SPACEPOINT™

SpacePoint Scout

User Manual

PNI SENSOR CORPORATION

Table of Contents

List of Figures

List of Tables

1 Copyright & Warranty Information

Revised January 2013: for the most recent version visit our website at www.pnicorp.com

PNI Sensor Corporation
2331 Circadian Way
Santa Rosa, CA 95407, USA
Tel: (707) 566-2260
Fax: (707) 566-2261

Warranty and Limitation of Liability. PNI Sensor Corporation ("PNI") manufactures its Products from parts and components that are new or equivalent to new in performance. PNI warrants that each Product to be delivered hereunder, if properly used, will, for ninety (90) days following the date of shipment unless a different warranty time period for such Product is specified: (i) in PNI's Price List in effect at time of order acceptance; or (ii) on PNI's web site (www.pnicorp.com) at time of order acceptance, be free from defects in material and workmanship and will operate in accordance with PNI's published specifications and documentation for the Product in effect at time of order. PNI will make no changes to the specifications or manufacturing processes that affect form, fit, or function of the Product without written notice to the Customer, however, PNI may at any time, without such notice, make minor changes to specifications or manufacturing processes that do not affect the form, fit, or function of the Product. This warranty will be void if the Products' serial number, or other identification marks have been defaced, damaged, or removed. This warranty does not cover wear and tear due to normal use, or damage to the Product as the result of improper usage, neglect of care, alteration, accident, or unauthorized repair.
THE ABOVE WARRANTY IS IN LIEU OF ANY OTHER WARRANTY, WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE. PNI NEITHER ASSUMES NOR AUTHORIZES ANY PERSON TO ASSUME FOR IT ANY OTHER LIABILITY.

If any Product furnished hereunder fails to conform to the above warranty, Customer's sole and exclusive remedy and PNI's sole and exclusive liability will be, at PNI's option, to repair, replace, or credit Customer's account with an amount equal to the price paid for any such Product which fails during the applicable warranty period provided that (i) Customer promptly notifies PNI in writing that such Product is defective and furnishes an explanation of the deficiency; (ii) such Product is returned to PNI's service facility at Customer's risk and expense; and (iii) PNI is satisfied that claimed deficiencies exist and were not caused by accident, misuse, neglect, alteration, repair, improper installation, or improper testing. If a Product is defective, transportation charges for the return of the Product to Customer within the United States and Canada will be paid by PNI. For all other locations, the warranty excludes all costs of shipping, customs clearance, and other related charges. PNI will have a reasonable time to make repairs or to replace the Product or to credit Customer's account. PNI warrants any such repaired or replacement Product to be free from defects in material and workmanship on the same terms as the Product originally purchased.

Except for the breach of warranty remedies set forth herein, or for personal injury, PNI shall have no liability for any indirect or speculative damages (including, but not limited to, consequential, incidental, punitive and special damages) relating to the use of or inability to use this Product, whether arising out of contract, negligence, tort, or under any warranty theory, or for infringement of any other party's intellectual property rights, irrespective of whether PNI had advance notice of the possibility of any such damages, including, but not limited to, loss of use, revenue or profit. In no event shall PNI's total liability for all claims regarding a Product exceed the price paid for the Product. PNI neither assumes nor authorizes any person to assume for it any other liabilities.

Some states and provinces do not allow limitations on how long an implied warranty lasts or the exclusion or limitation of incidental or consequential damages, so the above limitations or exclusions may not apply to you. This warranty gives you specific legal rights and you may have other rights that vary by state or province.

2 Introduction

Thank you for purchasing PNI Sensor Corporation's SpacePoint Scout motion-tracking module, pn 13317. We're certain you'll be impressed with its performance.

The SpacePoint Scout outputs precise data regarding the module's orientation. This can be used for a variety of motion-tracking applications, most notably for motion-based controllers for TV and set-top box navigation, and for video gaming. The impressive performance of the SpacePoint Scout allows a manufacturer to competitively differentiate their products by easily incorporating intuitive, accurate, and low-latency motion control into their systems.

