22D
15	ETH2_TX+	Ethernet Transmit	Output	#22D
16	ETH2_TX-	Ethernet Transmit	Output	#22D
17	ETH2_RX+	Ethernet Receive	Input	#22D
18	ETH2_RX-	Ethernet Receive	Input	#22D
19	SHIELD	Signal Ground	Ground	#22D
20	WSS_LI+	Left In-phase Current Sense	Bidirectional	#22D
21	WSS_LI-	Left In-phase Current Return	Ground	#22D
22	WSS_LQ+	Left Quadrature Current Sense	Bidirectional	#22D
23	WSS_LQ-	Left Quadrature Current Return	Ground	#22D
24	WSS_RI+	Right In-phase Current Sense	Bidirectional	#22D
25	WSS_RI-	Right In-phase Current Return	Ground	#22D

Pin	Name	Description	I/O	Contact
26	WSS_RQ+	Right Quadrature Current Sense	Bidirectional	#22D
27	WSS_RQ-	Right Quadrature Current Return	Ground	#22D
28	PPS_232	RS-232 Pulse-Per-Second	Output	#22D
29	GND	Signal Ground	Ground	#22D
30	NMEA_232	RS-232 NMEA Transmit	Output	#22D
31	PPS_5V+	Non-Inverted Pulse-Per-Second	Output	#22D
32	PPS_5V-	Inverted Pulse-Per-Second	Output	#22D
33	SERIAL_TX-	RS-232 Transmit / RS-422 TX-	Output	#22D
34	SERIAL_TX+	RS-422 TX+	Output	#22D
35	SERIAL_RX-	RS-232 Receive / RS-422 RX-	Input	#22D
36	SERIAL_RX+	RS-422 RX+	Input	#22D
37	GND	Signal Ground	Ground	#22D

11.2 PinPoint™ Precision IMU

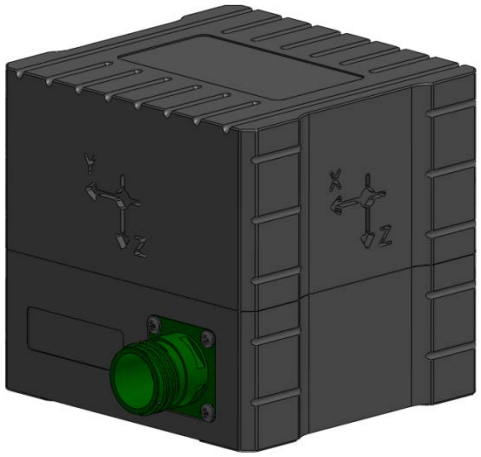


Figure 17: PinPoint™ Precision IMU

11.2.1 J1 – Precision IMU connector

The external IMU connector is used for connection the External Precision IMU to the PinPoint™ enclosure.

Pin	Name	Description	I/O	Contact
1	RX_DATA_OUT+	Connection to PinPoint™	Output	#22D
2	RX_DATA_OUT-	Connection to PinPoint™	Output	#22D
3	TX_DATA_IN+	Connection to PinPoint™	Input	#22D
4	TX_DATA_IN-	Connection to PinPoint™	Input	#22D
5	SYNC_OUT+	Connection to PinPoint™	Output	#22D
6	SYNC_OUT-	Connection to PinPoint™	Output	#22D
7	CLOCK_IN+	Connection to PinPoint™	Input	#22D
8	CLOCK_IN-	Connection to PinPoint™	Input	#22D
9	IMU_RESET_IN	Connection to PinPoint™	Input	#22D
10	IMU_MODE_IN	Connection to PinPoint™	Input	#22D
11	POWER_IN	Connection to PinPoint™	Power	#22D
12	POWER_GND	Connection to PinPoint™	Ground	#22D
13-21	(reserved)	Not Used		#22D
22	SHIELD	Chassis Ground	Ground	#22D

12. Appendix: Calculations

12.1 Coordinate System

The relationship between target points in the two frames, given the current local and global position, is as follows:

$$Target_{North} = Local_{North} + (Target_{Latitude} - Global_{Latitude})\,(R_M + Global_{Height})$$

$$Target_{East} = Local_{East} + (Target_{Longitude} - Global_{Longitude})\,(R_N + Global_{Height})\cos(Global_{Latitude})$$

$$Target_{Down} = Local_{Down} + Global_{Height} - Target_{Height}$$

Where:

Radius of curvature in the prime vertical $$R_N = \frac{a}{\sqrt{1 - e^2 \sin^2(Global_{Latitude})}}$$