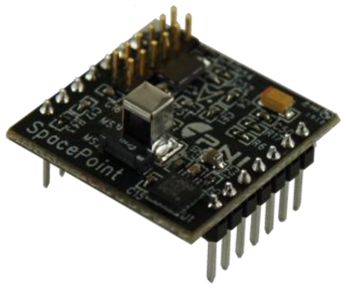


Figure 2-1: SpacePoint Scout Module

Today's controllers are becoming more intuitive with the addition of gyros and accelerometers, but these sensors alone result in drift and inaccurate pointing. As shown in Figure 2-2, the SpacePoint Scout incorporates PNI's Geomagnetic Sensor Suite with a 3-axis gyroscope and 3-axis accelerometer, resulting in accurate motion tracking with no gyro drift. The system outputs are internally processed using PNI's SpacePoint algorithm to provide calculated orientation data in the form of quaternions, and horizontal and vertical cursor positions.

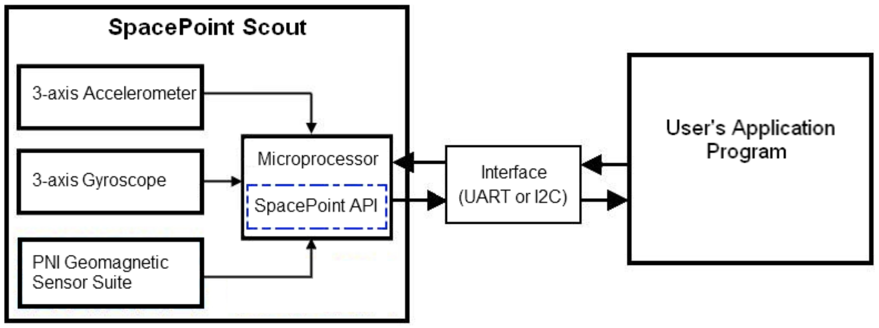


Figure 2-2: SpacePoint Scout System Functional Diagram

3 Specifications

3.1 Characteristics & Requirements

Table 3-1: Sensor Components

Sensor Type	Manufacturer	Model
3-Axis Gyroscope	ST Microelectronics	L3G4200D
3-Axis Accelerometer	ST Microelectronics	LIS3LV02DL
3-Axis Magnetic Sensor	PNI Sensor Corporation	RM3000-f

Table 3-2: I/O Characteristics

Parameter	Value
Supply Voltage (V_{in})	3.8 to 10 VDC
Communication Interface	UART & I^2C
Communication Protocol	PNI Binary
Communication Data Rate	115200 baud
Output Data Rate	125 Hz
High-Level Input	+2 V to +3.8 V
Low-Level Input	-0.5 V to +1.1 V
High-Level Output	2.4 V min. @ 8 mA
Low-Level Output[1]	0.4 V max. @ 8 mA

Footnote:

1. For I^2C communication, the Scout incorporates an internal 4.7 kΩ pull-up resistor.

Table 3-3: Environmental Requirements

Parameter	Value
Operating Temperature[2]	-40C to +85C
Storage Temperature	-40C to +85C

Footnote:

2. The SpacePoint Scout can operate across this temperature range, but performance will vary across the range.

Table 3-4: Mechanical Characteristics

Parameter	Value
Dimensions (*l* x *w* x *h*)	25.4 x 25.4 x 16.0 mm
Weight	6 gm
Connector	7 pin SIP, 0.1” Header