Radius of curvature in the meridian $$R_M = \left(\frac{a(1-e^2)}{(1 - e^2 \sin^2(Global_{Latitude}))^{(3/2)}} \right)$$

WGS 84 equatorial radius (meters) $a = 6378137.0$

WGS 84 first eccentricity squared $e^2 = 0.00669438$

12.2 Vibration

While PinPoint's™ accuracy is dependent on a number of different factors, simply looking at long-term drift of the corrections:

$$\sigma^2_{distance} = \left(\sigma_v^2 + r^2\sigma_a^2\right) t$$

Where σ_v is the RMS linear velocity vibration in m/s, σ_a is the RMS angular velocity vibration in rad/s, r is the length in m of the moment arm between the IMU and the measurement, and t is time in s.

While this is a gross simplification of the problem, it does give an idea of the magnitude of drift from different levels of vibration.

12.3 WSS Error

The maximum velocity error, in m/s, is given by

$$error = \sqrt{\frac{a}{2 \cdot ppm}}$$

Where a is the acceleration of the vehicle and ppm is the pulses per meter.

13. Appendix: Examples

NOTE: The fixed-point I32 angles used by PinPoint to represent latitude and longitude allow for calculations to be performed with millimeter level precision. If converting these angles to a floating-point representation, make sure to use a format with enough bits to accurately represent the data, as a standard, single-precision float will only give meter-level accuracy.

13.1 Local Coordinates for Global Navigation

The global and local frame serve two different purposes. The global frame provides a georeferenced coordinate system that when averaged over time, is repeatable and reliable as is common with GPS based solutions. For navigation around nearby objects, the transient effects of GPS drifts and "pops" can be detrimental to smooth path planning and execution. The local frame is meant to address this, providing a stable and consistent frame for immediate planning at the cost of longer term repeatability.

The stability of the local frame and immunity to GPS "pops" make the local frame ideal for low-level navigation around nearby objects. On the other hand, globally referenced features, such as GPS waypoints, require the repeatability and extended position accuracy of global frame. The recommended global navigation schema is to store the global waypoints and features in global frame and transform them into local frame when needed for the immediate level planning. This should be performed on every iteration of the navigation loop using the most up-to-date vehicle positions in both frames, because over time the local and global frames may drift with respect to each other.

Given the following:

- Global (LLH) vehicle location: (37.189350˚ N, 80.391360˚ W, 635.0 m)
- Local (NED) vehicle location: (100.0 m, 300.0 m, 0.0 m)
- Global (LLH) waypoint location: (37.189000˚ N, 80.394400˚ W, 638.0 m)

First, convert degrees and direction to signed radians:

- Global (LLH) vehicle location: (0.64907660, -1.4030939, 635 m)
- Global (LLH) waypoint location: (0.64907050, -1.4031470, 638 m)

Next, calculate radii of curvature at the vehicle using the equations in section 12.1:

- $R_N = \frac{6378137.0}{\sqrt{1-0.00669438(\sin(0.64907660))^2}} = 6385951.4$
- $R_M = \frac{6378137.0(1-0.00669438)}{(1-0.00669438(\sin(0.64907660))^2)^{(3/2)}} = 6358754.0$

Then, calculate the local frame coordinates of the waypoint using the equations in section 12.1:

- $North = 100.0 + (0.64907050 - 0.64907660)(6358754.0 + 635.0) = 61.2m$
- $East = 300.0 + (-1.4031470 + 1.4030939)(6385951.4 + 635.0)\cos(0.64907660) = 30.8m$
- $Down = 0.0 + 635.0 - 638.0 = -3.0m$

13.2 Global Coordinates of Sensed Object

Often, it is desirable to store the global location of an object for future reference. Sensors that are hard-mounted to a vehicle and therefore see objects in vehicle frame (or a sensor-frame with a fixed transformation to vehicle frame), and must be rotated to local frame to account for the attitude of the vehicle before being converted to global frame.