3.2 Mechanical Drawing

Dimensions in mm

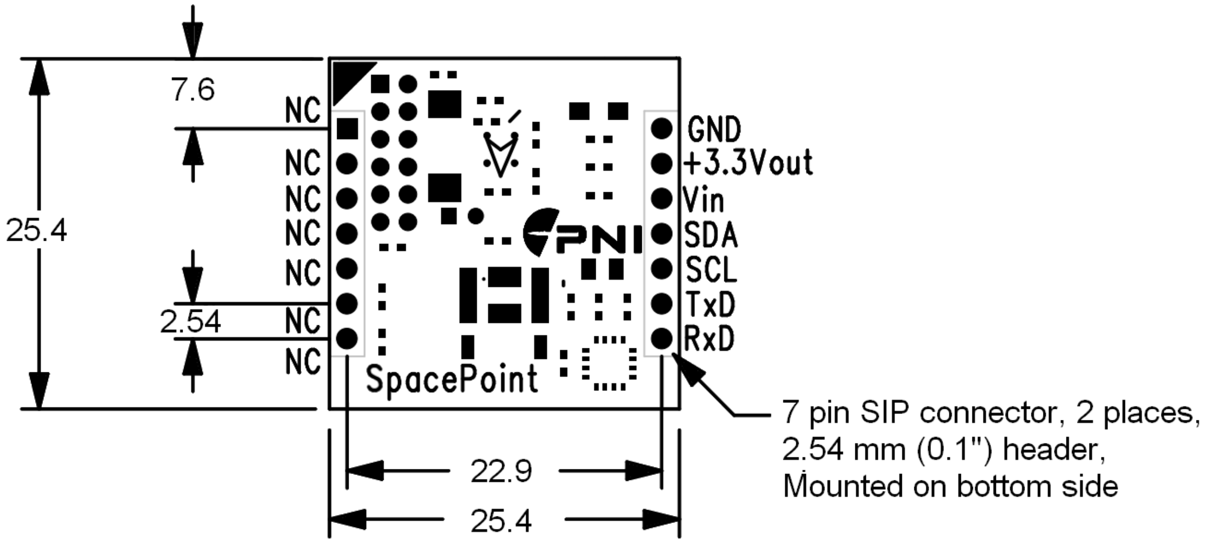


Figure 3-1: Scout Mechanical Drawing

4 Using the SpacePoint Scout

The SpacePoint Scout supports both UART and I^2C interfaces for communicating with the user's host system. These interfaces are discussed in Sections 4.2 and 4.3, respectively, which follow a discussion on setting up the SpacePoint Scout.

PNI's CommBoard (pn 13466) can be mated with the SpacePoint Scout and this, in combination with the SpacePoint Scout Test Program, provides a quick way to start working with the SpacePoint Scout. See the PNI CommBoard User Manual and Section 6 for additional information.

4.1 Setup

First, identify the arrowhead on the SpacePoint Scout PCB, then mount the Scout within the host system such that the arrowhead points in the intended direction-of-travel or line-of-sight. Figure 3-1 shows the pin-out. Note that TxD and RxD should be used for UART communication, while SCL and SDA should be used for I^2C communication. A couple of points regarding mounting:

- Mounting should be such that the SpacePoint Scout is isolated from persistent accelerations, such as vibration or rapid directional changes. The Scout is intended to provide accurate motion tracking while experiencing intermittent accelerations. However, the Scout will have difficulty accurately tracking motion over extended periods of time if it is constantly subjected to accelerations.
- While the Scout can compensate for transient changes in the local magnetic field, it is good design practice to keep the Scout away from sources of local magnetic distortion that knowingly will change with time; such as speakers, electrical equipment that will be turned on and off, or ferrous bodies that will move.

Finally, it is important to ensure the Scout is at rest when initially powered on, and remains at rest for 3 seconds after powering on. The gyroscopes initialize during this period, and movement of the module will distort the initialization process.

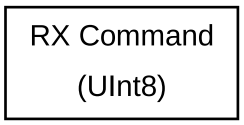
4.2 UART Communication

The SpacePoint Scout allows for communication with the host system via a UART interface. Through the UART , the user's system requests the output from the SpacePoint Scout and the module subsequently transmits this data. The outputs include quaternions, which prescribe the orientation of the module, and the horizontal and vertical position for a cursor. See Section 5 for how to interpret the outputs.

Table 4-1: UART Configuration

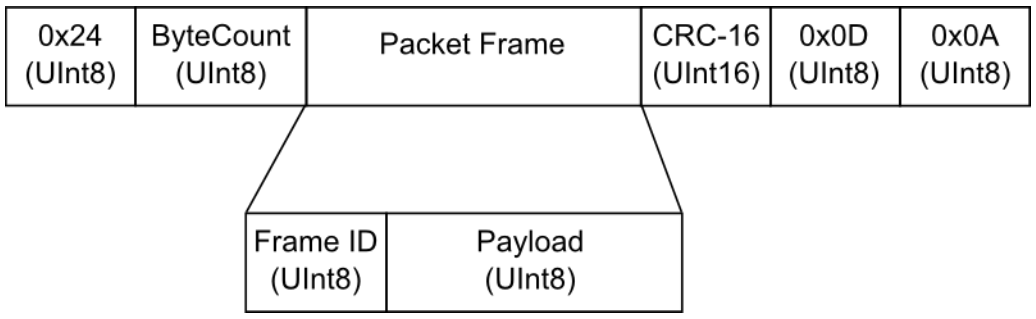
Parameter	Value
Baud Rate	115200
Data Bits	8
Parity	none
Stop Bits	1
Flow Control	none

The receive (Rx) and transmit (Tx) data structures and data frames for the UART interface are shown below in Figure 4-1 and Figure 4-2. All data is presented as unsigned integers in little Endian format. (For I^2C communication the format is different, being big Endian.) Note that the receive command is a single byte frame.



	Rx Command	Description
kRequestQ	0x32	Request quaternion data from SpacePoint algorithm.
kResetRef	0x34	Reset Reference Frame

Figure 4-1: Receive Data Structure



	Frame ID	Description
kRequestQResp	0x32	Output SpacePoint quaternion data.

Figure 4-2: Transmit Data Structure

kRequestQ & kRequestQResp

The command to receive data (kRequestQ) is 0x32. Once the SpacePoint algorithm receives the kRequestQ command, the algorithm will operate in push mode, such that the response frame is constantly transmitted at 125 Hz. The structure of the kRequestQResp frame is given below in Table 4-2.

Table 4-2: Structure of UART kRequestQResp frame

Byte	Description
1	Leading Byte = 0x24
2	Byte Count = 0x1A
3	Frame ID = 0x32
4	Accel X LSB
5	Accel X MSB
6	Accel Y LSB
7	Accel Y MSB
8	Accel Z LSB
9	Accel Z MSB
10	Quaternion X LSB
11	Quaternion X MSB
12	Quaternion Y LSB
13	Quaternion Y MSB
14	Quaternion Z LSB
15	Quaternion Z MSB
16	Quaternion W LSB
17	Quaternion W MSB
18	Horizontal Position (Hpos) LSB
19	Horizontal Position (Hpos) MSB
20	Vertical Position (Vpos) LSB
21	Vertical Position (Vpos) MSB
22	Reserved
23	CRC – 16 Upper Byte
24	CRC – 16 Lower Byte
25	Upper Trailing Byte = 0x0D
26	Lower Trailing Byte = 0x0A

Note: *The CRC mentioned in this document adheres to a 16-bit Fletcher algorithm. See http://en.wikipedia.org/wiki/Fletcher%27s_checksum.*

kResetRef

Cursor position tracking is relative to a reference established by the user. Normally a user will point the SpacePoint Scout at the center of a screen, then press a keyboard button or a button on the user's device that resets the cursor to the center of the screen. The UART command to set the cursor at the center of the screen is 0x34.

4.3 I^{2}C Communication

The SpacePoint Scout also supports communication with the host system via an I^{2}C interface. The module acts as the Slave device and the user's processor acts as the Master device. The slave address (HwADDR) is 0x18. All data is presented as unsigned integers in big Endian format. (For UART communication the format is different, being little Endian.) Command sequences are given below for polling data and resetting the cursor reference frame.

Polling Quaternion Data

Master	ST	HwADDR		0x31		SR	HwADDR+1			MAK
Slave			SAK		SAK			SAK	QX MSB	

Master		MAK		MAK		MAK		MAK	
Slave	QX LSB		QY MSB		QY LSB		QZ MSB		QZ LSB

Master	MAK		MAK		NMAK	SP
Slave		QW MSB		QW LSB		

Polling Cursor Position (Hpos and Vpos) Data

Master	ST	HwADDR		0x33		SR	HwADDR+1			MAK
Slave			SAK		SAK			SAK	Reserved	

Master		MAK		MAK		MAK		MAK
Slave	Reserved		Reserved		Reserved		Reserved	

Master		MAK		MAK		MAK		MAK
Slave	Reserved		Reserved		Reserved		Reserved	

Master		MAK		MAK		MAK		MAK

Slave	Reserved		Reserved		Reserved		HPOS MSB	

Master		MAK		MAK		MAK	
Slave	HPOS MSB-1		HPOS LSB+1		HPOS LSB		VPOS MSB

Master	MAK		MAK		MAK		NMAK	SP
Slave		VPOS MSB-1		VPOS LSB+1		VPOS LSB		

Establishing the Cursor Reference Frame (ResetRef)

Cursor position tracking is relative to a reference established by the user. Normally the user will point the SpacePoint Scout at the center of a screen, then press a keyboard button or a button on the user's device that will reset the cursor to the center of the screen. The command sequence to set the cursor at the center of the screen is given below.