Given the following:

- Global (LLH) vehicle location: (37.18935˚ N, 80.39136˚ W, 635.0 m)
- Local (NED) vehicle location: (100.0 m, 300.0 m, 0.0 m)
- Global / Local (RPY) attitude: (0.0˚, 10.0˚, -90.0˚)
- Vehicle (FRD) object location: (6.0 m, 0.0 m, 0.0 m)

First, transform object coordinates from vehicle frame to local frame using standard homogeneous transformations:

$$\begin{bmatrix}100.0\\300.0\\0.0\end{bmatrix}+\begin{bmatrix}\cos(-90) & -\sin(-90) & 0\\ \sin(-90) & \cos(-90) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\cos(10) & 0 & \sin(10)\\ 0 & 1 & 0\\ -\sin(10) & 0 & \cos(10)\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(0) & -\sin(0)\\ 0 & \sin(0) & \cos(0)\end{bmatrix}\begin{bmatrix}6.0\\0.0\\0.0\end{bmatrix}$$
$$=\begin{bmatrix}100.0\\294.1\\-1.0\end{bmatrix}$$

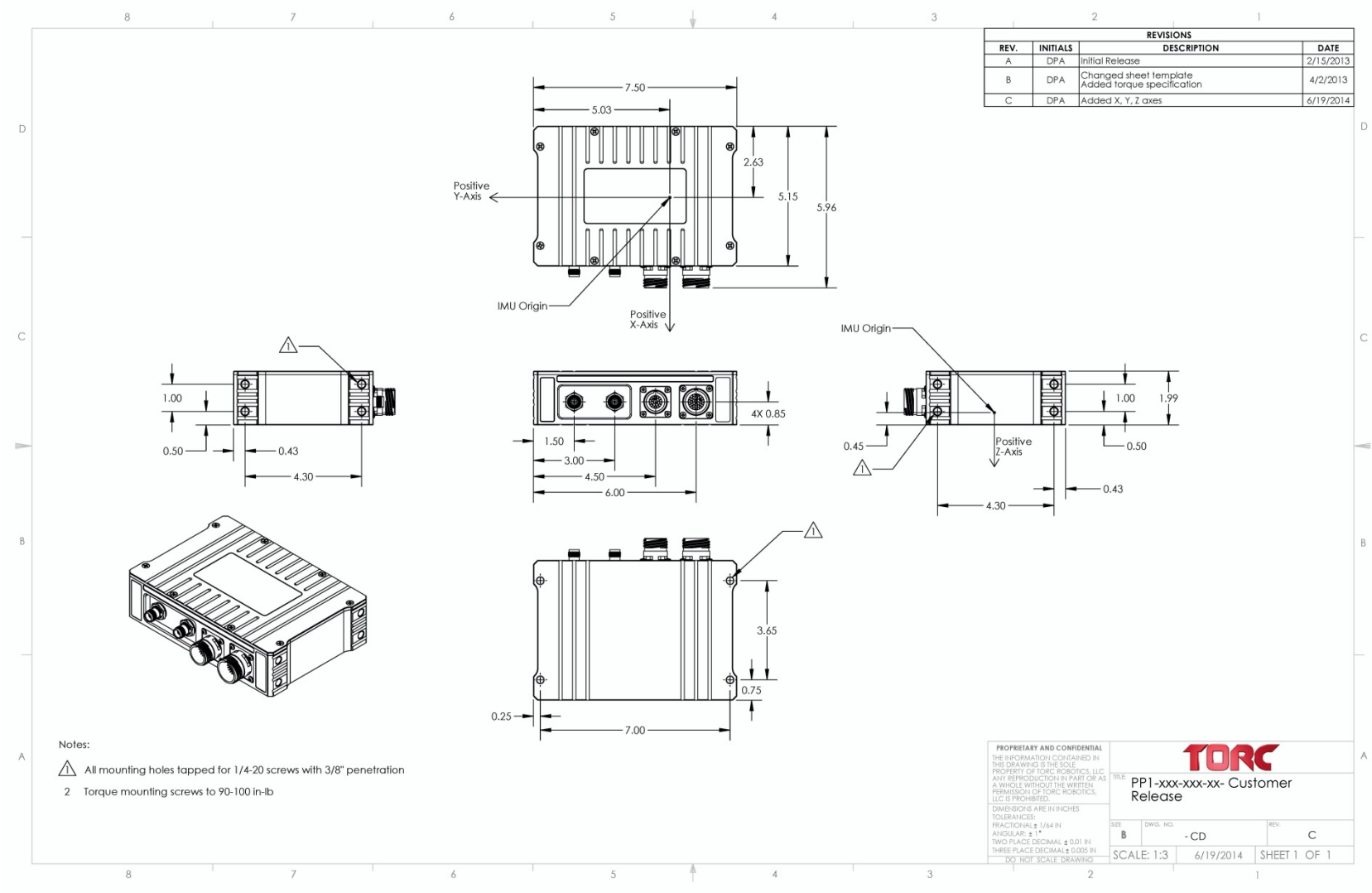
Next, calculate the global coordinates of the point in local frame using the equations in section 12.1:

- $Lat = 0.64907660 + \frac{100.0-100.0}{6358754.0+635.0} = 0.64907660$
- $Lon = -1.4030939 + \frac{294.1-300.0}{(6385951.4+635.0)\cos(0.64907660)} = -1.4030951$
- $Alt = 635.0 + (0.0 + 1.0) = 636.0$

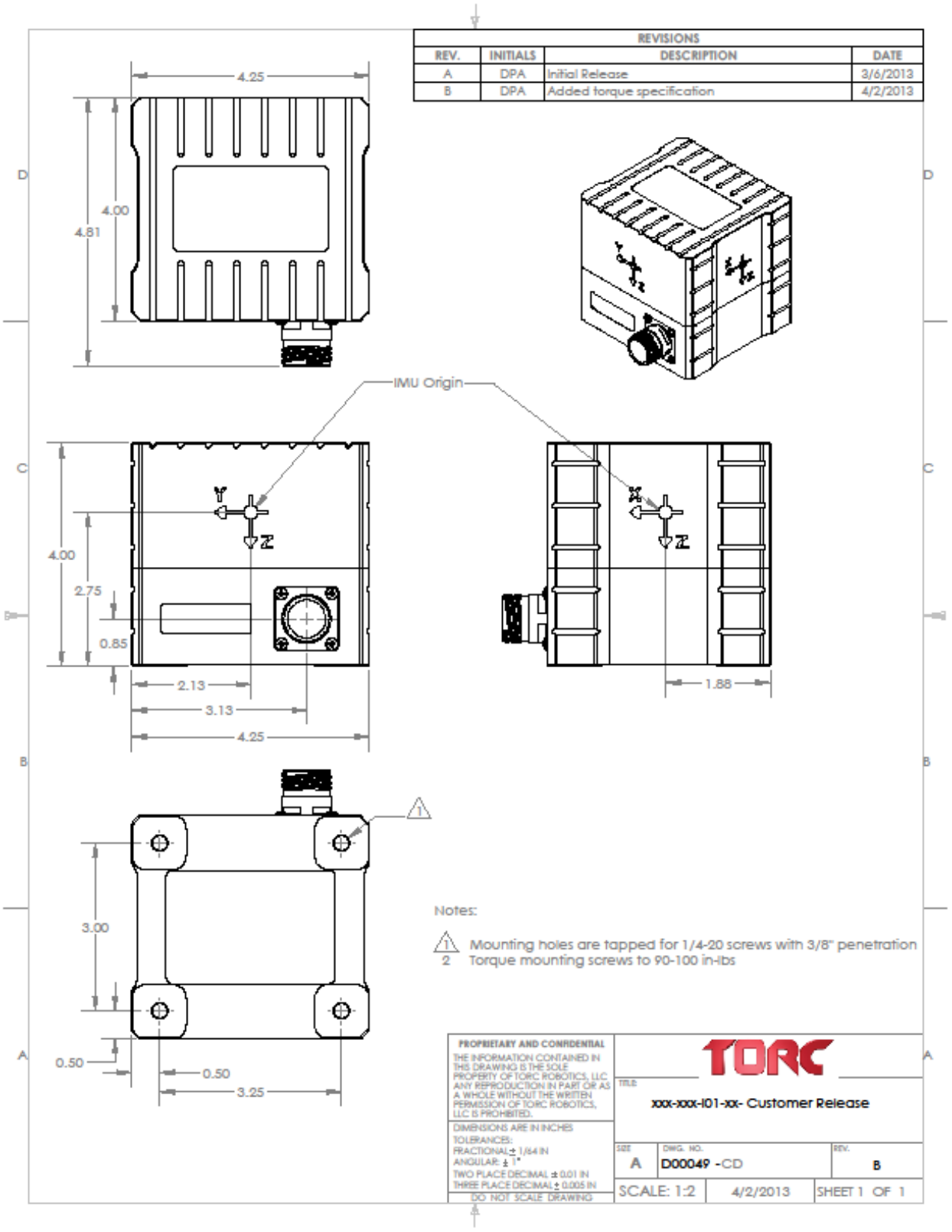
Then, switch back to degrees:

- Global (LLH) object location: (37.189350˚ N, 80.391425˚ W, 636.0 m)

14. Appendix: Mechanical Dimensions



15. Appendix: IMU Mechanical Dimensions



16. Appendix: Antenna Dimensions