Master	ST	HwADDR		0xB4		SP
Slave			SAK		SAK	

5 Converting and Interpreting the Output

The various outputs provided by the SpacePoint Scout are in an unscaled format. Table 5-1 provides conversion and scaling information for both the UART and I^{2}C interfaces. Further discussion on SpacePoint Quaternions and Cursor Position follows.

Table 5-1: Scaling of Output Data

Output	Interface	Equation	Units or Range
Quaternions	UART & I^{2}C	q_scaled = (q_output - 32768)/32768	Range: –1.0 to 1.0
Acceleration	UART	Accel_g = (Accel_output –32768)*6/32768	g

Hpos & Vpos	UART	Hpos_scaled = (Hpos_output -32768)/32768
		Vpos_scaled = (Vpos_output -32768)/32768
			Range: –0.5 to +0.5

Hpos & Vpos	I^{2}C	See IEEE – 754	Range: –0.5 to +0.5

5.1 SpacePoint Quaternions

Rotation quaternions are 4 element vectors which describe the rotation of an object with a single angle, Φ, and a rotation axis, v ≡ [vx vy vz]. For the SpacePoint algorithm, the rotation quaternion is defined as:

$$q \equiv [qx\ qy\ qz\ qw] = [q_0\ q_1\ q_2\ q_3], \text{ where}$$
$$qx = vx * \sin(\Phi/2)$$
$$qy = vy * \sin(\Phi/2)$$
$$qz = vz * \sin(\Phi/2)$$
$$qw = \cos(\Phi/2)$$

Note that *qw* often is referred to as the scalar term of the quaternion. Also, other definitions of a rotation quaternion exist, such as *q ≡ [qw qx qy qz]*.

For the SpacePoint algorithm, the rotation quaternion represents the absolute rotation from the NED (North East Down) reference orientation. The NED reference orientation has the x axis pointed north, the y axis pointed east, and the z axis pointing down. If the SpacePoint Scout is held level with the front of the device pointing towards magnetic north, then q = [0 0 0 1], since Φ = 0.

5.2 Cursor Position

The cursor position values, Hpos and Vpos, are used to map the SpacePoint Scout's orientation to a cursor's position on a TV or computer screen.

Standard practice is for Hpos and Vpos values to range from –0.5 to +0.5, such that "0.0" is the center of the screen, Hpos = –0.5 represents the left side of the screen, and Vpos = 0.5 represents the bottom of the screen.

The formulas in Table 5-1 show how to covert the raw Hpos and Vpos output to properly scaled values, assuming a 16:9 screen ratio. Consequently, an ~60° rotation in the horizontal plane will move the cursor from the left side of the screen to the right side, while an ~34° rotation in the vertical plane will move the cursor from the top to the bottom of the screen.

Before using the position data, it is first necessary to set the SpacePoint Scout orientation with respect to the screen. This is done by pointing the device at the center of the screen, then sending the ResetRef command using either the UART or I^2C interface.

6 SpacePoint Scout Test Program

The SpacePoint Scout Test Program is intended to demonstrate the functionality and performance of the SpacePoint Scout. The program is built with the Unity3D Game Engine and will run on Windows XP, Windows Vista, and Windows7 computers. To help ensure optimal performance, a dedicated graphics card is recommended.

PNI generally recommends mating the Scout to PNI's CommBoard (pn 13466) to run the SpacePoint Scout Test Program, as this simplifies the set up. However this is not necessary as the program operates via the UART interface, and the user can directly run the program using the Scout's UART interface.

6.1 Set Up

6.1.1 Software

Ensure the following conditions are met:

- The "SpacePointTestProgram.exe" program and its associated file folder "SpacePointTestProgram_Data" are located on your computer's local hard drive and are in the same directory.
- Microsoft's .NET Framework and Visual C++ Redistributable are installed on your computer. If you are uncertain if they are installed, go to the "Control Panel", then "Add or Remove Programs". You should see "Microsoft Visual C++ Redistributable – x86" and either "Microsoft .NET Framework 3.5" or "Microsoft .NET Framework 4 Client Profile" in the list of currently installed programs. If either is missing, they can be downloaded at Microsoft's website: http://www.microsoft.com/downloads/en/default.aspx. In the event the Test program does not work properly, uninstall all versions of Microsoft .NET Framework and reinstall it.
- Close all other applications when running the program to ensure optimal performance. Also, it is advisable to operate the program on a computer with a single screen.

6.1.2 Hardware – Using PNI CommBoard

If using the PNI CommBoard, plug the Scout into the CommBoard and configure the jumpers as shown in Figure 6-1.

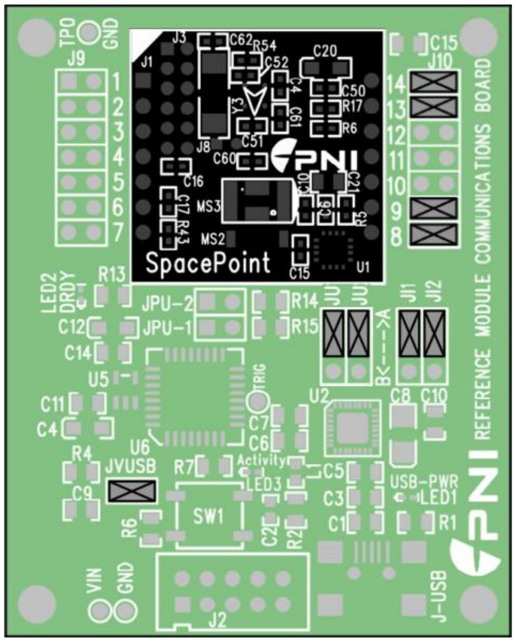


Figure 6-1: PNI CommBoard Jumper Configuration

6.1.3 Hardware – Direct UART Connection

If incorporating the SpacePoint Scout directly into a host system, the UART interface should be directly implemented. This involves following the set-up instructions discussed in Sections 4.1 and 4.2, and specifically requires soldering or otherwise connecting the GND, Vin, TxD, and RxD pins on the Scout to the host system, as indicated in Figure 3-1.

6.2 Powering Up the SpacePoint Scout

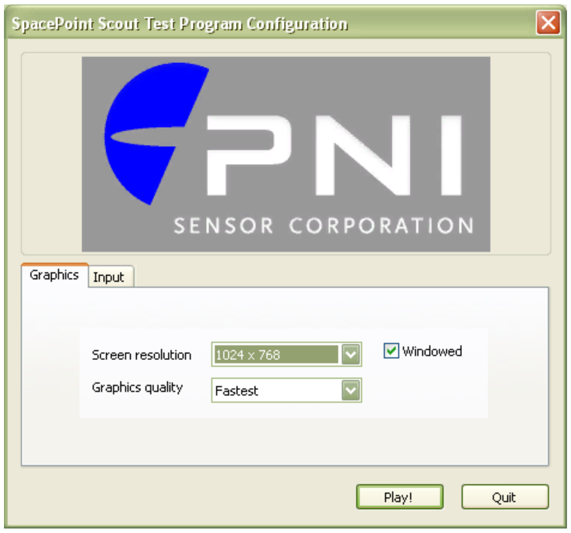
If you are using the PNI CommBoard, connect the CommBoard to your computer using a USB-to-mini-USB cable. The red and blue LEDs on the CommBoard should light up. Press the Reset button (SW1) on the CommBoard to ensure the computer is correctly interpreting your device's communication protocol

If the Scout is integrated into a host system and the UART is directly implemented, then supply power to Vin.

Note: *Ensure the SpacePoint Scout is fully at rest when powering up, and that it remains so for at least 3 seconds afterwards. This is so the gyro bias properly stabilizes during the initial power-on.*

6.3 Launching the Test Program

Launch the program by double-clicking on "SpacePointTestProgram.exe". The program Configuration window will appear, as shown below. The default settings normally work well, but you can change them if you prefer. Note that if "Windowed" is unchecked the program will launch in full screen mode. Click <Play!> to proceed.


SpacePoint Scout Test Program Configuration

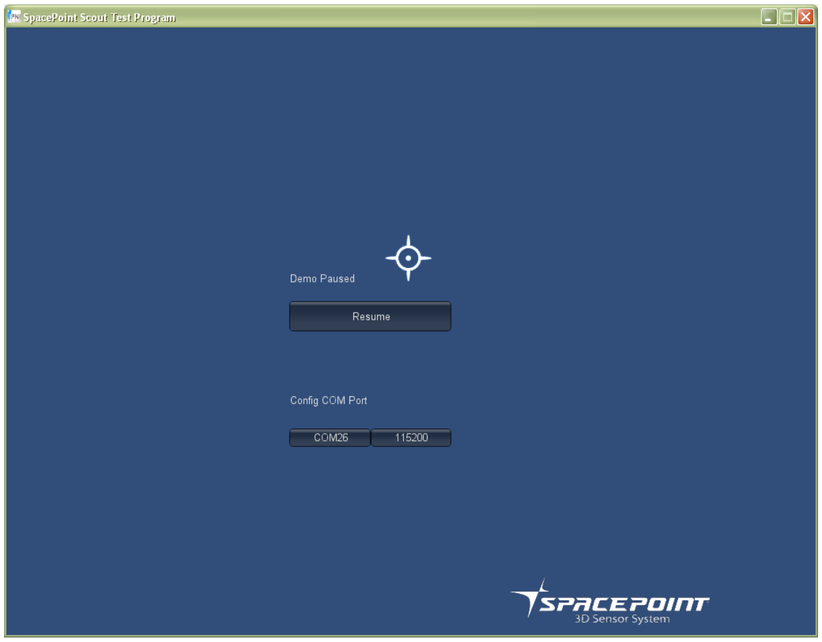
Graphics | Input

Screen resolution: 1024 x 768

Graphics quality: Fastest

Windowed ✓

Now the port configuration screen will appear, as shown below. Set the COM Port to the appropriate port. If you are uncertain which port this is, in Windows go to the Device Manager and click on Ports to see which Port shows a USB Serial Port. Set the baud rate to 115200. Click <Resume>.

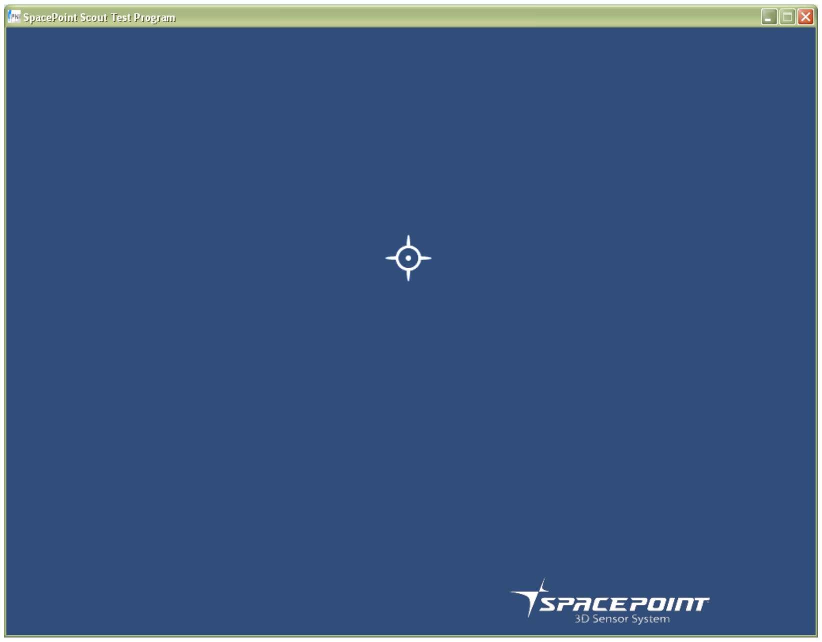

SpacePoint Scout Test Program

Demo Paused

Config COM Port: COM26 | 115200

6.4 Running the Point Tracking Screen

The Point Tracking screen will now open, as shown below.



Initialization: Point the SpacePoint Scout at the center of the screen, and then press the '4' key. This establishes the relationship between the orientation of the device and the cursor position (ResetRef). Next press the '2' key to start outputting data.

Viewing data: To view output data, press "H" so the data window appears. In the data box you will see data outputs from the calculated accelerometer readings (AOut), the quaternions (QOut), and the horizontal and vertical positions (hvPos). Pressing "H" again will toggle back to the Point Tracking screen without the data window.

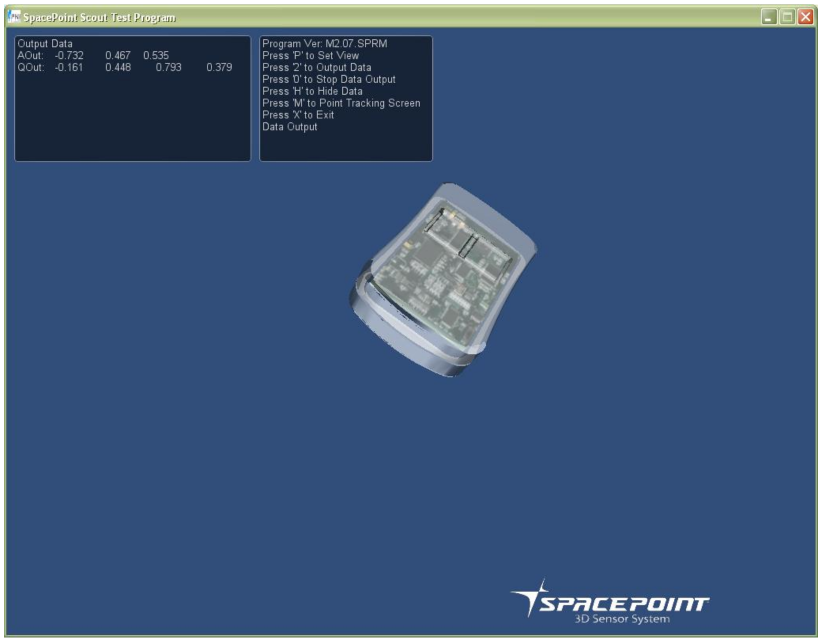
Running the Program: After pressing '4' and '2', the cursor will track the orientation of the SpacePoint Scout. Point the device to the left or right, and the cursor will move to the left or right. This will happen with very low latency and precise motion detection. If you point the device so the cursor goes off screen, the device will continue tracking the absolute orientation of the device such that when you later point the device at the center of the screen the cursor will return to the center of the screen.

If you want to re-establish the relationship between the orientation of the SpacePoint Scout and the cursor position, press '4' again. This will stop data from outputting and the cursor will freeze where it was when you pressed '4'. Press '2' again to start outputting data. If you did not change the orientation of the device since pressing the '4' key, the cursor will move to the center of the screen.

To stop outputting data press ‘0’.

6.5 Running the Motion Tracking Screen

The SpacePoint Scout Test Program also provides a 3-D rendering of the SpacePoint Demonstration Module. This is accessed by pressing “M” to toggle to the Motion Tracking screen. Pressing the “H” key brings up the data window, just as on the Point Tracking screen. Press the ‘2’ key to start outputting data. Note that the rendered image will not change, even though data is being output. Initialize and align the viewpoint in the Motion Tracking screen by pressing ‘P’. Now the rendered image will track the orientation of the device.



```
Output Data
AOut:   -0.732      0.467   0.535
QOut:   -0.161      0.448       0.793       0.379
```

```
Program Ver: M2.07.SPRM
Press 'P' to Set View
Press '2' to Output Data
Press '0' to Stop Data Output
Press 'H' to Hide Data
Press 'M' to Point Tracking Screen
Press 'X' to Exit
Data Output
```

Pressing ‘M’ again toggles you back to the Point Tracking Screen.

6.6 Exiting the Program & Summary of Commands

To exit the program, press the “x” key on your computer’s keyboard. Table 6-1 summarizes the various commands for the SpacePoint Scout Test Program.

Table 6-1: Summary of SpacePoint Scout Test Program Commands

Key	Command
“2”	Start serial communication between SpacePoint Scout and computer
“0”	Stop serial communication between SpacePoint Scout and computer
“H”	Toggle to show or hide the data screen
“M”	Toggle between the Point Tracking and Motion Tracking screens
“P”	Set orientation in Motion Tracking screen
“4”	Center cursor in Point Tracking screen
“X”	Exit program